A Brief Tutorial On Recursive Estimation With Examples From Intelligent Vehicle Applications (Part IV): Sampling Based Methods And The Particle Filter

Hao Li

A Brief Tutorial On Recursive Estimation With Examples From Intelligent Vehicle Applications (Part IV): Sampling Based Methods And The Particle Filter

Hao Li

Abstract

Following the third article of the series "*A brief tutorial on recursive estimation*", in this article (the fourth article) we continue to focus on the problem of how to handle model nonlinearity in recursive estimation. We will review the particle filter a.k.a. a sequential Monte Carlo method which has the potential to handle recursive estimation problems with an system model and a measurement model of arbitrary types and with data statistics of arbitrary types. We will explain basic principles that underlie the particle filter, and demonstrate its performance with examples from intelligent vehicle applications. We will explain its advantage as well as limitation.

Keywords: recursive estimation, model nonlinearity, particle filter (PF), sequential importance sampling (SIS), multimodal ambiguity, intelligent vehicles

1 Introduction

In the third article [1] of the series "*A brief tutorial on recursive estimation*", we have reviewed the *unscented Kalman filter* (UKF) [2] [3] which may be treated as another "extension" of the original Kalman filter besides the already existing *extended Kalman filter* (EKF); we have explained its idea of how to handle nonlinear factors in the estimation and maintain the consistency of data statistics—note that the EKF may result in inconsistent data statistics in handling a nonlinear estimation problem.

On the other hand, the UKF is still a KF and one fundamental requirement for applying it is that all data statistics are modelled as Gaussian distributions. In other words, the distribution of a random variable is approximated by an **ellipsoid** (a high dimensional analogue of an *ellipse*). Although data statistics in reality can hardly be strictly Gaussian, yet this practice is fairly effective for modelling **unimodal** data statistics. A question arises naturally: if data statistics of a variable are **multimodal** and it is not suitable at all to approximate them as a Gaussian distribution (see Fig.1), then what recursive estimation method can we use?

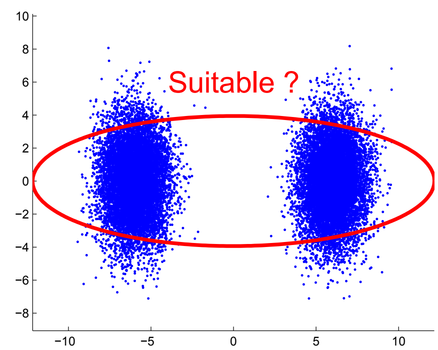

Figure 1: Multimodal statistics

Or imagine that data statistics of a state are still unimodal but the shape of its distribution is not so like an ellipsoid (see Fig.2), then besides choices of applying variants of the KF, what estimation method else can we use so that the distribution shape may be better captured?

We do have another choice besides the KF family and this choice is the **particle filter** (PF) a.k.a. a **sequential Monte Carlo method** (SMC) [4] [5] [6] [7]. The PF has the potential to handle recursive estimation problems with an system model and a measurement model of arbitrary types and with data statistics of arbitrary types.

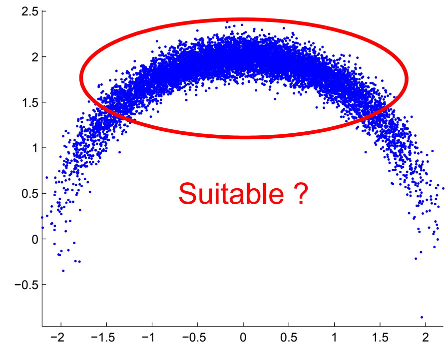

Figure 2: unimodal statistics unlike an ellipsoid

2 Sequential Monte Carlo (SMC) Methods

In fact, contents in this section are not necessarily related to the PF; however, with understanding of these contents, on can naturally understand the mechanism of the PF which is explained in section 3.

2.1 Sampling: Monte Carlo (MC) methods

We begin with a question: Suppose we have a random variable whose **probability density function** (PDF) $p(x)$ is not exactly known, but we can draw samples x^i $(i = 1, 2, 3, ...)$ according to the distribution of this random variable i.e. $x^i \sim p(x)$—one can imagine that a *black box* process outputs random samples from time to time, but we do not have any idea on the distribution i.e. the PDF according to which the black box process outputs random samples. An example of such kind of black box processes is the process of throwing a dice again and again—Then can we approximate this PDF unknown with samples drawn? The answer is **yes**. We can approximate $p(x)$ as in (1) (suppose we have already drawn N samples):

$$p(x) \approx \frac{1}{N} \sum_{i=1}^{N} \delta_{x^i}(x)|_{x^i \sim p(x)} \tag{1}$$

where $\delta_{x^i}(x)$ is the **Dirac *delta* function** with the **Dirac measure** or **Dirac unit mass** at x^i. The quality of the approximation increases as the number of samples increases, and we have:

$$p(x) = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \delta_{x^i}(x)|_{x^i \sim p(x)} \tag{2}$$

We give a rough analysis for (2): given a generic set of x denoted as A, we have

$$\begin{aligned}
&\int_A \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \delta_{x^i}(x)|_{x^i\sim p(x)} dx \\
&= \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \int_A \delta_{x^i}(x)|_{x^i\sim p(x)} dx \\
&= \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} I(x^i|_{x^i\sim p(x)} \in A) \\
&= \lim_{N\to\infty} \frac{\#(x^i|_{x^i\sim p(x)} \in A,\ \text{for } i = 1, ..., N)}{N} \\
&= p(x \in A) = \int_A p(x) dx
\end{aligned}$$

where $I(...)$ denotes the **indicator function**

$$I(x \in A) \text{ or } I_A(x) = \begin{cases} 1 & \text{if is true } x \in A \\ 0 & \text{if not true} \end{cases}$$

and $\#(...)$ means "the number of elements specified by (...)". Because of the generality of A in above derivation, we let $\|A\| \to 0$ and the equality will converge to a limit form as specified in (2). Above, we have only given a rough proof for (2) because we have no intention to go into too much pure mathematical details here. To understand the PF latter, one only needs to know a fact here: *normally, a PDF can be fairly approximated by a large number of samples generated according to this PDF.*

The approximation method described in (1) and theoretically in (2) is the **Monte Carlo** (MC) method—"Monte Carlo" refers to the name of a city famous for its casino i.e. a place for lots of "sampling" processes, so one can easily see the implicit meaning of "Monte Carlo" as used in "Monte Carlo method" or other terms in statistics background. Privately, one may also call it "Las Vegas" method or "Ao Men (Macao)" method if one likes.

If we have an explicit description of the PDF $p(x)$, is the Monte Carlo method still useful in this case? The answer is also **yes**. In real applications, we rarely examine a PDF itself; instead, we usually examine certain integral associated with the PDF, such as the **expectation** of a random state

$$\bar{x} = \int_{-\infty}^{\infty} x p(x) dx$$

or the variance or covariance of the state

$$D(x) = \int_{-\infty}^{\infty} (x - \bar{x})^2 p(x) dx$$

or the expectation of certain cost function $c(x)$

$$\bar{c} = \int_{-\infty}^{\infty} c(x) p(x) dx$$

We know that computing an integral explicitly is not always easy and many times even impossible. The Monte Carlo method provides a tractable and generic way to handle integrals numerically. Given an arbitrary function $f(x)$, its expectation can be approximately computed using the Monte Carlo method as

$$\begin{aligned}
\bar{f} &= \int_{-\infty}^{\infty} f(x) p(x) dx \\
&\approx \int_{-\infty}^{\infty} f(x) [\frac{1}{N} \sum_{i=1}^{N} \delta_{x^i}(x)|_{x^i \sim p(x)}] dx \\
&= \frac{1}{N} \sum_{i=1}^{N} f(x^i)|_{x^i \sim p(x)}
\end{aligned}$$

2.2 Importance sampling

In the previous subsection, we have reviewed the basic Monte Carlo method. Now suppose it is difficult to directly draw samples according to the objective PDF $p(x)$, but it is easy to draw samples according to another PDF $q(x)$ (satisfying $q(x) > 0$) as $x^i \sim q(x)$, $i = 1, 2, ...$ Question: can we approximate $p(x)$ with samples drawn according to $q(x)$? The answer is **yes**. We can approximate $p(x)$ as

$$\begin{aligned}
p(x) &= \frac{p(x)}{q(x)} q(x) \\
&\approx \frac{p(x)}{q(x)} [\frac{1}{N} \sum_{i=1}^{N} \delta_{x^i}(x)|_{x^i \sim q(x)}] \text{ (recall (1))} \\
&= \frac{1}{N} \sum_{i=1}^{N} \frac{p(x^i)}{q(x^i)} \delta_{x^i}(x)|_{x^i \sim q(x)}
\end{aligned}$$

i.e. as in (3)

$$\begin{aligned}
p(x) &= \sum_{i=1}^{N} w^i \delta_{x^i}(x)|_{x^i \sim q(x)} \\
w^i &= \frac{1}{N} \frac{p(x^i)}{q(x^i)}
\end{aligned} \tag{3}$$

where w^i is called an **importance weight** or a **weight** for short. We can normalize w^i ($i = 1, 2, ..., N$) to guarantee that the integral of the approximation function (3) is 1. Weights before the normalization step are called **unnormalized (importance) weights**. The PDF $q(x)$, according to which samples are drawn, is called the **proposal distribution function**, the **proposal density**, or the **importance density**.

2.3 Sequential sampling

Given a set of variables $x_{0:t}$ i.e. $\{x_0, x_1, ..., x_t\}$ and its associated **joint probability density function** (joint PDF) $p(x_{0:t})$, we can approximate this joint PDF as in (1) and (2):

$$p(x_{0:t}) \approx \frac{1}{N} \sum_{i=1}^{N} \delta_{x_{0:t}^i}(x)|_{x_{0:t}^i \sim p(x_{0:t})}$$

However, as t increases, it will become more and more difficult to draw samples directly from the joint PDF $p(x_{0:t})$. Then how to draw samples to approximate $p(x_{0:t})$?

Although it is difficult to draw a sample for the whole $x_{0:t}$ directly from the joint PDF, it is much easier to draw a sample for only one element in

$x_{0:t}$. To draw a sample $x^i_{0:t}$, we may follow the spirit of “divide and conquer” and draw x^i_0, x^i_1, ..., x^i_t one by one i.e. in a sequential way. Besides, recall the chain rule

$$p(x_{0:t}) = p(x_0)p(x_1|x_0)p(x_2|x_{0:1})...p(x_t|x_{0:t-1})$$

Then a strategy of sampling can be formulated as in (4):

$$\begin{aligned}
&x^i_0 \sim p(x_0)\\
&x^i_1 \sim p(x_1|x^i_0)\\
&x^i_2 \sim p(x_2|x^i_{0:1})\\
&......\\
&x^i_t \sim p(x_t|x^i_{0:t-1})
\end{aligned} \tag{4}$$

This sampling method described in (4) is the **sequential sampling** (SS) method. A question arises naturally, can a group of samples $x^i_{0:t}$ ($i = 1, ..., N$ with N large enough) drawn in the sequential way as in (4) fairly approximate the statistics reflected by the joint PDF $p(x_{0:t})$?

The answer is **yes**. Here, we only give a rough proof for cases of $t = 1$ and one can extend the result to cases of general t via *(mathematical) induction*—We recommend readers to jump this proof part for the moment and go directly to the next subsection.

Given a generic set of x_0 denoted as A_0 and a generic set of x_1 denoted as A_1; let a large number of samples drawn sequentially via (4) be denoted as $x^i_{0:1}$ ($i = 1, ..., N$). Define

$$s(A_0, A_1) = \lim_{N\to\infty} \int_{A_0}\int_{A_1} \frac{1}{N}\sum_{i=1}^{N} \delta_{x^i_{0:1}}(x_{0:1})dx_0dx_1$$

What we want to prove is

$$\lim_{\|A_{0:1}\|\to 0} \frac{s(A_0, A_1)}{p(x_0 \in A_0, x_1 \in A_1)} = 1 \tag{5}$$

First, define two functions in terms of x_1

$$p_{min}(x_1|A_0) = \min_{x_0\in A_0} p(x_1|x_0),\ \forall x_1 \in A_1$$
$$p_{max}(x_1|A_0) = \max_{x_0\in A_0} p(x_1|x_0),\ \forall x_1 \in A_1$$

We have

$$
\begin{aligned}
s(A_0, A_1) &= \lim_{N\to\infty} \int_{A_0}\int_{A_1} \frac{1}{N}\sum_{i=1}^{N} \delta_{x^i_{0:1}}(x_{0:1})dx_0dx_1 \\
&= \lim_{N\to\infty} \frac{1}{N}\#(x^i_0 \in A_0, x^i_1 \in A_1|_{x^i_1\sim p(x_1|x^i_0)}) \\
&= \lim_{N\to\infty} \frac{\#(x^i_0 \in A_0)}{N}\frac{\#(x^i_0 \in A_0, x^i_1 \in A_1|_{x^i_1\sim p(x_1|x^i_0)})}{\#(x^i_0 \in A_0)} \\
&\geq \lim_{N\to\infty} \frac{\#(x^i_0 \in A_0)}{N}\frac{\#(x^i_0 \in A_0, x^i_1 \in A_1|_{x^i_1\sim p_{min}(x_1|A_0)})}{\#(x^i_0 \in A_0)} \\
&= p(x_0 \in A_0)p_{min}(x_1 \in A_1|A_0)
\end{aligned}
$$

Note in above derivation that $x^i_1 \sim p_{min}(x_1|A_0)$ is independent of $x^i_0 \sim p(x_0)$. Similarly we have

$$
s(A_0, A_1) \leq p(x_0 \in A_0)p_{max}(x_1 \in A_1|A_0)
$$

If we assume $p(x)$ is continuous, then we have

$$
\lim_{\|A_{0:1}\|\to 0} \frac{p_{max}(x_1 \in A_1|A_0)}{p_{min}(x_1 \in A_1|A_0)} = 1
$$

Also note that

$$
\begin{aligned}
p(x_1 \in &A_1|x_0 \in A_0) \\
&= \int_{A_0} \frac{p(x_1 \in A_1|x_0)p(x_0)}{p(x_0 \in A_0)}dx_0 \\
&\geq \frac{\int_{A_0} p_{min}(x_1 \in A_1|A_0)p(x_0)dx_0}{p(x_0 \in A_0)} \\
&= p_{min}(x_1 \in A_1|A_0)
\end{aligned}
$$

and similarly that

$$
p(x_1 \in A_1|x_0 \in A_0) \leq p_{max}(x_1 \in A_1|A_0)
$$

and thus we have

$$
\lim_{\|A_{0:1}\|\to 0} \frac{p_{max}(x_1 \in A_1|A_0)}{p(x_1 \in A_1|x_0 \in A_0)} = \lim_{\|A_{0:1}\|\to 0} \frac{p_{min}(x_1 \in A_1|A_0)}{p(x_1 \in A_1|x_0 \in A_0)} = 1
$$

and we further have

$$\lim_{\|A_{0:1}\|\to 0} \frac{p(x_0 \in A_0)p_{max}(x_1 \in A_1|A_0)}{p(x_0 \in A_0)p(x_1 \in A_1|x_0 \in A_0)}$$
$$= \lim_{\|A_{0:1}\|\to 0} \frac{p(x_0 \in A_0)p_{min}(x_1 \in A_1|A_0)}{p(x_0 \in A_0)p(x_1 \in A_1|x_0 \in A_0)} = 1$$

i.e.

$$\lim_{\|A_{0:1}\|\to 0} \frac{p(x_0 \in A_0)p_{max}(x_1 \in A_1|A_0)}{p(x_0 \in A_0, x_1 \in A_1)}$$
$$= \lim_{\|A_{0:1}\|\to 0} \frac{p(x_0 \in A_0)p_{min}(x_1 \in A_1|A_0)}{p(x_0 \in A_0, x_1 \in A_1)} = 1$$

Then with the already obtained result

$$p(x_0 \in A_0)p_{min}(x_1 \in A_1|A_0) \leq s(A_0, A_1)$$
$$\leq p(x_0 \in A_0)p_{max}(x_1 \in A_1|A_0)$$

we can have the result in (5).

2.4 Sequential importance sampling (SIS)

Now suppose we have a set of variables $x_{0:t}$ i.e. $\{x_0, x_1, ..., x_t\}$ and its associated joint PDF $p(x_{0:t})$. Further suppose it is even difficult to draw samples for only one element in $x_{0:t}$ according to p, not to mention to draw samples directly for the whole $x_{0:t}$ according to the joint PDF. Then how can we draw samples of $x_{0:t}$?

Based on the importance sampling method introduced in section 2.2 and the sequential sampling method introduced in section 2.3, we can naturally introduce a solution to this problem, which is a combination of the importance sampling method and the sequential sampling method. The solution is usually referred to as the **sequential importance sampling** (SIS) method in literature. The idea is:

Draw samples $x_{0:t}^i$ according to a proposal joint PDF $q(x_{0:t})$ using the sequential sampling method as

$$\begin{aligned}
&x_0^i \sim q(x_0) \\
&x_1^i \sim q(x_1|x_0^i) \\
&x_2^i \sim q(x_2|x_{0:1}^i) \\
&\ldots\ldots \\
&x_t^i \sim q(x_t|x_{0:t-1}^i)
\end{aligned} \tag{6}$$

and then represent $p(x_{0:t})$ with these samples using the importance sampling method as

$$\begin{aligned}
p(x_{0:t}) &= \frac{p(x_{0:t})}{q(x_{0:t})} q(x_{0:t}) \\
&\approx \frac{p(x_{0:t})}{q(x_{0:t})} \frac{1}{N} \sum_{i=1}^{N} \delta_{x_{0:t}^i}(x_{0:t}) \\
&= \frac{1}{N} \sum_{i=1}^{N} \frac{p(x_{0:t}^i)}{q(x_{0:t}^i)} \delta_{x_{0:t}^i}(x_{0:t}) \\
&= \frac{1}{N} \sum_{i=1}^{N} [\frac{p(x_0^i)}{q(x_0^i)} \prod_{k=1}^{t} \frac{p(x_k^i|x_{0:k-1}^i)}{q(x_k^i|x_{0:k-1}^i)}] \delta_{x_{0:t}^i}(x_{0:t})
\end{aligned} \tag{7}$$

where samples $x_{0:t}^i$ $(i = 1, ..., N)$ are drawn according to $q(x_{0:t})$ as in (6). Here, we define the (unnormalized) weights as

$$w_t^i = \frac{1}{N} \frac{p(x_0^i)}{q(x_0^i)} \prod_{k=1}^{t} \frac{p(x_k^i|x_{0:k-1}^i)}{q(x_k^i|x_{0:k-1}^i)} \tag{8}$$

Note that

$$\begin{aligned}
w_0^i &= \frac{1}{N} \frac{p(x_0^i)}{q(x_0^i)} \\
w_k^i &= w_{k-1}^i \frac{p(x_k^i|x_{0:k-1}^i)}{q(x_k^i|x_{0:k-1}^i)} \text{ for } k = 1, 2, ..., t
\end{aligned}$$

So we can reformulate (6) and (8) in a recursive way as follows:

Sequential Importance Sampling (SIS)

Initialization ($t = 0$):
• Sample x_0^i $(i = 1, ..., N)$ as

$$x_0^i \sim q(x_0)$$

• Compute weights w_0^i $(i = 1, ..., N)$ as

$$w_0^i = \frac{p(x_0^i)}{q(x_0^i)}$$

and normalize the weights.
Iteration ($t \geq 1$):
• Sample x_t^i $(i = 1, ..., N)$ as

$$x_t^i \sim q(x_t | x_{0:t-1}^i)$$

• Compute weights w_t^i $(i = 1, ..., N)$ as

$$w_t^i = w_{t-1}^i \frac{p(x_t^i | x_{0:t-1}^i)}{q(x_t^i | x_{0:t-1}^i)}$$

and normalize the weights.

2.5 Resampling

The effectiveness of the SIS method described in section 2.4 relies on the condition that the number of samples i.e. N is large enough for all the $t + 1$ dimensions of $x_{0:t}$—a generic x_k ($k \in \{0, 1, ..., t\}$) may itself be a multidimensional state, yet we treat it here as an representative for an "abstract" dimension—As t increases, more samples are naturally needed to "cover" the extra dimensions. For example, suppose we need 10 samples for each dimension to guarantee a fair distribution approximation for this dimension, then we need a total number of 10^{t+1} samples to approximate the distribution of the joint states $x_{0:t}$. The number of samples needed grows exponentially with t and will quickly become forbiddingly large. In reality, we can only set a limited number of samples despite the increase of t. Given a fixed number of samples, the approximation quality of these samples deteriorates quickly as t increases and before long the samples can not represent the distribution of $x_{0:t}$ at all. This is the **degeneracy problem**.

Since samples are limited, we hope that they can represent distribution areas of large importance instead of being wasted to represent areas of trivial importance. The **resampling** technique provides a way to concentrate samples more towards areas of large importance.

The idea of the resampling technique is: given N samples x^i $(i = 1, ..., N)$ with their corresponding weights w^i $(i = 1, ..., N)$, generate N new samples $\bar{x}^i$ $(i = 1, ..., N)$ from the old samples. Each new sample is generated by selecting randomly an old sample in $x^{1:N}$ with the chance of this old sample being selected proportional to its corresponding weight. After the new samples are generated, set all the new weights to be the same to $1/N$ i.e. $\bar{w}^i = 1/N$ $(i = 1, ..., N)$. Replace $x^{1:N}$ and $w^{1:N}$ by $\bar{x}^{1:N}$ and $\bar{w}^{1:N}$.

2.6 Sequential importance sampling with resampling (SIS/R)

We may perform the resampling introduced in section 2.5 at the end of each iteration in the sequential importance sampling procedures introduced in section 2.4. On the other hand, we may choose to perform the resampling occasionally only under certain conditions. For example, if the weights are distributed not uniformly enough, which means most importance are concentrated to few samples, then we will perform the resampling so that in the next iteration samples are more likely to be drawn from areas of large importance. An indicator for the uniformness of the weights can be defined as in (9):

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} (w^i)^2} \tag{9}$$

The more uniform are the weights, the larger is the indicator N_{eff}. In special cases where $w^i = 1/N$ $(i = 1, ..., N)$ i.e. the weights are most uniform, N_{eff} achieves the largest value of N. We can set a threshold for N_{eff} denoted as N_{thr}: the resampling will be applied if N_{eff} is below N_{thr}.

Now we can reformulate the SIS procedures with adaptive resampling in a more generic framework as follows

---

Sequential Importance Sampling with Resampling (SIS/R)

Initialization $(t = 0)$:
• Sample x_0^i $(i = 1, ..., N)$ as

$$x_0^i \sim q(x_0)$$

• Compute weights w_0^i $(i = 1, ..., N)$ as

$$w_0^i = \frac{p(x_0^i)}{q(x_0^i)}$$

and normalize the weights.
Iteration $(t \geq 1)$:
• Sample x_t^i $(i = 1, ..., N)$ as

$$x_t^i \sim q(x_t|x_{0:t-1}^i)$$

• Compute weights w_t^i $(i = 1, ..., N)$ as

$$w_t^i = w_{t-1}^i \frac{p(x_t^i|x_{0:t-1}^i)}{q(x_t^i|x_{0:t-1}^i)}$$

and normalize the weights.
• Compute N_{eff} of $w_t^{1:N}$ via (9). If $N_{eff} < N_{thr}$, then resample $x_{0:t}^i$ and w_t^i $(i = 1, ..., N)$ as described in section 2.5.

With the SIS/R method introduced, we can go to the particle filter, explanations on which would be rather natural now.

3 The Particle Filter

Recall the generic formulation of estimation problems in the first tutorial article [8]. Given *a priori* knowledge on x_0, measurements $z_{1:t}$ (from time 0 to time t), a system model $p(x_t|x_{t-1})$ (here we omit explicit representation of the system input), and a measurement model $p(z_t|x_t)$, the estimation problem consists in estimating the *a posteriori* distribution $p(x_{0:t}|z_{1:t})$. Using Bayesian inference we have (based on the Markov assumption)

$$
\begin{aligned}
p(x_{0:t}|z_{1:t}) &= \frac{p(z_t|x_{0:t}, z_{1:t-1})p(x_{0:t}|z_{1:t-1})}{p(z_t|z_{1:t-1})} \\
&= \frac{1}{Z}p(z_t|x_{0:t}, z_{1:t-1})p(x_t|x_{0:t-1}, z_{1:t-1})p(x_{0:t-1}|z_{1:t-1}) \\
&= \frac{1}{Z}p(z_t|x_t)p(x_t|x_{t-1})p(x_{0:t-1}|z_{1:t-1})
\end{aligned}
$$

where Z is a normalization constant $Z = \int p(z_t|x_t)p(x_t|x_{t-1})p(x_{0:t-1}|z_{1:t-1})dx_{0:t}$. In other words, we have a recursive formalism to compute $p(x_{0:t}|z_{1:t})$ from $p(x_{0:t-1}|z_{1:t-1})$ as in (10):

$$
p(x_{0:t}|z_{1:t}) \propto p(z_t|x_t)p(x_t|x_{t-1})p(x_{0:t-1}|z_{1:t-1}) \tag{10}
$$

If we substitute $p(x_{0:t}|z_{1:t})$ for $p(x_{0:t})$ and substitute $p(z_t|x_t)p(x_t|x_{t-1})$ for $p(x_t^i|x_{0:t-1}^i)$ in the SIS/R method presented in section 2.6, and if we follow the Markov assumption, then the SIS/R method will become the particle filter as follows

---

Particle Filter (PF)

Initialization ($t = 0$):
• Sample x_0^i ($i = 1, ..., N$) as

$$x_0^i \sim q(x_0)$$

• Compute weights w_0^i ($i = 1, ..., N$) as

$$w_0^i = \frac{p(x_0^i)}{q(x_0^i)}$$

and normalize the weights.
Iteration ($t \geq 1$):
• Sample x_t^i ($i = 1, ..., N$) as

$$x_t^i \sim q(x_t|x_{t-1}^i)$$

• Compute weights w_t^i $(i = 1, ..., N)$ as

$$w_t^i = w_{t-1}^i \frac{p(z_t|x_t^i)p(x_t^i|x_{t-1}^i)}{q(x_t^i|x_{t-1}^i)}$$

and normalize the weights.
• Compute N_{eff} of $w_t^{1:N}$ via (9). If $N_{eff} < N_{thr}$, then resample $x_{0:t}^i$ and w_t^i $(i = 1, ..., N)$ as described in section 2.5.

In the particle filter, samples are usually no longer referred to as "samples" but figuratively as **particles**. Each particle represents a sample or a hypothesis for the whole trajectory of the state, i.e. for $x_{0:t}$.

Discussion

As we can see in above formalism of the particle filter, there is no specific condition or assumption imposed on the system model $p(x_t|x_{t-1})$ and the measurement model $p(z_t|x_t)$—they can refer to arbitrary functions, linear or nonlinear; they can take data of arbitrary statistics, Gaussian or non-Gaussian, unimodal or multimodal etc—Besides, they are used as "black boxes", which brings implementation convenience (recall that the UKF [2] [3] [1] also possesses this advantage compared with the EKF). Therefore, the particle filter is a rather generic method and can be used to handle arbitrary recursive estimation problems, at least *theoretically*—why say "theoretically"? Reasons will be given latter.

In many applications, we can fairly set the proposal density $q(x_t|x_{t-1})$ to be the same to the system model density $p(x_t|x_{t-1})$ and this will simplify the weight update step to

$$w_t^i = w_{t-1}^i p(z_t|x_t^i)$$

It is worth noting that setting $q(x_t|x_{t-1})$ to be the same to $p(x_t|x_{t-1})$ is not always a desirable choice. In cases where it is difficult to draw samples according to $p(x_t|x_{t-1})$ because of its complexity or where samples drawn according to $p(x_t|x_{t-1})$ may not well capture areas of large importance, we had better choose certain proposal density other than $p(x_t|x_{t-1})$.

4 Vehicle Localization In A 2D Case

4.1 Application description

We consider the same application of vehicle localization in a 2D case as described in [8] [1]. Suppose a vehicle is navigating on a 2D plane and needs to estimate its pose i.e. the position (x, y) and the orientation θ. In other words, we treat the pose of the vehicle as its state to be estimated; this state is denoted compact as $\mathbf{p}$ i.e. $\mathbf{p} = (x, y, \theta)$. The system model is given as the following kinematic model:

$$\begin{cases} x_t & = x_{t-1} + v_t \Delta T cos(\theta_{t-1} + \phi_t \Delta T/2) \\ y_t & = y_{t-1} + v_t \Delta T sin(\theta_{t-1} + \phi_t \Delta T/2) \\ \theta_t & = \theta_{t-1} + \phi_t \Delta T \end{cases} \tag{11}$$

where ΔT denotes the system period; v and ϕ denote respectively the speed and the yawrate of the vehicle. Suppose the vehicle is equipped with devices that monitor its speed and its yawrate. Speed measurements are denoted as $\hat{v}$, and yawrate measurements are denoted as $\hat{\phi}$. Their errors are assumed to follow the Gaussian distribution as $\Delta v_t \sim N(0, \Sigma_v)$ and $\Delta \phi_t \sim N(0, \Sigma_\phi)$.

Suppose the vehicle is also equipped with a component that outputs measurements on the vehicle position (x, y). Let vehicle position measurements be denoted as $\mathbf{z}$ and the measurement model is given as:

$$\mathbf{z}_t = \mathbf{H}\mathbf{p}_t + \gamma_t \tag{12}$$

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

where γ denotes the measurement error which is assumed to follow the Gaussian distribution with zero mean and covariance Σ_γ, i.e. $\gamma \sim N(\mathbf{0}, \Sigma_\gamma)$. The measurement model given in (12) is a partial measurement model.

One can refer to [8] [1] for details of implementing the EKF and the UKF. Details of implementing the PF are given as follows.

$q(x_t|x_{t-1})$ is chosen to be the same to $p(x_t|x_{t-1})$. A total number of $N = 100$ samples $\mathbf{p}^i = (x^i, y^i, \theta^i)$ $(i = 1, ..., N)$ are used to characterize the statistics of the vehicle state. In each iteration, perform

- Sample $\mathbf{p}_t^i = (x_t^i, y_t^i, \theta_t^i)$ $(i = 1, ..., N)$ as

$$\begin{cases} x_t^i & = x_{t-1}^i + (\hat{v}_t + \Delta v_t^i) \Delta T cos(\theta_{t-1}^i + (\hat{\phi}_t + \Delta \phi_t^i) \Delta T/2) \\ y_t^i & = y_{t-1}^i + (\hat{v}_t + \Delta v_t^i) \Delta T sin(\theta_{t-1}^i + (\hat{\phi}_t + \Delta \phi_t^i) \Delta T/2) \\ \theta_t^i & = \theta_{t-1}^i + (\hat{\phi}_t + \Delta \phi_t^i) \Delta T \end{cases}$$

where $\Delta v_t^i \sim N(0, \Sigma_v)$ and $\Delta \phi_t^i \sim N(0, \Sigma_\phi)$ $(i = 1, ..., N)$.
• Compute weights w_t^i $(i = 1, ..., N)$ as

$$w_t^i = w_{t-1}^i \exp(-\frac{1}{2}(\mathbf{z}_t - \mathbf{H}\mathbf{p}_t^i)^T \boldsymbol{\Sigma}_\gamma^{-1}(\mathbf{z}_t - \mathbf{H}\mathbf{p}_t^i))$$

and normalize the weights.
• Compute N_{eff} of $w_t^{1:N}$ via (9). If $N_{eff} < N_{thr}$, then resample $p_{0:t}^i$ and w_t^i $(i = 1, ..., N)$ as described in section 2.5.

4.2 Simulation

We tested performances of the EKF, the UKF, and the PF using same synthetic data generated according to the system model (11) and the measurement model (12). In the simulation, let $\Delta T = 1(s)$; let $\Sigma_v = 0.2^2(m^2/s^2)$; let $\Sigma_\phi = 0.05^2(rad^2/s^2)$; let $\Sigma_\gamma = diag(5.0^2, 5.0^2)(m^2)$. Set the ground-truth $p_0 = [0(m), 0(m), 0(rad)]^T$; $v_t = 10(m/s)$ and $\phi_t = 0.0(rad/s)$. The speed measurements and the yawrate measurements were synthesized according to $\hat{v}_t \sim N(v_t, \Sigma_v)$ and $\hat{\phi}_t \sim N(\phi_t, \Sigma_\phi)$. The vehicle position measurements were synthesized according to $z_t \sim N(p_t, \Sigma_\gamma)$.

The EKF, the UKF, and the PF were applied to the same synthetic data and their estimates on the vehicle state were obtained respectively. The results of 100 Monte Carlo trials are shown in Fig.3 and Fig.4, in both of which the red lines , the blue lines, and the green lines represent respectively the errors of the PF estimates (after convergence), those of the UKF estimates (after convergence), and those of the EKF estimates (after convergence). The black crosses in Fig.3 represent the position measurement errors.

As we can see, the PF performed similarly with the EKF and the UKF, yet slightly outperformed by the latter two. As discussed in section 3, the particle filter is a rather generic method and can be used to handle arbitrary recursive estimation problems; one may ask why the PF performed worse (though slightly) than the two KFs which are even less generic than the PF. One crucial reason lies in the number of particles N.

In fact, the ubiquitous applicability of the PF relies on an ideal condition that N is as large as possible i.e. $N \to \infty$. In real practice, however, we can never achieve this. For example, in the tests we used only 100 particles. Using limited number of particles to approximate the state statistics will inevitably result in approximation error. On the other hand, despite its performance slightly inferior to those of the KFs in above tests, the PF still

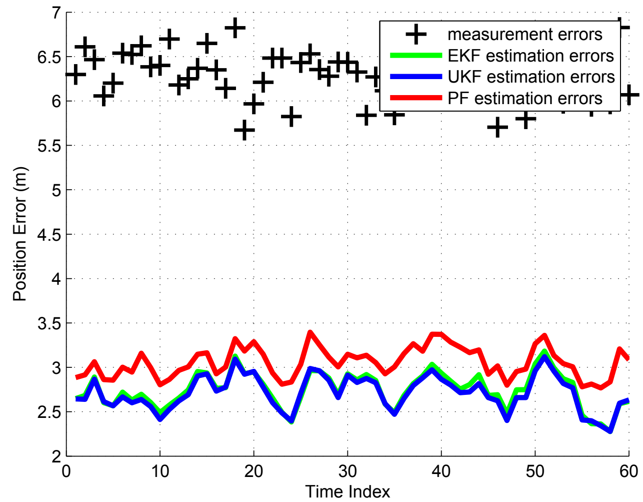

Figure 3: Position estimate errors for 100 Monte Carlo trials

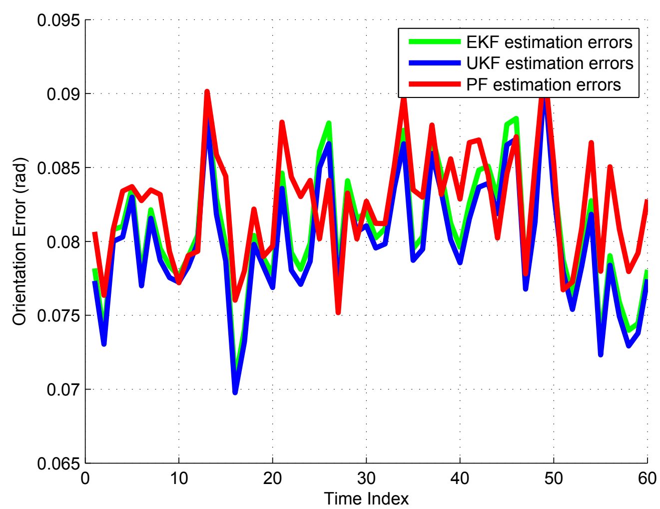

Figure 4: Orientation estimate errors for 100 Monte Carlo trials

fulfilled the estimation task well and performed similarly with the KFs on the whole. This shows that probabilistic methods can be effective in handling estimation problems.

4.3 Discussion

Theoretically, the PF can be used to handle arbitrary recursive estimation problems. We have already posed the question: why say "theoretically"? The reason lies in the dimension of the state. In above application example, the vehicle state has a dimension of only three (i.e. has three dimensions), so the practice of using a set of particles to approximate the state statistics is tractable. Imagine we need to estimate a state with fifty dimensions—note that a dimension of fifty is still moderate for many applications—Further suppose we only need two particles to "cover" the statistics on each dimension, one for the area above the mean, and one for the area below the mean—this is a rather rough approximation, just like we use a positive number and a negative number to approximate the entire real axis—then we need a total number of 2^{50} particles to cover all the fifty dimensions.

What does this 2^{50} mean? It means that we need at least $2^{50} \approx 100...00$ (there are 15 zeros at the end) operations to evaluate the weights of the particles at only one iteration. Given a normal 4G CPU and suppose it performs one operation at each system period, then 2^{50} operations will take about three days of computation!

Whereas an recursive estimation problem with a state of fifty dimensions is still tractable for the KF, it is already far beyond the capability of the PF as shown above. Generally, computational inefficiency is a disadvantage of the PF compared with the KF. Besides, as demonstrated in previous test results, the PF may be outperformed by the KF in handling estimation problems of unimodal data statistics.

Then what is the utility of the PF (family) compared with the KF (family)? The true power or advantage of the PF consists in its ability to handle multimodal ambiguity. For example, suppose we have two (or even more) measurement outputs or detected positives at the same time and these two measurement outputs are both plausible. We know there can be only one true positive where the other (or others) is (or are) false positives. To apply the KF, first we need to **decide** (usually using certain **data association** technique) which measurement output is the true one. However, what if our decision or data association is wrong? This can happen and can mislead the estimation result.

On the other hand, to apply the PF, we do not need to make a decisive choice at the moment; we can model the two (or several) plausible measurement outputs with a two-modal (or multimodal) distribution and perform the PF as we normally do. Although particles which coincide with both plausible measurement outputs can survive for the moment, particles that coincide with the true measurement output tend to survive finally because true measurement outputs usually possess more temporal consistency than false positives do. One can refer to [9] for a good demonstration of how the PF gradually removes multimodal ambiguity. Because of its advantage, the PF has been used in plenty of intelligent vehicle applications, such as lane detection [10] [11] [12], vehicle localization [13] [14] etc.

One may ask: if false positives are temporally even more consistent than true measurements, will the PF still succeed in the estimation? The answer is no. The estimates may be misled by the false positives because the PF does not know *a priori* whether they are true or false. In such case, however, one had better reflect on the quality of their measuring component instead of on the PF.

5 Conclusion

In this fourth article of the series "*A brief tutorial on recursive estimation*", we focus on the particle filter (PF) a.k.a. a sequential Monte Carlo method. We have explained the basic Monte Carlo method, the importance sampling (IS) method, the sequential sampling (SS) method, and the sequential importance sampling (SIS) method (which can be treated as a combination of the IS method and the SS method). We have also reviewed the degeneracy problem and the resampling method that handles this problem. Based on the sequential importance sampling method with resampling (SIS/R), we have reviewed the formalism of the PF.

The PF has the potential to handle recursive estimation problems with an system model and a measurement model of arbitrary types and with data statistics of arbitrary types. On the other hand, the PF is not computationally efficient compared with the KF (including its variants). Besides, the PF may be outperformed by the KF due to the approximation error of its particles. Thus, we had better not exaggerate the applicability of the PF just because of its generic formalism. If we intend to use the PF in certain application, we had better bear in mind its true power, especially its potential to handle multimodal ambiguity.

References

[1] H. Li. *A Brief Tutorial On Recursive Estimation With Examples From Intelligent Vehicle Applications (Part III): Handling Nonlinear Estimation Problems And The Unscented Kalman Filter*. HAL Open Archives, 2014.

[2] S.J. Julier and J.K. Uhlmann. A new extension of the kalman filter to nonlinear systems. *Int. symp. aerospace defense sensing, simul. and controls*, 3(26), 1997.

[3] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[4] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F (Radar and Signal Processing)*, 140(2):107–113, 1993.

[5] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[6] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. New York, USA: Springer-Verlag, 2001.

[7] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[8] H. Li. *A Brief Tutorial On Recursive Estimation With Examples From Intelligent Vehicle Applications (Part I): Basic Spirit And Utilities*. HAL Open Archives, 2014.

[9] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT Press, 2005.

[10] Z.W. Kim. Robust lane detection and tracking in challenging scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 9(1):16–26, 2008.

[11] H. Li and F. Nashashibi. Robust real-time lane detection based on lane mark segment features and general a priori knowledge. In *IEEE International Conference on Robotics and Biomimetics*, pages 812–817, 2011.

[12] H. Li and F. Nashashibi. Lane detection (part i): Mono-vision based method. *INRIA Tech Report*, RT-433, 2013.

[13] F. Chausse, J. Laneurit, and R. Chapuis. Vehicle localization on a digital map using particles filtering. In *IEEE Intelligent Vehicles Symposium*, pages 243–248, 2005.

[14] H. Li, F. Nashashibi, and G. Toulminet. Localization for intelligent vehicle by fusing mono-camera, low-cost gps and map data. In *IEEE International Conference on Intelligent Transportation Systems*, pages 1657–1662, 2010.