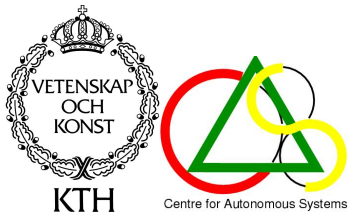
EL2320 - Nonparametric Filters

John Folkesson

Nonparameteric Filters
(Chap 4 and 8 in Thrun)



Nonparameteric Misnomer

"As the number of *parameters* goes to infinity, *nonparametric* techiques tend to converge uniformly to the correct posterior" - The Book

Nonparameteric Misnomer

Better name:
Distribution Free, there is no assumption on the functional form of the underling probability distribution.

Nonparameteric?

We have upto now estimated parameters of our a posteriori distribution. Mainly the mean and variance of a Gaussian.
Many real distributions are not even remotely Gaussian and neither any other analytic distribution.
We now turn to these problems. They typically represent the posteriori by estimates of its value at a finite number of states.
Two of the ways to chose the states to represent:

- break up the state space into fixed regions, histogram filters.
- break up the state space adaptively where regions of highest posteriori probability have the posteriori estimated at more closely spaced points, particle filters (or even a sum of Gausssians).

Histogram vs. Discrete Bayes Filter

The Discrete Bayes Filter estimates a distribution for a Discrete random variable X taking on values from some set $\{x_i\}$, $p_k = P(X = x_k)$

The Histogram Filter estimates the distribution for a Continuous Random Variable X, by turning it into a discrete one. It breaks up the continuous space into a finite number of bins.

The value of X then transforms into the index of the bin it falls into.

One then uses the Discrete Bayes Filter on the problem.

Discrete Bayes Filter

Predict:

$\bar{p}_{k,t} = \sum_i P(X_t = x_k | u_t, X_{t-1} = x_i) p_{i,t-1}$

Update:

$p_{k,t} \propto p(z_t | X_t = x_k) \bar{p}_{k.t}$

Must do above for every k then:

$\sum_k p_{k,t} = 1$

Finishes the t^{th} estimate

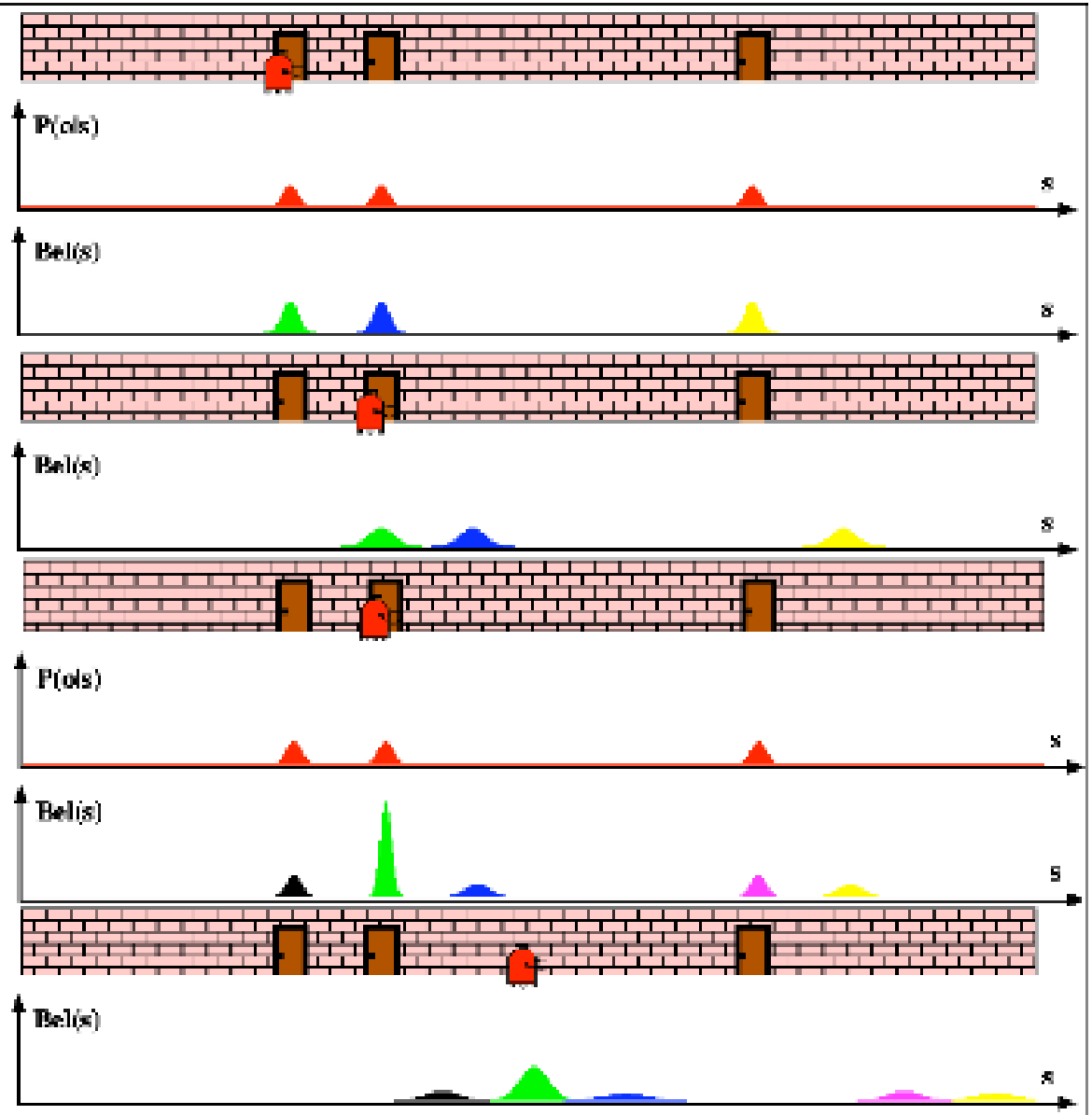
Discrete Bayes Filter

Predict: $\bar{p}_{k,t} = \sum_i P(X_t = x_k | u_t, X_{t-1} = x_i) p_{i,t-1}$
Update: $p_{k,t} \propto p(z_t | X_t = x_k) \bar{p}_{k.t}$ $\sum_k p_{k,t} = 1$

You are cooking a stew which has an amount of salt in it already. It is either 1 or 2 grams.
You are able to add salt in exact gram amounts but can not be sure if the amount added is 1 or 2 grams (equal probability).
You want exactly 4 grams of salt.
You can taste the stew and are sure to tell if the amount of salt is low or high by 2 or more grams but can not tell the difference between 3, 4, or 5 (they all taste right).
What are all the above $p, P, \bar{p}, u, X, x$ and z for the start?
If you taste that it is not enough salt after adding salt at t=1 what does $p_{k,1}$ look like?
Is there any chance that by adding and tasting you can know how much salt is in the stew?

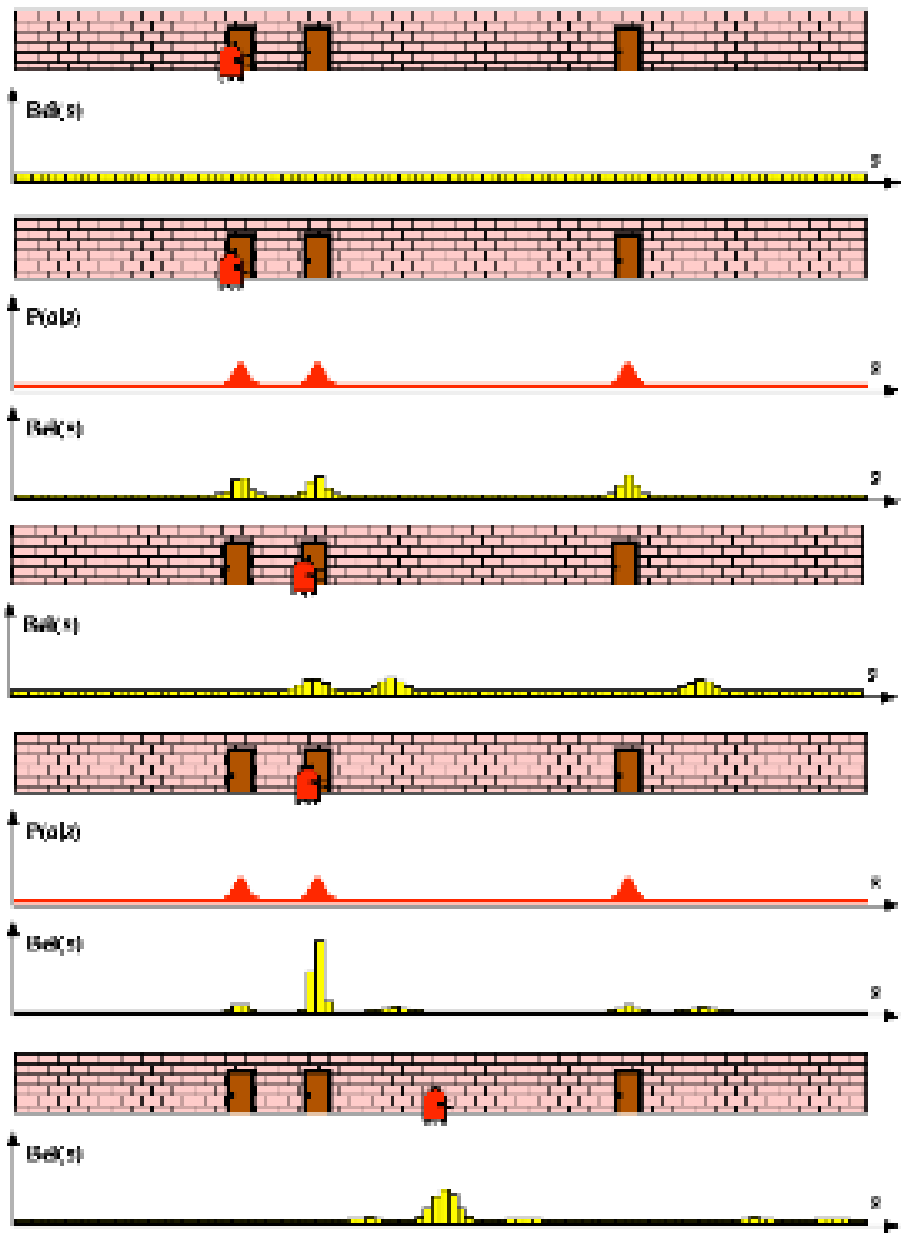
Histogram Filter

A Simple Example



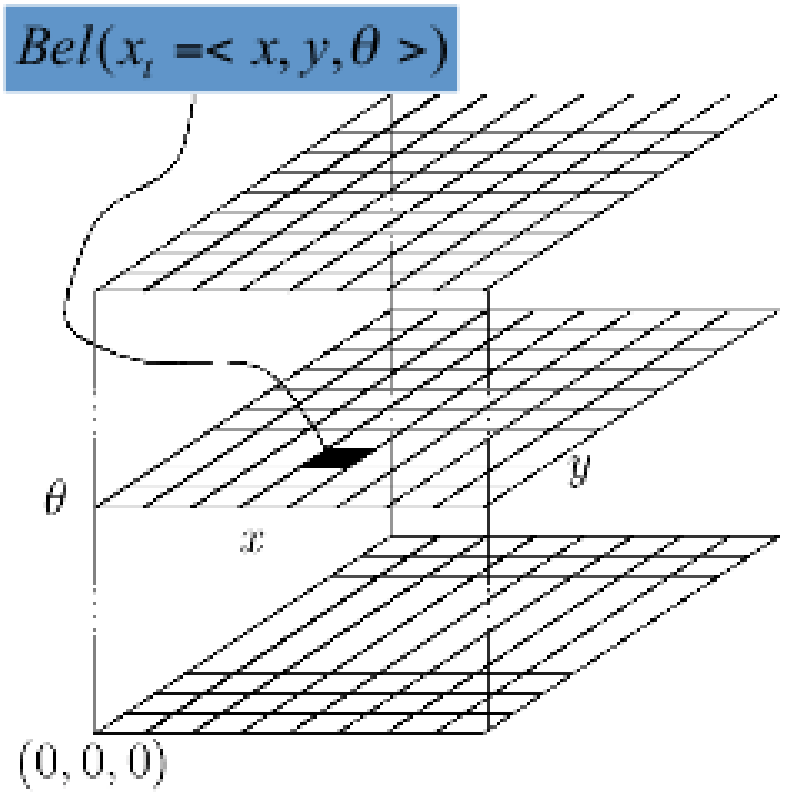
Histogram Filter

Piecewise Constant Representation

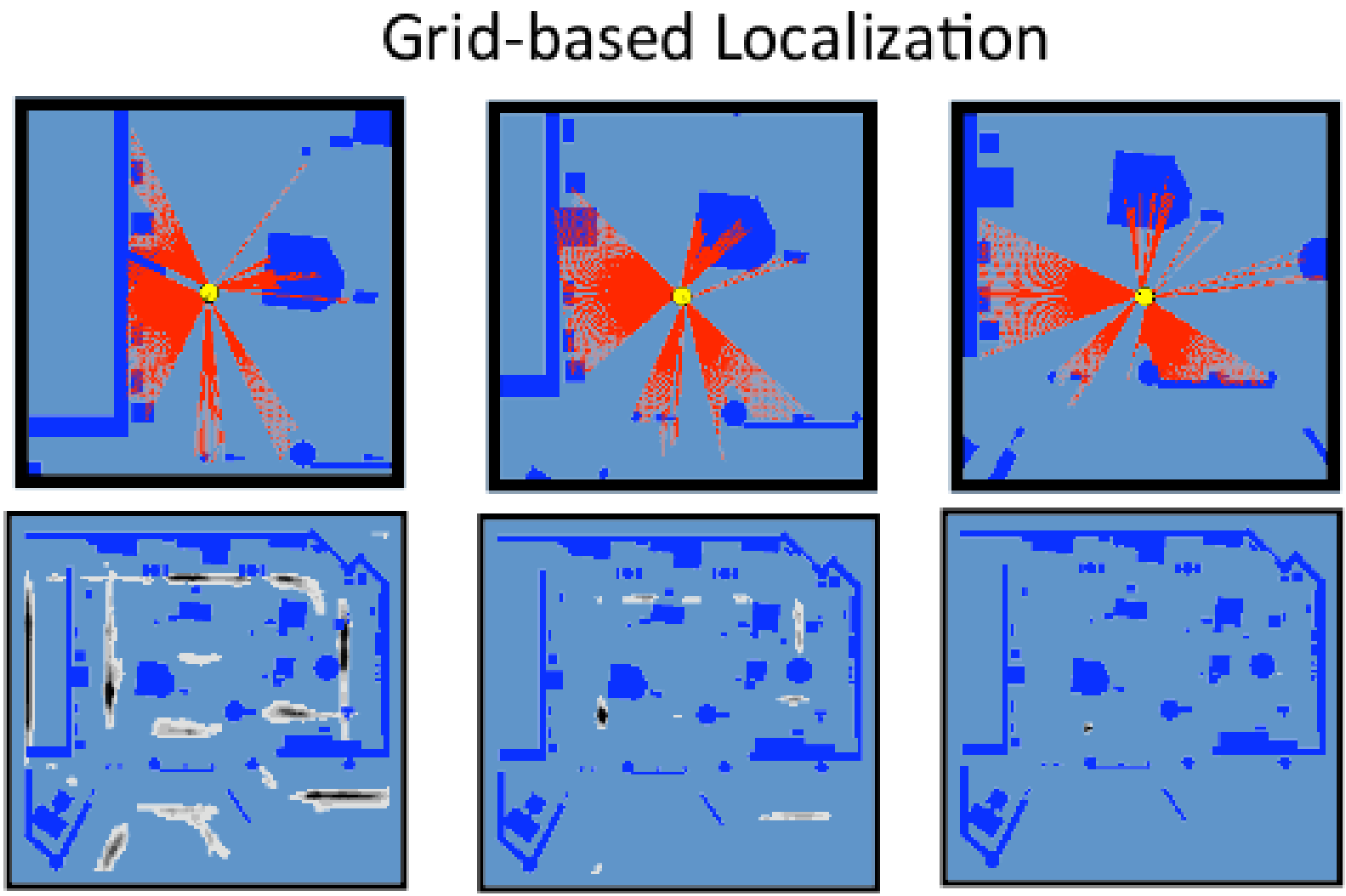


Histogram Filter as Grid based localization

Piecewise Constant Representation



Histogram Filter as Grid based localization



Histogram Filter

The idea of the Histogram filter is to partition the state space up into regions. Then use a single value to represent the value of the posterori over the whole region.
So if we need to know the value of the probability density function at point $\mathbf{x}_t$, which is within the k^{th} region we use:

$$p(\mathbf{x}_t) \approx \frac{p_{k,t}}{v_{k,t}}$$

where $v_{k,t}$ is the volume of the k^{th} region.
$p_{k,t}$ is the probability that the state is within region k.

Histogram Filter

We can approximate the measurement model by using its value at the mean state

$$\hat{\mathbf{x}}_{\mathbf{k},\mathbf{t}} = \frac{1}{v_{k,t}} \int_{v_{k,t}} \mathbf{x} d\mathbf{x}$$

$$p(\mathbf{z}_t|\mathbf{x}_{k,t}) = \frac{p(\mathbf{z}_t,\mathbf{x}_{k,t})}{P(\mathbf{x}_{k,t})}$$

$$p(\mathbf{z}_t|\mathbf{x}_{k,t}) \approx p(\mathbf{z}_t|\hat{\mathbf{x}}_{\mathbf{k},\mathbf{t}})$$

Histogram Filter

We can also approximate the prediction model by using its value at the mean state

$$p(\mathbf{x}_{k,t}|\mathbf{u}_t, \mathbf{x}_{i,t-1}) = \eta v_{k,t} p(\hat{\mathbf{x}}_{\mathbf{k,t}}|\mathbf{u}_t, \hat{\mathbf{x}}_{i,t-1})$$

Histogram Filter

These techniques work for 'small' state spaces.
The denser we make the bins, the better accuracy we get.
Uniformly covering the state space makes sense for compact spaces with spread distributions.

We can do better partitioning if we know the important regions. This leads to a recursive re-partioning of the space to put more bins where the probability is highest. So called density trees.

We can also just have a fine grid over all the space but ignore big parts of the grid that have low probability. We do so called selective updating.

Particle Filter aka Sequential Monte Carlo Filter

Particle filter and histogram filter are similar in that they represent the continous distribution with a discrete one.

Histogram filter is deterministic while particle Filter is random.

One imprecise way to think of particle filters is a simulation.

Another is that we have a collection of samples (particles) drawn from the posterior distribution.

Particle Filter

Input is a set of 'particles' $x_{t-1}^{(m)}$ drawn from the $p(x_{t-1}|z_1.., z_{t-1}, u_1..u_{t-1})$.
m runs from $1...M$

Phase I: propagate to new time using the model.

1. sample $p(x_t|x_{t-1}^{(m)}, u_t) \rightarrow \hat{x}_t^{(m)}$
2. compute $w_t^{(m)} = p(z_t|\hat{x}_t^{(m)})$ (importance factor)

Phase II: resample or importance sampling

1. draw from the set of $\hat{x}_t^{(m)}$ with probability $w_t^{(m)} \rightarrow x_t^{(m)}$

Sample? Whats that?

sample $p(x_t|x_{t-1}^{(m)}, u_t) \rightarrow \hat{x}_t^{(m)}$?

Let us say that the amount of salt was continuous and that we are instead using a particle filter. The inital distribution of particles is over the grams of salt in the stew. The inital M=3 particle set could look like $\{x_0^{(m)}\} = \{1.2, 0.5, 2.3\}$.
We assume the amount added in gram $\sim N(1.5, 1.0)$.
We have generated random numbers from a (uniform) random number generator as integers between 1 and 10,000: $\{5000, 8621, 3483\}$.

What is $p(x_1|x_0^{(m)}, u_t)$ and how shall we use the random numbers to sample it and get $\hat{x}_1^{(m)}$?

Sample?

$\{x_0^{(m)}\} = \{1.2, 0.5, 2.3\}$ salt adding$\sim N(1.5, 1.0)$

uniform random with max=10,000: $\{5000, 8621, 3483\}$.

Cumulative normal distribution : $\Phi(z) = P(Z \leq z)$

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.00	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.10	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.20	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.30	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.40	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.50	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.60	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.70	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.80	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.90	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.00	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.10	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.20	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.30	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.40	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.50	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.60	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.70	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.80	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706

Particle Filter

Input is a set of 'particles' $x_{t-1}^{(m)}$ drawn from the $p(x_{t-1}|z_1.., z_{t-1}, u_1..u_{t-1})$.
m runs from $1...M$

Phase I: propagate to new time using the model.

1. sample $p(x_t|x_{t-1}^{(m)}, u_t) \rightarrow \hat{x}_t^{(m)}$
2. compute $w_t^{(m)} = p(z_t|\hat{x}_t^{(m)})$ (importance factor)

So after Phase I step 1 we have $\hat{x}_t^{(m)}$ which we hope are drawn from the distribution $p(\mathbf{x}_t|\{\mathbf{u}_t\}, \{\mathbf{z}_{t-1}\})$
We then compute these weights $w_t^{(m)}$. Why? Then
Phase II: resample or importance sampling

1. draw from the set of $\hat{x}_t^{(m)}$ with probability $w_t^{(m)} \rightarrow x_t^{(m)}$

Particle Filter

Compare to Bayes Filter:

$$p(\mathbf{x}_t|\{\mathbf{z}_t\},\{\mathbf{u}_t\}) \propto p(\mathbf{z}_t|\mathbf{x}_t,\{\mathbf{u}_t\},\{\mathbf{z}_{t-1}\})p(\mathbf{x}_t|\{\mathbf{u}_t\},\{\mathbf{z}_{t-1}\})$$

$$\propto p(\mathbf{z}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{u}_t,\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\{\mathbf{z}_{t-1}\},\{\mathbf{u}_{t-1}\})(d\mathbf{x}_{t-1})^n$$

$w_t^{(m)} = p(z_t|\hat{x}_t^{(m)})$ (importance factor)

So this is needed to adjust for the difference between our sampled distribution:

$p(\mathbf{x}_t|\{\mathbf{u}_t\},\{\mathbf{z}_{t-1}\})$ (called the **proposal distribution**)

and the true posterior:

$p(\mathbf{x}_t|\{\mathbf{z}_t\},\{\mathbf{u}_t\})$ (called the **target distribution**).

Density Extraction

Having a set of (possibly weighted) particles drawn from the posteori distribution allows us to estimate the continuous distribution.

1. we can fit a Gaussian to the mean and variance of the particle set.
2. we can create bins and count how many particles are in each bin to form a histogram.
   - problems with larger numbers of dimensions.
   - can represent multimodal distributions and are easy to compute.
3. We can place a so called kernel around each particle. For example a Gaussian Kernel:
   $p(x) \propto \sum_{m=1}^{M} w^{(m)} G(x - x^{(m)}, \sigma^2)$