ECE276A: Sensing & Estimation in Robotics
Lecture 6: Bayesian and Particle Filtering

Instructor:

Nikolay Atanasov: natanasov@ucsd.edu

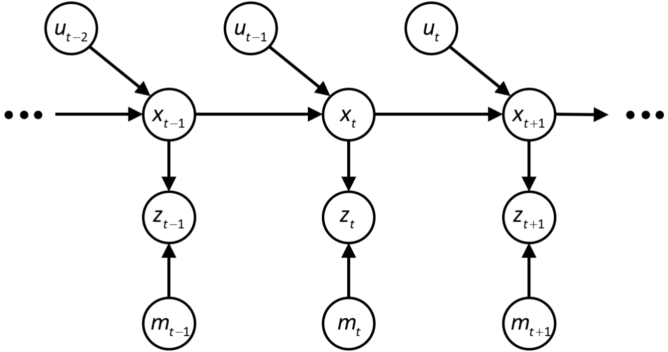
Teaching Assistants:

Qiaojun Feng: qif007@eng.ucsd.edu
Tianyu Wang: tiw161@eng.ucsd.edu
Ibrahim Akbar: iakbar@eng.ucsd.edu
You-Yi Jau: yjau@eng.ucsd.edu
Harshini Rajachander: hrajacha@eng.ucsd.edu

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Electrical and Computer Engineering

Structure of Robotics Problems

- **Time**: t (discrete or continuous)
- **Robot state**: x_t (e.g., position, orientation, velocity, etc.)
- **Control input**: u_t (e.g., quadrotor thrust and moment of rotation)
- **Observation**: z_t (e.g., image, laser scan, inertial measurements)
- **Environment state**: m_t (e.g., map of the occupancy of space)



Structure of Robotics Problems

- The sequences of control inputs $u_{0:t}$ and observations $z_{0:t}$ are assumed known/observed
- The sequences of robot states $x_{0:t}$ and environment states $m_{0:t}$ are unknown/hidden
- **Markov Assumptions**
  - The state x_{t+1} only depends on the previous input u_t and state x_t
  - The observation z_t only depends on the robot state x_t and the environment state m_t
- **Motion Model**: a function f (or equivalently a probability density function p_f) that describes the motion of the robot to a new state x_{t+1} after applying control input u_t at state x_t

$$x_{t+1} = f(x_t, u_t, w_t) \sim p_f(\cdot \mid x_t, u_t) \qquad w_t = \text{motion noise}$$

- **Observation Model**: a function h (or a probability density function p_h) that describes the observation z_t of the robot depending on x_t and m_t

$$z_t = h(x_t, m_t, v_t) \sim p_h(\cdot \mid x_t, m_t) \qquad v_t = \text{observation noise}$$

Bayes Filter

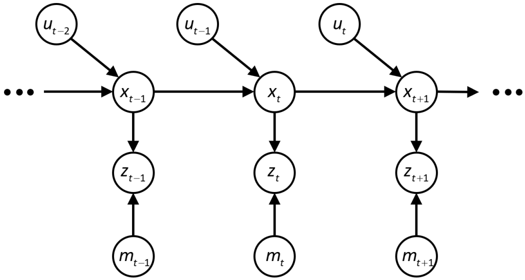
- A **Bayes filter** is a probabilistic tool for estimating the state of dynamical systems (robot and/or environment) that combines evidence from control inputs and observations using **Markov assumptions** and **Bayes rule**:
  - **Total probability**: $p(x) = \int p(x,y)dy$
  - **Conditioning**: $p(x,y) = p(y \mid x)p(x)$
  - **Bayes rule**: $p(x \mid y,z) = \frac{p(y \mid x,z)p(x \mid z)}{\int p(y,s \mid z)ds} = \frac{p(y \mid x,z)p(z \mid x)p(x)}{p(y \mid z)p(z)}$
- Special cases of the Bayes filter:
  - Kalman filter
  - Particle filter
  - Forward algorithm for Hidden Markov Models (HMMs)

Filtering Examples

- Track the center $c_t \in \mathbb{R}^2$ and radius $r_t \in \mathbb{R}$ of a ball in images: `http://www.pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/`
- Track the position $p_t \in \mathbb{R}^3$ and orientation $R_t \in SO(3)$ of a camera: `https://www.youtube.com/watch?v=CsJkci5lfco`
- Estimate the probability of occupancy of the environment: `https://www.youtube.com/watch?v=RhPlzIyTT58`

Filtering Problem

- The Markov assumptions are used to decompose the joint pdf of the states $x_{0:T}$ (robot and map combined), observations $z_{0:T}$, and controls $u_{0:T-1}$



- **Joint distribution**:

$$p(x_{0:T}, z_{0:T}, u_{0:T-1}) = \underbrace{p_{0|0}(x_0)}_{\text{prior}} \prod_{t=0}^{T} \underbrace{p_h(z_t \mid x_t)}_{\text{observation model}} \prod_{t=1}^{T} \underbrace{p_f(x_t \mid x_{t-1}, u_{t-1})}_{\text{motion model}}$$

- **Filtering**: keeps track of

$$p_{t|t}(x_t) := p(x_t \mid z_{0:t}, u_{0:t-1})$$
$$p_{t+1|t}(x_{t+1}) := p(x_{t+1} \mid z_{0:t}, u_{0:t})$$

- **Smoothing**: keeps track of

$$p_{t|t}(x_{0:t}) := p(x_{0:t} \mid z_{0:t}, u_{0:t-1})$$
$$p_{t+1|t}(x_{0:t+1}) := p(x_{0:t+1} \mid z_{0:t}, u_{0:t})$$

Bayes Filter

- **Prediction step**: given a prior density $p_{t|t}$ over x_t and the control input u_t, uses the motion model p_f to compute the predicted density $p_{t+1|t}$ over x_{t+1}:

$$\boxed{p_{t+1|t}(x) = \int p_f(x \mid s, u_t) p_{t|t}(s) ds}$$

- **Update step**: given the predicted density $p_{t+1|t}$ over x_{t+1} and the measurement z_{t+1}, uses the observation model p_h to incorporate the measurement information and obtain the posterior $p_{t+1|t+1}$ over x_{t+1}:

$$\boxed{p_{t+1|t+1}(x) = \frac{p_h(z_{t+1} \mid x) p_{t+1|t}(x)}{\int p_h(z_{t+1} \mid s) p_{t+1|t}(s) ds}}$$

Bayes Filter

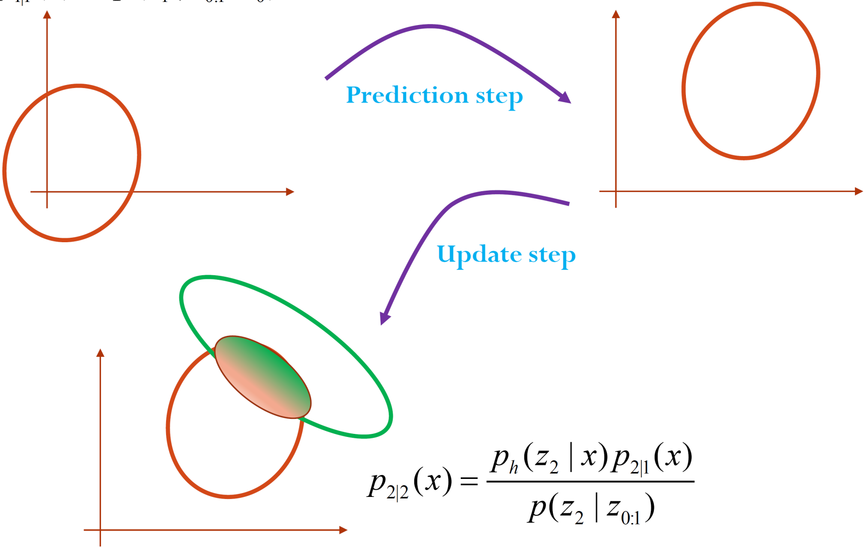
$$
\begin{aligned}
p_{t+1|t+1}(x_{t+1}) =& p(x_{t+1} \mid z_{0:t+1}, u_{0:t}) \\
\overset{\text{Bayes}}{=\!=\!=} & \frac{1}{\eta_{t+1}} p(z_{t+1} \mid x_{t+1}, z_{0:t}, u_{0:t}) p(x_{t+1} \mid z_{0:t}, u_{0:t}) \\
\overset{\text{Markov}}{=\!=\!=} & \frac{1}{\eta_{t+1}} p_h(z_{t+1} \mid x_{t+1}) p(x_{t+1} \mid z_{0:t}, u_{0:t}) \\
\overset{\text{Total prob.}}{=\!=\!=} & \frac{1}{\eta_{t+1}} p_h(z_{t+1} \mid x_{t+1}) \int p(x_{t+1}, x_t \mid z_{0:t}, u_{0:t}) dx_t \\
\overset{\text{Cond. prob.}}{=\!=\!=} & \frac{1}{\eta_{t+1}} p_h(z_{t+1} \mid x_{t+1}) \int p(x_{t+1} \mid z_{0:t}, u_{0:t}, x_t) p(x_t \mid z_{0:t}, u_{0:t}) dx_t \\
\overset{\text{Markov}}{=\!=\!=} & \frac{1}{\eta_{t+1}} p_h(z_{t+1} \mid x_{t+1}) \int p_f(x_{t+1} \mid x_t, u_t) p(x_t \mid z_{0:t}, u_{0:t-1}) dx_t \\
=& \boxed{\frac{1}{\eta_{t+1}} p_h(z_{t+1} \mid x_{t+1}) \int p_f(x_{t+1} \mid x_t, u_t) p_{t|t}(x_t) dx_t}
\end{aligned}
$$

- **Normalization constant**: $\eta_{t+1} := p(z_{t+1} \mid z_{0:t}, u_{0:t})$

Bayes Filter

$$p_{1|1}(x) := p(x_1 \mid z_{0:1}, u_0)$$

$$p_{2|1}(x) = \int p_f\,(x \mid s, u_1)\, p_{1|1}(s)\, ds$$



$$p_{2|2}(x) = \frac{p_h(z_2 \mid x)\, p_{2|1}(x)}{p(z_2 \mid z_{0:1})}$$

Bayes Filter Summary

- **Motion model**: $x_{t+1} = f(x_t, u_t, w_t) \sim p_f(\cdot \mid x_t, u_t)$
- **Observation model**: $z_t = h(x_t, v_t) \sim p_h(\cdot \mid x_t)$
- **Joint distribution**:

$$p(x_{0:T}, z_{0:T}, u_{0:T-1}) = \underbrace{p_{0|0}(x_0)}_{\text{prior}} \prod_{t=0}^{T} \underbrace{p_h(z_t \mid x_t)}_{\text{observation model}} \prod_{t=0}^{T} \underbrace{p_f(x_t \mid x_{t-1}, u_{t-1})}_{\text{motion model}}$$

- **Filtering**: recursive implementation that keeps track of

$$p_{t|t}(x_t) := p(x_t \mid z_{0:t}, u_{0:t-1})$$
$$p_{t+1|t}(x_{t+1}) := p(x_{t+1} \mid z_{0:t}, u_{0:t})$$

- **Bayes filter**:

$$p_{t+1|t+1}(x_{t+1}) = \underbrace{\overbrace{\frac{1}{p(z_{t+1}|z_{0:t}, u_{0:t})}}^{\frac{1}{\eta_{t+1}}} p_h(z_{t+1} \mid x_{t+1}) \overbrace{\int p_f(x_{t+1} \mid x_t, u_t) p_{t|t}(x_t) dx_t}^{\textbf{Predict: } p_{t+1|t}(x_{t+1})}}_{\textbf{Update}}$$

Bayes Smoother

- **Smoothing**: keeps track of

$$p_{t|t}(x_{0:t}) := p(x_{0:t} \mid z_{0:t}, u_{0:t-1})$$
$$p_{t+1|t}(x_{0:t+1}) := p(x_{0:t+1} \mid z_{0:t}, u_{0:t})$$

- Forward pass (**Bayes filter**): compute $p(x_{t+1} \mid z_{0:t+1}, u_{0:t})$ and $p(x_{t+1} \mid z_{0:t}, u_{0:t})$ for $t = 0, \ldots, T$

- Backward pass (**Bayes smoother**): for $t = T-1, \ldots, 0$ compute:

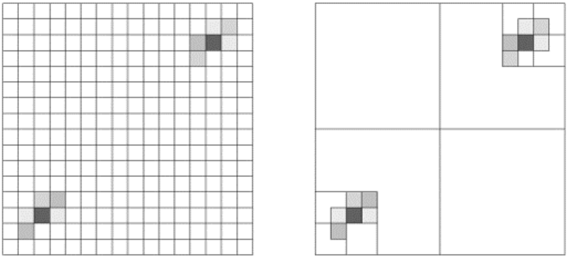
$$p(x_t \mid z_{0:T}, u_{0:T-1}) \overset{\text{Total}}{\underset{\text{Probability}}{=\!=}} \int p(x_t \mid x_{t+1}, z_{0:T}, u_{0:T-1}) p(x_{t+1} \mid z_{0:T}, u_{0:T-1}) dx_{t+1}$$

$$\overset{\text{Markov}}{\underset{\text{Assumption}}{=\!=}} \int p(x_t \mid x_{t+1}, z_{0:t}, u_{0:t}) p(x_{t+1} \mid z_{0:T}, u_{0:T-1}) dx_{t+1}$$

$$\overset{\text{Bayes}}{\underset{\text{Rule}}{=\!=}} \underbrace{p(x_t \mid z_{0:t}, u_{0:t-1})}_{\text{forward pass}} \int \left[\frac{\overbrace{p_f(x_{t+1} \mid x_t, u_t)}^{\text{motion model}} p(x_{t+1} \mid z_{0:T}, u_{0:T-1})}{\underbrace{p(x_{t+1} \mid z_{0:t}, u_{0:t})}_{\text{forward pass}}} \right] dx_{t+1}$$

Histogram Filter

- Represents the pdfs $p_{t|t}$ and $p_{t+1|t}$ via a histogram over a discrete set of possible values
- The accuracy is limited by the grid size
- A small grid becomes very computationally expensive in high dimensional state spaces because the number of cells is exponential in the number of dimensions
- **Adaptive Histogram Filter**: represents the pdf via adaptive discretization, e.g., octrees



Histogram Filter

- **Prediction step**
  - Assumes bounded Gaussian noise in the motion model
  - Realizes the prediction step by shifting the data in the grid according to the control input and convolving the grid with a **separable** Gaussian kernel:

$$\begin{array}{|c|c|c|}\hline 1/16 & 1/8 & 1/16 \\ \hline 1/8 & 1/4 & 1/8 \\ \hline 1/16 & 1/8 & 1/16 \\ \hline \end{array} \quad \cong \quad \begin{array}{|c|}\hline 1/4 \\ \hline 1/2 \\ \hline 1/4 \\ \hline \end{array} + \begin{array}{|c|c|c|}\hline 1/4 & 1/2 & 1/4 \\ \hline \end{array}$$

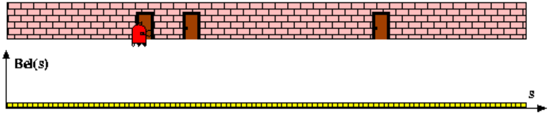
  - This reduces the prediction step cost from $O(n^2)$ to $O(n)$ where n is the number of cells
- **Update step**
  - To update and normalize the pdf upon sensory input, one has to iterate over all cells
  - Is it possible to monitor which part of the state space is affected by the observations and only update that?

Markov Localization

- **Robot Localization Problem**: Given a map m, a sequence of control inputs $u_{0:t-1}$, and a sequence of measurements $z_{0:t}$, infer the state of the robot x_t
- **Approach**: use a Bayes filter with a multi-modal distribution in order to capture multiple hypotheses about the robot state, e.g.:
  - Histogram filter
  - Particle filter
  - Gaussian mixture filter
- **Pruning**: need to keep the number of hypotheses/components under control
- **Important considerations**: What are the motion and observation models and how is the map represented?

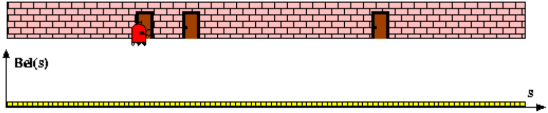
Histogram Filter Localization (1-D)

Prior:

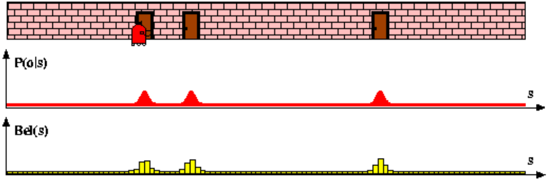


Histogram Filter Localization (1-D)

Prior:

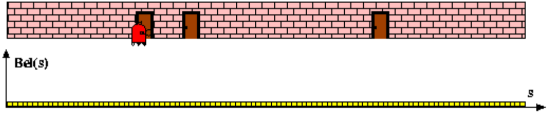


Update:

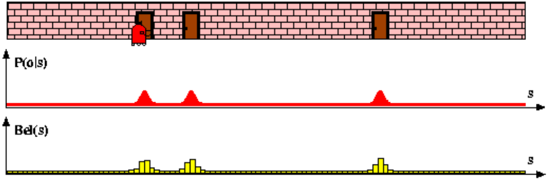


Histogram Filter Localization (1-D)

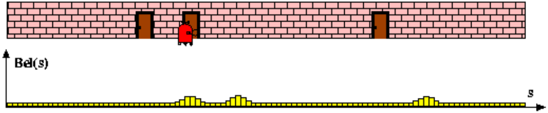
Prior:



Update:



Predict:



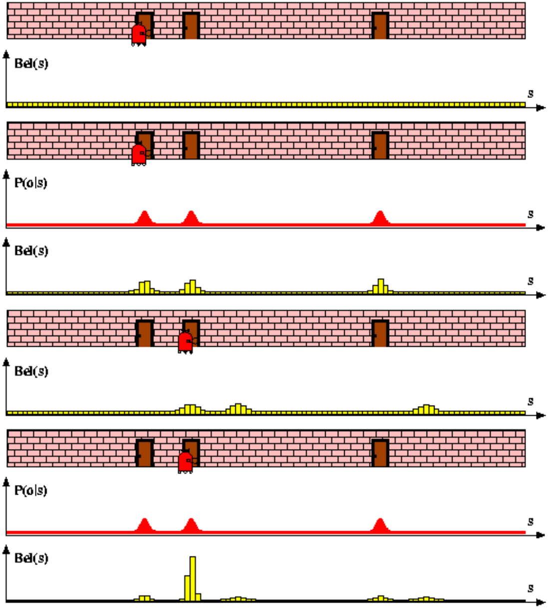
Histogram Filter Localization (1-D)

Prior:

Update:

Predict:

Update:



Particle Filter

- Uses a mixture of delta functions (**particles**):
$$\delta(x;\mu^{(k)}) := \begin{cases} 1 & x = \mu^{(k)} \\ 0 & \text{else} \end{cases} \qquad \text{for } k = 1,\ldots,N$$
with weights $\alpha^{(k)}$ to represent the pdfs $p_{t|t}$ and $p_{t+1|t}$
- To derive the filter, substitute the delta mixture pdf in the Bayes filter prediction and update steps
- **Prior distribution**: $x_t \mid z_{0:t}, u_{0:t-1} \sim p_{t|t}(x_t) := \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} \delta\left(x_t; \mu_{t|t}^{(k)}\right)$
- **Prediction**:
$$p_{t+1|t}(x) = \int p_f(x \mid s, u_t) \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} \delta\left(s; \mu_{t|t}^{(k)}\right) ds \overset{??}{\approx} \sum_{k=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(k)} \delta\left(x; \mu_{t+1|t}^{(k)}\right)$$
- **Update**:
$$p_{t+1|t+1}(x) = \frac{p_h\left(z_{t+1} \mid x\right) \sum_{k=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(k)} \delta\left(x; \mu_{t+1|t}^{(k)}\right)}{\int p_h\left(z_{t+1} \mid s\right) \sum_{j=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(j)} \delta\left(s; \mu_{t+1|t}^{(j)}\right) ds} \overset{??}{\approx} \sum_{k=1}^{N_{t+1|t+1}} \alpha_{t+1|t+1}^{(k)} \delta\left(x; \mu_{t+1|t+1}^{(k)}\right)$$

Particle Filter Prediction

- How do we approximate the prediction step as a delta-mixture pdf?

$$
\begin{aligned}
p_{t+1|t}(x) &= \int p_f(x \mid s, u_t) \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} \delta\left(s; \mu_{t|t}^{(k)}\right) ds \\
&= \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} p_f(x \mid \mu_{t|t}^{(k)}, u_t) \overset{??}{\approx} \sum_{k=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(k)} \delta\left(x; \mu_{t+1|t}^{(k)}\right)
\end{aligned}
$$

- Since $p_{t+1|t}(x)$ is a mixture pdf, we can approximate it with particles by drawing samples directly from it
- Let $N_{t+1|t}$ be the number of particles in the approximation (usually, $N_{t+1|t} = N_{t|t}$)
- **Bootstrap approximation**: repeat $N_{t+1|t}$ times and normalize the weights at the end:
  - Draw $j \in \{1, \ldots, N_{t|t}\}$ with probability $\alpha_{t|t}^{(j)}$
  - Draw $\mu_{t+1|t}^{(j)} \sim p_f\left(\cdot \mid \mu_{t|t}^{(j)}, u_t\right)$
  - Add the weighted sample $\left(\mu_{t+1|t}^{(j)}, p_{t+1|t}\left(\mu_{t+1|t}^{(j)}\right)\right)$ to the new particle set

Particle Filter Update

- Update step: evaluates Bayes rule with the delta mixture pdf

$$
\begin{aligned}
p_{t+1|t+1}(x) &= \frac{p_h\left(z_{t+1} \mid x\right) \sum_{k=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(k)} \delta\left(x; \mu_{t+1|t}^{(k)}\right)}{\int p_h\left(z_{t+1} \mid s\right) \sum_{j=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(j)} \delta\left(s; \mu_{t+1|t}^{(j)}\right) ds} \\
&= \sum_{k=1}^{N_{t+1|t}} \left[\frac{\alpha_{t+1|t}^{(k)} p_h\left(z_{t+1} \mid \mu_{t+1|t}^{(k)}\right)}{\sum_{j=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(j)} p_h\left(z_{t+1} \mid \mu_{t+1|t}^{(j)}\right)} \right] \delta\left(x; \mu_{t+1|t}^{(k)}\right)
\end{aligned}
$$

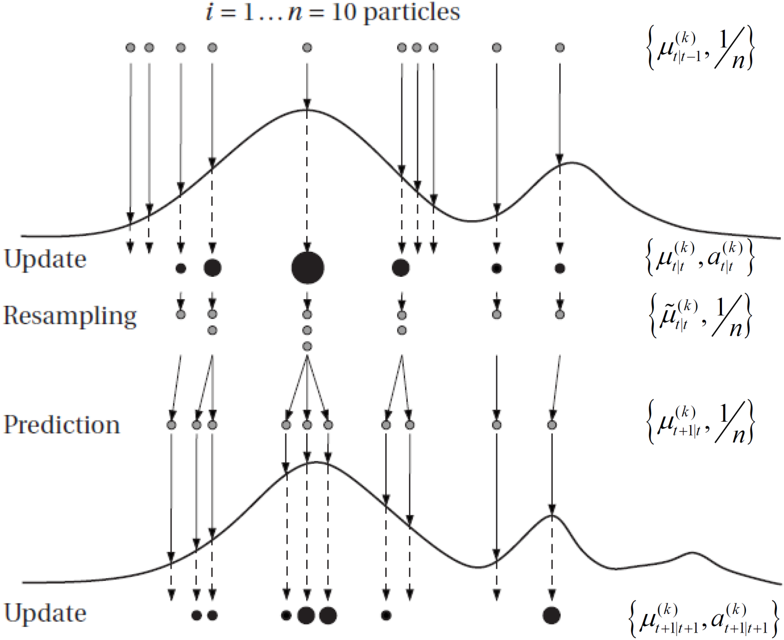
- The resulting pdf turns out to be a delta mixture so no approximation is necessary

- The update step does not update the particle positions but only their weights

Particle Filter Resampling

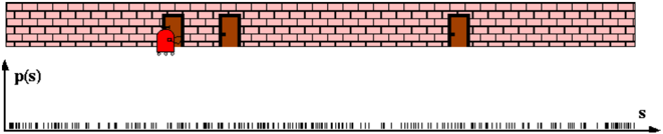
- How do we avoid **particle depletion** - a situation in which most of the updated particle weights become close to zero because the finite set of particles are not accurate hypotheses, i.e., the observation likelihoods $p_h\left(z_{t+1} \mid \mu_{t+1|t}^{(k)}\right)$ are small at all $k = 1, \ldots, N_{t+1|t}$?
- The particle filter uses a procedure called **resampling** to avoid particle depletion during the update step
- Given a weighted set of particles, resampling creates a new particles set with equal weights by adding many particles to the locations that had high weight and few particles to the locations that had low weights
- Resampling focuses the representation power of the particles to likely regions, while leaving unlikely regions with only few particles
- Resampling is applied at time t if the **effective number of particles**:
$\boxed{N_{eff} := \frac{1}{\sum_{k=1}^{N_{t|t}} \left(\alpha_{t|t}^{(k)}\right)^2}}$ is less than a threshold

Particle Filter Resampling



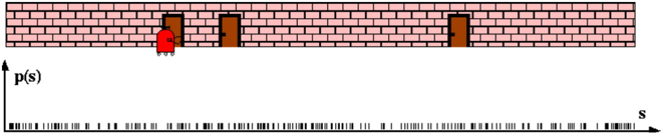
Particle Filter Localization (1-D)

Prior:

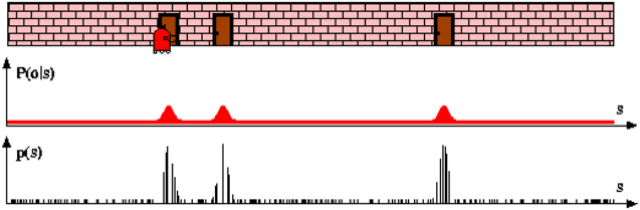


Particle Filter Localization (1-D)

Prior:



Update:

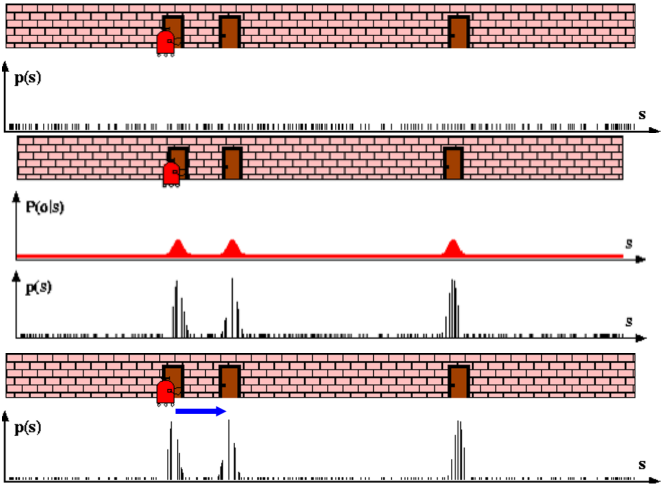


Particle Filter Localization (1-D)

Prior:

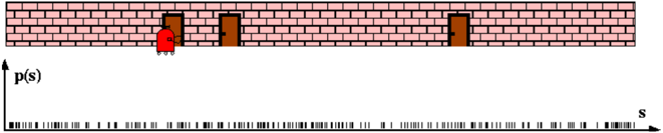
Update:

Predict:

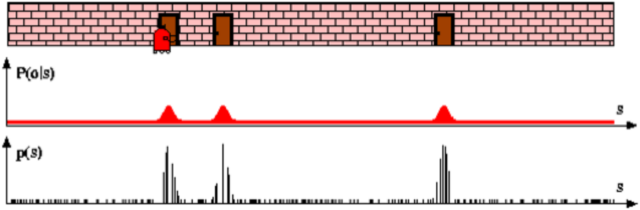


Particle Filter Localization (1-D)

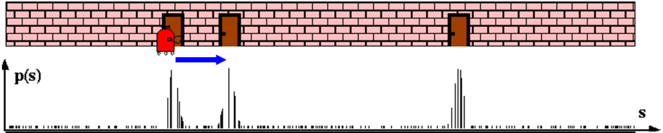
Prior:



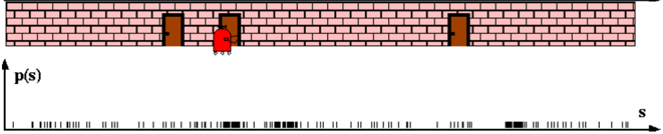
Update:



Predict:

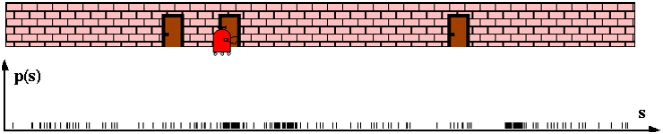


Resample:



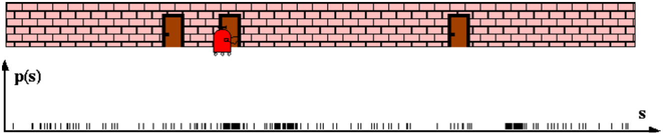
Particle Filter Localization (1-D)

Prior:

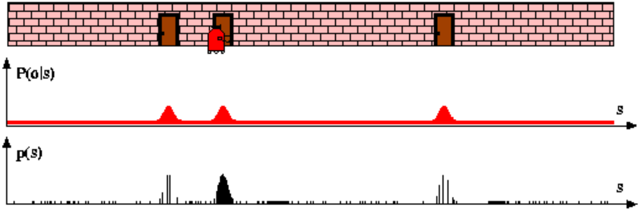


Particle Filter Localization (1-D)

Prior:

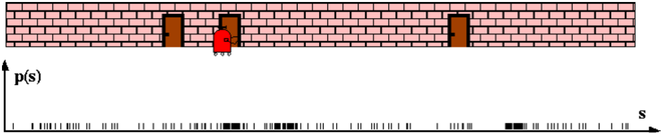


Update:

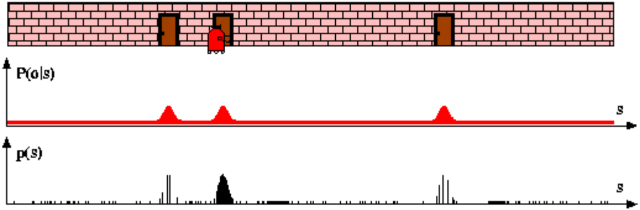


Particle Filter Localization (1-D)

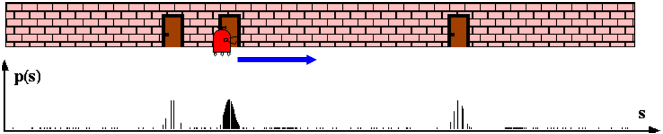
Prior:



Update:



Predict:



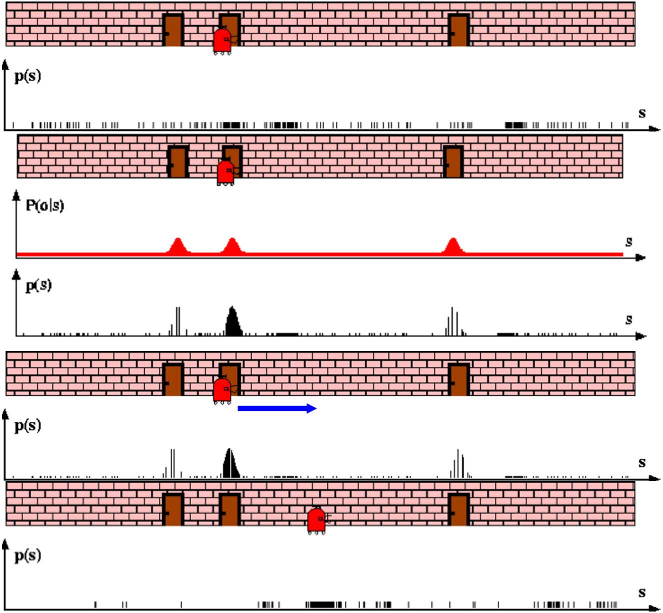
Particle Filter Localization (1-D)

Prior:

Update:

Predict:

Resample:

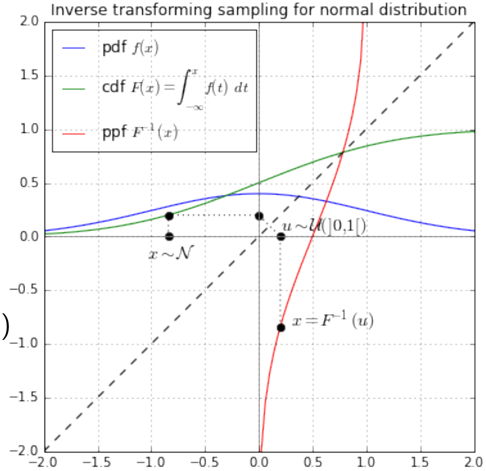


Inverse Transform Sampling

- **Target distribution**: How do we sample from a distribution with pdf $p(x)$ and CDF $F(x) = \int_{-\infty}^{x} p(s)ds$?

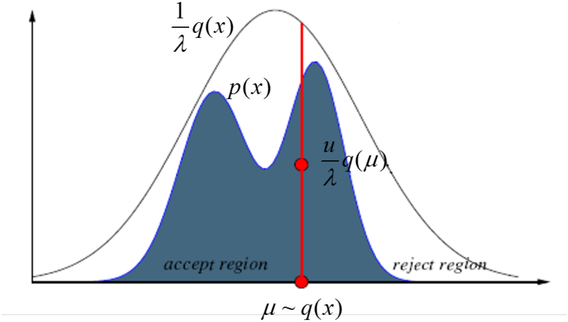
- **Inverse Transform Sampling**:
  1. Draw $u \sim \mathcal{U}(0, 1)$
  2. Return inverse CDF value: $\mu = F^{-1}(u)$
  3. The CDF of $F^{-1}(u)$ is:

$$\begin{aligned} \mathbb{P}(F^{-1}(u) \leq x) &= \mathbb{P}(u \leq F(x)) \\ &= F(x) \end{aligned}$$



Rejection Sampling

- **Target distribution**: How do we sample from a complicated pdf $p(x)$?
- **Proposal distribution**: use another pdf $q(x)$ that is easy to sample from (e.g., Uniform, Gaussian) and: $\lambda p(x) \leq q(x)$ with $\lambda \in (0, 1)$
- **Rejection Sampling**:
  1. Draw $u \sim \mathcal{U}(0, 1)$ and $\mu \sim q(\cdot)$
  2. Return μ only if $u \leq \frac{\lambda p(\mu)}{q(\mu)}$. If λ is small, many rejections are necessary
- Good $q(x)$ and λ are **hard to choose** in practice



Sample Importance Resampling (SIR)

- How about rejection sampling without λ?
- **Sample Importance Resampling** for a target distribution $p(\cdot)$ with proposal distribution $q(\cdot)$
  1. Draw $\mu^{(1)}, \dots, \mu^{(N)} \sim q(\cdot)$
  2. Compute importance weights $\alpha^{(k)} = \frac{p(\mu^{(k)})}{q(\mu^{(k)})}$ and normalize: $\alpha^{(k)} = \frac{\alpha^{(k)}}{\sum_j \alpha^{(j)}}$
  3. Draw $\mu^{(k)}$ independently with replacement from $\{\mu^{(1)}, \dots, \mu^{(N)}\}$ with probability $\alpha^{(k)}$ and add to the final sample set with weight $\frac{1}{N}$
- If $q(\cdot)$ is a poor approximation of $p(\cdot)$, then the best samples from q are not necessarily good samples for resampling
- **SIR applied to the particle filter**:
  - draw $\mu^{(k)}$ independently with replacement from $\{\mu^{(1)}, \dots, \mu^{(N)}\}$ with probability $\alpha^{(k)}$ and add to the final sample set with weight $\frac{1}{N}$

Markov Chain Monte Carlo Resampling

- The main drawback of rejection sampling and SIR is that choosing a good proposal distribution $q(\cdot)$ is hard
- **Idea**: let the proposed samples μ depend on the last accepted sample μ', i.e., obtain correlated samples from a conditional proposal distribution $\mu^{(k)} \sim q\left(\cdot \mid \mu^{(k-1)}\right)$
- Under certain conditions, the samples generated from $q(\cdot \mid \mu')$ form an ergodic Markov chain with $p(\cdot)$ as its stationary distribution
- MCMC methods include Metropolis-Hastings and Gibbs sampling

Stratified Resampling

- In the last step of SIR, the weighted sample set $\{\mu^{(k)}, \alpha^{(k)}\}$ is resampled independently with replacement

- This might result in high variance resampling, i.e., sometimes some samples with large weights might not be selected or samples with very small weights may be selected multiple times

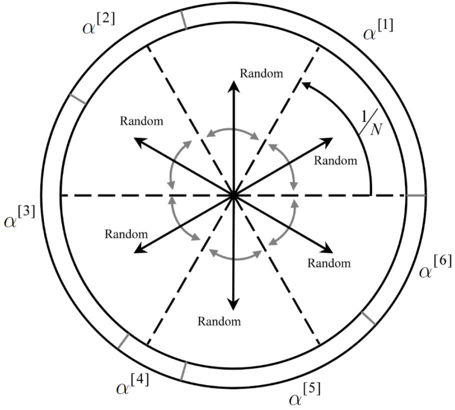
- **Stratified resampling**: guarantees that samples with large weights appear at least once and those with small weights – at most once. Stratified resampling is **optimal in terms of variance** (Thrun et al. 2005)

- Instead of selecting samples independently, use a sequential process:
  - Add the weights along the circumference of a circle
  - Divide the circle into N equal pieces and sample a uniform on each piece
  - Samples with large weights are chosen at least once and those with small weights – at most once

Stratified and Systematic Resampling

Stratified (low variance) resampling

1: **Input**: particle set $\left\{\mu^{(k)}, \alpha^{(k)}\right\}_{k=1}^{N}$
2: **Output**: resampled particle set
3: $j \leftarrow 1,\ c \leftarrow \alpha^{(1)}$
4: **for** $k = 1, \ldots, N$ **do**
5: $\quad u \sim \mathcal{U}\left(0, \frac{1}{N}\right)$
6: $\quad \beta = u + \frac{k-1}{N}$
7: $\quad$ **while** $\beta > c$ **do**
8: $\quad\quad j = j + 1,\ c = c + \alpha^{(j)}$
9: $\quad$ add $\left(\mu^{(j)}, \frac{1}{N}\right)$ to the new set



- **Systematic resampling**: the same as stratified resampling except that the **same** uniform is used for each piece, i.e., $u \sim \mathcal{U}\left(0, \frac{1}{N}\right)$ is sampled only once before the for loop above.

Particle Filter Summary

- **Prior**: $x_t \mid z_{0:t}, u_{0:t-1} \sim p_{t|t}(x_x) := \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} \delta\left(x_t; \mu_{t|t}^{(k)}\right)$
- **Prediction**: approximate the mixture by sampling:

$$p_{t+1|t}(x) = \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} p_f(x \mid \mu_{t|t}^{(k)}, u_t) \approx \sum_{k=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(k)} \delta\left(x; \mu_{t+1|t}^{(k)}\right)$$

- **Update**: rescale the particles based on the observation likelihood:

$$p_{t+1|t+1}(x) = \sum_{k=1}^{N_{t+1|t}} \left[\frac{\alpha_{t+1|t}^{(k)} p_h\left(z_{t+1} \mid \mu_{t+1|t}^{(k)}\right)}{\sum_{j=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(j)} p_h\left(z_{t+1} \mid \mu_{t+1|t}^{(j)}\right)} \right] \delta\left(x; \mu_{t+1|t}^{(k)}\right)$$

- If $N_{eff} := \frac{1}{\sum_{k=1}^{N_{t|t}} \left(\alpha_{t|t}^{(k)}\right)^2} \leq N_{threshold}$, **resample** the particle set $\left\{\mu_{t+1|t+1}^{(k)}, \alpha_{t+1|t+1}^{(k)}\right\}$ via stratified or sample importance resampling