Introduction to Mobile Robotics

Bayes Filter – Particle Filter and Monte Carlo Localization

Wolfram Burgard
Motivation

- Estimating the state of a dynamical system is a fundamental problem
- The Recursive Bayes Filter is an effective approach to estimate the belief about the state of a dynamical system
  - How to represent this belief?
  - How to maximize it?
- Particle filters are a way to efficiently represent an arbitrary (non-Gaussian) distribution
- Basic principle
  - Set of state hypotheses (“particles”)
  - Survival-of-the-fittest
Bayes Filters

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

**Bayes**

\[ \begin{align*}
\text{Bel}(x_t) &= \eta \ P(z_t \mid x_t, u_1, z_1, \ldots, u_t) P(x_t \mid u_1, z_1, \ldots, u_t) \\
&= \eta \ P(z_t \mid x_t) P(x_t \mid u_1, z_1, \ldots, u_t) \\
&= \eta \ P(z_t \mid x_t) \int P(x_t \mid u_1, z_1, \ldots, u_t, x_{t-1}) \ P(x_{t-1} \mid u_1, z_1, \ldots, u_t) \, dx_{t-1} \\
&= \eta \ P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) P(x_{t-1} \mid u_1, z_1, \ldots, u_t) \, dx_{t-1} \\
&= \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) P(x_{t-1} \mid u_1, z_1, \ldots, z_{t-1}) \, dx_{t-1} \\
&= \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) \text{Bel}(x_{t-1}) \, dx_{t-1}
\end{align*} \]

**Markov**

\[ z = \text{observation} \]
\[ u = \text{action} \]
\[ x = \text{state} \]
Probabilistic Localization

\[ Bel(x | z, u) = \alpha p(z | x) \int_{x'} p(x | u, x') Bel(x') dx' \]
Function Approximation

- Particle sets can be used to approximate functions

- The more particles fall into an interval, the higher the probability of that interval

- How to draw samples from a function/distribution?
Let us assume that $f(x) < a$ for all $x$

- Sample $x$ from a uniform distribution
- Sample $c$ from $[0, a]$
- if $f(x) > c$ keep the sample otherwise reject the sample

Rejection Sampling
Importance Sampling Principle

- We can even use a different distribution $g$ to generate samples from $f$.
- Using an importance weight $w$, we can account for the “differences between $g$ and $f$”.
- $w = f / g$
- $f$ is called target.
- $g$ is called proposal.
- Pre-condition: $f(x) > 0 \ \Rightarrow \ g(x) > 0$
Particle Filter Representation

- Set of weighted samples

\[ S = \{ \langle s[i], w[i] \rangle \mid i = 1, \ldots, N \} \]

State hypothesis \hspace{2cm} \text{Importance weight}

- The samples represent the posterior

\[ p(x) = \sum_{i=1}^{N} w_i \cdot \delta_{s[i]}(x) \]
Importance Sampling with Resampling: Landmark Detection Example
Distributions
Distributions

Wanted: samples distributed according to 
\[ p(x \mid z_1, z_2, z_3) \]
This is Easy!

We can draw samples from $p(x|z_l)$ by adding noise to the detection parameters.
Importance Sampling

Target distribution \( f: p(x \mid z_1, z_2, \ldots, z_n) = \frac{\prod_{k} p(z_k \mid x) \ p(x)}{p(z_1, z_2, \ldots, z_n)} \)

Sampling distribution \( g: p(x \mid z_l) = \frac{p(z_l \mid x) p(x)}{p(z_l)} \)

Importance weights \( w: \frac{f}{g} = \frac{p(x \mid z_1, z_2, \ldots, z_n)}{p(x \mid z_l)} = \frac{p(z_l) \prod_{k \neq l} p(z_k \mid x)}{p(z_1, z_2, \ldots, z_n)} \)
Importance Sampling with Resampling

Weighted samples

After resampling
Particle Filter Localization

\[
Bel(x | z, u) = \alpha p(z | x) \int_{x'} p(x | u, x') Bel(x') dx'
\]

1. Draw \(x'\) from \(Bel(x)\)
2. Draw \(x\) from \(p(x | u, x')\)
3. Importance factor for \(x\) : \(w = \alpha p(z | x)\)
4. Re-sample
Rejection Sampling

- Let us assume that \( f(x) < a \) for all \( x \)
- Sample \( x \) from a uniform distribution
- Sample \( c \) from \([0,a]\)
- if \( f(x) > c \) keep the sample
  otherwise reject the sample
Importance Sampling Principle

- We can even use a different distribution \( g \) to generate samples from \( f \).
- Using an importance weight \( w \), we can account for the “differences between \( g \) and \( f \)”.
- \( w = \frac{f}{g} \)
- \( f \) is called target
- \( g \) is called proposal
- Pre-condition: \( f(x) > 0 \Rightarrow g(x) > 0 \)
Particle Filters
Sensor Information: Importance Sampling

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

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

\[ Bel^-(x) \leftarrow \int p(x \mid u, x') Bel(x') \, dx' \]
Sensor Information: Importance Sampling

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

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

$$Bel^-(x) \leftarrow \int p(x | u, x') Bel(x') \, dx'$$
Particle Filter Algorithm

- Sample the next generation for particles using the proposal distribution

- Compute the importance weights:
  \[
  \text{weight} = \frac{\text{target distribution}}{\text{proposal distribution}}
  \]

- Resampling: “Replace unlikely samples by more likely ones”
Particle Filter Algorithm

1. Algorithm `particle_filter(S_{t-1}, u_t, z_t)`:

2. \( S_t = \emptyset, \quad \eta = 0 \)

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 \( p(x_t | x_{t-1}, u_t) \) using \( x_{t-1}^{j(i)} \) and \( u_t \)

6. \( w_t^i = p(z_t | x_t^i) \)
   - *Compute importance weight*

7. \( \eta = \eta + w_t^i \)
   - *Update normalization factor*

8. \( S_t = S_t \cup \{ < x_t^i, w_t^i > \} \)
   - *Add to new particle set*

9. For \( i = 1, \ldots, n \)
10. \( w_t^i = w_t^i / \eta \)
    - *Normalize weights*
Particle Filter Algorithm

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

- Draw \( x_{t-1}^i \) from \( Bel(x_{t-1}) \)
- Draw \( x_t^i \) from \( p(x_t \mid x_{t-1}^i, u_t) \)
- Importance factor for \( x_t^i \):

\[
w_t^i = \frac{\text{target distribution}}{\text{proposal distribution}} = \eta \ p(z_t \mid x_t) \ p(x_t \mid x_{t-1}, u_t) \ Bel(x_{t-1}) / \ p(x_t \mid x_{t-1}, u_t) \ Bel(x_{t-1}) \propto p(z_t \mid x_t)
\]
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 $S'$. 
Resampling

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

1. Algorithm **systematic_resampling**(S,n):

2. \( S' = \emptyset, \ c_1 = \omega^1 \)

3. **For** \( i = 2 \ldots n \)**

4. \( c_i = c_{i-1} + \omega^i \)

5. \( u_1 \sim U[0,n^{-1}], i = 1 \) **Generate cdf**

6. **For** \( j = 1 \ldots n \)**

7. **While** ( \( u_j > c_i \) ) **Draw samples** …

8. \( i = i + 1 \) **Skip until next threshold reached**

9. \( S' = S' \cup \left\{ < x^i, n^{-1} > \right\} \) **Insert**

10. \( u_{j+1} = u_j + n^{-1} \) **Increment threshold**

11. **Return** \( S' \)

Also called **stochastic universal sampling**
Particle Filters for Mobile Robot Localization

- Each particle is a potential pose of the robot
- Proposal distribution is the motion model of the robot (prediction step)
- The observation model is used to compute the importance weight (correction step)
Motion Model
Proximity Sensor Model (Reminder)

Laser sensor

Sonar sensor
Mobile Robot Localization Using Particle Filters (1)

- Each particle is a potential pose of the robot.

- The set of weighted particles approximates the posterior belief about the robot’s pose (target distribution).
Mobile Robot Localization Using Particle Filters (2)

- Particles are drawn from the motion model (proposal distribution)
- Particles are weighted according to the observation model (sensor model)
- Particles are resampled according to the particle weights
Why is resampling needed?

- We only have a finite number of particles
- Without resampling: The filter is likely to loose track of the “good” hypotheses
- Resampling ensures that particles stay in the meaningful area of the state space
Sample-based Localization (Sonar)
Using Ceiling Maps for Localization

[Dellaert et al. 99]
Vision-based Localization

\[ P(z|x) \]

\[ h(x) \]
Under a Light

Measurement $z$: $P(z|x)$:
Next to a Light

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

Measurement $z$: $P(z|x)$:
Global Localization Using Vision
Limitations

- The approach described so far is able
  - to track the pose of a mobile robot and
  - to globally localize the robot

- How can we deal with localization errors (i.e., the kidnapped robot problem)?
Approaches

- Randomly insert a fixed number of samples with randomly chosen poses
- This corresponds to the assumption that the robot can be teleported at any point in time to an arbitrary location
- Alternatively, insert such samples inversely proportional to the average likelihood of the observations (the lower this likelihood, the higher the probability that the current estimate is wrong).
Summary – Particle Filters

- Particle filters are an implementation of recursive Bayesian filtering
- They represent the posterior by a set of weighted samples
- They can model arbitrary and thus also non-Gaussian distributions
- Proposal to draw new samples
- Weights are computed to account for the difference between the proposal and the target
- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter
Summary – PF Localization

- In the context of localization, the particles are propagated according to the motion model.
- They are then weighted according to the likelihood model (likelihood of the observations).
- In a re-sampling step, new particles are drawn with a probability proportional to the likelihood of the observation.
- This leads to one of the most popular approaches to mobile robot localization.