Recursive Bayes Filtering

Advanced AI

Wolfram Burgard
Tutorial Goal

To familiarize you with probabilistic paradigm in robotics

- Basic techniques
  - Advantages
  - Pitfalls and limitations
- Successful Applications
- Open research issues
Robotics Yesterday
Robotics Today
RoboCup
Physical Agents are Inherently Uncertain

- Uncertainty arises from four major factors:
  - Environment stochastic, unpredictable
  - Robot stochastic
  - Sensor limited, noisy
  - Models inaccurate
Nature of Sensor Data

Odometry Data

Range Data
Probabilistic Techniques for Physical Agents

Key idea: Explicit representation of uncertainty using the calculus of probability theory

Perception = state estimation
Action = utility optimization
Advantages of Probabilistic Paradigm

- Can accommodate inaccurate models
- Can accommodate imperfect sensors
- Robust in real-world applications
- Best known approach to many hard robotics problems
Pitfalls

- Computationally demanding
- False assumptions
- Approximate
Outline

- Introduction
- Probabilistic State Estimation
- Robot Localization
- Probabilistic Decision Making
  - Planning
  - Between MDPs and POMDPs
  - Exploration
- Conclusions
Axioms of Probability Theory

\[ \Pr(A) \] denotes probability that proposition \( A \) is true.

\[ \begin{align*}
0 & \leq \Pr(A) \leq 1 \\
\Pr(True) & = 1 \\
\Pr(False) & = 0 \\
\Pr(A \lor B) & = \Pr(A) + \Pr(B) - \Pr(A \land B)
\end{align*} \]
A Closer Look at Axiom 3

\[ \Pr(A \lor B) = \Pr(A) + \Pr(B) - \Pr(A \land B) \]
Using the Axioms

\[
\Pr(A \lor \neg A) = \Pr(A) + \Pr(\neg A) - \Pr(A \land \neg A)
\]

\[
\Pr(True) = \Pr(A) + \Pr(\neg A) - \Pr(False)
\]

\[
1 = \Pr(A) + \Pr(\neg A) - 0
\]

\[
\Pr(\neg A) = 1 - \Pr(A)
\]
Discrete Random Variables

- $X$ denotes a random variable.
- $X$ can take on a finite number of values in \( \{x_1, x_2, ..., x_n\} \).
- \( P(X=x_i) \), or \( P(x_i) \), is the probability that the random variable $X$ takes on value $x_i$.
- \( P(\cdot) \) is called probability mass function.

E.g. \[ P(Room) = \langle 0.7, 0.2, 0.08, 0.02 \rangle \]
Continuous Random Variables

- \( X \) takes on values in the continuum.
- \( p(X=x) \), or \( p(x) \), is a probability density function.

\[
\Pr(x \in [a, b]) = \int_a^b p(x) \, dx
\]

- E.g.
Joint and Conditional Probability

- $P(X=x \text{ and } Y=y) = P(x,y)$

- If $X$ and $Y$ are independent then
  
  $P(x,y) = P(x) \cdot P(y)$

- $P(x \mid y)$ is the probability of $x$ given $y$
  
  $P(x \mid y) = \frac{P(x,y)}{P(y)}$
  
  $P(x,y) = P(x \mid y) \cdot P(y)$

- If $X$ and $Y$ are independent then
  
  $P(x \mid y) = P(x)$
Law of Total Probability, Marginals

**Discrete case**

\[ \sum_{x} P(x) = 1 \]

\[ P(x) = \sum_{y} P(x, y) \]

\[ P(x) = \sum_{y} P(x \mid y)P(y) \]

**Continuous case**

\[ \int p(x) \, dx = 1 \]

\[ p(x) = \int p(x, y) \, dy \]

\[ p(x) = \int p(x \mid y) p(y) \, dy \]
Bayes Formula

\[ P(x, y) = P(x \mid y)P(y) = P(y \mid x)P(x) \]

\[ \Rightarrow \]

\[ P(x \mid y) = \frac{P(y \mid x) P(x)}{P(y)} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}} \]
Normalization

\[ P(x \mid y) = \frac{P(y \mid x) P(x)}{P(y)} = \eta \cdot P(y \mid x) P(x) \]

\[ \eta = P(y)^{-1} = \sum_x \frac{1}{P(y \mid x)P(x)} \]

Algorithm:

\[ \forall x : \text{aux}_{x \mid y} = P(y \mid x) \cdot P(x) \]

\[ \eta = \sum_x \frac{1}{\text{aux}_{x \mid y}} \]

\[ \forall x : P(x \mid y) = \eta \cdot \text{aux}_{x \mid y} \]
Conditioning

- Total probability:

\[ P(x \mid y) = \int P(x \mid y, z) P(z \mid y) \, dz \]

- Bayes rule and background knowledge:

\[ P(x \mid y, z) = \frac{P(y \mid x, z) \, P(x \mid z)}{P(y \mid z)} \]
Simple Example of State Estimation

- Suppose a robot obtains measurement $z$
- What is $P(\text{open}|z)$?
Causal vs. Diagnostic Reasoning

- $P(open|z)$ is diagnostic.
- $P(z|open)$ is causal.
- Often causal knowledge is easier to obtain.  
- Bayes rule allows us to use causal knowledge:

$$P(open|z) = \frac{P(z|open)P(open)}{P(z)}$$
Example

- \( P(z|open) = 0.6 \quad P(z|\neg open) = 0.3 \)
- \( P(open) = P(\neg open) = 0.5 \)

\[
P(open|z) = \frac{P(z|open)P(open)}{P(z|open)p(open) + P(z|\neg open)p(\neg open)}
\]

\[
P(open|z) = \frac{0.6 \cdot 0.5}{0.6 \cdot 0.5 + 0.3 \cdot 0.5} = \frac{2}{3} = 0.67
\]

- \( z \) raises the probability that the door is open.
Combining Evidence

- Suppose our robot obtains another observation $z_2$.

- How can we integrate this new information?

- More generally, how can we estimate $P(x|z_1...z_n)$?
Recursive Bayesian Updating

\[ P(x \mid z_1, \ldots, z_n) = \frac{P(z_n \mid x, z_1, \ldots, z_{n-1}) P(x \mid z_1, \ldots, z_{n-1})}{P(z_n \mid z_1, \ldots, z_{n-1})} \]

**Markov assumption:** \( z_n \) is independent of \( z_1, \ldots, z_{n-1} \) if we know \( x \).

\[ P(x \mid z_1, \ldots, z_n) = \frac{P(z_n \mid x) P(x \mid z_1, \ldots, z_{n-1})}{P(z_n \mid z_1, \ldots, z_{n-1})} \]

\[ = \eta P(z_n \mid x) P(x \mid z_1, \ldots, z_{n-1}) \]

\[ = \eta_{1\ldots n} \prod_{i=1\ldots n} P(z_i \mid x) P(x) \]
Example: Second Measurement

- \( P(z_2|\text{open}) = 0.5 \quad P(z_2|\neg\text{open}) = 0.6 \)
- \( P(\text{open}|z_1) = \frac{2}{3} \)

\[
P(\text{open}|z_2, z_1) = \frac{P(z_2|\text{open}) P(\text{open}|z_1)}{P(z_2|\text{open}) P(\text{open}|z_1) + P(z_2|\neg\text{open}) P(\neg\text{open}|z_1)}
\]

\[
= \frac{\frac{1}{2} \cdot \frac{2}{3}}{\frac{1}{2} \cdot \frac{2}{3} + \frac{3}{5} \cdot \frac{1}{3}} = \frac{5}{8} = 0.625
\]

- \( z_2 \) lowers the probability that the door is open.
A Typical Pitfall

- Two possible locations $x_1$ and $x_2$
- $P(x_1) = 1 - P(x_2) = 0.99$
- $P(z|x_2) = 0.09$ $P(z|x_1) = 0.07$
Actions

- Often the world is **dynamic** since
  - actions carried out by the robot,
  - actions carried out by other agents,
  - or just the **time** passing by
  change the world.

- How can we **incorporate** such **actions**?
Typical Actions

- The robot turns its wheels to move
- The robot uses its manipulator to grasp an object
- Plants grow over time...

- Actions are never carried out with absolute certainty.
- In contrast to measurements, actions generally increase the uncertainty.
Modeling Actions

- To incorporate the outcome of an action $u$ into the current “belief”, we use the conditional pdf

$$P(x|u,x')$$

- This term specifies the pdf that executing $u$ changes the state from $x'$ to $x$. 
Example: Closing the door
State Transitions

\[ P(x|u,x') \] for \( u = \) “close door”:

If the door is open, the action “close door” succeeds in 90% of all cases.
Integrating the Outcome of Actions

Continuous case:

\[ P(x \mid u) = \int P(x \mid u, x') P(x') dx' \]

Discrete case:

\[ P(x \mid u) = \sum P(x \mid u, x') P(x') \]
Example: The Resulting Belief

\[ P(\text{closed} \mid u) = \sum P(\text{closed} \mid u, x')P(x') \]
\[ = P(\text{closed} \mid u, \text{open})P(\text{open}) \]
\[ + P(\text{closed} \mid u, \text{closed})P(\text{closed}) \]
\[ = \frac{9}{10} \times \frac{5}{8} + \frac{1}{1} \times \frac{3}{18} = \frac{15}{16} \]

\[ P(\text{open} \mid u) = \sum P(\text{open} \mid u, x')P(x') \]
\[ = P(\text{open} \mid u, \text{open})P(\text{open}) \]
\[ + P(\text{open} \mid u, \text{closed})P(\text{closed}) \]
\[ = \frac{1}{10} \times \frac{5}{8} + \frac{0}{1} \times \frac{3}{18} = \frac{1}{16} \]
\[ = 1 - P(\text{closed} \mid u) \]
Bayes Filters: Framework

Given:
- Stream of observations $z$ and action data $u$: $d_t = \{u_1, z_2, \ldots, u_{t-1}, z_t\}$
- Sensor model $P(z|x)$.
- Action model $P(x|u, x')$.
- Prior probability of the system state $P(x)$.

Wanted:
- Estimate of the state $X$ of a dynamical system.
- The posterior of the state is also called **Belief**:

$$Bel(x_t) = P(x_t | u_1, z_2 \ldots, u_{t-1}, z_t)$$
Markov Assumption

\[ p(d_t, d_{t-1}, \ldots, d_0 | x_t, d_{t+1}, d_{t+2}, \ldots) = p(d_t, d_{t-1}, \ldots, d_0 | x_t) \]
\[ p(d_t, d_{t+1}, \ldots | x_t, d_1, d_2, \ldots, d_{t-1}) = p(d_t, d_{t+1}, \ldots | x_t) \]
\[ p(x_t | u_{t-1}, x_{t-1}, d_{t-2}, \ldots, d_0) = p(x_t | u_{t-1}, x_{t-1}) \]

Underlying Assumptions

- Static world
- Independent noise
- Perfect model, no approximation errors
**Bayes Filters**

\[ \text{Bel}(x_t) = P(x_t \mid u_1, z_2, \ldots, u_{t-1}, z_t) \]

\[ = \eta \ P(z_t \mid x_t, u_1, z_2, \ldots, u_{t-1}) \ P(x_t \mid u_1, z_2, \ldots, u_{t-1}) \]

\[ = \eta \ P(z_t \mid x_t) \ P(x_t \mid u_1, z_2, \ldots, u_{t-1}) \]

**Markov**

\[ = \eta \ P(z_t \mid x_t) \int P(x_t \mid u_1, z_2, \ldots, u_{t-1}, x_{t-1}) \]
\[ \quad \quad \quad P(x_{t-1} \mid u_1, z_2, \ldots, u_{t-1}) \ dx_{t-1} \]

\[ = \eta \ P(z_t \mid x_t) \int P(x_t \mid u_{t-1}, x_{t-1}) \ P(x_{t-1} \mid u_1, z_2, \ldots, u_{t-1}) \ dx_{t-1} \]

\[ = \eta \ P(z_t \mid x_t) \int P(x_t \mid u_{t-1}, x_{t-1}) \text{Bel}(x_{t-1}) \ dx_{t-1} \]

\[ z = \text{observation} \]
\[ u = \text{action} \]
\[ x = \text{state} \]
\[
Bel(x_t) = \eta \ P(z_t \mid x_t) \int P(x_t \mid u_{t-1}, x_{t-1}) \ Bel(x_{t-1}) \ dx_{t-1}
\]

1. Algorithm **Bayes_filter**(\( Bel(x), d \)):
2. \( \eta = 0 \)
3. if \( d \) is a perceptual data item \( z \) then
   4. For all \( x \) do
   5. \( Bel'(x) = P(z \mid x) Bel(x) \)
   6. \( \eta = \eta + Bel'(x) \)
   7. For all \( x \) do
   8. \( Bel'(x) = \eta^{-1} Bel'(x) \)

9. else if \( d \) is an action data item \( u \) then
10. For all \( x \) do
11. \( Bel'(x) = \int P(x \mid u, x') Bel(x') \ dx' \)
12. return \( Bel'(x) \)
Bayes Filters are Familiar!

\[ Bel(x_t) = \eta \, P(z_t \mid x_t) \int P(x_t \mid u_{t-1}, x_{t-1}) \, Bel(x_{t-1}) \, dx_{t-1} \]

- Kalman filters
- Particle filters
- Hidden Markov models
- Dynamic Bayes networks
- Partially Observable Markov Decision Processes (POMDPs)
Application to Door State Estimation

- Estimate the opening angle of a door
- and the state of other dynamic objects
- using a laser-range finder
- from a moving mobile robot and
- based on Bayes filters.
Result
Lessons Learned

- Bayes rule allows us to compute probabilities that are hard to assess otherwise.

- Under the Markov assumption, recursive Bayesian updating can be used to efficiently combine evidence.

- Bayes filters are a probabilistic tool for estimating the state of dynamic systems.
Tutorial Outline

- Introduction
- Probabilistic State Estimation
  - Localization
- Probabilistic Decision Making
  - Planning
  - Between MDPs and POMDPs
  - Exploration
- Conclusions
The Localization Problem

“Using sensory information to locate the robot in its environment is the most fundamental problem to providing a mobile robot with autonomous capabilities.” [Cox ’91]

- **Given**
  - Map of the environment.
  - Sequence of sensor measurements.

- **Wanted**
  - Estimate of the robot’s position.

- **Problem classes**
  - Position tracking
  - Global localization
  - Kidnapped robot problem (recovery)
Representations for Bayesian Robot Localization

**Discrete approaches ('95)**
- Topological representation ('95)
  - uncertainty handling (POMDPs)
  - occas. global localization, recovery
- Grid-based, metric representation ('96)
  - global localization, recovery

**Kalman filters (late-80s?)**
- Gaussians
- approximately linear models
- position tracking

**Particle filters ('99)**
- sample-based representation
- global localization, recovery

**Multi-hypothesis ('00)**
- multiple Kalman filters
- global localization, recovery
What is the Right Representation?

- Kalman filters
- Multi-hypothesis tracking
- Grid-based representations
- Topological approaches
- Particle filters
Gaussians

\[ p(x) \sim N(\mu, \sigma^2) : \]
\[ p(x) = \frac{1}{\sqrt{2\pi \sigma}} e^{-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}} \]

Univariate

\[ p(\mathbf{x}) \sim N(\mu, \Sigma) : \]
\[ p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2} (\mathbf{x}-\mu)' \Sigma^{-1} (\mathbf{x}-\mu)} \]

Multivariate
Kalman Filters

Estimate the state of processes that are governed by the following linear stochastic difference equation.

\[ x_{t+1} = Ax_t + Bu_t + v_t \]
\[ z_t = Cx_t + w_t \]

The random variables \( v_t \) and \( w_t \) represent the process measurement noise and are assumed to be independent, white and with normal probability distributions.
Kalman Filters

[Schiele et al. 94], [Weiß et al. 94], [Borenstein 96], [Gutmann et al. 96, 98], [Arras 98]
Kalman Filter Algorithm

1. Algorithm **Kalman_filter** ( \langle \mu, \Sigma \rangle, d \rangle):

2. If \( d \) is a perceptual data item \( z \) then

3. \[ K = \Sigma C^T \left( C \Sigma C^T + \Sigma_{obs} \right)^{-1} \]

4. \[ \mu = \mu + K(z - C\mu) \]

5. \[ \Sigma = (I - KC)\Sigma \]

6. Else if \( d \) is an action data item \( u \) then

7. \[ \mu = A\mu + Bu \]

8. \[ \Sigma = A\Sigma A^T + \Sigma_{act} \]

9. Return \( \langle \mu, \Sigma \rangle \)
Non-linear Systems

- Very strong assumptions:
  - Linear state dynamics
  - Observations linear in state
- What can we do if system is not linear?
  - Linearize it: EKF
  - Compute the Jacobians of the dynamics and observations at the current state.
  - Extended Kalman filter works surprisingly well even for highly non-linear systems.
Kalman Filter-based Systems (1)

- [Gutmann et al. 96, 98]:
  - Match LRF scans against map
  - Highly successful in RoboCup mid-size league

Courtesy of S. Gutmann
Kalman Filter-based Systems (2)

Courtesy of S. Gutmann
Kalman Filter-based Systems (3)

- [Arras et al. 98]:
  - Laser range-finder and vision
  - High precision (<1cm accuracy)

Courtesy of K. Arras
Multi-hypothesis Tracking

[Cox 92], [Jensfelt, Kristensen 99]
Localization With MHT

- Belief is represented by multiple hypotheses
- Each hypothesis is tracked by a Kalman filter

- **Additional problems:**
  - **Data association:** Which observation corresponds to which hypothesis?
  - **Hypothesis management:** When to add / delete hypotheses?
  - Huge body of literature on target tracking, motion correspondence etc.

See e.g. [Cox 93]
MHT: Implemented System (1)

- [Jensfelt and Kristensen 99,01]
  - Hypotheses are extracted from LRF scans
  - Each hypothesis has probability of being the correct one:
    \[ H_i = \{ \hat{x}_i, \Sigma_i, P(H_i) \} \]
  - Hypothesis probability is computed using Bayes’ rule
    \[ P(H_i | s) = \frac{P(s | H_i) P(H_i)}{P(s)} \]
  - Hypotheses with low probability are deleted
  - New candidates are extracted from LRF scans
    \[ C_j = \{ z_j, R_j \} \]
MHT: Implemented System (2)

Robot view | Pose candidates

Sensor data → Feature extraction → Generate pose candidates

MATCH existing? NO → Creative feature?

YES → Update hypothesis

YES → Create hypothesis

Courtesy of P. Jensfelt and S. Kristensen
MHT: Implemented System (3) Example run

Map and trajectory

Hypotheses vs. time

Courtesy of P. Jensfelt and S. Kristensen
Piecewise Constant

[Burgard et al. 96,98], [Fox et al. 99], [Konolige et al. 99]
Piecwise Constant Representation

\[ \text{bel}(x_t = \langle x, y, \theta \rangle) \]
Grid-based Localization
Tree-based Representations (1)

**Idea**: Represent density using a variant of Octrees
Xavier:
Localization in a Topological Map

Probabilistic Robot Navigation in Partially Observable Environments
Reid G. Simmons
Sven Koenig
Computer Science Department
Carnegie Mellon University

[Simmons and Koenig 96]
Particle Filters

- Represent density by random samples
- Estimation of non-Gaussian, nonlinear processes
- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter, Particle filter

- Filtering: [Rubin, 88], [Gordon et al., 93], [Kitagawa 96]
- Computer vision: [Isard and Blake 96, 98]
- Dynamic Bayesian Networks: [Kanazawa et al., 95]
MCL: Global Localization
MCL: Sensor Update

\[ Bel(x) \leftarrow \alpha p(z \mid x) Bel^{-}(x) \]

\[ w \leftarrow \frac{\alpha p(z \mid x) Bel^{-}(x)}{Bel^{-}(x)} = \alpha p(z \mid x) \]
MCL: Robot Motion

\[ \text{Bel}^{-}(x) \leftarrow \int p(x | u, x') \text{Bel}(x') \, dx' \]
MCL: Sensor Update

\[ Bel(x) \leftarrow \alpha p(z \mid x) Bel^-(x) \]

\[ w \leftarrow \frac{\alpha p(z \mid x) Bel^-(x)}{Bel^-(x)} = \alpha p(z \mid x) \]
MCL: Robot Motion

\[
Bel^{-}(x) \leftarrow \int p(x \mid u, x') Bel(x') \, dx'
\]
Particle Filter Algorithm

1. Algorithm **particle_filter**( $S_{t-1}$, $u_{t-1}$, $z_t$):
2. $S_t = \emptyset$, $\eta = 0$
3. **For** $i = 1 \ldots n$ **Generate new samples**
4. Sample index $j(i)$ from the discrete distribution given by $w_{t-1}$
5. Sample $x_t^i$ from $p(x_t | x_{t-1}, u_{t-1})$ using $x_t^{j(i)}$ and $u_{t-1}$
6. $w_t^i = p(z_t | x_t^i)$ **Compute importance weight**
7. $\eta = \eta + w_t^i$ **Update normalization factor**
8. $S_t = S_t \cup \{< x_t^i, w_t^i >\}$ **Insert**
9. **For** $i = 1 \ldots n$
10. $w_t^i = w_t^i / \eta$ **Normalize weights**
Resampling

- **Given**: Set $S$ of weighted samples.

- **Wanted**: Random sample, where the probability of drawing $x_i$ is given by $w_i$.

- Typically done $n$ times with replacement to generate new sample set $S'$.  
Resampling

- Roulette wheel
- Binary search, log n

- Stochastic universal sampling
- Systematic resampling
- Linear time complexity
- Easy to implement, low variance
Motion Model \[ p(x_t \mid a_{t-1}, x_{t-1}) \]

Model odometry error as Gaussian noise on \( \alpha, \beta, \) and \( \delta \)
Motion Model \( p(x_t \mid a_{t-1}, x_{t-1}) \)
Model for Proximity Sensors

The sensor is reflected either by a **known** or by an **unknown** obstacle:

\[ P(d_i \mid l) = 1 - (1 - (1 - \sum_{j<i} P_u(d_j)) c_d P_m(d_i \mid l))) \cdot (1 - (1 - \sum_{j<i} P(d_j)) c_r) \]

---

**Laser sensor**

**Sonar sensor**
MCL: Global Localization (Sonar)

[Fox et al., 99]
Using Ceiling Maps for Localization

[Dellaert et al. 99]
Vision-based Localization
MCL: Global Localization Using Vision
Localization for AIBO robots
Adaptive Sampling
**KLD-sampling**

- **Idea:**
  - Assume we know the true belief.
  - Represent this belief as a multinomial distribution.
  - Determine number of samples such that we can guarantee that, with probability \((1 - \delta)\), the KL-distance between the true posterior and the sample-based approximation is less than \(\epsilon\).

- **Observation:**
  - For fixed \(\delta\) and \(\epsilon\), number of samples only depends on number \(k\) of bins with support:

\[
 n = \frac{1}{2\epsilon} X^2(k - 1, 1 - \delta) \cong \frac{k - 1}{2\epsilon} \left\{ \frac{2}{9(k - 1)} + \sqrt{\frac{2}{9(k - 1)}} z_{1 - \delta} \right\}^3
\]
MCL: Adaptive Sampling (Sonar)
Particle Filters for Robot Localization (Summary)

- Approximate Bayes Estimation/Filtering
  - Full posterior estimation
  - Converges in $O(1/\sqrt{\text{#samples}})$ [Tanner’93]
  - Robust: multiple hypotheses with degree of belief
  - Efficient in low-dimensional spaces: focuses computation where needed
  - Any-time: by varying number of samples
  - Easy to implement
# Localization Algorithms - Comparison

<table>
<thead>
<tr>
<th></th>
<th>Kalman filter</th>
<th>Multi-hypothesis tracking</th>
<th>Topological maps</th>
<th>Grid-based (fixed/variable)</th>
<th>Particle filter</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sensors</strong></td>
<td>Gaussian</td>
<td>Gaussian</td>
<td>Features</td>
<td>Non-Gaussian</td>
<td>Non-Gaussian</td>
</tr>
<tr>
<td><strong>Posterior</strong></td>
<td>Gaussian</td>
<td>Multi-modal</td>
<td>Piecewise constant</td>
<td>Piecewise constant</td>
<td>Samples</td>
</tr>
<tr>
<td><strong>Efficiency (memory)</strong></td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>-/+</td>
<td>++/+++</td>
</tr>
<tr>
<td><strong>Efficiency (time)</strong></td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>0/+</td>
<td>++/+++</td>
</tr>
<tr>
<td><strong>Implementation</strong></td>
<td>+</td>
<td>0</td>
<td>+</td>
<td>+/0</td>
<td>++</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>++</td>
<td>++</td>
<td>-</td>
<td>+++</td>
<td>++</td>
</tr>
<tr>
<td><strong>Robustness</strong></td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td>++/+++</td>
</tr>
<tr>
<td><strong>Global localization</strong></td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Localization: Lessons Learned

- Probabilistic Localization = Bayes filters
- Particle filters: Approximate posterior by random samples
- Extensions:
  - Filter for dynamic environments
  - Safe avoidance of invisible hazards
  - People tracking
  - Recovery from total failures
  - Active Localization
  - Multi-robot localization