Wireless Sensor Network, **2009, 2, 61-121**
doi:10.4236/wsn.2009.12014 Published Online July 2009 (http://www.SciRP.org/journal/wsn/).

Gaussian Convolution Filter and its Application to Tracking

Qing LIN, Jianjun YIN, Jianqiu ZHANG, Bo HU
Electronic Engineering Department, Fudan University, Shanghai, China
E-mail: *qlin98@fudan.edu.cn*
Received April 29, 2009; *revised May* 11, 2009; *accepted May* 12, 2009

Abstract

A new recursive algorithm, called the Gaussian convolution filter (GCF), is proposed for nonlinear dynamic state space models. Based on the convolution filter (CF) and similar to the Gaussian filters, the GCF approximates the posterior density of the states by Gaussian distribution. The analytical results show the ability to deal with complex observation model and small observation noise of the GCF over the Gaussian particle filter (GPF) and the lower complexity, more amenable for parallel implementation than the CF. The Simulation in the Tracking domain demonstrates the good performance of the GCF.

Keywords: Signal Processing, Tracking, Nonlinear Estimation

1. Introduction

To estimate the dynamic state from the history of noisy observations is the main objective of filtering, which arise in many fields including statistics, economics and engineering such as tracking and navigation. Based on the difference of the dynamic state space models, usually filtering can be divided into two categories: linear and nonlinear, which correspond to the linear Gaussian models and nonlinear and/or non-Gaussian models respectively. For linear filtering, Kalman filter (KF) [1] usually gives the optimal results. For nonlinear filtering, the extended Kalman filter (EKF) [2] is most popular. However, the linearization process of the EKF is liable to large errors threatening the convergence of the algorithm, particularly for models with high nonlinearity. A recently-popularized technique for numerical approximation, termed as the particle filter (PF) [3−5], offers a general tool for the state estimation of nonlinear non-Gaussian systems. The core idea behind the PF is to use samples (particles) to approximate the concerned distributions. Usually the PF gives better results than the EKF and the unscented Kalman filter (UKF) [6]. However, it also has drawbacks. Firstly, the algorithm is complex and difficult to parallel implementation [7]. Secondly it is prone to divergence when the observation noise is too small [8]. Thirdly, it does not work when the likelihood function can not be obtained analytically [8]. The first drawback has been overcome by the Gaussian particle filter (GPF) [7], which approximates the posterior distributions by single Gaussians, and avoid the resampling step, which reduces the complexity and is more amenable for fully parallel implementation in VLSI. However the second and third shortages are still within the GPF. The convolution filter (CF) [8] has circumvented the second and third drawbacks by using convolution kernels, however, the first one still remains.

In this paper, we propose a new algorithm, namely the Gaussian convolution filter (GCF), which can overcome all the three drawbacks above. The GCF is based on the convolution filtering concept, and it approximates the posterior distributions by single Gaussians. It is shown that the GCF is asymptotically optimal in the number of particle under the Gaussianity assumption. The Simulation results in the Tracking demonstrate the performance of the GCF when the observation noise is too small and the GPF fails.

2. Background

In this section, we first describe the problem formulation. The convolution filter is then recalled.

2.1. Problem Formulation

Let the following general model of a state space system be considered [3]:

$$x_t = f_t(x_{t-1}, \omega_t) \quad \textbf{(1)}$$

$$y_t = h_t(x_t, \upsilon_t) \quad \textbf{(2)}$$

where ω_t and υ_t denote the known independent process and measurement noise respectively, $\{x_t\}$ the state of the system complying with the Markov process, $\{y_t\}$ the measurements of the system, and both f_t and h_t are known nonlinear or linear functions. Again, let $X_t = \{x_1, \cdots x_t\}$, $Y_t = \{y_1, \cdots y_t\}$, and $p(x_0|x_{-1}) = p(x_0)$ be the initial density. Here, the purpose is supposed to estimate the posterior probability density function (pdf) $p(x_t|Y_t)$ by the Bayes recursion

$$p(x_t \mid Y_{t-1}) = \int p(x_t \mid x_{t-1})\, p(x_{t-1} \mid Y_{t-1})\, dx_{t-1} \quad \textbf{(3)}$$

$$p(x_t|Y_t) = p(y_t|x_t)p(x_t|Y_{t-1}) \Big/ \left(\int p(y_t|x_t)p(x_t|Y_{t-1})dx_t\right) \quad \textbf{(4)}$$

2.2. The Convolution Filter

Usually in PF scheme, the weight of each particle is given by the likelihood function. The observations thus operate the filter through the likelihood function which is assumed to exist and to be known. This assumption is rather restricting in practice. Moreover it rules out the non-noisy case and will also cause trouble when the observation noise is too small and also when the noise is non-additive as in the general system (1) and (2). These drawbacks can be circumvented by using convolution kernels to weight the particles in the CF [8]. We can approximate the weights consistently by simulating the observations, and use this approximation in place of the true function in the PF, i.e.

$$w_t^{(i)} \propto p\left(y_t \mid x_t^{(i)}\right) \approx K_{hn}^{z}\left(y_t - y_t^{(i)}\right) \quad \textbf{(5)}$$

where $w_t^{(i)}$ denote particle weights, K_{hn}^{z} is Parzen-Rosenblatt kernel of appropriate dimensions (A Parzen-Rosenblatt kernel is a bounded positive and symmetric function for which $\int K d\lambda = 1$, where λ is the Lebesgue measure, and $\lim \|x\|^d K(x) = 0$ as $\|x\| \to \infty$, where d is the dimension of variable y and $\|\cdot\|$ denotes the squared norm), h_n is called the kernel bandwidth, and $\hat{y}_t^{(i)}$ are the samples from observation. Then we can get the estimation as

$$p(x_t \mid Y_t) = \sum_{i=1}^{n} w_t^{(i)} K_{hn}^{x}\left(x_t - \hat{x}_t^{(i)}\right) \Big/ \sum_{i=1}^{n} w_t^{(i)} \quad \textbf{(6)}$$

A brief description of the resampled CF (RCF) is given in Table 1.

Table 1. The RCF algorithm.

```
For $t = 0$
    Given $p_0$ be the probability density of the initial state distribution
For $t \ge 1$
    (1) resample: $\{x_{t-1}^{(i)}\}_{i=1}^{n} \sim p_{t-1}$
    (2) evolving: $x_t^{(i)} \sim f_t(x_{t-1}^{(i)}, \cdot)$, $y_t^{(i)} \sim h_t(x_t^{(i)}, \cdot)$, $i = 1, \cdots, n$
    (3) estimation:
        $p_t(x_t \mid Y_t) = \dfrac{\sum_{i=1}^{n} K_{hn}^{y}\left(y_t - y_t^{(i)}\right) K_{hn}^{x}\left(x_t - x_t^{(i)}\right)}{\sum_{i=1}^{n} K_{hn}^{y}\left(y_t - y_t^{(i)}\right)}$.
```

3. The Gaussian Convolution Filter

In this section, we present the main results of the paper, the GCF recursion. The main idea behind the GCF is to estimate the posterior distributions by CF, and then approximate them by Gaussians.

3.1. The Measurement Update

Assume the density of prediction is approximated by a Gaussian [7], i.e.,

$$p(x_t \mid Y_{t-1}) \approx \mathrm{N}\left(x_t; \bar{\mu}_t, \bar{\Sigma}_t\right) \quad \textbf{(7)}$$

where $\mathrm{N}(x; \mu, \Sigma)$ denotes the Gaussian distribution with the mean μ and covariance Σ. Take (7) as the importance density and get samples from it, i.e.,

$$x_t^{(i)} \sim \mathrm{N}\left(x_t; \bar{\mu}_t, \bar{\Sigma}_t\right), \quad i = 1, \cdots, N \quad \textbf{(8)}$$

Obtain the observations

$$y_t^{(i)} \sim h_t(x_t^{(i)}, \cdot), \quad i = 1, \cdots, N \quad \textbf{(9)}$$

and the particle weights

$$w_t^{(i)} \propto \hat{p}\left(y_t \mid x_t^{(i)}\right) = K_{hn}^{z}\left(y_t - y_t^{(i)}\right) \quad \textbf{(10)}$$

By substituting (7) into (4) we get

$$p(x_t \mid Y_t) \approx m_t \cdot p(y_t \mid x_t) \cdot \mathrm{N}\left(x_t; \bar{\mu}_t, \bar{\Sigma}_t\right) \quad \textbf{(11)}$$

where

$$m_t = 1 \Big/ \int p(y_t \mid x_t)\, p(x_t \mid Y_{t-1})\, dx_t \quad \textbf{(12)}$$

We approximate (11) by a Gaussian, i.e.,

$$p(x_t \mid Y_t) \approx \mathrm{N}\left(x_t; \mu_t, \Sigma_t\right) \quad \textbf{(13)}$$

where

$$\begin{aligned} \mu_t &= \sum_{i=1}^{N} w_t^{(i)} x_t^{(i)} \Big/ \sum_{i=1}^{N} w_t^{(i)} \\ \Sigma_t &= \sum_{i=1}^{N} w_t^{(i)} \left(x_t^{(i)} - \mu_t\right)\left(x_t^{(i)} - \mu_t\right)^T \Big/ w_t^{(i)} \end{aligned} \quad \textbf{(14)}$$

where A^T denotes the transpose of matrix A. We now give the corollary to verify the convergence of the algorithm above. Let us note that the form of corollary here is similar to that in GPF [7], however the difference is that the one here is based on the CF.

Corollary 1: Assume that at time t, the prediction distribution is Gaussian, i.e. $p(x_t \mid Y_{t-1}) \approx \mathrm{N}(x_t; \bar{\mu}_t, \bar{\Sigma}_t)$. The GCF measurement updates the filtering distribution by the algorithm above. Then, μ_t computed in (14) converges to the MMSE estimate of x_t almost surely as $N \to \infty$, and the MMSE estimate given by Σ_t in (14) converges to the true MMSE estimate almost surely as $N \to \infty$.

Proof: 1) mean

$$\begin{aligned}
\mu_t &= \hat{E}(x_t \mid Y_t) = \frac{\sum_{i=1}^N x_t^{(i)} w_t^{(i)}}{\sum_{i=1}^N w_t^{(i)}} = \frac{\sum_{i=1}^N x_t^{(i)} \hat{p}\left(y_t \mid x_t^{(i)}\right)}{\sum_{i=1}^N \hat{p}\left(y_t \mid x_t^{(i)}\right)} \\
&\xrightarrow{x_t^{(i)} \sim p(x_t \mid Y_{t-1})} \frac{\int x_t \hat{p}(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t}{\int \hat{p}(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t} \\
&\approx \frac{\int x_t p(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t}{\int p(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t} \\
&= \frac{\int x_t p(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t}{p(y_t \mid Y_{t-1})} \\
&= \int x_t p(x_t \mid Y_t) dx_t = E(x_t \mid Y_t)
\end{aligned} \tag{15}$$

2) covariance

$$\begin{aligned}
\Sigma_t &= \hat{E}\left(\left(x_t - \hat{E}(x_t \mid Y_t)\right)\left(x_t - \hat{E}(x_t \mid Y_t)\right)^T \mid Y_t\right) \\
&= \frac{\sum_{i=1}^N \left(x_t^{(i)} - \mu_t\right)\left(x_t^{(i)} - \mu_t\right)^T w_t^{(i)}}{\sum_{i=1}^N w_t^{(i)}} \\
&= \frac{\sum_{i=1}^N \left(x_t^{(i)} - \mu_t\right)\left(x_t^{(i)} - \mu_t\right)^T \hat{p}\left(y_t \mid x_t^{(i)}\right)}{\sum_{i=1}^N \hat{p}\left(y_t \mid x_t^{(i)}\right)} \\
&\xrightarrow{x_t^{(i)} \sim p(x_t \mid Y_{t-1})} \frac{\int \left(x_t - E(x_t \mid Y_t)\right)\left(x_t - E(x_t \mid Y_t)\right)^T \cdot \hat{p}(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t}{\int \hat{p}(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t} \\
&\approx \frac{\int \left(x_t - E(x_t \mid Y_t)\right)\left(x_t - E(x_t \mid Y_t)\right)^T p(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t}{\int p(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t} \\
&= \frac{\int \left(x_t - E(x_t \mid Y_t)\right)\left(x_t - E(x_t \mid Y_t)\right)^T p(y_t \mid x_t) p(x_t \mid Y_{t-1}) dx_t}{p(y_t \mid Y_{t-1})} \\
&= \int \left(x_t - E(x_t \mid Y_t)\right)\left(x_t - E(x_t \mid Y_t)\right)^T p(x_t \mid Y_t) dx_t \\
&= E\left(\left(x_t - E(x_t \mid Y_t)\right)\left(x_t - E(x_t \mid Y_t)\right)^T \mid Y_t\right)
\end{aligned} \tag{16}$$

3.2. The Time Update

By substituting (13) to (3) we have

$$\begin{aligned}
p(x_{t+1} \mid Y_t) &= \int p(x_{t+1} \mid x_t) p(x_t \mid Y_t) dx_t \\
&\approx \int p(x_{t+1} \mid x_t) \mathrm{N}(x_t; \mu_t, \Sigma_t) dx_t
\end{aligned} \tag{17}$$

Draw samples

$$x_t^{(i)} \sim \mathrm{N}(x_t; \mu_t, \Sigma_t), \quad i = 1, \cdots, N \tag{18}$$

and then a Monte Carlo approximation of the predictive distribution is given by

$$p(x_{t+1} \mid Y_t) \approx \frac{1}{N} \sum_{i=1}^N p\left(x_{t+1} \mid x_t^{(i)}\right) \tag{19}$$

Obtain samples at time t+1 from the process model by

$$x_{t+1}^{(i)} \sim f_{t+1}(x_t^{(i)}, \cdot) \tag{20}$$

from which the mean and covariance of $p(x_{t+1} \mid Y_t)$ are computed as

$$\begin{aligned}
\bar{\mu}_{t+1} &= \frac{1}{M} \sum_{i=1}^N x_{t+1}^{(i)} \\
\bar{\Sigma}_{t+1} &= \frac{1}{M} \sum_{i=1}^N \left(x_{t+1}^{(i)} - \bar{\mu}_{t+1}\right)\left(x_{t+1}^{(i)} - \bar{\mu}_{t+1}\right)
\end{aligned} \tag{21}$$

Recall that the prediction distribution is approximated as a Gaussian, we have

$$p(x_{t+1} \mid Y_t) \approx \mathrm{N}(x_{t+1}; \bar{\mu}_{t+1}, \bar{\Sigma}_{t+1}) \tag{22}$$

3.3. Summary of the GCF

We summarize the algorithm above in Table 2.

The GCF does away with the need of resampling step, this means that the GCF is more amenable for fully parallel implementation in VLSI than the CF. Moreover, because of the use of convolution kernels the GCF can deal with scenarios that the observation noise is too small or the likelihood function can not be obtained analytically.

4. Tracking Simulation Results

An example: Consider the problem of tracking a maneuvering target [9], whose position and velocity at instant t are given by a continuous random vector $x_{2t} \in R^{n-1}$, and where the maneuver/regime of the target is represented by the discrete random variable $x_{1k} \in R$. The state to be estimated is $x_t = \{x_{1t} ; x_{2t}\}$. The model is as follows:

$$X_{2t} = Fx_{2t-1} + Bx_{1t} + w_t ;$$
$$Z_t = Cx_{2t} + e_t$$

Table 2. The GCF algorithm.

```
For  $t = 0$
  Given  $p(x_0) = \mathrm{N}(x_0; \mu_0, \Sigma_0)$
For  $t \ge 1$
  (1) Time update
    (i) Draw samples from posterior density
    $x_{t-1}^{(i)} \sim \mathrm{N}(x_{t-1}; \mu_{t-1}, \Sigma_{t-1}), \quad i = 1, \cdots, N$
    (ii) Draw samples from the process model
    $x_t^{(i)} \sim f_t(x_{t-1}^{(i)}, \cdot), \quad i = 1, \cdots, N$
    (iii) Compute the mean and covariance
    $\bar{\mu}_t = \frac{1}{M} \sum_{i=1}^{N} x_{t1}^{(i)}$
    $\bar{\Sigma}_t = \frac{1}{M} \sum_{i=1}^{N} (x_t^{(i)} - \bar{\mu}_t)(x_t^{(i)} - \bar{\mu}_t)^T$
  (2) Measurement update
  (i) Draw samples from importance density
    $x_t^{(i)} \sim \mathrm{N}(x_t; \bar{\mu}_t, \bar{\Sigma}_t), \quad i = 1, \cdots, N$
  (ii) Draw samples from the observation model
    $y_t^{(i)} \sim h_t(x_t^{(i)}, \cdot)$
  (iii) Compute and normalized the weights
    $w_t^{(i)} \propto K_{hn}^{z}(y_t - \hat{y}_t^{(i)})$
    $w_t^{(i)} = w_t^{(i)} \Big/ \sum_{i=1}^{N} w_t^{(i)}$
  (iv) Compute the mean and covariance
    $\mu_t = \sum_{i=1}^{N} w_t^{(i)} x_t^{(i)}$
    $\Sigma_t = \sum_{i=1}^{N} w_t^{(i)} (x_t^{(i)} - \mu_t)(x_t^{(i)} - \mu_t)$
```

Additionally, given

$$p(x_{1t} \mid x_{1t-1}) = \sqrt{x_{1t-1}^{\ 2} + 0.01t} + \theta_t, \quad \theta_t \sim N(0,10).$$

w_t and e_t are zero mean Gaussian noises, with covariance matrices Q and R. Since given X_{1t}, the dynamics of x_{2t} are linear-Gaussian. In our model, we use $x_{2t}=[x_t\ y_t\ x_t'\ y_t']^T$ where (x; y) is the position of the target in a cartesian plane. We take :

F=[1,0,0.3,0;0,1,0,0.3;0,0,1,0;0,0,0,1]T,

B=[1.25;-1.25;0.25;-0.25]T,

C=[1,0,0,0;0,1,0,0].

We use the simulation data as follows:

$\omega_t \sim N(0,(0.01,0;0,0.01))$, $e^t \sim N(0,R)$, $\hat{x}_{0|1}$ = (20, 30, 0.5, 0.5)^T where the measurement noise variance varies during simulation. Both GCF and GPF are adopted in the simulation. Figure 1 shows the absolute value of errors of the state estimates given by the GPF (denoted by asterisk-solid line) and GCF (denoted by dashed line) respectively when the observation noise variance R=5. In this case both the GPF and the GCF works well, also shown in Table 3. However, when R is reduced, e.g. R=0.1, the GPF diverges while the proposed GCF still works well, as shown in Table 3, where ϕ means the divergence of the results as shown in [8], and the RMSE is calculated according to

$$\sqrt{\sum_{i=1}^{t} (\hat{x}_i - x_i)^2 \Big/ N},$$

where $\hat{x}_i$ and x_i are the estimated and true values respectively, N denotes the total estimation times.

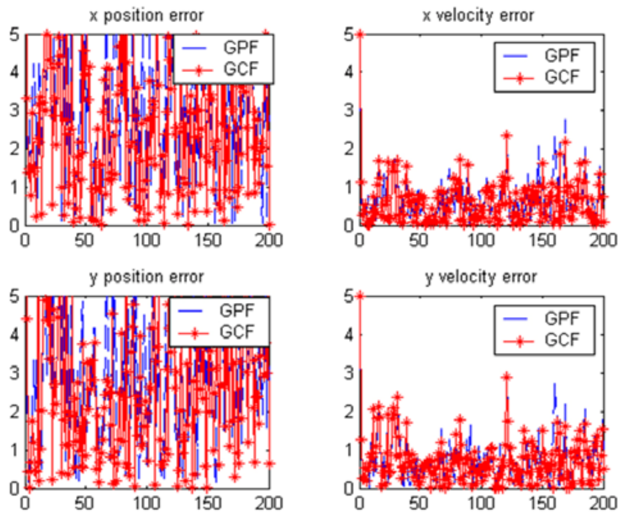

Figure 1. Absolute value of estimated error (R=5).

Table 3. Detailed simulation results.

meth-ods	*R*	x position RMSE	y position RMSE	x velocity RMSE	y velocity RMSE
GPF	10	8.691727	9.132639	43.681761	42.659641
	1	0.936217	0.898894	4.448537	5.993423
	0.1	ϕ	ϕ	ϕ	ϕ
GCF	10	8.777454	9.123733	46.752717	47.554381
	1	0.951737	0.901504	3.882222	5.185661
	0.1	0.189300	0.198540	1.274176	1.247551

5. Conclusions

The proposed Gaussian convolution filter (GCF) overcomes the drawbacks of the existing GPF, which is limited in the applications to scenarios that have non-noisy or near non-noisy observations or lack the knowledge of the likelihood function. Moreover the parallelizability of the GCF and the absence of resampling step makes it more convenient for VLSI implementation and, hence, feasible for practical real-time applications than the existing CF. Simulation results are also presented to demonstrate the performance of the GCF when the observation noise is small and the GPF fails.

6. References

[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," Transactions of the ASME–Journal

of Basic Engineering, Vol. 82-D, pp. 35–45, 1960.

[2] M. K. Steven, “Fundamentals of statistical signal processing,” PTR Prentice-Hall, Englewood Cliffs, N.J., 1993.

[3] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, “Novel approach to nonlinear/non-Gaussian Bayesian state estimation,” IEE Proceedings-F, Vol. 140, No. 2, pp. 107–113, 1993.

[4] A. Doucet, F. Nando, and N. J. Gordon, “Sequential Monte Carlo in practice,” Springer, New York, 2001.

[5] T. Schon, F. Gustafsson, and P. J. Nordlund, “Marginalized particle filters for mixed linear/nonlinear state-space models,” IEEE Transactions on Signal Processing 2005, Vol. 53, No. 7, pp. 2279–2289.

[6] J. J. Simon and K. U. Jeffrey, “Unscented filtering and nonlinear estimation,” Proceedings of the IEEE, Vol. 92, No. 3, pp. 401–422, 2004.

[7] J. H. Kotecha and P. M. Djuric, “Gaussian particle filtering,” IEEE Transactions on Signal Processing, Vol. 51, No. 10, pp. 2592–2601, 2003.

[8] V. Rossi and J.-P. Vila, “Nonlinear filter in discreet time: A particle convolution approach,” Biostatic Group of Monetepellier, Technical Report 04-03, 2004. http://vrossi.free.fr/recherche.html.

[9] F. Mustiere, M. Bolic, and M. Bouchard, “A modified Rao-blackwellised particle filter[C],” IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2006, Toulouse, France, Vol. 3, pp. 21–24, May 2006.