

Lecture 16: Particle Filters

CS 344R/393R: Robotics
Benjamin Kuipers

Markov Localization

$$Bel(x_t) = \eta P(z_t | x_t) \int P(x_t | u_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

- The integral is evaluated over all x_{t-1} .
 - It computes the probability of reaching x_t from any location x_{t-1} , using the action u_{t-1} .
- The equation is evaluated for every x_t .
 - It computes the posterior probability distribution of x_t .
- Computational efficiency is a problem.
 - $o(k^2)$ if there are k poses x_t .
 - k reflects resolution of position and orientation.

Action and Sensor Models

- Action model: $P(x_t | u_{t-1}, x_{t-1})$
- Sensor model: $P(z_t | x_t)$

- Distributions over possible values of x_t or z_t given specific values of other variables.

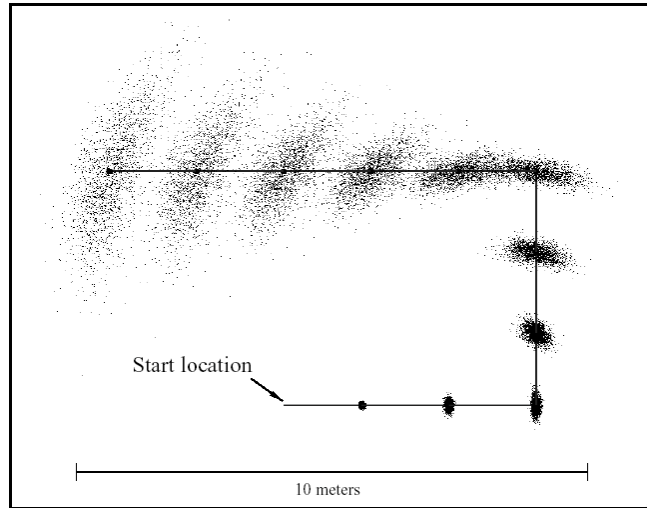
- We discussed these last time.

Monte Carlo Simulation

- Given a probability distribution over inputs, computing the distribution over outcomes can be hard.
 - Simulating a concrete instance is easy.
- Sample concrete instances (“particles”) from the input distribution.
- Collect the outcomes.
 - The distribution of sample outcomes approximates the desired distribution.
- This has been called “particle filtering.”

Actions Disperse the Distribution

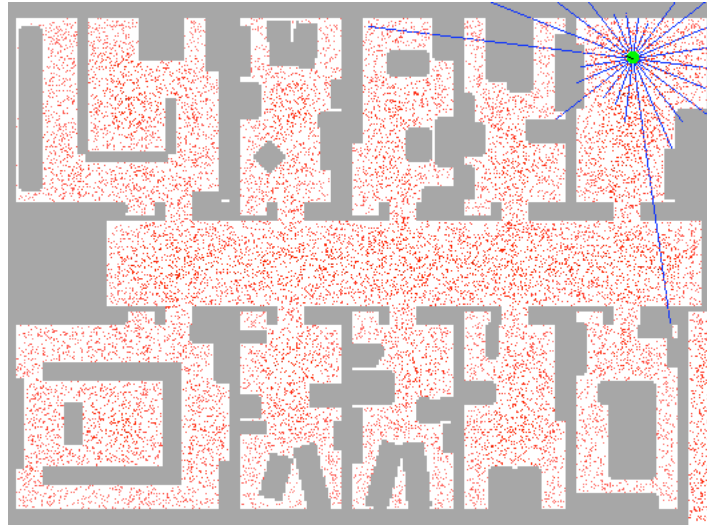
- N particles approximate a probability distribution.
- The distribution disperses under actions



Monte Carlo Localization

- A concrete instance is a particular *pose*.
 - A *pose* is position plus orientation.
- A probability distribution is represented by a collection of N poses.
 - Each pose has an importance factor.
 - The importance factors sum to 1.
- Initialize with
 - N uniformly distributed poses.
 - Equal importance factors of N^{-1} .

Localization Movie (known map)



Representing a Distribution

- The distribution $Bel(x_t)$ is represented by a set S_t of N weighted samples:

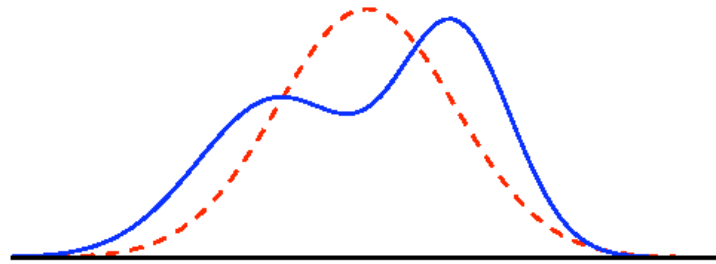
$$S_t = \left\{ \left\langle x_t^{(i)}, w_t^{(i)} \right\rangle \mid i = 1, \dots, N \right\}$$

where $\sum_{i=1}^N w_t^{(i)} = 1$

- A *particle filter* is a Bayes filter that uses this sample representation.

Importance Sampling

- Sample from a *proposal* distribution.
 - Correct to approximate a *target* distribution.



Samples from proposal distribution

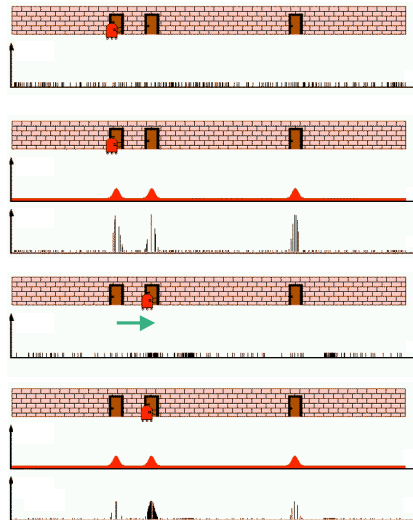


Weighted samples



Simple Example

- Uniform distribution
- Weighting by sensor model
- Prediction by action model
- Weighting by sensor model



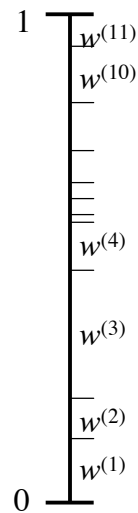
The Basic Particle Filter Algorithm

- **Input:** $u_{t-1}, z_t, S_{t-1} = \left\{ \langle x_{t-1}^{(i)}, w_{t-1}^{(i)} \rangle \mid i = 1, \dots, N \right\}$
 - $S_t := \emptyset, i := 1, \alpha := 0$
- **while** $i \leq N$ **do**
 - sample j from the discrete distribution given by the weights in S_{t-1}
 - sample $x_t^{(i)}$ from $p(x_t \mid u_{t-1}, x_{t-1})$ given $x_{t-1}^{(j)}$ and u_{t-1} .
 - $w_t^{(i)} := p(z_t \mid x_t^{(i)})$
 - $\alpha := \alpha + w_t^{(i)}; i := i + 1$
 - $S_t := S_t \cup \{ \langle x_t^{(i)}, w_t^{(i)} \rangle \}$
- **for** $i := 1$ to N **do** $w_t^{(i)} := w_t^{(i)} / \alpha$
- **return** S_t

Sampling from a Weighted Set of Particles

- Given $S_t = \left\{ \langle x_t^{(i)}, w_t^{(i)} \rangle \mid i = 1, \dots, N \right\}$
- Draw α from a uniform distribution over $[0,1]$.
- Find the minimum k such that

$$\sum_{i=1}^k w_t^{(i)} > \alpha$$
- Return $x_t^{(k)}$



KLD Sampling

- The number N of samples needed can be adapted dynamically, based on the discrete χ^2 (chi-squared) statistic.
- *At each iteration of the particle filter, determine the number of samples such that, with probability $1-\delta$, the error between the true posterior and the sample-based approximation is less than ε .*
- See the handout [Fox, IJRR, 2003].

Kullbach-Liebler Distance

- Consider an unknown distribution $p(x)$ that we approximate with the distribution $q(x)$.
 - How much extra information is required?

$$\begin{aligned} KL(p \parallel q) &= -\sum_x p(x) \log q(x) - \left(-\sum_x p(x) \log p(x) \right) \\ &= \sum_x p(x) \log \frac{p(x)}{q(x)} \end{aligned}$$

- KL distance is non-negative, and zero only when $q(x)=p(x)$, but it's not symmetric.
 - So it's not a metric.

KLD Sampling

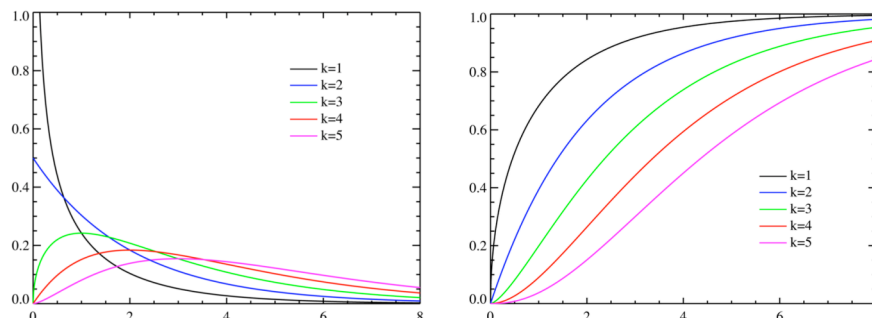
- Let $p(x)$ be the true distribution over k bins.
- Let $q(x)$ be the maximum likelihood estimate of $p(x)$ given n samples.
- We can guarantee $P(KL(p||q) \leq \varepsilon) = 1 - \delta$ by choosing the number of samples n according to the Chi-square distribution with $k-1$ degrees of freedom.

$$n = \frac{1}{2\varepsilon} \chi_{k-1, 1-\delta}^2 = \frac{k-1}{2\varepsilon} \left\{ 1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}} z_{1-\delta} \right\}$$

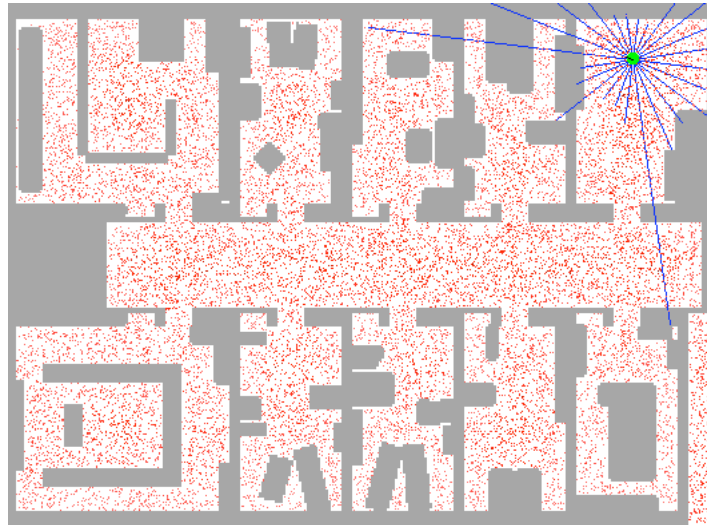
Chi-Square Distribution

- If $X_i \sim N(0,1)$ are k independent random variables, then the random variable $Q = \sum_{i=1}^k X_i^2$

is distributed according to the Chi-square distribution with k degrees of freedom: $Q \sim \chi_k^2$



Localization Movie (known map)



MCL Algorithm

- Repeat to collect N samples.
 - Draw a sample x_{t-1} from the distribution $Bel(x_{t-1})$, with likelihood given by its importance factor.
 - Given an action u_{t-1} and the action model distribution $P(x_t | u_{t-1}, x_{t-1})$, sample state x_t .
 - Assign the importance factor $P(z_t | x_t)$ to x_t .
- Normalize the importance factors.
- Repeat for each time-step.

$$Bel(x_t) = \eta P(z_t | x_t) \int P(x_t | u_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

MCL works quite well

- $N=1000$ seems to work OK.
- Straight MCL works best for sensors that are **not** highly accurate.
 - For very accurate sensors, $P(z_t | x_t)$ is very narrow and highly peaked.
 - Poses x_t that are nearly (but not exactly) correct can get low importance values.
 - They may be under-represented in the next generation.

An Alternative: Mixture Proposal Distribution

- In Monte Carlo, the *proposal distribution* is the distribution for selecting the concrete hypotheses (“particles”).
- For MCL, the proposal distribution for $\langle x_t, x_{t-1} \rangle$ is $P(x_t | u_{t-1}, x_{t-1}) \times Bel(x_{t-1})$
- Instead, we can use a mixture of several different proposal distributions.

Dual Proposal Distribution

- Draw sample particles x_t based on the sensor model distribution $P(z_t | x_t)$.
 - This is *not* straight-forward.
- Draw sample particles x_{t-1} based on $Bel(x_{t-1})$.
 - The proposal distribution for $\langle x_t, x_{t-1} \rangle$ is $P(z_t | x_t) \times Bel(x_{t-1})$
- Each pair gets importance factor $P(x_t | u_{t-1}, x_{t-1})$
- Normalize to sum to 1.

$$Bel(x_t) = \eta P(z_t | x_t) \int P(x_t | u_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

Mixture Proposal

- Some particles are proposed based on prior position and the *action* model.
 - Vulnerable to problems with highly accurate sensors!
- Some particles are proposed based on prior position and the *sensor* model.
 - Vulnerable to problems due to sensor noise.
- A mixture does better than either.
 - Good results with as few as $N = 50$ particles!
 - Use $k M_1 + (1-k) M_2$ for $0 < k < 1$.

A mixture proposal distribution for the mapping assignment

- Use a mixture of
 - 90% MCL proposal distribution based on the sensor and action models given;
 - 10% broader Gaussian distribution, spreading particles around in case you have a major error.
- The dual proposal distribution is too hard to implement.

Make a Good Graphical Display

- Show the evolving occupancy grid map at each step.
- Show the distribution of particles during localization.
 - The display can only show (x, y) , but
 - The particle is really (x, y, θ)
- Start at the origin of a big array (memory is cheap!).
 - Keep a bounding box around the useful part, and only compute and display that.
 - Update the bounding box as the map grows.