Particle Filters
an overview

Matthias Mühlich

Institut für Angewandte Physik
J.W.Goethe-Universität Frankfurt
muehlich@iap.uni-frankfurt.de
Particle Filters
a tutorial

Matthias Mühlich
Institut für Angewandte Physik
J.W.Goethe-Universität Frankfurt
muehlich@iap.uni-frankfurt.de
1 Introduction

An increasing number of researchers is using a family of techniques and algorithms called

- condensation algorithms
- bootstrap filtering
- particle filters
- interacting particle approximations
- sequential Monte Carlo methods
- SIS, SIR, ASIR, RPF, . . .

Time scale: last 10 years [e.g. Isard & Blake 1996; Kitagawa 1996; Gordon, Salmond & Smith 1993]

The question of this talk is: What is behind all that?
General Classification of Filter Strategies

Gaussian models:

- Kalman filter
- extended Kalman filter
- linear-update filter / linear regression filter / statistical linearization filter
  - unscented filter
  - central difference filter
  - divided difference filter
- assumed density filter / moment matching
Mixture of Gaussian models:

- assumed density filter / pseudo-Bayes
- Gaussian-sum filter

Nonparametric models:

- particle filter class
- histogram filter
Some Basic Remarks

- various applications: computer vision (i.e. tracking), control theory, econometrics (stock markets, monetary flow, interest rates), ...
- we deal with discrete time systems only
- no out-of-sequence measurements
- we are mainly interested in estimating the state at time \( k \) from measurements up to time \( k' = k \) (opposite: smoothing \( k' > k \) and prediction \( k' < k \); furthermore \( k' \) need not be fixed...) 
- no restrictions to linear processes or Gaussian noise!
Overview of this Talk

- The Dynamic System Model
- Bayesian Filter Approach
- Optimal and Suboptimal Solutions
- The Particle Filter
- Experiments and Summary

- states of a system and state transition equation
- measurement equation
Overview of this Talk

- The Dynamic System Model
- Bayesian Filter Approach
- Optimal and Suboptimal Solutions
- The Particle Filter
- Experiments and Summary

- estimation of the state
- probabilistic modelling
- Bayesian filter
Overview of this Talk

- The Dynamic System Model
- Bayesian Filter Approach
- Optimal and Suboptimal Solutions
- The Particle Filter
- Experiments and Summary

- filtered pdf can be written down easily, but it is not always tractable (→ ugly integrals . . .)
- conditions under which optimal solutions exist: Kalman filter and grid-based filter
- what can be done in other cases: suboptimal approaches
Overview of this Talk

- The Dynamic System Model
- Bayesian Filter Approach
- Optimal and Suboptimal Solutions
- The Particle Filter
- Experiments and Summary

- standard particle filter
- various improved versions
Overview of this Talk

- The Dynamic System Model
- Bayesian Filter Approach
- Optimal and Suboptimal Solutions
- The Particle Filter
- Experiments and Summary

– some experimental data and conclusion
2 Dynamic System

A dynamic system can be modelled with two equations:

State Transition or Evolution Equation

\[ \mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}_{k-1}) \]

- \( f(\cdot, \cdot, \cdot) \): evolution function (possible non-linear)
- \( \mathbf{x}_k, \mathbf{x}_{k-1} \in \mathbb{IR}^{n_x} \): current and previous state
- \( \mathbf{v}_{k-1} \in \mathbb{IR}^{n_y} \): state noise (usually not Gaussian)
- \( \mathbf{u}_{k-1} \in \mathbb{IR}^{n_u} \): known input

Note: state only depends on previous state, i.e. first order Markov process
Measurement Equation

\[ z_k = h_k(x_k, u_k, n_k) \]

- \( h(\cdot, \cdot, \cdot) \): measurement function (possible non-linear)
- \( z_k \in \mathbb{IR}^{n_z} \): measurement
- \( x_k \in \mathbb{IR}^{n_x} \): state
- \( n_k \in \mathbb{IR}^{n_n} \): measurement noise (usually not Gaussian)
- \( u_k \in \mathbb{IR}^{n_u} \): known input

(dimensionality of state, measurement, input, state noise, and measurement noise can all be different!)
measurements
(observed)

states
(cannot be observed and have to be estimated)
measurements (observed)

states
(cannot be observed and have to be \textit{estimated})

labels:
- $z_{k-1}$: measurement
- $x_{k-1}$: state
- time

In the context of dynamic systems, measurements are observed quantities, while states are internal quantities that cannot be directly observed and thus must be estimated. The diagram illustrates the process of updating states with measurements over time.
measurements
(observed)

states
(cannot be observed and have to be estimated)
measurements
(observed)

states
(cannot be observed and have to be estimated)

\begin{align*}
  z_{k-1} & \quad \rightarrow \quad z_k \\
  x_{k-1} & \quad \rightarrow \quad x_k \\
  x_k & \quad \rightarrow \quad x_{k+1}
\end{align*}
Measurements (observed)

States (cannot be observed and have to be estimated)
Assumptions:

The observations are conditionally independent given the state: \( p(z_k | x_k) \).

Hidden Markov Model (HMM):
\( p(x_0) \) given and \( p(x_k | x_{k-1}) \) defines state transition probability for \( k \geq 1 \).
3 Bayesian Filters

Estimating the Posterior

Bayesian approach: We attempt to construct the posterior pdf of the state given all measurements.

⇒ can be termed a complete solution to the estimation problem because all available information is used; from the pdf, an optimal estimate can theoretically be found for any criterion.

in detail: We seek estimates of $x_k$ based on all available measurements up to time $k$ (abbreviated as $z_{1:k}$) by constructing the posterior $p(x_k|z_{1:k})$.

Assumption: initial state pdf (prior) $p(x_0)$ is given
The Use of Knowing the Posterior

Let $f_k : \mathbb{R}^{(k+1) \times n_x} \rightarrow \mathbb{R}$ be any arbitrary (integrable) function that can depend

- on all components of the state $x$
- on the whole trajectory in state-space

Examples: This function can be an estimator for the current state or for future observations.

Then we can compute its expectation using

$$
E[f_k(x_{0:k})] = \int f(x_{0:k}) p(x_{0:k} | z_{1:k}) dx_{0:k}
$$

MMSE estimate of state: $\hat{x} = E[x_k]$. Other estimates that can be computed: median, modes, confidence intervals, kurtosis, ...
Recursive Filters

recursive filters (i.e. sequential update of previous estimate) ↔ batch processing (computation with all data in one step)

not only faster: allow on-line processing of data (lower storage costs, rapid adaption to changing signals characteristics)

essentially consist of two steps:

**prediction step:** \( p(x_{k-1}|z_{1:k-1}) \rightarrow p(x_k|z_{1:k-1}) \)
(usually deforms / translates / spreads state pdf due to noise)

**update step:** \( p(x_k|z_{1:k-1}, z_k) \rightarrow p(x_k|z_{1:k}) \)
(combines likelihood of current measurement with predicted state; usually concentrates state pdf)
General Prediction-Update Framework

Assume that pdf $p(x_{k-1}|z_{1:k-1})$ is available at time $k-1$.

Prediction step: (using Chapman-Kolmogoroff equation)

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \quad (1)$$

This is the prior of the state $x_k$ at time $k$ without knowledge of the measurement $z_k$, i.e. the probability given only previous measurements.

Update step: (compute posterior pdf from predicted prior pdf and new measurement)

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \quad (2)$$
Let us prove formula (2) (just in order to train calculations with joint and conditional probabilities . . . )

\[
p(x_k|z_{1:k}) = \frac{p(z_{1:k}|x_k)p(x_k)}{p(z_{1:k})}
\]

(Bayes rule)
Let us prove formula (2) (just in order to train calculations with joint and conditional probabilities...)

\[ p(x_k|z_{1:k}) = \frac{p(z_{1:k}|x_k)p(x_k)}{p(z_{1:k})} = \frac{p(z_k, z_{1:k-1}|x_k)p(x_k)}{p(z_k, z_{1:k-1})} \]

(separate \( p(z_{1:k}) \) into \( p(z_k, z_{1:k-1}) \))
Let us prove formula (2) (just in order to train calculations with joint and conditional probabilities...)

\[
p(x_k | z_{1:k}) = \frac{p(z_{1:k} | x_k)p(x_k)}{p(z_{1:k})} = \frac{p(z_k, z_{1:k-1} | x_k)p(x_k)}{p(z_k, z_{1:k-1})} = \frac{p(z_k | z_{1:k-1}, x_k)p(z_{1:k-1} | x_k)p(x_k)}{p(z_k | z_{1:k-1})p(z_{1:k-1})}
\]

(factorize joint probability: \( p(a, b | c) = p(a | b, c) \cdot p(b | c) \) and \( p(a, b) = p(a | b) \cdot p(b) \))
Let us prove formula (2) (just in order to train calculations with joint and conditional probabilities...)

\[
p(x_k | z_{1:k}) = \frac{p(z_{1:k} | x_k) p(x_k)}{p(z_{1:k})} = \frac{p(z_k, z_{1:k-1} | x_k) p(x_k)}{p(z_k, z_{1:k-1})} = \frac{p(z_k | z_{1:k-1}, x_k) p(z_{1:k-1} | x_k) p(x_k)}{p(z_k | z_{1:k-1}) p(z_{1:k-1})} = \frac{p(z_k | z_{1:k-1}, x_k) p(x_k | z_{1:k-1}) p(z_{1:k-1}) p(x_k)}{p(z_k | z_{1:k-1}) p(z_{1:k-1}) p(x_k)}
\]

(Bayes rule)
Let us prove formula (2) (just in order to train calculations with joint and conditional probabilities...)

\[
p(x_k | z_{1:k}) = \frac{p(z_{1:k} | x_k)p(x_k)}{p(z_{1:k})} = \frac{p(z_{k}, z_{1:k-1} | x_k)p(x_k)}{p(z_{k}, z_{1:k-1})} = \frac{p(z_k | z_{1:k-1}, x_k)p(z_{1:k-1} | x_k)p(x_k)}{p(z_k | z_{1:k-1})p(z_{1:(k-1)})} = \frac{p(z_k | z_{1:k-1}, x_k)p(x_k | z_{1:k-1})p(z_{1:k-1})p(x_k)}{p(z_k | z_{1:k-1})p(z_{1:k-1})p(x_k)} = \frac{p(z_k | x_k)p(x_k | z_{1:k-1})}{p(z_k | z_{1:k-1})}
\]

(indindependence of observations; cancelling out terms)
The Structure of the Update Equation

\[ p(x_k|z_{1:k}) = \frac{p(z_k|x_k) \cdot p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \]

\[ \text{posterior} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}} \]

prior: given by prediction equation

likelihood: given by observation model

evidence: the normalizing constant in the denominator

\[ p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k \]
This theoretically allows an optimal Bayesian solution (in the sense of computing the posterior pdf).

Problem: only a conceptual solution; integrals are not tractable.

But: in some restricted cases, an optimal solution is possible. Two optimal solutions (under restrictive assumptions):

- (standard) Kalman filter
- grid-based filter
4 Kalman Filter

Introduction

Assumptions:

- posterior at time $k-1$, i.e. $p(x_{k-1}|z_{k-1})$, is Gaussian
- dynamic system characterized by

$$x_k = F_k x_{k-1} + G_k v_{k-1}$$

$$z_k = H_k x_k + J_k n_k$$

- both noise vectors Gaussian (covariance matrices are $Q_{k-1}$ and $R_k$)

Then new posterior $p(x_k|z_k)$ is Gaussian, too, and can be computed using simple linear equations.
optimal solution, but *highly restrictive* assumptions must hold

**Prediction Equation**

At time $k-1$: $p(x_{k-1}|z_{1:k-1}) = \mathcal{N}(m_{k-1|k-1}, P_{k-1|k-1})$

Inserting into (1) yields

$$p(x_k|z_{1:k-1}) = \mathcal{N}(m_{k|k-1}, P_{k|k-1})$$

with

$$m_{k|k-1} = F_k m_{k-1|k-1}$$

and

$$P_{k|k-1} = G_k Q_{k-1} G_k^T + F_k P_{k-1|k-1} F_k^T$$
Update Equation

Inserting into (2) yields

\[ p(x_k|z_{1:k}) = \mathcal{N}(m_{k|k}, P_{k|k}) \]

with

\[ m_{k|k} = m_{k|k-1} + K_k(z_k - H_k m_{k|k-1}) \]

estimated \( \hat{z}_k \)

and

\[ P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1} \]

Kalman Gain:

\[ K_k = P_{k|k-1} H_k^T \left( H_k P_{k|k-1} H_k^T + J_k R_k J_k^T \right)^{-1} \]

\[ \text{Cov} [\hat{z}_k] \]
5 Grid-Based Filter

Introduction

Assumptions:

- state space is discrete
- number of different states ($N_s$) is limited
  (Note: implicitly includes discreteness)

Suppose at time $k - 1$ we have states $x^i$ with $i = 1, \ldots, N_s$. Conditional probability of these states:

$$Pr(x_{k-1} = x^i | z_{1:k-1}) = w^i_{k-1|k-1}$$

Then the (old) posterior at time $k - 1$ is given by:

$$p(x_{k-1} | z_{1:k-1}) = \sum_{i=1}^{N_s} w^i_{k-1|k-1} \delta(x_{k-1} - x^i)$$
Results (Summary)

Both the (new) prior and the (new) posterior have the same structure: a sum of weighted Dirac peaks:

\[
p(x_k|z_{1:k-1}) = \sum_{i=1}^{N_S} w_{k|k-1}^i \delta(x_{k-1} - x^i)
\]

\[
p(x_k|z_{1:k}) = \sum_{i=1}^{N_S} w_{k|k}^i \delta(x_{k-1} - x^i)
\]

Note: extension to different sets of states for each time step

\[\{x^i\} : i = 1, \ldots, N_S \rightarrow \{x^i_k\} : i = 1, \ldots, N_s,k\]

with time-varying index \(k\) is easily possible; the ‘allowed’ states need not be constant.
Prediction Equation (in Detail)

Inserting into (1) yields

\[ p(x_k | z_{1:k-1}) = \sum_{i=1}^{N_s} \sum_{j=1}^{N_s} w_{k-1|k-1}^i p(x^i | x^j) \delta(x_{k-1} - x^i) \]

\[ = \sum_{i=1}^{N_s} w_{k|k-1}^i \delta(x_{k-1} - x^i) \]

where \( w_{k|k-1}^i = \sum_{j=1}^{N_s} w_{k-1|k-1}^j p(x^i | x^j) \)

(new) prior weights = old posterior weights, reweighted using state transition probabilities
Update Equation (in Detail)

Inserting into (2) yields

\[ p(x_k|z_{1:k}) = \sum_{i}^{N_s} w_{k|k}^i \delta(x_{k-1} - x^i) \]

where \( w_{k|k}^i = \frac{w_{k|k-1}^i p(z_k|x^i)}{\sum_{j}^{N_s} w_{k|k-1}^j p(z_k|x^j)} \).

Note: denominator only needed for normalization

posterior weights = prior weights, reweighted using likelihoods
6 Particle Filter

Suboptimal Approximations

If we want to preserve Kalman filter principle...

- Extended Kalman Filter (EKF)
- Unscented Kalman Filter (UKF)

...we get better results,

BUT: we cannot get rid off Gaussian approximations
EKF / UKF:

All these approaches fail if we have

- bimodal / multimodal pdfs
- heavily skewed pdfs

We need a more general scheme to tackle these problems.
Particle Filter – General Concept

Many different names (do you remember the introduction?) but the general concept is rather simple:

PARTICLE FILTER:
If we cannot solve the integrals required for a Bayesian recursive filter analytically ... we represent the posterior probabilities by a set of randomly chosen weighted samples.

Note: “randomly chosen” ≡ “Monte Carlo” (we are playing roulette / throwing the dice)
Increasing number of samples ⇒ (almost sure) convergence to true pdf
Sequential Importance Sampling (SIS)

SIS is the basic framework for most particle filter algorithms. Let

\[ \{x^i_{0:k}\} : \text{set of support points (samples, particles)} \]
\[ i = 1, \ldots, N_s \]
\[ \text{(whole trajectory for each particle!)} \]
\[ w^i_k : \text{associated weights, normalized to } \sum_i w^i_k = 1 \]

Then:

\[ p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(x_{0:k} - x^i_{0:k}) \]

(discrete weighted approximation to the true posterior)
SIS (continued)

Usually we cannot draw samples \( x_k^i \) from \( p(\cdot) \) directly. Assume we sample directly from a (different) importance function \( q(\cdot) \). Our approximation is still correct (up to normalization) if

\[
    w_k^i \propto \frac{p(x_{0:k}^i|z_{1:k})}{q(x_{0:k}^i|z_{1:k})}
\]

The trick: we can choose \( q(\cdot) \) freely!

If the importance function is chosen to factorize such that

\[
    q(x_{0:k}|z_{1:k}) = q(x_k|x_{0:k-1}, z_{1:k}) q(x_{0:k-1}|z_{1:k-1})
\]

then one can augment old particles \( x_{0:k-1}^i \) by \( x_k \sim q(x_k|x_{0:k-1}, z_{1:k}) \) to get new particles \( x_{0:k}^i \).
SIS (continued)

Weight update (after some lengthy computations...):

\[
    w^i_k = w^i_{k-1} \frac{p(z^i_k | x^i_k) \ p(x^i_k | x^i_{k-1})}{q(x^i_k | x^{0:k-1}_i, z^{1:k})}
\]  

(3)

Furthermore, if \( q(x_k | x^{0:k-1}_k, z^{1:k}) = q(x_k | x^{k-1}_k, z^{1:k}) \) (only dependent on last state and observations):

\[
    p(x | z^{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(x_k - x^i_k)
\]

(and we need not preserve trajectories \( x^{i:0:k-1}_i \) and history of observations \( z^{1:k-1} \))
SIS Algorithm – Pseudo Code

\[
\{\{x_i^k, w_i^k\}_{i=1}^{N_s}\} = \text{SIS}(\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, z_k)
\]

\text{FOR } i = 1 : N_s
\text{draw } x_k^i \sim q(x_k|x_{k-1}^i, z_k)
\text{update weights according to (3)}
\text{END FOR}

\text{normalize weights to } \sum_{i=1}^{N_s} w_k^i = 1
PROBLEM: Degeneracy Problem

Problem with SIS approach: after a few iterations, most particles have negligible weight (the weight is concentrated on a few particles only)

Counter measures:

- brute force: many, many samples $N_s$
- good choice of importance density
- resampling

Note: amount of degeneracy can be estimated based on variance of weights [Liu 1996].
Optimal Importance Density:

It can be shown that the optimal importance density is given by

\[ q(x_k| x_{k-1}, z_k)_{opt} = p(x_k| x_{k-1}, z_k) \]

Then

\[ w_k^i = w_{k-1}^i \int p(z_k| x_k') p(x_k'| x_{k-1}^i) dx_k' \]

Two major drawbacks: usually neither sampling from \( q_{opt} \) nor solving the integral in \( w_k^i \) is possible. . . (but in some special cases, it works)

Other alternative which is often convenient: \( q(\cdot) = p(x_k| x_{k-1}) \) (prior). Easy to implement, but does not take measurements into account.
Resampling Approaches

Basic idea of resampling:

Whenever degeneracy rises above threshold: replace old set of samples (+ weights) with new set of samples (+ weights), such that sample density better reflects posterior pdf.

This eliminates particles with low weight and chooses more particles in more probable regions.

Complexity: possible in $O(N_s)$ operations
The resampling principle:

\[ i=1, \ldots, N=10 \text{ particles} \]

\[ \{X_{t-1}^{(i)}, \tilde{W}_{t-1}^{(i)}\} \]

\[ \{X_t^{(i)}, N^{-1}\} \]

\[ \{\tilde{X}_{t-1}^{(i)}, \tilde{W}_{t-1}^{(i)}\} \]

\[ \{X_t^{(i)}, N^{-1}\} \]

\[ \{\tilde{X}_t^{(i)}, \tilde{W}_t^{(i)}\} \]

(graphics taken from Van der Merwe et al.)
General Particle Filter – Pseudo Code

\[
\left[ \{ x_i^k, w_i^k \}_{i=1}^{N_s} \right] = \text{PF} \left( \left\{ x_{k-1}^i, w_{k-1}^i \right\}_{i=1}^{N_s}, z_k \right)
\]

\text{FOR } i = 1 : N_s
\begin{align*}
&\text{draw } x_k^i \sim q(x_k|x_{k-1}^i, z_k) \\
&\text{update weights according to (3)}
\end{align*}
\text{END FOR}

\text{normalize weights to } \sum_{i=1}^{N_s} w_k^i = 1

\text{IF degeneracy too high}
\text{resample } \left\{ x_k^i, w_k^i \right\}_{i=1}^{N_s}
\text{END IF}
PROBLEM: Loss of Diversity

No degeneracy problem but new problem arises:

- Particles with high weight are selected more and more often, others die out slowly
  ⇒ *loss of diversity* or *sample impoverishment*

For small process noise, all particles can collapse into a single point within a few iterations.

Other problem: resampling limits the ability to parallelize algorithm.
Other Particle Filter Variants

Methods to counteract loss of diversity and degeneracy problem:

- resample-move algorithm
- regularization
- Rao-Blackwellisation
- multiple Monte-Carlo

Other particle filter variants found in the literature:

- sampling importance resampling (SIR)
- auxiliary sampling importance resampling (ASIR)
- regularized particle filter (RPF)
- ...
Experiments

see videos...
8 Summary

First of all: what I did *not* talk about…

- speed of convergence
- number of samples needed
- complexity issues / tricks for speed-up of algorithms
- advanced particle filter variants in detail

⇒ refer to the literature if you want to know more
Advantages of particle filters (PFs):

- can deal with non-linearities
- can deal with non-Gaussian noise
- can be implemented in $O(N_s)$
- mostly parallelizable
- easy to implement

- in contrast to HMM filters (state-space discretized to $N$ fixed states): PFs focus adaptively on probable regions of state-space
Thesis:

If you want to solve a filtering problem, then particle filters are the best filters you can use, much better than e.g. Kalman filters.

Right or wrong?
WRONG!

Particle filters include a random element; they only convergence to the true posterior pdf (almost surely) if $N_s \to \infty$. Therefore: *If the assumptions for Kalman filters or grid-based filters are valid, no PF can outperform them!*

Additionally: depending on the dynamic model, Gaussian sum filters, unscented Kalman filters or extended Kalman filters may produce satisfactory results at lower computational cost.

(But you should at least try a PF; it is usually better than other suboptimal methods!)
PF approaches proved their usefulness in a variety of applications.

But:

- choice of importance function $q(\cdot)$ is crucial in PF design
- large sample number $N_s$ increases computational effort
- potential problems: *degeneracy* and *loss of diversity*

If these points are taken into account, then particle filters are an extremely powerful tool for filtering / estimation.

(“black box usage” vs “know what you’re doing!”)
Thank you!

This presentation was made with \LaTeX.
(try to write București in Powerpoint...)