Bayesian Filtering for Robot Localization

CSE-473

Probabilistic Robotics

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

• Perception = state estimation
• Control = utility optimization

Localization

“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)

Sample-based Localization (sonar)
Bayes Filters: Framework

• **Given:**
  - Stream of observations $z$ and action data $u$:
    $$ d_t = u_t, z_1, z_2, ..., u_{t-1}, z_t $$
  - 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**:
    $$ Bel(x_t) = p(x_t | u_1, z_2, ..., u_t, z_t) $$

Graphical Model of Localization

Markov Assumption

- $p(z_t | x_t, z_{t-1}, u_{t-1}) = p(z_t | x_t)$
- $p(x_t | x_{t-1}, z_{t-1}, u_{t-1}) = p(x_t | x_{t-1}, u_{t-1})$

Underlying Assumptions

- Static world
- Independent noise
- Perfect model, no approximation errors
Two Time Slice Representation of Dynamic Bayes Net

$$z_{t-1} \rightarrow x_{t-1} \rightarrow x_t \rightarrow z_t$$

Belief Update

$$p(x_t \mid u_t, z_t) = \int p(x_t \mid u_t, z_t, x_{t-1}) \, dx_{t-1}$$

$$= \alpha \int p(x_t \mid u_t, x_{t-1}) \, dz_t \int p(z_t \mid x_t) \, p(x_{t-1}) \, dx_{t-1}$$

$$= \alpha \int p(z_t \mid x_t) \, \int p(x_t \mid u_t, x_{t-1}) \, p(x_{t-1}) \, dx_{t-1}$$

Bayes Filters

$$Bel(x_t) = p(x_t \mid u_1, z_2, \ldots, u_{t-1}, z_t)$$

Bayes

$$= \eta \int p(z_t \mid x_t) \, p(x_t \mid u_1, z_2, \ldots, u_{t-1}) \, dx_{t-1}$$

Markov

$$= \eta \int p(z_t \mid x_t) \, p(x_t \mid u_1, z_2, \ldots, u_{t-1}) \, dx_{t-1}$$

Total prob.

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

Markov

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

$$= \eta \int p(z_t \mid x_t) \, \int p(x_t \mid u_1, z_2, \ldots, u_{t-1}, x_{t-1}) \, Bel(x_{t-1}) \, dx_{t-1}$$

Representations for Bayesian Robot Localization

Discrete approaches ('95)
- Topological representation ('95)
- Uncertainty handling (POMDPs)
- Occasional global localization, recovery

Grid-based, metric representation ('96)
- Global localization, recovery

Particle filters ('99)
- Sample-based representation
- Global localization, recovery

Kalman filters (late-80s?)
- Gaussians
- Approximately linear models
- Position tracking

Multi-hypothesis ('00)
- Multiple Kalman filters
- Global localization, recovery
Particle Filters

- Represent belief 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]d

Sample-based Density Representation

![Sample-based Density Representation](image1)

Importance Sampling

![Importance Sampling](image2)

Weight samples: $w = f/g$
Particle Filters

Robot Motion

\[ Bel^\sim(x) \leftarrow \int p(x|u,x') Bel(x') \, dx' \]

Sensor Information: Importance Sampling

\[ Bel(x) \leftarrow \alpha \gamma(z|x) Bel^\sim(x) \]

\[ w \leftarrow \frac{\alpha \gamma(z|x) Bel^\sim(x)}{Bel^\sim(x)} = \alpha \gamma(z|x) \]
Robot Motion

\[ \text{Bel } \tilde{x}(x) \leftarrow \int p(x | x') \text{Bel } (x') \, dx' \]

Particle Filter Algorithm

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

Motion Model Reminder

Proximity Sensor Model Reminder

Importance factor for \( x_t \):

\[ w_t^i = \frac{\text{target distribution}}{\text{proposal distribution}} \]

\[ = \frac{\eta \cdot p(z_t | x_t) p(x_t | x_{t-1}, u_{t-1}) \text{Bel } (x_{t-1})}{p(x_t | x_{t-1}, u_{t-1}) \text{Bel } (x_{t-1})} \]

\[ \propto p(z_t | x_t) \]

Laser sensor

Sonar sensor
Sample-based Localization (sonar)
Using Ceiling Maps for Localization

[Dellaert et al. 99]

Vision-based Localization

Under a Light

Next to a Light
Elsewhere

Measurement $z$: $P(z|x)$

Global Localization Using Vision

Localization for AIBO robots

Adaptive Sampling
Example Run Sonar

Example Run Laser

Example Run

Tracking with a Moving Robot
Ball Tracking in RoboCup

- Extremely noisy (nonlinear) motion of observer
- Inaccurate sensing, limited processing power
- Interactions between target and environment

Goal: Unified framework for modeling the ball and its interactions.

Rao-Blackwellised PF for Inference

- Represent posterior by random samples
- Each sample
  \[ s_i = \langle r_i, m_i, b_i \rangle = \langle x, y, \theta_j, m_i, \mu, \Sigma_i \rangle \]
  contains robot location, ball mode, ball Kalman filter
- Generate individual components of a particle stepwise using the factorization
  \[ p(b_i, m_{i|k}, r_{ik}, z_{ik}, u_{ik-}) = \]
  \[ p(b_i | m_{ik}, r_{ik}, z_{ik}, u_{ik-}) \cdot p(m_{ik} | r_{ik}, z_{ik}, u_{ik-}) \cdot p(r_{ik} | z_{ik}, u_{ik-}) \]
GPS Receivers We Used

- GeoStats wearable GPS logger
- Nokia 6600 Java Cell Phone with Bluetooth GPS unit

Geographic Information Systems

- Street map
  - Data source: Census 2000 Tiger/line data
- Bus routes and bus stops
  - Data source: Metro GIS

Adding Mode of Transportation

- Transportation mode
- Edge, velocity, position
- Data (edge) association
- GPS reading

Infer Location and Transportation

- Green
  - Bus mode
- Red
  - Car mode
- Blue
  - Foot mode
Hierarchical Model

\[ z_{k-1} \rightarrow z_k \rightarrow z_k \]

Goal

Trip segment

Transportation mode

Edge, velocity, position

Data (edge) association

GPS reading

Predict Goal and Path

Application: Opportunity Knocks

Detect User Errors

Untrained

Trained

Instantiated
Application: Opportunity Knocks

Particle Filter Algorithm

1: Algorithm Particle_filter(\(X_{t-1}, u_t, z_t\)):
2: \(\hat{X}_t \leftarrow X_t = \emptyset\)
3: for \(m = 1\) to \(M\) do
4: sample \(x_t^{[m]} \sim p(x_t | u_t, z_t^{[1:m]})\)
5: \(w_t^{[m]} = p(z_t | x_t^{[m]})\)
6: \(\hat{X}_t \leftarrow \hat{X}_t + (x_t^{[m]}, w_t^{[m]})\)
7: endfor
8: for \(m = 1\) to \(M\) do
9: draw \(i\) with probability \(w_t^{[i]}\)
10: add \(x_t^{[i]}\) to \(X_t\)
11: endfor
12: return \(X_t\)

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 \(S'\).

- Roulette wheel
- Binary search, \(\log n\)
- Stochastic universal sampling
- Systematic resampling
- Linear time complexity
- Easy to implement, low variance
Resampling Algorithm

1. Algorithm low_variance_sample\(\bar{X}, \bar{W})\):
   \[
   \bar{X}_i = 0
   \]
2. \(r = \text{rand}(0; \sigma^{-1})\)
3. \(c = w_i^0\)
4. \(i = 1\)
5. for \(m = 1\) to \(M\) do
   \[
   U = r + (m - 1) \cdot \sigma^{-1}
   \]
   while \(U > c\)
   \[
   i = i + 1
   \]
   end while
7. \(c = c + w_i^0\)
8. endfor
9. add \(x_i^0\) to \(\bar{X}_i\)
10. endfor
11. return \(\bar{X}_i\)

Probabilistic Kinematics

- Robot moves from \(\langle \bar{z}, \bar{y}, \theta \rangle\) to \(\langle \bar{z}', \bar{y}', \theta \rangle\).
- Odometry information \(u = \hat{\delta}_{rot1}, \hat{\delta}_{rot2}, \hat{\delta}_{trans}\).

\[
\begin{align*}
\hat{\delta}_{trans} & = \sqrt{(\bar{y} - \bar{y}')^2 + (\bar{y} - \bar{y})^2} \\
\hat{\delta}_{rot1} & = \text{atan2} (\bar{y}' - \bar{y}, \bar{y}' - \bar{y}) = \hat{\gamma} \\
\hat{\delta}_{rot2} & = \hat{\gamma} - \hat{\gamma}_{rot1}
\end{align*}
\]

Noise Model for Motion

- The measured motion is given by the true motion corrupted with noise.
- To predict sample position, just sample from ”noisy version” of measured motion.

\[
\begin{align*}
\hat{\delta}_{rot1} & = \hat{\gamma}_{rot1} + \hat{\gamma}_{rot}\hat{\delta}_{rot1} + \hat{\delta}_{rot} \hat{\delta}_{rot1} \\
\hat{\delta}_{trans} & = \hat{\delta}_{trans} + \hat{\gamma}_{trans}\hat{\delta}_{trans} + \hat{\delta}_{trans} \hat{\delta}_{trans} \\
\hat{\delta}_{rot2} & = \hat{\gamma}_{rot2} + \hat{\gamma}_{rot2}\hat{\delta}_{rot2} + \hat{\delta}_{rot2} \hat{\delta}_{rot2}
\end{align*}
\]

Gaussian Noise Model

Normal distribution

\[
x \sim \mathcal{N}(\mu, \sigma^2)
\]

\[
\begin{align*}
\mathcal{E}_p(x) & = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x - \mu)^2}{2\sigma^2}}
\end{align*}
\]
Examples (odometry based)

Motion Model with Map

\[ P(x | u, x') = \frac{1}{P(x | m) P(x | u, x')} \]

- When does this approximation fail?