Localization II

Roland Siegwart, Margarita Chli, Martin Rufli
Probabilistic Reasoning (e.g. Bayesian)

- Reasoning in the presence of uncertainties and incomplete information
- Combining preliminary information and models with learning from experimental data

\[
p(x|y) = \frac{p(y|x)p(x)}{p(y)}
\]
Map Representation | Continuous Line-Based

a) Architecture map
b) Representation with set of finite or infinite lines
Map Representation | *Exact cell decomposition*

- Exact cell decomposition - Polygons
Map Representation | Approximate cell decomposition

- Fixed cell decomposition (occupancy grid)
  - Narrow passages disappear
Map Representation | *Adaptive cell decomposition*

- Fixed cell decomposition
  - Narrow passages disappear

![Map Diagram](image)
Kalman Filter Localization | in summary

1. **Prediction (ACT)** based on previous estimate and odometry
2. **Observation (SEE)** with on-board sensors
3. **Measurement prediction** based on prediction and map
4. **Matching** of observation and map
5. **Estimation** → position update (posteriori position)

Observation: Probability of making this observation

Prediction: Robot’s belief before the observation

Estimation: Robot’s belief update

Observation: Probability of making this observation

\[ p(x) \]
Localization: Probabilistic Position Estimation
(Kalman Filter: continuous, recursive and very compact)
**ACT** | using motion model and its uncertainties

\[
\overline{\text{bel}}(x_t) = \sum_{x_{t-1}} p(x_t | u_t, x_{t-1}) \text{bel}(x_{t-1})
\]

**prior belief**

\[
\text{bel}(x_{t-1})
\]

**uncertain motion (odometry)**

\[
p(u_t)
\]

**prediction update**

\[
\text{ACT}
\]

**convolution**
estimation of position based on perception and map

\[ \text{bel}(x_t) \]  

\text{prediction update}

\[ p(z_t \mid x_t, M) \]  

\text{perception}

\[ \text{bel}(x_t) = \eta p(z_t \mid x_t, M) \text{bel}(x_t) \]  

\text{measurement update}

\[ \text{Multiplication and normalization (}\eta\text{)} \]
# Kalman Filter Localization

*Markov versus Kalman localization*

<table>
<thead>
<tr>
<th></th>
<th><strong>Markov</strong></th>
<th><strong>Kalman</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PROS</strong></td>
<td>- localization starting from any unknown position</td>
<td>- Tracks the robot and is inherently very precise and efficient</td>
</tr>
<tr>
<td></td>
<td>- recovers from ambiguous situation</td>
<td></td>
</tr>
<tr>
<td><strong>CONS</strong></td>
<td>- However, to update the probability of all positions within the whole state space at any time requires a discrete representation of the space (grid). The required memory and calculation power can thus become very important if a fine grid is used.</td>
<td>- If the uncertainty of the robot becomes too large (e.g. collision with an object) the Kalman filter will fail and the position is definitively lost</td>
</tr>
</tbody>
</table>
Map Representation | Approximate cell decomposition

- Occupancy grid example
  - 0 indicates that the cell has not been hit by any ranging measurements (free space)
  - 1 indicates that the cell has been hit one or multiple times by ranging measurements (occupied space)
  - Can change over time (e.g. dynamic obstacles)

Courtesy of S. Thrun
Markov Localization
Case Study – Grid Map

- Example 2: Museum
  - Laser scan 1

Courtesy of W. Burgard
Markov Localization
Case Study – Grid Map

- Example 2: Museum
  - Laser scan 2

Courtesy of W. Burgard
Markov Localization
Case Study – Grid Map

- Example 2: Museum
  - Laser scan 3

Courtesy of W. Burgard
Markov Localization
Case Study – Grid Map

- Example 2: Museum
  - Laser scan 13

Courtesy of
W. Burgard
Markov Localization
Case Study – Grid Map

- Example 2: Museum
  - Laser scan 21

Courtesy of W. Burgard
Drawbacks of Markov localization

- Planar motion case
  - is a three-dimensional grid-map array
  - cell size must be chosen carefully.

- During each prediction and measurement steps
  - all the cells are updated
  - the computation can become too heavy for real-time operations.

- Example
  - 30x30 m environment;
    - cell size of 0.1 m x 0.1 m x 1 deg
      → 300 x 300 x 360 = 32.4 million cells!
      → Important processing power needed
      → Large memory requirement
Drawbacks of Markov localization

- Reducing complexity
  - Various approaches have been proposed for reducing complexity
    - One possible solution would be to increase the cell size at the expense of localization accuracy.
    - Another solution is to use an adaptive cell decomposition instead of a fixed cell decomposition.

- Randomized Sampling / Particle Filter
  - The main goal is to reduce the number of states that are updated in each step
  - Approximated belief state by representing only a ‘representative’ subset of all states (possible locations)
  - E.g. update only 10% of all possible locations
  - The sampling process is typically weighted, e.g. put more samples around the local peaks in the probability density function
  - However, you have to ensure some less likely locations are still tracked, otherwise the robot might get lost
Map Representation | *Topological map*

- A topological map represents the environment as a graph with nodes and edges.
  - Nodes correspond to spaces
  - Edge correspond to physical connections

- Topological maps lack scale and distances, but topological relationships (e.g., left, right, etc.) are maintained.

![Diagram of a topological map showing nodes and edges connected to represent the environment's layout.](image-url)
Map Representation | *Topological map*

- London underground map
Map Representation | Example: Automatic Map building of topological map using **vision**

- Use line and color features
Map Representation | Example: Automatic Map building of topological map using vision
Markov Localization: Case Study - Topological Map (1)

- The Dervish Robot
- Topological Localization
Markov Localization: Case Study - Topological Map (2)

- Topological map of office-type environment

<table>
<thead>
<tr>
<th></th>
<th>Wall</th>
<th>Closed door</th>
<th>Open door</th>
<th>Open hallway</th>
<th>Foyer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nothing detected</td>
<td>0.70</td>
<td>0.40</td>
<td>0.05</td>
<td>0.001</td>
<td>0.30</td>
</tr>
<tr>
<td>Closed door detected</td>
<td>0.30</td>
<td>0.60</td>
<td>0</td>
<td>0</td>
<td>0.05</td>
</tr>
<tr>
<td>Open door detected</td>
<td>0</td>
<td>0</td>
<td>0.90</td>
<td>0.10</td>
<td>0.15</td>
</tr>
<tr>
<td>Open hallway detected</td>
<td>0</td>
<td>0</td>
<td>0.001</td>
<td>0.90</td>
<td>0.50</td>
</tr>
</tbody>
</table>
Markov Localization: Case Study - Topological Map (3)

- Update of believe state for position $n$ given the percept-pair $i$
  
  \[ p(n|i) = p(i|n)p(n) \]

- $p(n|i)$: new likelihood for being in position $n$
- $p(n)$: current believe state
- $p(i|n)$: probability of seeing $i$ in $n$ (see table)

- No action update!
  
  However, the robot is moving and therefore we can apply a combination of action and perception update

  \[ p(n_t|i_t) = \int p(n_t|n'_{t-1}, i_t)p(n'_{t-1})dn'_{t-1} \]

- $t-i$ is used instead of $t-1$ because the topological distance between $n'$ and $n$ is very depending on the specific topological map
Markov Localization: Case Study - Topological Map (4)

- The calculation

\[ p(n_t|n'_{t-i}, i_t) \]

is realized by multiplying the probability of generating perceptual event \( i \) at position \( n \) by the probability of having failed to generate perceptual event \( s \) at all nodes between \( n' \) and \( n \).

\[ p(n_t|n'_{t-i}, i_t) = p(i_t, n_t) \cdot p(\emptyset, n_{t-1}) \cdot p(\emptyset, n_{t-2}) \cdot \ldots \cdot p(\emptyset, n_{t-i+1}) \]
Markov Localization: Case Study - Topological Map (5)

- Example calculation
  - Assume that the robot has two nonzero belief states
    - \( p(1-2) = 1.0 \); \( p(2-3) = 0.2 \)
    and that it is facing east with certainty
  - **Perceptual event**: open hallway on its left and open door on its right
  - State 2-3 will progress potentially to 3, 3-4 or 4.
  - State 3 and 3-4 can be eliminated because the likelihood of detecting an open door is zero.
  - The likelihood of reaching state 4 is the product of the initial likelihood \( p(2-3) = 0.2 \), (a) the likelihood of detecting anything at node 3 and the likelihood of detecting a hallway on the left and a door on the right at node 4 and (b) the likelihood of detecting a hallway on the left and a door on the right at node 4. (for simplicity we assume that the likelihood of detecting nothing at node 3-4 is 1.0)
  - (a) occurs only if Dervish fails to detect the door on its left at node 3 (either closed or open), \( [0.6 \cdot 0.4 + (1-0.6) \cdot 0.05] \) and correctly detects nothing on its right, 0.7.
  - (b) occurs if Dervish correctly identifies the open hallway on its left at node 4, 0.90, and mistakes the right hallway for an open door, 0.10.
  - This leads to:
    - \( 0.2 \cdot [0.6 \cdot 0.4 + 0.4 \cdot 0.05] \cdot 0.7 \cdot [0.9 \cdot 0.1] \) \( \rightarrow p(4) = 0.003 \).
    - Similar calculation for progress from 1-2 \( \rightarrow p(2) = 0.3 \).
Markov Localization: *Case Study - Topological Map (5)*

<table>
<thead>
<tr>
<th></th>
<th>Wall</th>
<th>Closed door</th>
<th>Open door</th>
<th>Open hallway</th>
<th>Foyer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nothing detected</td>
<td>0.70</td>
<td>0.40</td>
<td>0.05</td>
<td>0.001</td>
<td>0.30</td>
</tr>
<tr>
<td>Closed door detected</td>
<td>0.30</td>
<td>0.60</td>
<td>0</td>
<td>0</td>
<td>0.05</td>
</tr>
<tr>
<td>Open door detected</td>
<td>0</td>
<td>0</td>
<td>0.90</td>
<td>0.10</td>
<td>0.15</td>
</tr>
<tr>
<td>Open hallway detected</td>
<td>0</td>
<td>0</td>
<td>0.001</td>
<td>0.90</td>
<td>0.50</td>
</tr>
</tbody>
</table>