Commun Nonlinear Sci Numer Simulat 31 (2016) 69–75

Contents lists available at ScienceDirect

Commun Nonlinear Sci Numer Simulat

journal homepage: www.elsevier.com/locate/cnsns

Classifying the weights of particle filters in nonlinear systems

Mahtab Sadat Sharifian [a], Abdolah Rahimi [b,*], Naser Pariz [a]

[a] Electrical and Robotic Department, Ferdowsi University of Mashhad, Mashhad, Iran
[b] Electrical and Robotic Department, Shiraz University of Technology, Shiraz, Iran

ARTICLE INFO

Article history:
Received 7 February 2015
Revised 5 May 2015
Accepted 23 May 2015
Available online 9 July 2015

Keywords:
Particle impoverishment
Particle filter
Fission particle

ABSTRACT

Among other methods, state estimation using filters is currently used to increase the accuracy of systems. These filters are categorized into the following three branches: linear, nonlinear with Gaussian noise, and nonlinear with non-Gaussian. In this paper, the performance of particle filters is investigated via nonlinear, non-Gaussian methods. The main aim of this paper was to improve the performance of particle filters by eliminating particle impoverishment. A resampling step was proposed to overcome this limitation. However, resampling usually leads to a dearth of samples. Therefore, to virtually increase the number of samples, the available samples were broken into smaller parts based on their respective weights via a clustering approach. The experimental results indicate that the proposed procedure improves the accuracy of state estimations without increasing computational burden.

1. Introduction

The majority of physical systems, such as thermal systems, astronomical systems, and weather and chaotic systems, demonstrate nonlinear behavior. Accordingly, nonlinear systems are of increasing research interest [1-2]. In particular, several studies have predicted and estimated parameters of nonlinear systems with the aim of increasing the accuracy of and removing noise from the system. The Kalman filter is used to estimate the parameters of linear systems, whereas extended Kalman filter (EKF) and unscented Kalman filter (UKF) are used to estimate the parameters of nonlinear systems with Gaussian noise. Finally, particle filters are used to estimate the parameters of nonlinear systems with non-Gaussian noise.

Particle filter methods use Monte Carlo approaches to estimate parameters. These methods have been increasingly studied for their simplicity of implementation, particularly in the fields of image processing, automatic control, navigation, etc. They reduce computational complexity and increase the accuracy of estimations by choosing reliable samples from several available samples [3-11]. It should be noted that the Bayesian estimation algorithm is used in the Monte Carlo procedure to estimate the prior probability density function for $k = 1,2,\ldots$, instead of the state x_k [12–14].

In particle filters, each generated particle is assigned a certain weight, which indicates the value of the generated sample in state estimation. The variance of the weights over time tends to zero. This phenomenon is called "particle degeneracy." In this study, a resampling step is applied to solve this problem. Several different resampling steps are presented in the literature [15-17]. However, resampling in particle filter methods might sometimes lead to sample impoverishment, as heavier samples with higher weights will be chosen many times while lighter particles will be removed from the sample pool after several steps. The EKFPF filter, which uses both a particle filter and an EKF, is used to solve this problem. The EKFPF approach uses the EKF to estimate the prior probability density function, thus overcoming particle impoverishment to some extent [18]. Particle impoverishment

* Corresponding author. Tel.: +98 9134112916.
E-mail address: ab.rahimi@sutech.ac.ir (A. Rahimi).

http://dx.doi.org/10.1016/j.cnsns.2015.05.021
1007-5704/

can also be overcome with the fission bootstrap particle filter (FBPF) [19–21]. This filter breaks particles into smaller samples, with a new weight attributed to each particle. Moreover, several classic approaches to overcoming particle impoverishment have been proposed in the literature, such as adaptive particle filters (APFs) [22–25], FBPFs [15], auxiliary particle filters [26], and UKF-based particle filters [27] . In addition, several methods based on artificial neural networks and fuzzy systems have been used in studies [28–29] to overcome particle impoverishment with low computational burden [30–31]. The paper focuses on the classification of particles into three groups by their weights, based on which decisions are made. Hopefully, this approach will increase estimation accuracy without decreasing its computational speed.

The paper is organized as follows: particle filters are introduced in Section 2. The taxonomy of particle filters is presented in Section 3. The simulation results are reported in Section 4. Finally, the concluding remarks are presented in Section 5.

2. Particle filter

Particle filters are generally used to estimate a random signal in a time period. The signals are estimated by sampling and each sample is called a particle, hence the term "particle filters" [25,32]. The dynamic equation of the system is sampled, and the obtained samples are weighted using the measurement equation. Then, based on this set of samples and their weights, the random signal is optimally estimated. However, particle degeneracy is observed during sampling and signal estimation. The resampling algorithm is used to overcome this phenomenon. Generally, the simplified form of the equations for particle filters is as follows:

$$\begin{aligned} &x\_(k+1) = f(x_k, v\_k) \\ &y_k = h(x_k) + w\_k \end{aligned} \tag{1}$$

where w and v represent the non-Gaussian noise of the process with covariances of R and Q, respectively. Further, f(.) and h(.) are the nonlinear functions of the system, $x_k \in R^n$ the system states, and $y_k \in R^m$ the outputs of the system measured at the kth step. Provided that all states and values observed at step t are available, the estimated states are calculated as follows (Eq. (2)):

$$p(x_{1:t}|y_{1:t}) = \frac{p(y_t|x_t)P(x_t|y_{t-1})}{p(y_t|y_{1:t-1})}. \tag{2}$$

Considering the probability theory,

$$p(x_t|y_{t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1} \tag{3}$$

In general, the abovementioned integral cannot be easily calculated because the dimensions of the state space are large. Monte Carlo simulation can be used to resolve this limitation. In this approach, instead of calculating the integral at all points, its value is calculated by sampling the points with the greatest share in the value of the integral. To calculate the integral

$$I = \int h(x)dx \tag{4}$$

it is decomposed into h(x)=f(x).π(x), where $\pi(x)$ denotes a probability function that satisfies the following conditions: $\int \pi(x)dx = 1, \pi(x) \geq 0$. If N samples are taken from this function, the Monte Carlo estimation of the integral will be equal to the sum of the values of function f(.) for the obtained samples:

$$I_N = \sum_{i=1}^{N} f(x^{[i]}) \tag{5}$$

The values of the system states are calculated according to Eq. (6):

$$x_t = E(f(x_{t-1}, v_{t-1})) = \frac{1}{N}\sum_{i=1}^{N} f\left(x_{t-1}^{[i]}, v_{t-1}\right). \tag{6}$$

If samples cannot be taken from the probability density function, another samplable function with similar properties to the density function of interest is used. After several calculations and simplifications, the final equation is calculated as Eq. (7):

$$x_t = E(f(x_{t-1})) = \frac{\frac{1}{N}\sum_{i=1}^{N} f(x_{t-1}^{[i]})w_{t(x_t)}}{\frac{1}{N}\sum_{i=1}^{N} w_{t(x_t)}} \tag{7}$$

where W's denote the importance weights of the system, which are calculated based on the following recursive equation:

$$w_t = w_{t-1}\frac{p(y_t|x_t^{[i]})p(x_t|x_{t-1}^{[i]})}{q(x_t^{[i]}|x_{0:t-1}^{[i]}, y_{1:t})} \tag{8}$$

where the probability density function, q, is a samplable density function. Then, the obtained weights are normalized using Eq. (9):

$$\tilde{w}_t^i(x_{0:t}) = \frac{w_t^i(x_{0:t})}{\sum_{j=1}^{N} w_t^j(x_{0:t})} \tag{9}$$