Particle Filters

Dr. Elli Angelopoulou

Pattern Recognition Lab (Computer Science 5)

University of Erlangen-Nuremberg

Concept

- Setup: Similar to Kalman Filtering, it explicitly predicts an estimate of the state of the dynamic system based on an uncertain system model. The prediction is then updated through the incorporation of information by noisy measurements.
- Goal: Minimize the difference between the updated estimate and the true state of the dynamic system.
- Means: Pick the updated estimate with the highest posterior probability, i.e. the highest Bayesian probability.
- Both Kalman Filter and Particle Filters express the state estimation problem in a Bayesian Framework.

Kalman Filter Setup

- A dynamic system is described at every time instance t_k by a state vector $\vec{x}_k$.
- At each time instance t_k there is also a measurement vector $\vec{z}_k$.
- There is a linear dynamic model $\vec{x}_k = \mathbf{\Phi}_{k-1}\vec{x}_{k-1} + \vec{w}_{k-1}$

  where $\vec{w}_{k-1}$ is a vector describing the random process noise and $\mathbf{\Phi}_{k-1}$ is the state transition matrix.
- The relationship between the true system state and the measurements is given by: $\vec{z}_k = \mathbf{H}_k\vec{x}_k + \vec{\mu}_k$

  where $\vec{\mu}_k$ is a vector describing the measurement noise and $\mathbf{H}_k$ is the measurement matrix.

Kalman Filter

- Prediction equations

$$\hat{x}_k^- = \Phi_{k-1}\hat{x}_{k-1}$$

$$\mathbf{P}_k^- = \Phi_{k-1}\mathbf{P}_{k-1}\Phi_{k-1}^T + \mathbf{Q}_k$$

  - Project state and covariance estimates forward in time

- Update equations

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + \mathbf{K}_k(\vec{z}_k - \mathbf{H}_k\hat{x}_k^-)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$$

  - Compute Kalman gain K
  - Include the measurement
  - Compute a posteriori estimate
  - Compute a posteriori covariance of the estimate

Bayesian Framework

- In a Bayesian framework, tracking is expressed as problem of estimating the probability of the state $\vec{x}_k$ given the previous states $(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_{k-1})$ and the measurements up to time t_k, $(\vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{k-1}, \vec{z}_k)$.
- We select the estimate with the highest probability:

$$\hat{x}_k = \max_{\vec{x}_k} p(\vec{x}_k | \vec{x}_1, \vec{x}_2, \ldots, \vec{x}_{k-1}, \vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{k-1}, \vec{z}_k)$$

- In a Markovian process, the future state depends only on the current state.
- Thus, we need to estimate $p(\vec{x}_k | \vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{k-1}, \vec{z}_k)$

Bayesian Framework - continued

- We estimate the posterior probability $p(\vec{x}_k|\vec{z}_1,\ldots,,\vec{z}_k)$ in a 2-step process:
- First estimate:

$$p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1}) = \int p(\vec{x}_k|\vec{x}_{k-1})\,p(\vec{x}_{k-1}|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1})\,dx_{k-1}$$

  This is the so-called *prediction step*.

- Then once the measurement $\vec{z}_k$ is obtained, compute

$$p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_k) = \frac{p(\vec{z}_k|\vec{x}_k)\,p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1})}{p(\vec{z}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1})}$$

  where $p(\vec{z}_k|\vec{z}_1,\ldots,\vec{z}_{k-1}) = \int p(\vec{z}_k|\vec{x}_k)\,p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_{k-1})\,dx_{k-1}$

  This the *update step*.

KF Assumptions in the Bayesian Framework

- The prediction pdf is Gaussian:

$$p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1}) \sim \mathcal{N}(\vec{x}_k|\hat{x}_k^-,\mathbf{P}_k^-)$$

- The update pdfs are Gaussian:

$$p(\vec{x}_{k-1}|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1}) \sim \mathcal{N}(\vec{x}_{k-1}|\hat{x}_{k-1},\mathbf{P}_{k-1})$$

$$p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_k) \sim \mathcal{N}(\vec{x}_k|\hat{x}_k,\mathbf{P}_k)$$

- Computing is equivalent to:

$$\hat{x}_k = \max_{\vec{x}_k} p(\vec{x}_k|\vec{x}_1,\vec{x}_2,\ldots,\vec{x}_{k-1},\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1},\vec{z}_k) \sim \mathcal{N}(\vec{x}_k,\mathbf{P}_k)$$

Kalman Filter Summary

- Kalman filter makes the following assumptions:
  1. Linear system model
  2. Gaussian noise
  3. Gaussian posterior distribution

- KF computes the optimal state estimate $\hat{x}_k$, as the max. probability density of $\vec{x}_k$ given the past estimates, the past measurements and the current measurement. The pdf is assumed to be Gaussian so its max. coincides with its mean.

- The Extended Kalman Filter tries to address some of these constraints but it is still based on:
  1. Local linearization of non-linear models
  2. Gaussian noise
  3. Gaussian posterior probability

Particle Filters

- Same dynamic framework (uncertain system model, noisy measurements that refine predicted estimates).
- No assumptions on system model (other than Markovian) or noise model.
- The estimation process is based on a Bayesian framework.
- Also known as:
  - Sequential Monte Carlo
  - Condensation Algorithm
  - Bootstrap Filtering
  - Survival of the Fittest
  - Interacting Particle Approximations
  - …

PF Dynamic System

- A dynamic system is described at every time instance t_k by a state vector $\vec{x}_k$.
- The underlying system model that describes how the system changes over time is:

$$\vec{x}_k = f_k(\vec{x}_{k-1}, \vec{v}_{k-1})$$

where f_k is a possibly non-linear transition function and $\vec{v}_{k-1}$ is a vector describing the random process noise.

- There are no restrictions or assumptions regarding the dynamic system, other than it is Markovian.

PF Measurements

- At every time instance t_k we also have observations of the dynamic system, $\vec{z}_k$.
- The measurement model is:

$$\vec{z}_k = h_k(\vec{x}_k, \vec{n}_k)$$

where h_k is a possibly non-linear measurement function and $\vec{n}_k$ is a vector describing the measurement noise.

- There are no restrictions or assumptions regarding the measurement model, other than it is Markovian.

PF Bayesian Framework

- Goal: Estimate $\vec{x}_k$ given the previous states $(\vec{x}_1,\ldots,\vec{x}_{k-1})$ and the measurements up to time t_k, $(\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1},\vec{z}_k)$.
- Bayesian perspective to the tracking problem: Recursively calculate some degree of belief in (probability of) the state $\vec{x}_k$ at time t_k, taking different values, given the data $(\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1},\vec{z}_k)$.
- This means, compute the pdf:

$$p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1},\vec{z}_k)$$

and pick the value of $\vec{x}_k$ with the highest probability.

Markovian Assumption

- Particle Filters make only one key assumption: The system and the measurement models are Markovian.
- $\vec{x}_k$ is Markovian: Its conditional probability density given the past states, depends only only the very last state through the transition density.

$$p(\vec{x}_k|\vec{x}_1,\vec{x}_2,\ldots,\vec{x}_{k-1}) = p(\vec{x}_k|\vec{x}_{k-1})$$

- $\vec{z}_k$ is Markovian: Its conditional probability density given the all states up to and including $\vec{x}_k$ and the past observations, depends only on $\vec{x}_k$ through the conditional likelihood

$$p(\vec{z}_k|\vec{x}_1,\vec{x}_2,\ldots,\vec{x}_k,\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1}) = p(\vec{z}_k|\vec{x}_k)$$

PF – 2-step Estimation Process

- Prediction Step:

$$p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1}) = \int p(\vec{x}_k|\vec{x}_{k-1})\,p(\vec{x}_{k-1}|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1})\,dx_{k-1}$$

- Once the measurement $\vec{z}_k$ is obtained, perform the Update Step:

$$p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_k) = \frac{p(\vec{z}_k|\vec{x}_k)\,p(\vec{x}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1})}{p(\vec{z}_k|\vec{z}_1,\vec{z}_2,\ldots,\vec{z}_{k-1})}$$

where $p(\vec{z}_k|\vec{z}_1,\ldots,\vec{z}_{k-1}) = \int p(\vec{z}_k|\vec{x}_k)\,p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_{k-1})\,dx_{k-1}$

- So if one knows the pdf $p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_k)$, then estimating $\vec{x}_k$ is straightforward. Just pick the max.

PDF Estimation

- However, $p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_k)$ can be a highly complex probability distribution, for which we may have no analytic description.
- What can we do?
- Approximate *p*() by generating N random samples from *p*():

$$\vec{x}^{(i)},\ 1 \le i \le N$$

  and then use these discrete samples when computing properties of *p*(), like the expected value or its variance.
- In PF these samples are called **particles**.

Example

- For example, instead of estimating the expected value of the measurement function analytically:

$$\bar{h} = \int h(\vec{x})\, p(\vec{x})\, d\vec{x}$$

use the discrete samples:

$$\bar{h} \approx \frac{1}{N} \sum_{i=1}^{N} h\left(\vec{x}^{(i)}\right)$$

Importance Sampling

- More often than not we can not sample directly from $p()$. So we sample from another distribution $q()$, which must have a larger support than $p()$.
- So take N random samples from $q()$ instead of $p()$:

$$\vec{x}^{(i)},\ 1 \le i \le N \quad \text{drawn from } q()$$

- A pdf like $q()$ that is used to obtain samples from $p()$ is called **importance distribution**.
- The technique of using an importance distribution like $q()$, to obtain samples for another distribution like $p()$, is called **importance sampling**.

Importance Sampling Correction

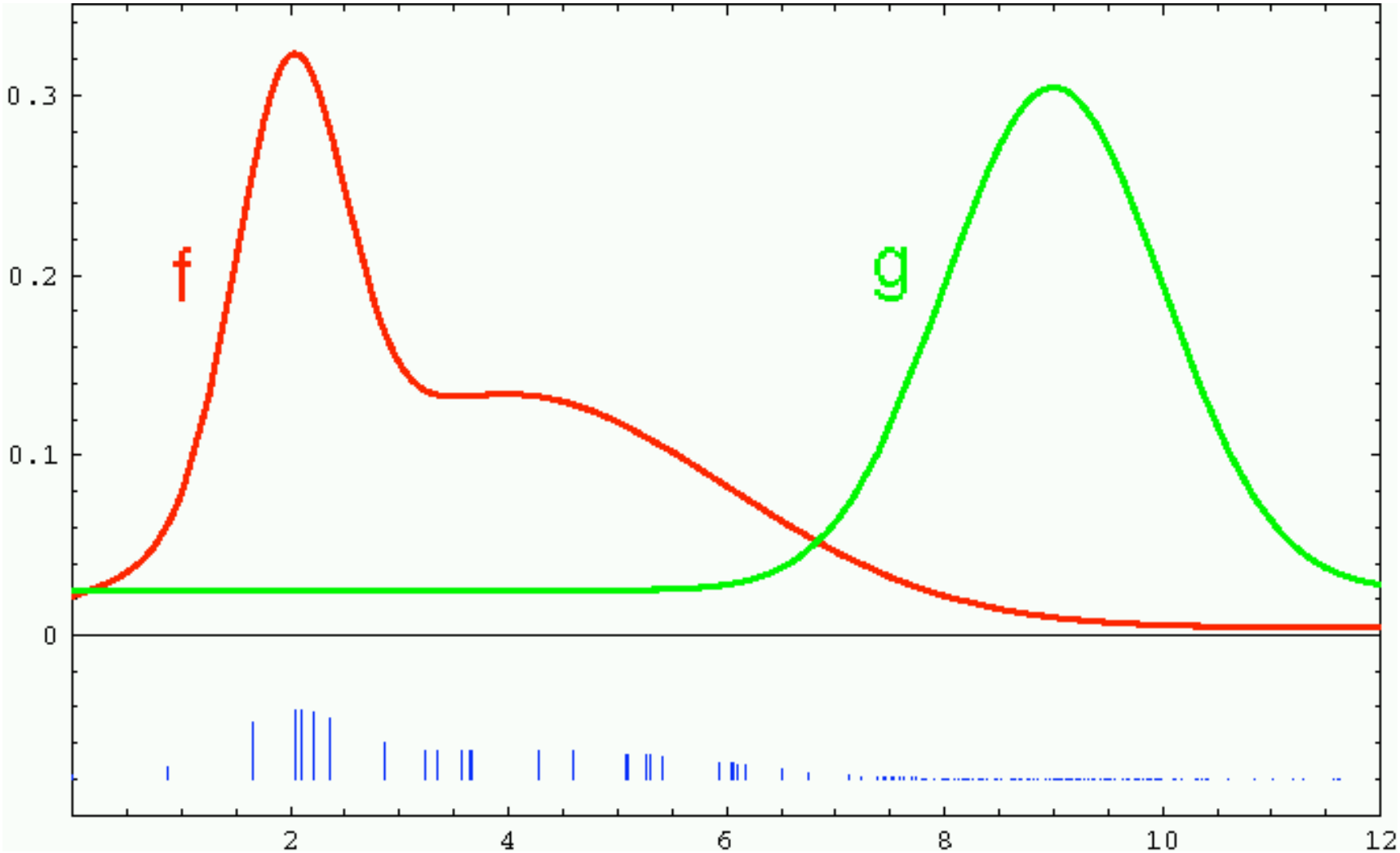
- Since *q*() is of course not *p*(), each sample must be corrected by being multiplied by an appropriate weight, so that one can obtain an unbiased estimation of the properties of *p*().
- So for each sample $\vec{x}^{(i)}$ there is a weight $\tilde{w}^{(i)}$ that handles the discrepancy between the two distributions:

$$\tilde{w}^{(i)} = \frac{p\left(\vec{x}^{(i)}\right)}{q\left(\vec{x}^{(i)}\right)}$$

- We normalize all the weights so that they sum up to 1.

$$w^{(i)} = \frac{\tilde{w}^{(i)}}{\sum_{i=1}^{N} \tilde{w}^{(i)}}$$

Importance Sampling Example



Weight samples: $w = f / g = p / q$

PDF Estimation via Importance Sampling

- We use importance sampling to get discrete samples of the pdf $p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_k)$.

$$\tilde{w}_k^{(i)} \propto \frac{p(\vec{x}_k^{(i)}|\vec{z}_1,\ldots,\vec{z}_k)}{q(\vec{x}_k^{(i)}|\vec{z}_1,\ldots,\vec{z}_k)}$$

- But within the Bayesian+Markovian framework we can compute variables and pdf's recursively

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} \frac{p(\vec{z}_k|\vec{x}_k^{(i)})\, p(\vec{x}_k^{(i)}|\vec{x}_{k-1}^{(i)})}{q(\vec{x}_k^{(i)}|\vec{x}_{k-1}^{(i)},\vec{z}_k)}$$

PF Estimation

- Our goal is to compute a good approximation of the posterior density: $p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_k)$
- We have established that:

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} \frac{p(\vec{z}_k|\vec{x}_k^{(i)})\, p(\vec{x}_k^{(i)}|\vec{x}_{k-1}^{(i)})}{q(\vec{x}_k^{(i)}|\vec{x}_{k-1}^{(i)},\vec{z}_k)}$$

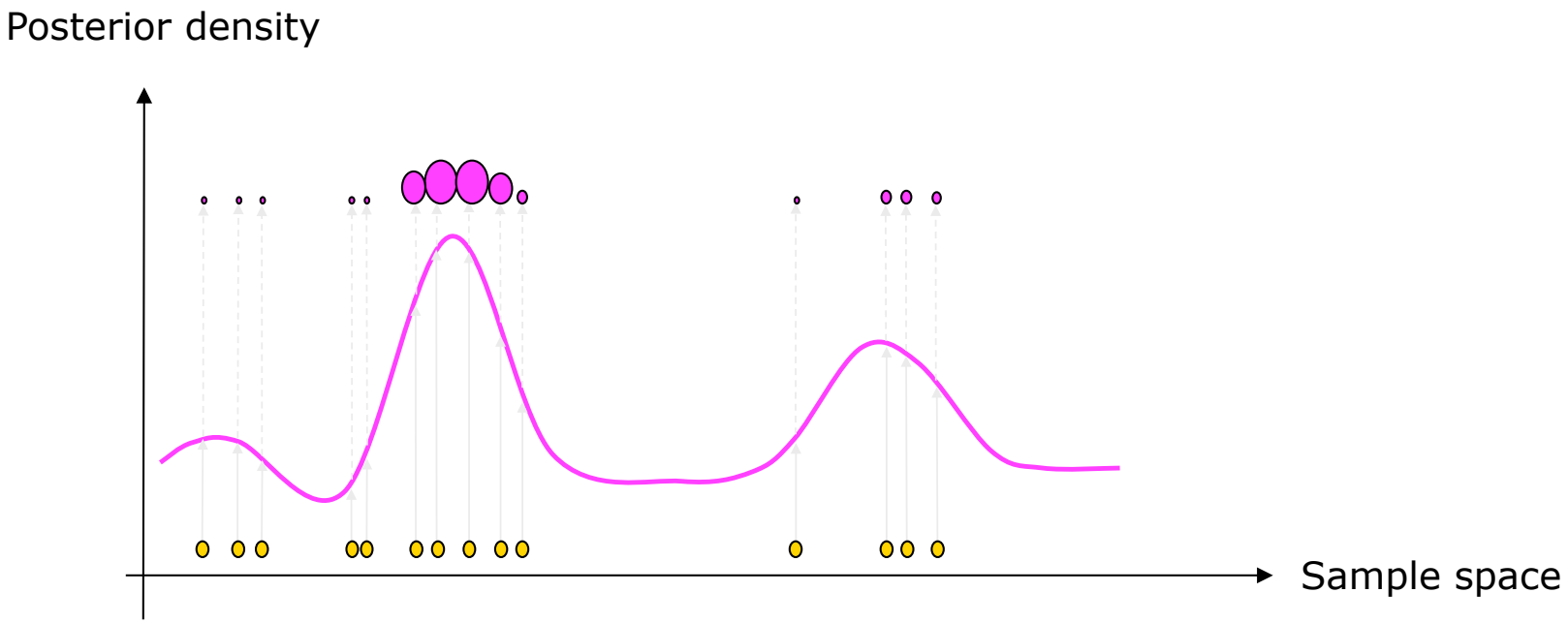
- Then $p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_k)$ can be approximated by:

$$p(\vec{x}_k|\vec{z}_1,\ldots,\vec{z}_k) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(\vec{x}_k - x_k^{(i)})$$

- As $N \rightarrow \infty$ the approximation approaches the true posterior probability function.

Weights and PDF

- The weight $w^{(i)}$ of each particle $\vec{x}^{(i)}$ can be thought of as an approximation of the probability that the specific particle value will occur.



Basic Particle Filter Algorithm

- For i = 1 to N
- Draw $\vec{x}_k^{(i)}$ from $q(\vec{x}_k^{(i)} \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.)$
- Compute $\tilde{w}^{(i)}$ using the Markovian update

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} \frac{p(\vec{z}_k \left| \vec{x}_k^{(i)} \right.) p(\vec{x}_k^{(i)} \left| \vec{x}_{k-1}^{(i)} \right.)}{q(\vec{x}_k^{(i)} \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.)}$$

- Compute the normalized weight, $w^{(i)}$, and assign it to the corresponding particle.
- End
- Out of all the particles, pick $\vec{x}_k^{(i)}$ with the maximum weight, or set $\vec{x}_k$ to the expected value of posterior density *p*().

The Importance of Importance Sampling

- The selection of the importance distribution can have a considerable impact on the accuracy of the estimation.
- If the importance distribution is very different from the pdf of the state variable, or is consistently uniformly distributed, it will lead to estimates with very high variance (i.e. pdfs that look like very wide Gaussians). Selecting the most probable value becomes unreliable.
- At the other extreme, consider the case where we have already converged to a peak of the pdf. After a few further iterations all but one particle will have negligible weight. Resulting again in very high variance.

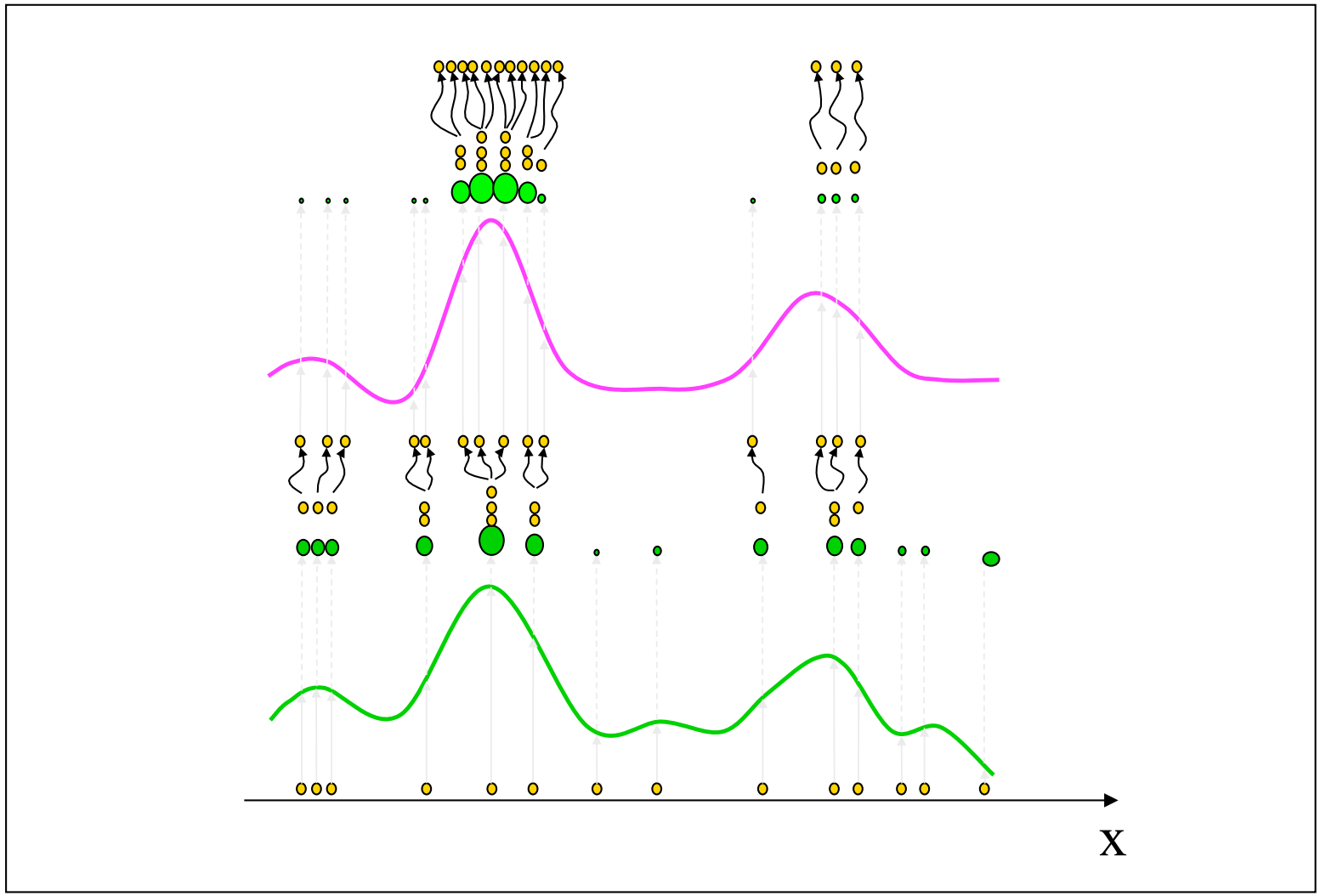
Degeneracy Problem

- When our sampling distribution closely matches the target distribution, the weights should be approximately constant. We want low variance in the weights across the different particles.
- Measure for degeneracy: *Effective sample size*

$$N_{eff} = \frac{N}{1 + Var(w_k^{(i)})}$$

- Small N_{eff} indicates severe degeneracy
- Solution 1: Use very large N – can be problematic.
- Solution 2: Resample, which means replicate particles in proportion to their weights. Particles with small weights are eliminated. Particles with large weights are replicated as many times as their weight.

Resampling Example

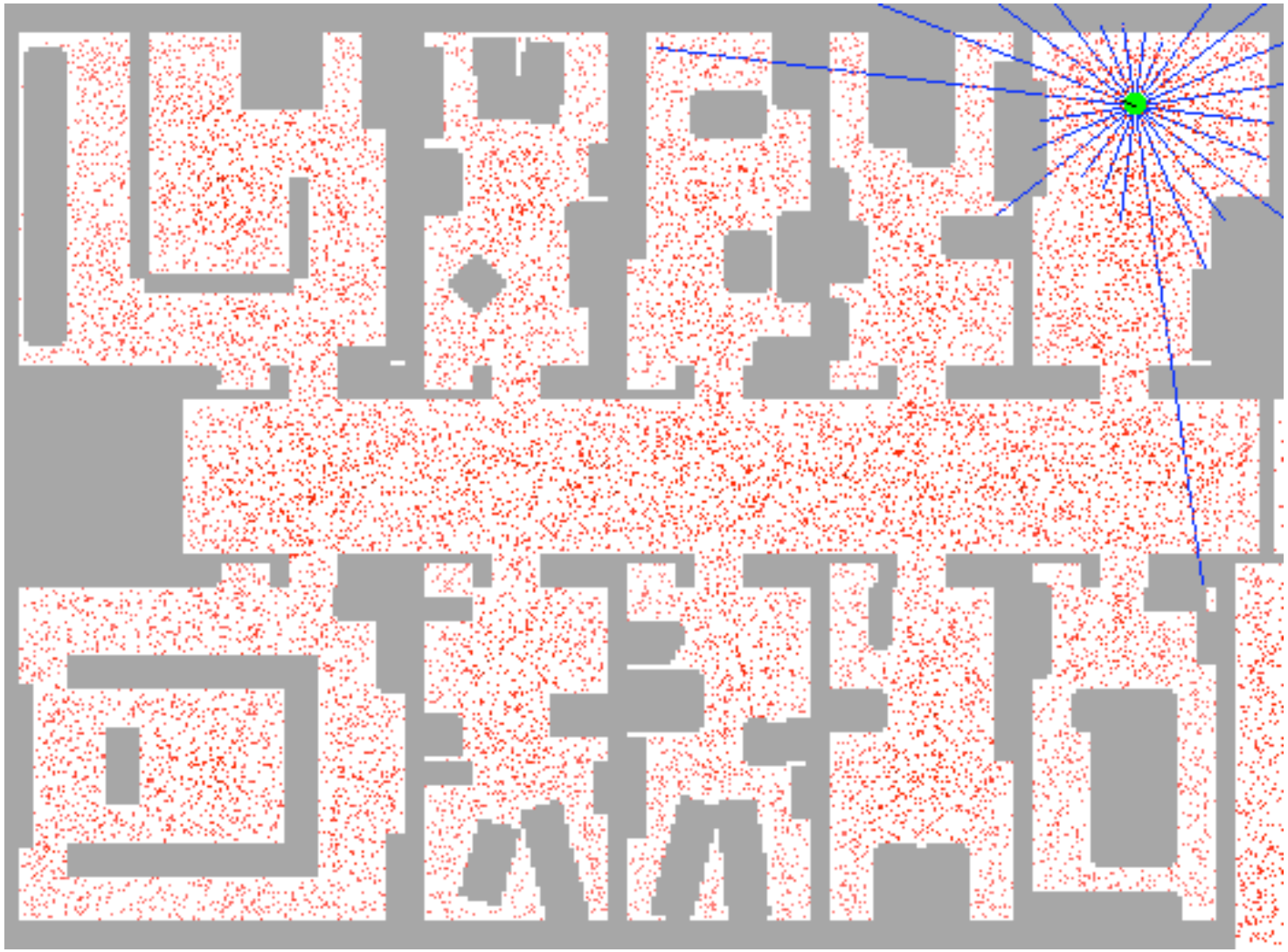


Particle Weights Adjust to PDF

First Condensation Demo

Well-Known Condensation Demo

PF in Robot Localization



Stanford Car - Urban Challenge ’07- Test A

Stanford Car - Urban Challenge ’07- Test B

Stanford Car - Urban Challenge ’07- Test C

Importance Distribution Function

- Which importance distribution function $q()$ should one use?
- Recall that one of the known problems of particle filters is degeneracy, which occurs when the effective sample size, N_{eff}, is small

$$N_{eff} = \frac{N}{1 + Var(w_k^{(i)})}$$

- One can choose an importance density $q()$ that minimizes the variance of the weights, $Var(w_k^{(i)})$.

Importance Distribution Function – cont.

- An importance density $q()$ that minimizes the variance of the weights $Var(w_k^{(i)})$ is:

$$q(\vec{x}_k^{(i)} \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.) = p(\vec{x}_k \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.) = \frac{p(\vec{z}_k \left| \vec{x}_k \right.) p(\vec{x}_k \left| \vec{x}_{k-1}^{(i)} \right.)}{p(\vec{z}_k \left| \vec{x}_{k-1}^{(i)} \right.)}$$

- Under this importance distribution function, the weight becomes:

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} p(\vec{z}_k \left| \vec{x}_{k-1}^{(i)} \right.) = \tilde{w}_{k-1}^{(i)} \int p(\vec{z}_k \left| \vec{x}'_k \right.) p(\vec{x}'_k \left| \vec{x}_{k-1}^{(i)} \right.) d\vec{x}'$$

Importance Weights:

- However, when we use the importance distribution:

$$q(\vec{x}_k^{(i)} \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.) = p(\vec{x}_k \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.)$$

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} \int p(\vec{z}_k \left| \vec{x}'_k \right.) p(\vec{x}'_k \left| \vec{x}_{k-1}^{(i)} \right.) d\vec{x}'$$

- Then, this means that:

a) Either we can somehow sample $p(\vec{x}_k \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.)$

b) Or we can evaluate the integral over the new state

$$\int p(\vec{z}_k \left| \vec{x}'_k \right.) p(\vec{x}'_k \left| \vec{x}_{k-1}^{(i)} \right.) d\vec{x}'$$

- Neither of which is straightforward in practice.

Importance Density in Practice

- In practice, we often use:

$$q(\vec{x}_k^{(i)} \left| \vec{x}_{k-1}^{(i)}, \vec{z}_k \right.) = p(\vec{x}_k^{(i)} \left| \vec{x}_{k-1}^{(i)} \right.)$$

- which leads to:

$$\tilde{w}_k^{(i)} \propto \tilde{w}_{k-1}^{(i)} p(\vec{z}_k \left| \vec{x}_k^{(i)} \right.)$$

- The particle filter then depends on what we use for:

$$p(\vec{z}_k \left| \vec{x}_k^{(i)} \right.)$$

  In other words how do we associate a measurement at time t_k with the state of the dynamic system at that time.

- This becomes a design decision for a developer/user particle filters.

Example

- Consider the case where we are tracking ball.
- We have chosen the velocity of the ball as a representation of the dynamic state of the ball.

$$\vec{x}_{k-1} = (v_{x_{k-1}}, v_{y_{k-1}})$$

- At each time instance we measure the position of the center of the ball.

$$\vec{z}_{k-1} = (c_{x_{k-1}}, c_{y_{k-1}})$$

- For each particle we can get an estimate of where the center of the ball should be at time t_k.

$$\hat{z}_k^{(i)} = (c_{x_{k-1}} + v_{x_{k-1}}^{(i)}, c_{y_{k-1}} + v_{y_{k-1}}^{(i)})$$

Example - continued

- We now have for each particle an estimate $\hat{\vec{z}}_k^{(i)}$
- At time t_k we measure the position of the center $\vec{z}_k$
- We can measure the error in the estimated ball position over all the particles:

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N}(\vec{z}_k - \hat{\vec{z}}_k^{(i)})^2$$

where N is the number of particles.

- We then use as conditional probability the function:

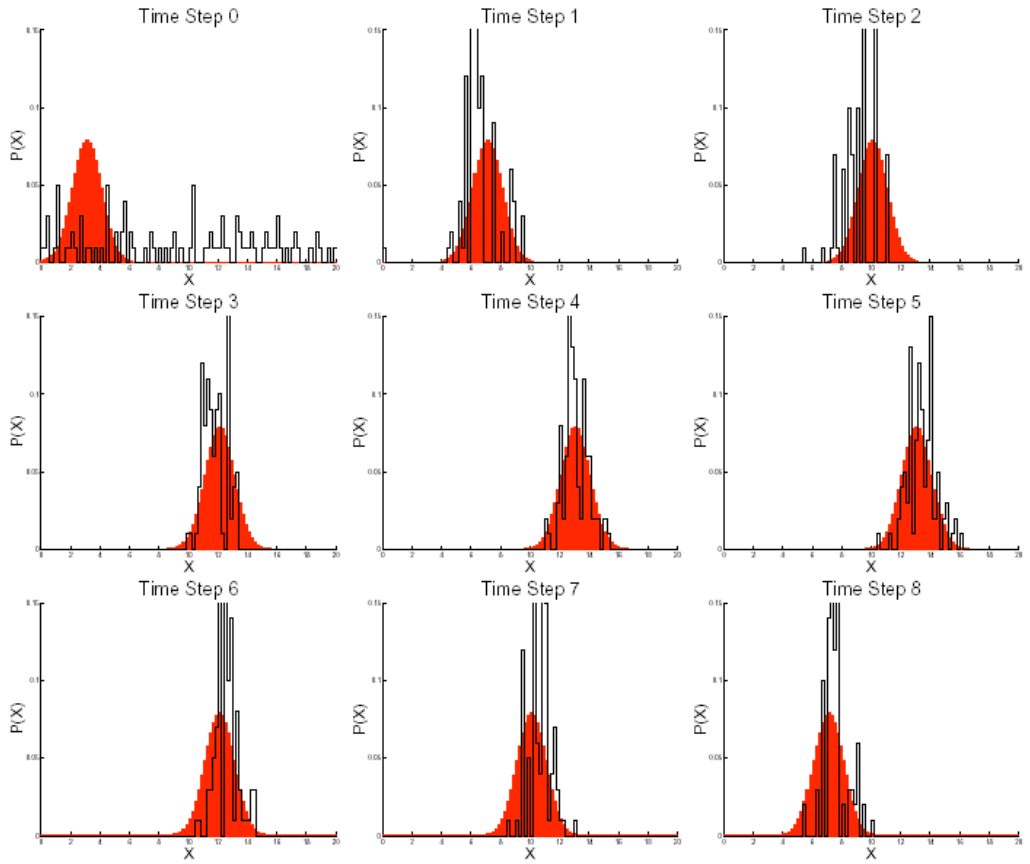
$$p(\vec{z}_k \left| \vec{x}_k^{(i)} \right.) \propto e^{-\left(\varepsilon/\sigma\right)}$$

where σ is empirically determined.

Further Resampling Improvements

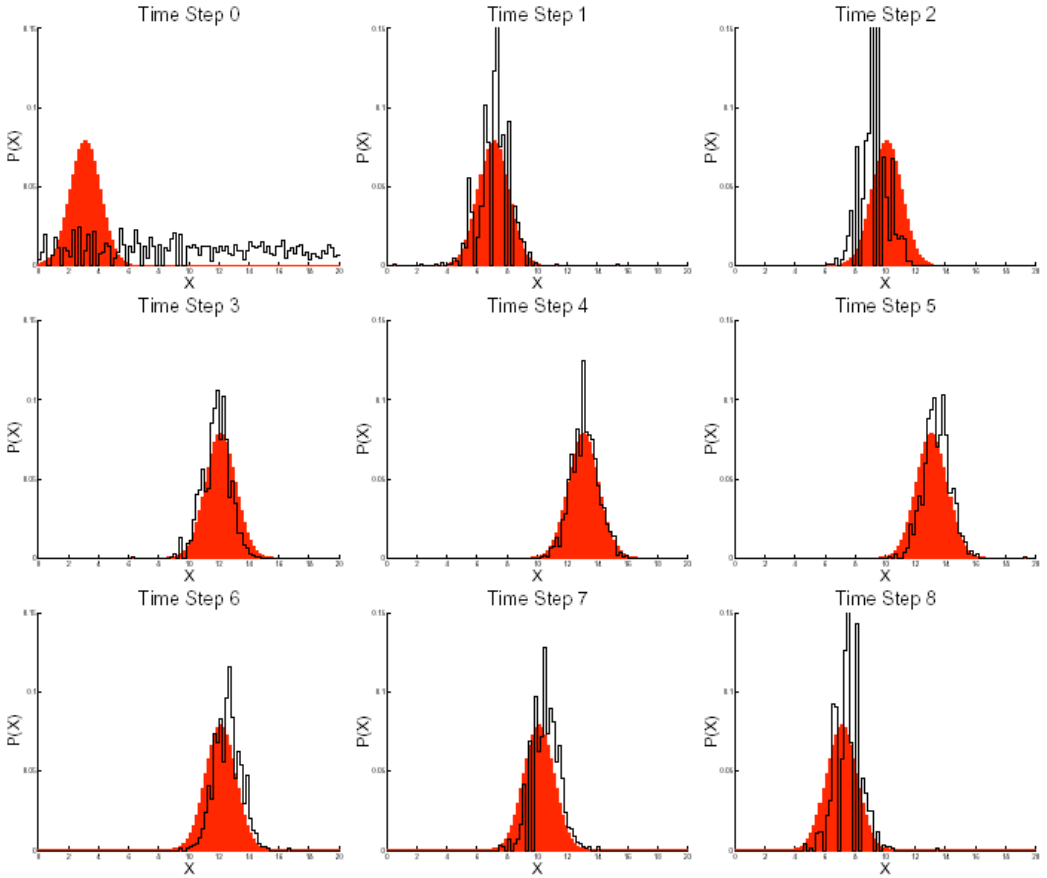
- Though the choice of $q()$ is critical, in practice we do not use very sophisticated importance distributions.
- As a consequence, despite resampling, degeneracy may still occur.
- To further alleviate this problem, sample smoothing methods can be used:
  - Roughening
    - Add an independent jitter to the resampled particles
  - Prior boosting
    - Increase the number of samples from the proposal distribution to M>N,
    - but in the resampling stage only draw N particles.

Particle Filter Example 1



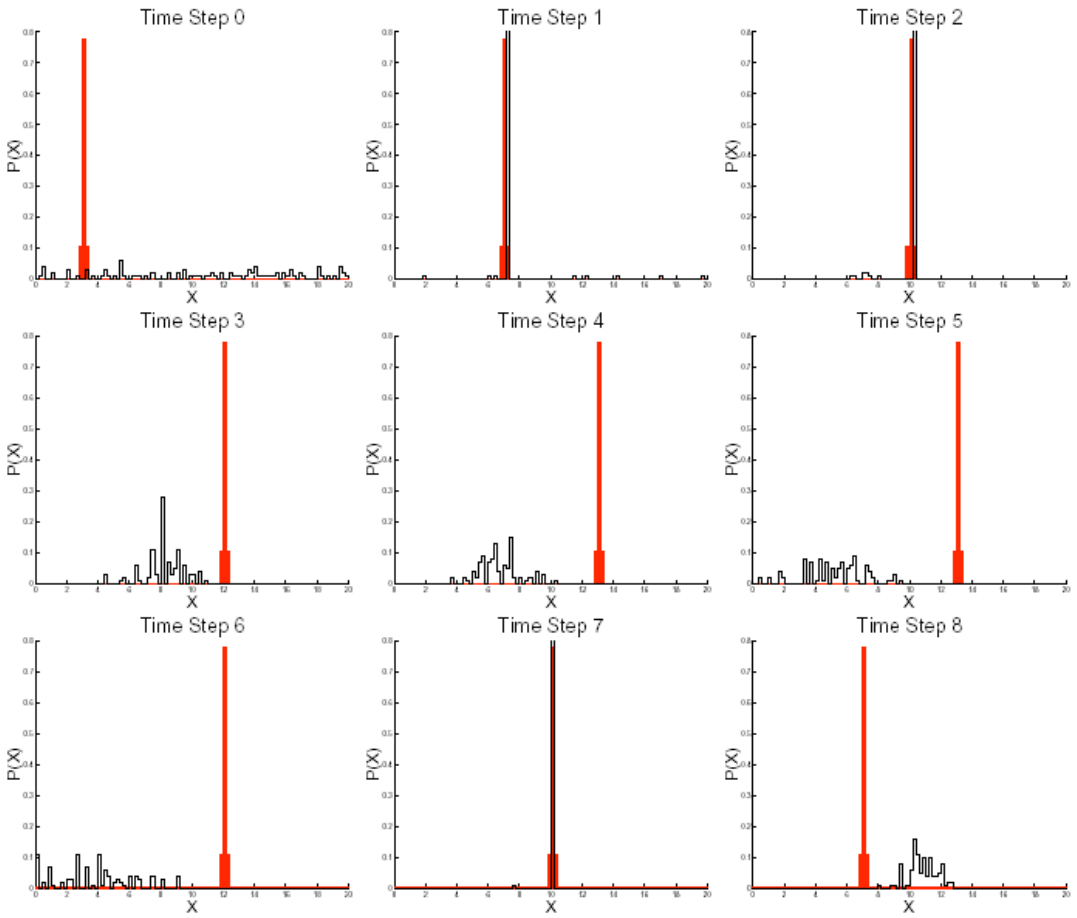
moving Gaussian + uniform, N=100 particles

Particle Filter Example 2



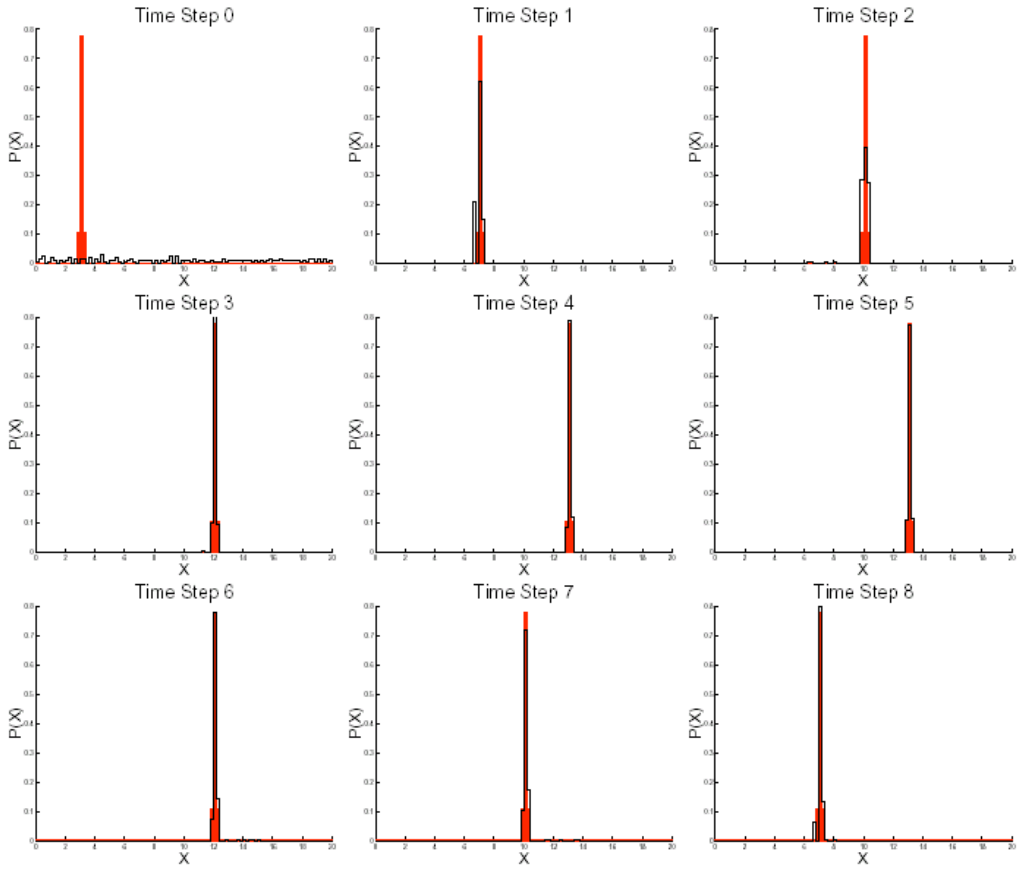
moving Gaussian + uniform, N=1000 particles

Particle Filter Example 3



moving (sharp) Gaussian + uniform, N=100 particles

Particle Filter Example 4

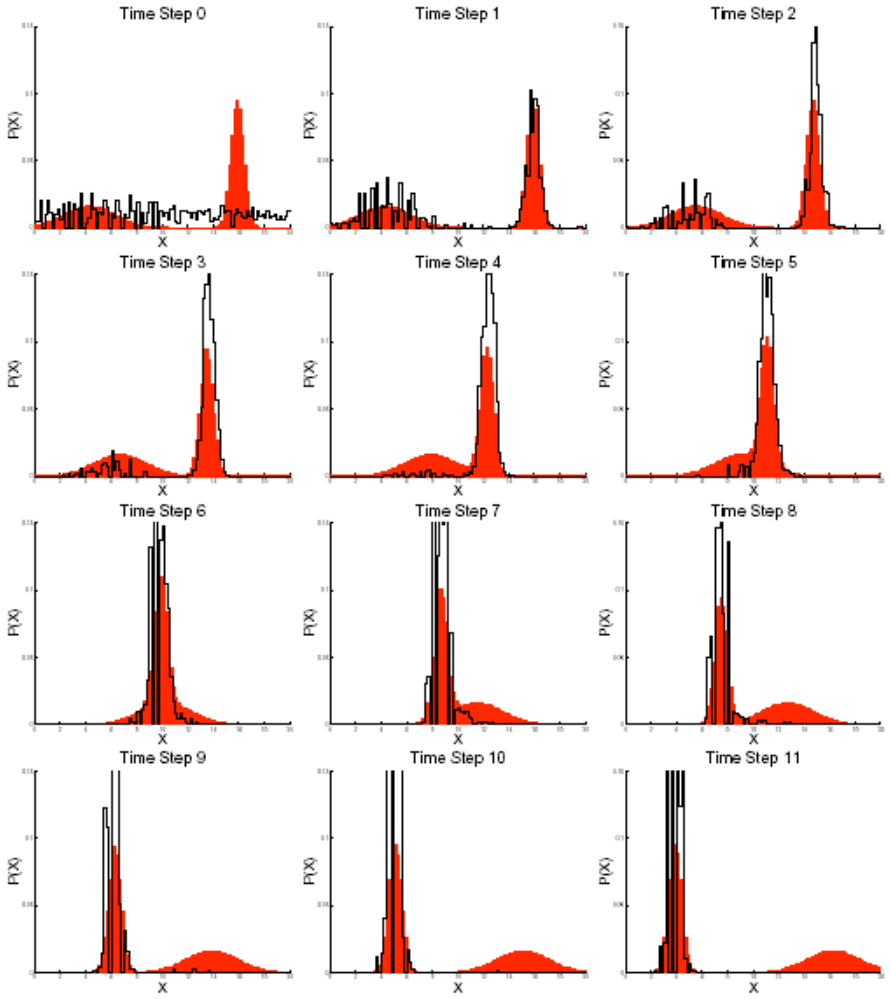


moving (sharp) Gaussian + uniform, N=1000 particles

Particle Filter Example 5

mixture of two Gaussians,

filter loses track of smaller and less pronounced peaks



Advantages of Particle Filters

- Ability to represent arbitrary densities, not just Gaussians. This is particularly important for multimodal distributions.
- Adaptive focusing on probable regions of state-space.
- No Gaussian noise assumptions.
- General state and measurement models (no linear assumption).
- The framework allows the inclusion of multiple models. For example, simultaneously tracking multiple pedestrians, cars and bicycles.

Disadvantages of Particle Filters

- High computational complexity (GPU and FPGA implementations).
- It is difficult to determine optimal number of particles.
- Number of particles increase with increasing model dimensionality.
- Potential problems: degeneracy and loss of diversity.
- The choice of importance density is crucial.

Particle Filters vs. Kalman Filter

- Same Bayesian framework and Markovian assumption.
- Recursive formulation composed of a prediction and update step.
- The Kalman Filter is fast, but is optimal only for linear systems with Gaussian distributions.
- Particle Filters are slow, but place no limitations on the system model or the distributions.
- Is there any way we can get the advantages of the Kalman Filter and the Particle Filters?

Marginalized Particle Filters

- There is a number of tracking applications where the system models are not entirely nonlinear and non-Gaussian.
- Some subset of the state vector *is linear and Gaussian*, conditional upon the other states.
- Idea: Use Kalman filtering for the linear Gaussian part of the dynamic system and particle filtering for the nonlinear part.
- This combination is known as:
  - Rao-Blackwellized particle filter
  - Marginalized particle filter
  - Mixture Kalman filter

MPF Setup:

- Partition the state vector into a linear component $\vec{x}_k^L$ and a nonlinear component $\vec{x}_k^N$:

$$\vec{x}_k = (\vec{x}_k^L, \vec{x}_k^N)$$

- The linear part of the model uses the Kalman filter formulation:

$$\vec{x}_k^L = A(\vec{x}_k^N)\vec{x}_{k-1}^L + \vec{w}_k^L$$

$$\vec{z}_k = B(\vec{x}_k^N)\vec{x}_k^L + \vec{\mu}_k^L$$

where the noise vectors $\vec{w}_k^L$ and $\vec{\mu}_k^L$ follow independent zero-mean Gaussian distributions.

- The state transition matrices *A*() and *B*() may depend on the non-linear component of the state vector.

MPF Setup -continued

- The non-linear part of the model is expressed using the more general particle filter formulation:

$$\vec{x}_k^N = f_k(\vec{x}_{k-1}^N, \vec{v}_{k-1}^N)$$

- The recursive formulation of the Kalman filter (5 eqs.) can be used for the linear part of the system.
- The linear part of the state vector is then marginalized in order to obtain the posterior distribution of the non-linear part of the state:

$$p(\vec{x}_k^N | \vec{z}_1, \ldots, \vec{z}_k) = \int p(\vec{x}_k^L, \vec{x}_k^N | \vec{z}_1, \ldots, \vec{z}_k) d\vec{x}_k^L$$

- The particle filter is then run on the non-linear part of the dynamic system.

Image Sources

1. The condensation animation and examples are by M. Isard, http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/ISARD1/condensation.html
2. The animation on robot localization based on laser range finders is courtesy of S. Thrun, http://robots.stanford.edu
3. The urban challenge videos of the Stanford team can be found in the web-page http://cs.stanford.edu/group/roadrunner/
4. The examples of how the particles are distributed for different types of pdf are courtesy of M. Pfeiffer http://www.igi.tugraz.at/pfeiffer/documents/particlefilters.ppt
5. The graphical example on Importance Sampling from the material on Probabilistic Robotics by S. Thurn, W. Burgard and D. Fox http://robots.stanford.edu/probabilistic-robotics/ppt/particle-filters.ppt
6. The resampling example is courtesy of M. Bolic http://www.site.uottawa.ca/research/spot/index_fichiers/Theory%20and%20Implementation%20of%20Particle%20Filters.ppt

Meaning of the Densities

Bearings-only tracking problem

- $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ posterior
  - What is the probability that the object is at the location $\mathbf{x}_k$ for all possible locations $\mathbf{x}_k$ if the history of measurements is $\mathbf{z}_{1:k}$**?**
- $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ prior
  - The motion model – where will the object be at time instant k given that it was previously at $\mathbf{x}_{k-1}$*?*
- $p(\mathbf{z}_k|\mathbf{x}_k)$ likelihood
  - The likelihood of making the observation $\mathbf{z}_k$ given that the object is at the location $\mathbf{x}_k$*.*