German Research Center for Artificial Intelligence

ROBOT LOCALIZATION

M. Sc. Mihaela Popescu
Robotics Innovation Center
DFKI Bremen

Prof. Dr. Dr. h.c. Frank Kirchner
Arbeitsgruppe Robotik, Universität Bremen
https://robotik.dfki-bremen.de/
robotik@dfki.de

23th November, 2022 – Bremen, Deutschland

Universität Bremen

Contents

Introduction

Mapping and Localization

Autonomous mobile robots are able to navigate through the environment.

In order to perform navigation, robots first need to answer two questions:

- Mapping - "How does the world look like?"
- Localization - "Where am I in this world?"

$\rightarrow$ Today's focus: Localization.

German Research Center for Artificial Intelligence

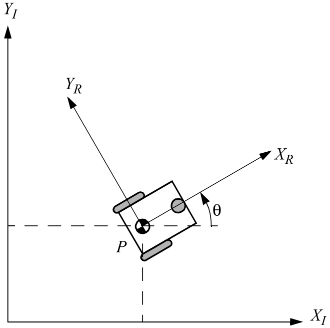
What is Localization?

- Given: Map (model) of the environment.
- Task: Estimate the robot pose relative to the given map.

German Research Center for Artificial Intelligence

What is Localization?

- Given: Map (model) of the environment.
- Task: Estimate the robot pose relative to the given map.
- **Robot pose:**
  - position (x_R, y_R)
  - orientation θ.



Universität Bremen

Why Do Robots Need Localization?

Autonomous mobile robots need to localize into the world in order to perform the following tasks:

- Map the environment.
- Plan a path to a goal position.
- Navigate autonomously.

Sensors for Localization

Sensors for Localization

German Research Center for Artificial Intelligence

Proprioceptive Sensors

- Wheel encoder → Wheel odometry
- Inertial Measurement Unit (IMU) → Inertial odometry

Exteroceptive Sensors

- Camera → Visual odometry
- Laser scanner, sonar → Scan matching

German Research Center for Artificial Intelligence

Proprioceptive Sensors

1. Wheel Odometry (Dead Reckoning)

- Wheel encoder: count the wheel rotations over time.

Pros:

- High frequency measurements.
- Cheap and lightweight sensor.

Cons:

- Drift due to wheel slippage.
- Only reliable on short term.

German Research Center for Artificial Intelligence

Proprioceptive Sensors

2. Inertial Odometry

- Inertial Measurement Unit (IMU): integrate the linear acceleration and angular velocity measurements to obtain robot position and orientation.

Pros:

- High frequency measurements.
- Low processing time.

Cons:

- Drift due to time integration.
- Only reliable on short term.

German Research Center for Artificial Intelligence

Exteroceptive Sensors

3. Visual Odometry

- Estimate robot pose use two (consecutive) camera images.
- Monocular / stereo camera.
- Direct / feature-based methods.

Pros:

- Provides rich information.
- Cheap and accessible sensor.

Cons:

- Sensitive to varying light conditions.
- Fails for fast camera motions.

German Research Center for Artificial Intelligence

Exteroceptive Sensors

4. Lidar-based Localization

- Laser scanner: perform scan matching to find the transformation between two point clouds. $\rightarrow$ See lecture on Robot Mapping.

Pros:

- Provides depth information.
- Robust under various world conditions.

Cons:

- Reduced frequency.
- Limited range.

Universität Bremen

German Research Center for Artificial Intelligence

External Sensors

5. Direct Localization

- Global Navigation Satellite System GNSS (e.g. GPS).
- Triangulation (camera motion tracking system).
- Trilateration (external beacons with known position).

Pros:

- Highly accurate measurements.
- Provide global localization.

Cons:

- GNSS signal obstructed by tall buildings or forests.
- Triangulation and trilateration require setup of the environment.

Algorithms for Probabilistic Localization

- Based on Bayesian statistics.
- Methods: Kalman filter, **particle filter (Monte Carlo Localization)**.
- Sensors measurements: wheel encoder, IMU, GPS, laser scanner, camera, etc.

Pros:

- Model sensor noise.
- Fuse multimodal sensor data.

Cons:

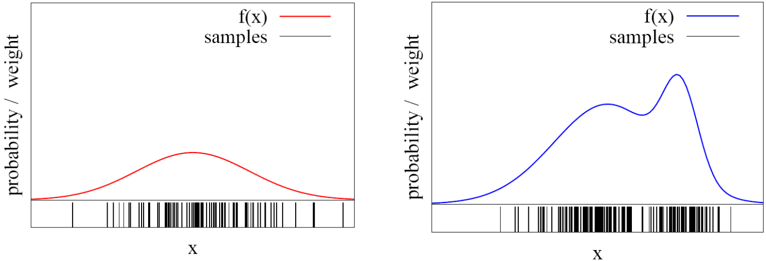
- Complex algorithms and models.
- Computationally expensive.

Particle Filter

Function Approximation

- Probabilistic method used for non-parametric **function approximation**.
- An arbitrary function can be described by a set of M **particles** at time t:

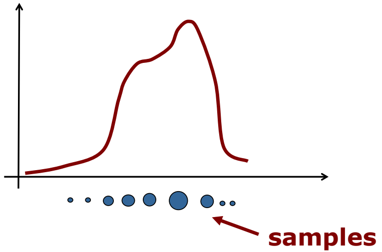
$$\mathcal{X}_t := \left\{ x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(M)} \right\}$$



German Research Center for Artificial Intelligence

Particle Filter for Localization

- Definition: non-parametric, **recursive Bayes filter** that estimates a posterior distribution based on noisy measurements.
- In robot localization, every sample (particle) in the particle filter represents a hypothesis of the **robot position**.
- The more samples, the better the robot position estimate.



Universität Bremen

Monte Carlo Localization

German Research Center for Artificial Intelligence

Introduction

- Localization method using **particle filter** to represent the posterior distribution. (Fox et al., 1999)
- One of the most popular localization algorithms.
- Applicable to both local and global localization problems.
- Various extensions exist to address shortcomings.

Universität Bremen

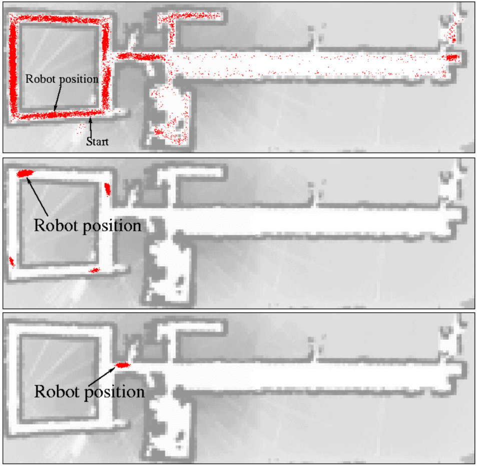
Prerequisites

- **Occupancy grid map** of the environment.
- **Wheel odometry** to estimate the robot motion model.
- **Laser scanner** sensor measurements.

German
Research Center
for Artificial
Intelligence

Prerequisites

- **Occupancy grid map** of the environment.
- **Wheel odometry** to estimate the robot motion model.
- **Laser scanner** sensor measurements.

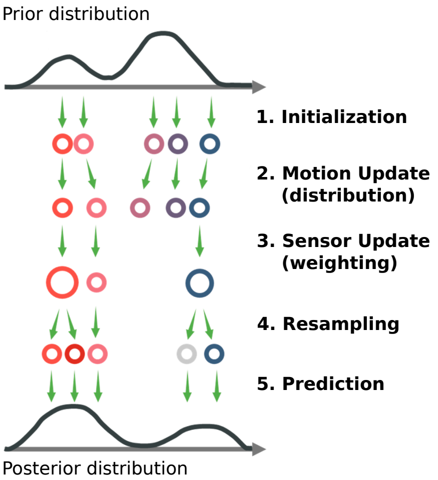


Universität
Bremen

Monte Carlo Localization

Particle Filter Steps

1. Initialization: draw samples from prior distribution.
2. Motion Update: distribute samples according to the **motion model**.
3. Sensor Update: assign weights to particles based on **sensor model**.
4. Resampling: draw particles according to their weights.
5. Prediction: output the estimated posterior distribution.



Monte Carlo Localization

Parameters

$\mathcal{X}_{t-1}$ = set of particles at time $t-1$

u_t = wheel odometry at time t

z_t = laser scan at time t

m = occupancy grid map

M = number of particles

$x_t^{[m]}$ = particle m at time t

$w_t^{[m]}$ = weight of particle m at time t

Algorithm MCL$(\mathcal{X}_{t-1}, u_t, z_t, m)$**:**
$\quad \bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
$\quad$ for $m = 1$ to M do
$\quad\quad x_t^{[m]} =$ **sample_motion_model**$(u_t, x_{t-1}^{[m]})$
$\quad\quad w_t^{[m]} =$ **measurement_model**$(z_t, x_t^{[m]}, m)$
$\quad\quad \bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
$\quad$ endfor
$\quad$ for $m = 1$ to M do
$\quad\quad$ draw i with probability $\propto w_t^{[i]}$
$\quad\quad$ add $x_t^{[i]}$ to $\mathcal{X}_t$
$\quad$ endfor
$\quad$ return $\mathcal{X}_t$

resampling

Pseudocode of MCL algorithm.

Monte Carlo Localization

Parameters

$\mathcal{X}_{t-1}$ = set of particles at time $t-1$

u_t = wheel odometry at time t

z_t = laser scan at time t

m = occupancy grid map

M = number of particles

$x_t^{[m]}$ = particle m at time t

$w_t^{[m]}$ = weight of particle m at time t

Algorithm MCL($\mathcal{X}_{t-1}, u_t, z_t, m$)**:**
    $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
    for $m = 1$ to M do
        $x_t^{[m]} =$ **sample_motion_model**$(u_t, x_{t-1}^{[m]})$
        $w_t^{[m]} =$ **measurement_model**$(z_t, x_t^{[m]}, m)$
        $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
    endfor
    for $m = 1$ to M do
        draw i with probability $\propto w_t^{[i]}$
        add $x_t^{[i]}$ to $\mathcal{X}_t$
    endfor
    return $\mathcal{X}_t$

sensor model

Pseudocode of MCL algorithm.

German Research Center for Artificial Intelligence

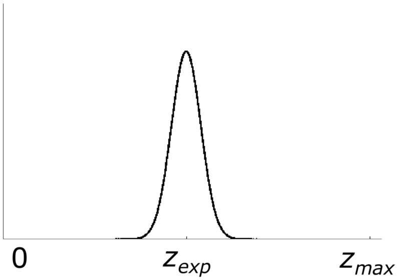
Sensor Model

In order to use the laser scans for localization, we have to define the sensor model and take the possible measurement errors into account:

- z_hit: Beams reflected by obstacles.
- z_short: Beams reflected by persons / caused by crosstalk.
- z_rand: Random measurements.
- z_max: Maximum range measurements.

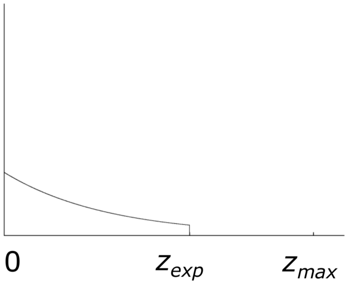
German Research Center for Artificial Intelligence

Measurement noise



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

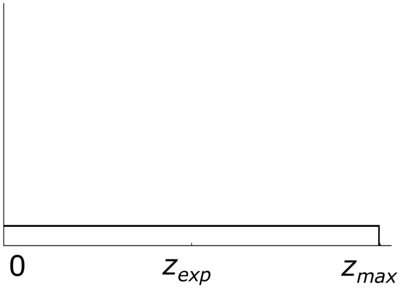
Unexpected obstacles



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

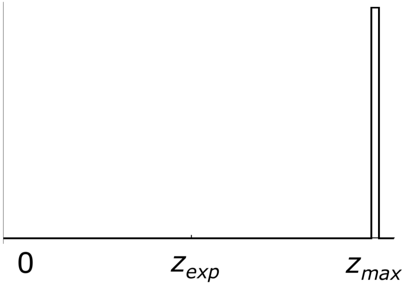
Monte Carlo Localization

Random measurement



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

Max range

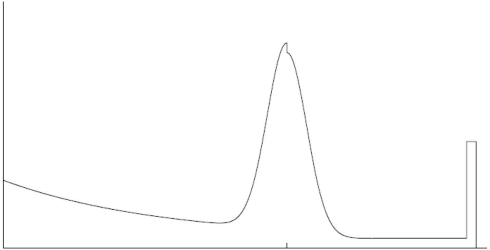


$$P_{\max}(z|x,m) = \begin{cases} 1 & z = z_{\max} \\ 0 & \text{otherwise} \end{cases}$$

German Research Center for Artificial Intelligence

Sensor Model

- The overall probability is given by the sum of the 4 probabilities.
- The parameters α are scaling factors which sum up to 1.



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

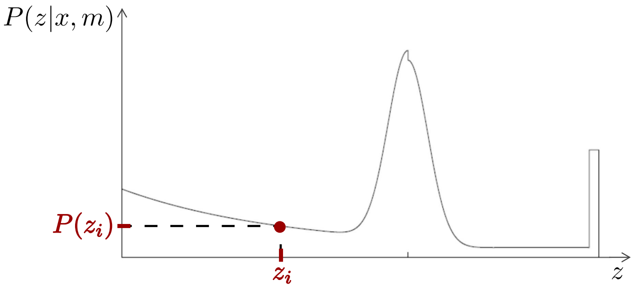
Sensor model after including all 4 types of measurement errors.

Universität Bremen

German Research Center for Artificial Intelligence

Sensor Model

- For every laser beam measurement z_i, look up its probability $P(z_i|x, m)$.
- The probability distribution of the measurement z is the product of all individual laser beam probabilities: $P(z|x, m) = \prod_i P(z_i|x, m)$.



Universität Bremen

Monte Carlo Localization

Algorithm MCL$(\mathcal{X}_{t-1}, u_t, z_t, m)$**:**

$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$

for $m = 1$ to M do

$x_t^{[m]} =$ **sample_motion_model**$(u_t, x_{t-1}^{[m]})$ motion model

$w_t^{[m]} =$ **measurement_model**$(z_t, x_t^{[m]}, m)$ sensor model

$\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$

endfor

for $m = 1$ to M do

draw i with probability $\propto w_t^{[i]}$ resampling

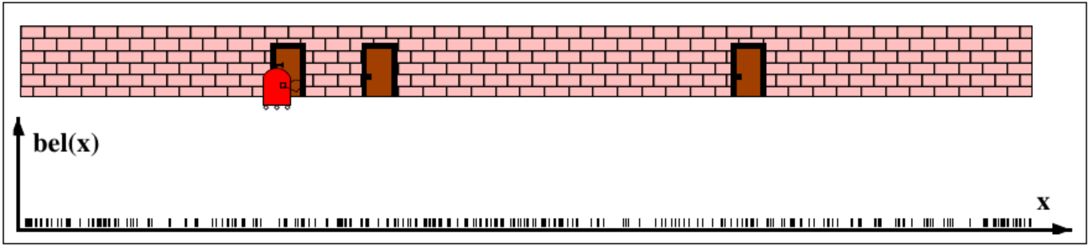
add $x_t^{[i]}$ to $\mathcal{X}_t$

endfor

return $\mathcal{X}_t$

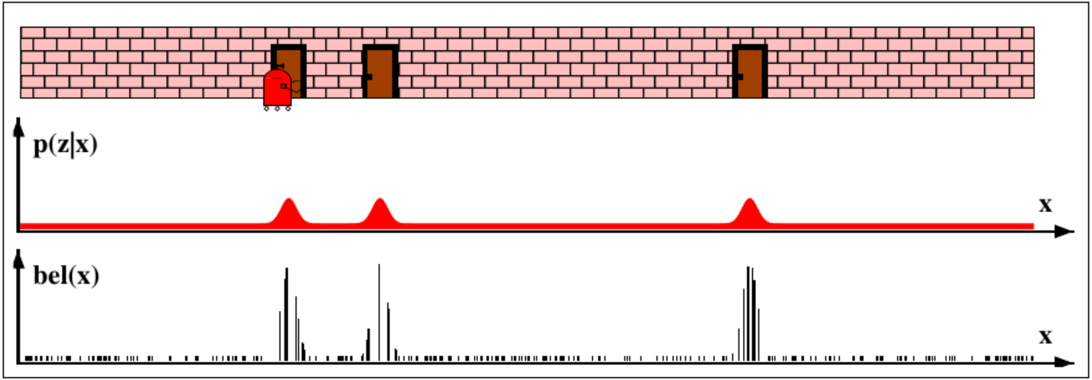
Pseudocode of MCL algorithm.

MCL Example (1)



Initialize by sampling poses from uniform distribution.

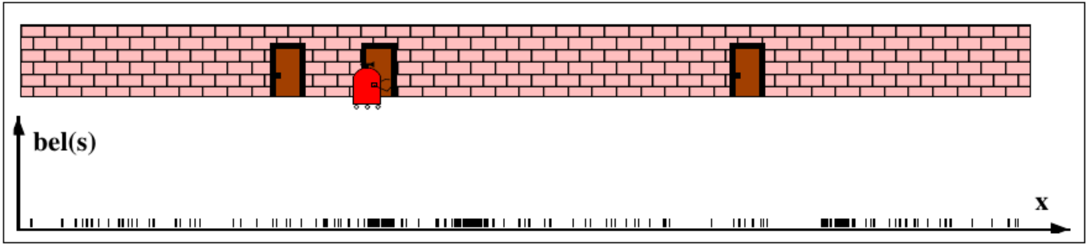
MCL Example (2)



Sensor update: assign importance weights to each particle.

Universität Bremen

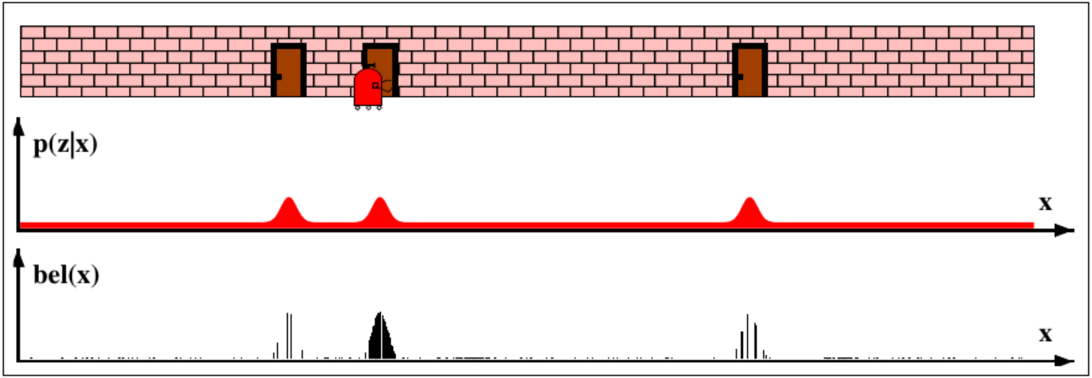
MCL Example (3)



Resample the particle set based on weights, then **motion update** (apply noisy motion transformation to each particle).

MCL Example (4)

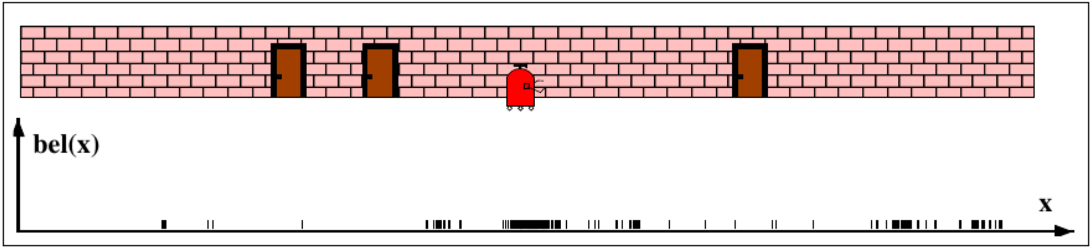
German Research Center for Artificial Intelligence



Next **sensor update**.

MCL Example (5)

German
Research Center
for Artificial
Intelligence



Resample and **motion update**.

Universität
Bremen

Monte Carlo Localization

German Research Center for Artificial Intelligence

Pros

- Estimates any posterior distribution (i.e. not limited to Gaussian distribution).
- Able to cope with noisy sensor data and inaccurate odometry.
- Easy to implement.

Cons

- Large number of particles slows down localization.
- Requires large storage space.
- High computational resources.

Universität Bremen

Further Problems

- We need to keep a random distribution throughout the state space by using a sufficient amount of particles.
- Without **addition of random particles**, MCL can fail if the particles converge to an incorrect pose.
- After all particles converged to the robot location they become redundant.
- Problematic for **high-dimensional spaces**, many particles slow down localization.

German Research Center for Artificial Intelligence

Further Problems

- We need to keep a random distribution throughout the state space by using a sufficient amount of particles.
- Without **addition of random particles**, MCL can fail if the particles converge to an incorrect pose.
- After all particles converged to the robot location they become redundant.
- Problematic for **high-dimensional spaces**, many particles slow down localization.

$\rightarrow$ Solution: **Adaptive** Monte Carlo Localization

German Research Center for Artificial Intelligence

Concept

- The Adaptive MCL is a variant of the standard MCL algorithm. (Fox, 2002)
- It **dynamically adjusts the number of particles** in the filter based on the certainty of the robot localization.
- The number of particles is decreased when the position estimate has higher certainty (when particles converge to robot pose).
- Allows a trade-off between processing speed and localization accuracy.

Universität Bremen

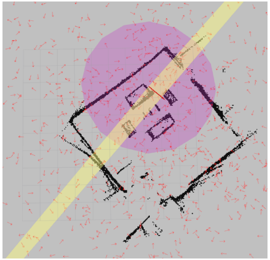
German Research Center for Artificial Intelligence

AMCL ROS Package

- We will use the amcl package from the ROS2 Navigation Stack.
- Parameters that can be configured:
  - Minimum and maximum number of particles.
  - Initial robot pose and covariance.
  - Laser scanner model parameters:
    - min/max range.
    - sensor model: z_hit, z_short, z_rand, z_max.
- Link: `https://navigation.ros.org/configuration/packages/configuring-amcl.html`

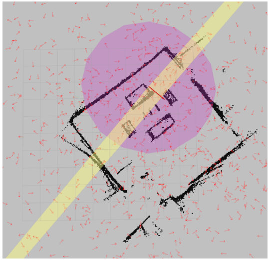
Particle Convergence

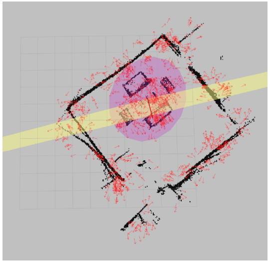
- Particles are initially spread out randomly over the entire map.
- Particles converge over time to the true robot location.
- The ellipses represent the uncertainty of the position and orientation estimates.



Particle Convergence

- Particles are initially spread out randomly over the entire map.
- Particles converge over time to the true robot location.
- The ellipses represent the uncertainty of the position and orientation estimates.

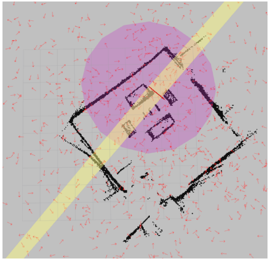


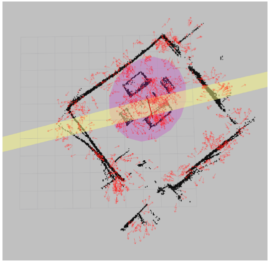


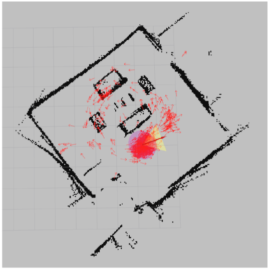
German Research Center for Artificial Intelligence

Particle Convergence

- Particles are initially spread out randomly over the entire map.
- Particles converge over time to the true robot location.
- The ellipses represent the uncertainty of the position and orientation estimates.



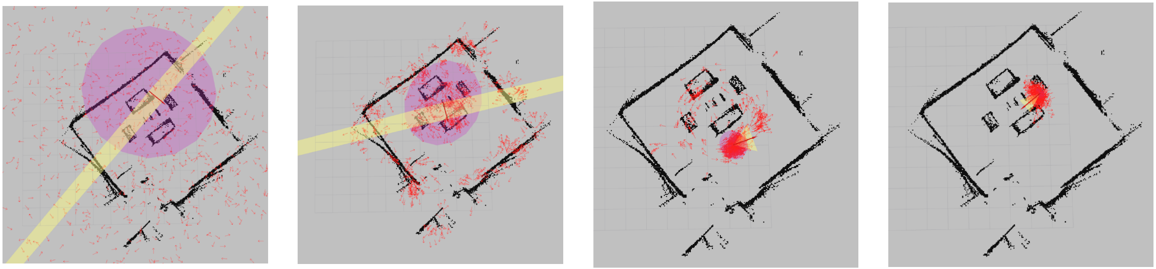




German Research Center for Artificial Intelligence

Particle Convergence

- Particles are initially spread out randomly over the entire map.
- Particles converge over time to the true robot location.
- The ellipses represent the uncertainty of the position and orientation estimates.



Challenges

German Research Center for Artificial Intelligence

Sensor Errors

- All sensors have errors (noise, bias, limited resolution/range).
- Sensor models are not perfect.
- Constraints:
  - Cost.
  - Size/Weight/Power.
  - Measurement frequency.
  - Usability of sensors depends on environment.

German Research Center for Artificial Intelligence

Global Localization

- Goal: estimate absolute robot position in the entire working space.
- Initial pose unknown (i.e. equally distributed over the map).
  $\rightarrow$ Determine robot pose under global uncertainty.
- Use sensor measurements to generate clusters of possible robot poses.
  $\rightarrow$ Hypotheses where the robot can be.
- Implement a strategy by which the robot can correctly eliminate all hypotheses except the right location (e.g. particle filter).

German Research Center for Artificial Intelligence

Position Tracking

- Assume initially known starting position.
- Update the current pose based on the previous pose.
- Errors add up over multiple iterations.
- Can lead to localization drifts.

Kidnapped Robot Problem

- It occurs when the robot is displaced to an unknown location in the map.
- Localization sensors (eg. wheel encoder, laser scanner) not aware of kidnapping. $\rightarrow$ Localization fails.
- This problem needs to be first recognized and then handled.
- Solution: add a few random uniformly distributed samples in the particle filter to recover, otherwise the robot will keep resampling from the wrong distribution.

Conclusion

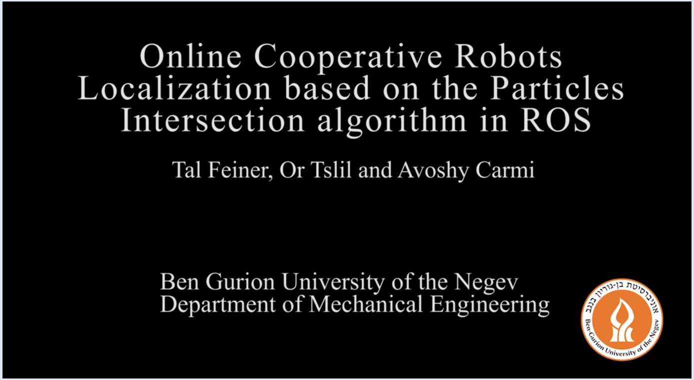
German Research Center for Artificial Intelligence

Collaborative Localization

- In a multiagent system, robots can cooperate to improve their localization.
- Algorithm proposed by the Ben Gurion University of the Negev (Israel) in 2019:
  - Robots use **particle filter** for localization and **Extended Kalman Filter (EKF)** to track the other robots.
  - When two or more robots are in each others' field of view, they fuse their particle filters with a method called **Particles Intersection**. (Tslil and Carmi, 2018)
  - Localization package available in ROS: `http://wiki.ros.org/mcl_pi`.

German Research Center for Artificial Intelligence

Collaborative Localization



Online Cooperative Robots Localization in ROS.
Source: `http://wiki.ros.org/mcl_pi`

Universität Bremen

German Research Center for Artificial Intelligence

Summary

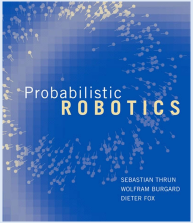
- Introduction to robot localization.
- Proprioceptive and exteroceptive sensors for localization.
- Particle filter for function approximation.
- Monte Carlo Localization & Adaptive MCL.
- Challenges (global localization, kidnapped robot problem).
- Collaborative localization.

Universität Bremen

Additional Literature

Probabilistic Robotics, Thrun et al.

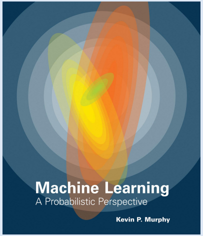
- Chapter 4.2: The Particle Filter.
- Chapter 8.3: MCL.



Source: `https://docs.ufpr.br/~danielsantos/ProbabilisticRobotics.pdf`

Machine Learning: A Probabilistic Perspective, Kevin P. Murphy

- Chapter 23.5: Particle filtering.



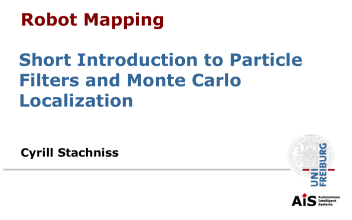
Source: `http://noiselab.ucsd.edu/ECE228/Murphy_Machine_Learning.pdf`

German Research Center for Artificial Intelligence

Additional Literature

Robot Mapping Course, Uni Freiburg

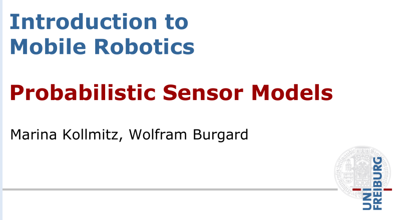
- Particle Filter and MCL.



Source: http://ais.informatik.uni-freiburg.de/teaching/ws12/mapping/pdf/slam09-particle-filter.pdf

Introduction to Mobile Robotics

- Probabilistic Sensor Models.



Source: http://ais.informatik.uni-freiburg.de/teaching/ss19/robotics/slides/07-sensor-models.pdf

References

Fox, D. (2002). « KLD-Sampling: Adaptive Particle Filters ». In: *Advances in Neural Information Processing Systems*. Ed. by T. Dietterich, S. Becker, and Z. Ghahramani. Vol. 14. MIT Press.

Fox, D., Burgard, W., Dellaert, F., and Thrun, S. (1999). « Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. ». In: *AAAI/IAAI*. Ed. by J. Hendler and D. Subramanian. AAAI Press / The MIT Press, pp. 343–349.

Tslil, O. and Carmi, A. (2018). « Information Fusion Using Particles Intersection ». In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4269–4273.

Next: Path Planning.