CSE 571
Probabilistic Robotics

Recap

Bayes Filters

\[ \text{Bel}(x_t) = P(x_t | u_1, z_1, \ldots, u_t, z_t) \]

Bayes

\[ = \eta \ P(z_t | x_t, u_t, z_1, \ldots, u_t) \ P(x_t | u_t, z_1, \ldots, u_t) \]

Markov

\[ = \eta \ P(z_t | x_t) \ P(x_t | u_t, z_1, \ldots, u_t) \]

Total prob.

\[ = \eta \ P(z_t | x_t) \int P(x_t | u_t, z_1, \ldots, u_t, x_{t-1}) \]

\[ \times \int P(x_{t-1} | u_t, z_1, \ldots, u_t) \ dx_{t-1} \]

Markov

\[ = \eta \ P(z_t | x_t) \int P(x_t | u_t, x_{t-1}) \ P(x_{t-1} | u_t, z_1, \ldots, u_t) \ dx_{t-1} \]

\[ + \eta \ P(z_t | x_t) \int P(x_t | u_t, x_{t-1}) \text{Bel}(x_{t-1}) \ dx_{t-1} \]

Parametric Sensor Model

\[ P(z | x, m) = \begin{pmatrix} a_{hit} \ a_{unexp} \\ a_{hit} \ a_{unexp} \end{pmatrix} \begin{pmatrix} P_{hit}(z | x, m) \\ P_{unexp}(z | x, m) \end{pmatrix} \]
**Parametric Kinematics Model**

- Robot moves from \( (x, y, \theta) \) to \( (x', y', \theta') \).
- Odometry information \( u = (\delta_{\text{rot}}, \delta_{\text{rot}2}, \delta_{\text{tran}}) \)

\[
\begin{align*}
\delta_{\text{tran}} &= \sqrt{(x' - x)^2 + (y' - y)^2} \\
\delta_{\text{rot}} &= \arctan2(y' - y, x' - x) - \bar{\theta} \\
\delta_{\text{rot}2} &= \bar{\theta}' - \bar{\theta} - \delta_{\text{rot}1}
\end{align*}
\]

---

**Alternative: Non-Parametric Gaussian Process Models**

- 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
    - Approximate linear models
    - Position tracking
  - Particle filters (’99)
    - Sample-based representation
    - Global localization, recovery
  - Multi-hypothesis (’00)
    - Multiple Kalman filters
    - Global localization, recovery

**The Prediction-Correction-Cycle of Kalman Filters**

- Prediction
  - \( \hat{x}_{k|k-1} = F \hat{x}_{k-1|k-1} + Bu_k \)
  - \( P_{k|k-1} = FF^T + Q \)

- Correction
  - \( \hat{x}_{k|k} = \hat{x}_{k|k-1} + K(z_k - H \hat{x}_{k|k-1}) \)
  - \( P_{k|k} = (I - KH)P_{k|k-1} \)
We can use a different distribution $g$ to generate samples from $f$.

By introducing an importance weight $w$, we can account for the "differences between $g$ and $f$":

$$w = \frac{f}{g}$$
**Types of SLAM-Problems**

- **Grid maps or scans**
  - [Lu & Milios, 97; Gutmann, 98; Thrun, 98; Burgard, 99; Konolige & Gutmann, 00; Thrun, 00; Arras, 99; Haehnel, 01;...]

- **Landmark-based**

**Why is SLAM a hard problem?**

- **SLAM**: robot path and map are both unknown
- Robot path error correlates errors in the map

**EKF-SLAM**

- Map with N landmarks: \((3+2N)\)-dimensional Gaussian

\[
\text{Bel}(x, m) = \begin{pmatrix}
x \\
y \\
\theta
\end{pmatrix}
\begin{pmatrix}
\sigma_{x} & \sigma_{y} & \sigma_{\theta} \\
\sigma_{x} & \sigma_{y} & \sigma_{\theta} \\
\sigma_{x} & \sigma_{y} & \sigma_{\theta}
\end{pmatrix}
\]

- Can handle hundreds of dimensions
**FastSLAM**

<table>
<thead>
<tr>
<th>Particle #1</th>
<th>Particle #2</th>
<th>Particle #3</th>
<th>Particle M</th>
</tr>
</thead>
<tbody>
<tr>
<td>x, y, z</td>
<td>x, y, z</td>
<td>x, y, z</td>
<td>x, y, z</td>
</tr>
<tr>
<td><strong>Landmark 1</strong></td>
<td><strong>Landmark 2</strong></td>
<td><strong>Landmark 2</strong></td>
<td><strong>Landmark N</strong></td>
</tr>
<tr>
<td><strong>Landmark 2</strong></td>
<td><strong>Landmark 2</strong></td>
<td><strong>Landmark 2</strong></td>
<td><strong>Landmark N</strong></td>
</tr>
<tr>
<td><strong>Landmark N</strong></td>
<td><strong>Landmark N</strong></td>
<td><strong>Landmark N</strong></td>
<td><strong>Landmark N</strong></td>
</tr>
</tbody>
</table>

FastSLAM uses 2 x 2 Kalman Filters for robot pose estimation.

**Graph-SLAM Idea**

- Graph-based SLAM
- Graph structure for constraints
- Bundle adjustment for optimization

**3D Outdoor Mapping**

- 10^6 features, 10^5 poses, only few secs using cg.

**RGB-D Mapping**

- 3D mapping using RGB-D sensors
- Real-time mapping and localization
RGB-D Mapping

MORE COMPLEX ESTIMATION

Ball-Environment Interaction
Inference: Posterior Estimation

\[
p(b_k, m_k, r_k | z^b_{1:k}, z^f_{1:k}, u_{k-1})
\]

Hierarchical Model

Goal
Trip segment
Transportation mode
Edge, velocity, position
GPS reading

Particles:

\[
s^{(i)} = \left\{ (g, t)^{(i)}, m^{(i)}, b^{(i)}, v^{(i)}, \theta^{(i)}, N^{(i)}(\mu, \sigma^2) \right\}
\]
Coordinated exploration with three robots from unknown start locations.

The robots are fully autonomous. All computation is performed on-board.

Shown is the perspective of one robot.

Planning via Cell Decomposition

- Graph construction:
  - lattice graph
  - pros: sparse graph, feasible paths
  - cons: possible incompleteness

action template

Rapidly exploring Random Tree (RRT)

Source: LaValle and Kuffner 01
Stochastic, Fully Observable

Manipulator Control Path

State space  Configuration space

Stochastic, Partially Observable

RL in Uncertain Environments: Converting Beliefs to Augmented States

Belief  Augmented state
The Belief Roadmap Algorithm

1. Sample means from $C_{mm}$, build graph and transfer functions
2. Propagate covariances by performing graph search

PILCO: GP Model-Based Learning

- Swing pendulum up and balance in inverted position
- Learn nonlinear control from scratch
- 4D state space, 300 control parameters
- 7 trials/17.5 sec experience
- Control freq.: 10 Hz

Inverse Optimal Control

2-D Planner

Further Examples and Discussion
**Goal:** General tool for real-time tracking of arbitrary articulated objects

**Input:** Shape models of parts along with joint structure

**Insight:** Efficient optimization via articulated signed distance functions

---

**DART: Dense Articulated Real-Time Tracking**
Using Articulated Signed-Distance Functions

---

**DART: Model-Based Tracking**
Hand (27 DoF) and Human Body (42 DoF)

---

**Fine-Grained Manipulation**

---

**Detection-Based Approach to Articulated Tracking**

---

Inspired by [Tompson-et al: SIGGRAPH-14]
### Model-Based / Detection-Based

<table>
<thead>
<tr>
<th></th>
<th>Model-based</th>
<th>Detection-based</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Generality</strong></td>
<td>Minimal assumptions, broad applicability</td>
<td>Only in trained regime</td>
</tr>
<tr>
<td><strong>Robustness</strong></td>
<td>Requires initialization and good model;</td>
<td>Robust in trained regime; Failure detection</td>
</tr>
<tr>
<td></td>
<td>Can detect failures</td>
<td>more difficult</td>
</tr>
<tr>
<td><strong>Training</strong></td>
<td>Minimal training, model building</td>
<td>Major training effort, Model-based for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>supervision</td>
</tr>
<tr>
<td><strong>Physics, contacts</strong></td>
<td>Explicitly modeled</td>
<td>Must be learned in data driven way</td>
</tr>
<tr>
<td><strong>Efficiency</strong></td>
<td>Efficient for local tracking;</td>
<td>Highly efficient once trained</td>
</tr>
<tr>
<td></td>
<td>Initialization extremely hard</td>
<td></td>
</tr>
</tbody>
</table>

### BakeBot

**BakeBot:**
Motion Planning for Cooking

Mario Bollini and Daniela Rus
CSAIL, MIT

[Bollini-Rus]

### Physics Simulation for Prediction

[Physicstolbeetz]

Logic Programming with Simulation-based Temporal Projection for Everyday Robot Object Manipulation

Lars Kunze, Mihai Emanuol Dolha and Michael Beetz

[Intelligent Autonomous Systems]

How Julia Does it

[HowJuliaDoesIt]
Gravity and Onions

Deep RL

Human-level control through deep reinforcement learning

Q-Network

Learning Models from Raw Perception

Deep Spatial Autoencoders for Visuomotor Learning

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, Pieter Abbeel
**Summary**

- Probabilistic robotics
  - Great framework for approaching robotics problems
  - Not always possible, or the most appropriate approach

- On models and learning
  - Models are great but never perfect
  - Especially when reasoning about messy stuff
  - Learning of residuals, shortcuts, everything?
  - Combination is key