CONTINUOUS-TIME NONLINEAR ESTIMATION FILTERS USING UKF-AIDED GAUSSIAN SUM REPRESENTATIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MURAT GOKCE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JANUARY 2014

Approval of the thesis:

CONTINUOUS-TIME NONLINEAR ESTIMATION FILTERS USING UKF-AIDED GAUSSIAN SUM REPRESENTATIONS

submitted by **MURAT GÖKCE** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronics Engineering Department, Middle East Technical University** by

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Gönül Turhan Sayan _____
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. Mustafa Kuzuoğlu _____
Supervisor, **Electrical and Electronics Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Mübeccel Demirekler _____
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Mustafa Kuzuoğlu _____
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Kemal Leblebicioğlu _____
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Asım Egemen Yılmaz _____
Electrical and Electronics Engineering Dept., Ankara University

Assoc. Prof. Dr. Umut Orguner _____
Electrical and Electronics Engineering Dept., METU

Date: 21.01.2014

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Murat Gökce

Signature :

ABSTRACT

CONTINUOUS-TIME NONLINEAR ESTIMATION FILTERS USING UKF-AIDED GAUSSIAN SUM REPRESENTATIONS

Gökce, Murat
Ph.D., Department of Electrical and Electronics Engineering, METU
Supervisor: Prof. Dr. Mustafa Kuzuoğlu

January 2014, 71 pages

A nonlinear filtering method is developed for continuous-time nonlinear systems with observations/measurements carried out in discrete-time by means of UKF-aided Gaussian sum representations. The time evolution of the probability density function (pdf) of the state variables (or the *a priori* pdf) is approximated by solving the Fokker-Planck equation numerically using Euler's method. At every Euler step, the values of the *a priori* pdf are evaluated at deterministic sample points. These values are used with Gaussian radial basis functions to obtain weighted sum of Gaussian approximation of *a priori* pdf. The locations of the sample points and mean and covariance values of Gaussian functions are found by the help of the prediction step of an Unscented Kalman Filter (UKF). The weights of the Gaussian functions are calculated using the method of least squares. The pdf of the updated state variables (or *a posteriori* pdf) is approximated similar to *a priori* case. This time Bayes rule and the help of the update step of UKF are used. In the developed filter, UKF acts as a one step look ahead mechanism to determine the high likelihood regions of the *a priori* and *a posteriori* pdfs and these pdfs are locally approximated around these high likelihood regions. As a second filtering method, particle flow is combined with UKF-aided Gaussian sum representations approach. Both filters are compared

with some of the known nonlinear filtering methods by means of computational load and error levels using various scenarios

Keywords:Continuous-time Systems; Nonlinear Filtering; Fokker-Planck Equation; Gaussian sum; Numerical Methods

ÖZ

UKF YARDIMLI GAUSS TOPLAMI GÖSTERİMİ KULLANAN SÜREKLİ ZAMAN LİNEER OLMAYAN KESTİRİM FİLTRELERİ

Gökce, Murat
Doktora., Elektrik ve Elektronik Mühendisliği Bölümü, ODTÜ
Tez Yöneticisi: Prof. Dr. Mustafa Kuzuoğlu

Ocak 2014, 71 sayfa

Bu çalışmada sürekli zaman lineer olmayan ve kesikli zamanda ölçümler içeren sistemler için UKF yardımlı Gauss toplamı yöntemi kullanan lineer olmayan bir filtre geliştirilmiştir.Durum değişkenlerinin olasılık yoğunluk fonksiyonunun(oyf) zaman devinimi(ya da önsel oyf) Fokker-Planck denkleminin Euler yöntemi kullanarak nümerik çözümü ile yaklaşık hesaplanmaktadır.Her Euler adımında önsel oyf'nun değeri belirli örnek noktalar için hesaplanmaktadır Bu değerlerGauss radyal fonksiyonları ile birlikte kullanılarak önsel oyf'nun ağırlıklı Gauss toplamyaklaşımı elde edilmektedir. Örnek noktaların konumu ve Gauss fonksiyonlarının ortalama ve kovaryans değerleri Kokusuz Kalman Filtre(UKF)'nin tahminleme adımı yardımıyla bulunmaktadır. Gauss fonksiyonlarının ağırlıkları en az karaler yöntemi ile hesaplanmaktadır. Durum değişkenlerinin güncellenmiş oyf'si(ya da sonsal oyf) de önsel oyf'ye benzer şekilde yaklaşık hesaplanmaktadır. Bu durumda Bayes kuralı ve UKF'nin güncelleme adımının yardımı kullanılmaktadır. Geliştirilen filtrede UKF önsel ve sonsal oyf'lerin yüksek olasılıklı bulunabilceği bölgelerin belirlenmesi için bir adım ötesini tahminleme mekanizması olarak kullanılmaktadır. Bahsedilen öyf'ler bu yüksek olasıklı bölgeler çevresinde yaklaşık olarak modellenmektedir. İkinci bir filtreleme yöntemi olarak parçaçık akışı UKF yardımlı Gauss toplam yöntemi ile birleştirilmiştir.Her iki filtre de bazı bilinen lineer olmayan filtreleme

yöntemleri ile hesaplama yükü ve hata seviyeleri açısından çeşitli senaryolar kullanılarak karşılaştırılmıştır

Anahtar Kelimeler: Sürekli Zaman Sistemleri; Lineer Olmayan Filtreleme; Fokker-Planck Denklemi; Gauss Toplamı; Nümerik Metotlar

To my family

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

The general problem of nonlinear filtering can be handled for systems that are represented either in continuous-time or discrete-time. Continuous-time systems are more general and can be approximated by discrete-time systems using suitable discretization methods. Except for a few special case there is no exact (i.e. optimal) nonlinear filters in continuous-time and there are several approximate filtering methods applied to both continuous-time and discrete-time systems in the literature.

The purpose of the thesis is to present the nonlinear filtering problem and to introduce new approximate approaches by means of UKF-aided Gaussian sum representations to the solution of the problem. Also these new solutions are compared with some of the known nonlinear filtering methods using various scenarios.

1.1 The Filtering Problem

The main focus of the nonlinear filtering problem is the estimation of the values of the states of a system at a specific time using noisy observations extracted from that system. The states can be modeled as stochastic processes, and the time evolution of these states can be expressed in terms of an ordinary stochastic differential equation represented in Ito form as;

$$dx = f(x, t)dt + G(x, t)d\beta(t) \tag{1.1}$$

with the initial condition, $x(t_0)=x_0$. This generic equation defines the dynamical model of the system in continuous-time, where the independent variable t represents time, and $t_0$denotes the initial time instant.$x(t) \in R^n$ denotes the markovian stochastic process representing the state vector, $f(x,t)\epsilon R^n$ is the drift function, $G(x, t)\epsilon R^{nxm}$ is a state dependent matrix and ß(t) ϵ R^m is the Wiener process and represents the process noise. The initial state x_0 is assumed to be a random vector with a known, not necessarily Gaussian pdf $p(x, t_0)$.

Measurements are generally obtained at specific time instants (which are also called sampling time instants). The relationship between the measurements and the states is given by a measurement function which is corrupted with a measurement noise. The measurement process can be represented by the following discrete-time measurement model;

$$y(t_k)=h(x(t_k), t_k) + v(t_k) \tag{1.2}$$

where $y \epsilon R^p$ is a vector of measurements, $h \epsilon R^p$ is the measurement function and depends on the state x and time t, $t_k$denotes the k-th sampling timeand $v \epsilon R^p$ is a zero-mean white measurement noise. It is assumed that the random variables x_0, ß and v are mutually independent. In order to simplify the notation, we will write x_k for $x(t_k)$, y_k for y(t_k), etc.

Estimation of the state values using these models involves two steps, namely prediction and update steps

When the initial state represented by the random vector x_0 at t_0 evolves up to the instant t_k using (1.1), it is represented with a new random vector x_k and its pdf evolves from $p(x, t_0|y_0)$to $p(x, t_k|y_0)$. The time evolved pdf $p(x, t_k|y_0)$isknown as *a priori* pdf at time t_k and the *a priori* pdf is found in the prediction step. If the increments of Wiener process in (1.1) areinfinitely divisible [6], the time evolution

of the state pdf for the above dynamical model can be given in terms of the Fokker-Planck equation expressed as

$$\frac{\partial p(x,t)}{\partial t} = -\frac{\partial p(x,t)}{\partial x} f(x,t) - p(x,t) Tr\left(\frac{\partial f(x,t)}{\partial x}\right) + \frac{1}{2} Tr\left(\frac{\partial^2 G Q_\beta G^T p(x,t)}{\partial x^2}\right) \quad (1.3)$$

Starting with an initial pdf, solution of the Fokker-Planck equation gives the *a priori* pdf.

In the update step, the *a priori* pdf is updated with the information coming from the current measurement to refine the state pdf. This improved state pdf is named as the *a posteriori* pdf. By using the *a priori* pdf at t_k, the *a posteriori* pdf $p(x, t_k|y_k)$ at t_k can be calculated by Bayes' rule and its value is proportional to the product of measurement *likelihood* and *a priori* pdf

$$p(x, t_k|y_k) \sim p(y_k | x, t_k).p(x, t_k|y_0) \quad (1.4)$$

The filtering problem tries to find these pdfs at every sampling time by prediction and update steps. When the *a posteriori* pdf is found at a sampling time, one can make estimations about the state (such as mean, covariance etc.) at that sampling time using the *a posteriori* pdf.

For the case where the dynamical and measurement models are linear, the filtering problem is easy to solve and the *a priori* and *a posteriori* pdfs can be found by using the Continuous-Discrete Kalman Filter [1].

However, in a nonlinear system representation, these pdfs cannot be found so easily as in the linear case. For a nonlinear dynamical system, Fokker-Planck equation cannot be analytically solved using a finite number of parameters except for a few special cases. Also, normalization problems may occur even though

analytical solutions are available or after Bayes' rule is applied using a nonlinear measurement model.

To overcome these diffuculties many approximation methods were suggested in literature. Examples of them are given in the Chapter 2.

In this thesis we devoloped a new approach for the approximate solution of the nonlinear filtering problem by means of UKF-aided Gaussian sum represantations. In our approach the *a priori* and the *a posteriori* pdfs are approximated by weighted Gaussian sums inside valid search regions These regions (or grids) are found by the help of prediction and update steps of an UKF. Number of Gaussians and their mean and covariance values are deterministically found in these regions. Weights of Gaussians are found by numerical solution of Fokker Planck equation for *a priori* pdf and by appliying Bayes' rule for *a posteriori* pdf. The developed approach has ability to approximate the whole parts of the pdfs or some parts of the pdfs close to the mean by adjusting the size of search regions.

Also as a second approach particle flow is combined with UKF-aided Gaussian sum representations. Here *a priori* pdf is approximated as in the first approach. But the mean of the *a posteriori* pdf is estimated using the flow of particles

The details of the developed approaches are given in Chapter 3. Comparison of the developed approaches with some of the exisiting nonlinear filtering algorithms is given in Chapter 4. Conslusions are drawn in Chapter 5

.

CHAPTER 2

EXAMPLES OF NONLINEAR FILTERING ALGORITHMS

As mentioned in Chapter 1, except for a few special cases there is no exact(i.e optimal) filtering methods for nonlinear systems. In [4], nonlinear filtering methods for some of these special cases are given. In order to overcome these difficulties, many approximation methods were suggested for nonlinear filtering [2], [3]. A comparison of some of these methods can be found in [5]. As mentioned in [3], two types of approximation methodologies exist in the literature. One tries to simplify the models and the other tries to find a global approximation for the pdfs. In this chapter a few of the existing filters using these methodologies are described. Some of them are also used in simulations for comperisons.

2.1 Converted Measurement Kalman Filter

Converted Measurement Kalman Filter(CMKF) is a popular technique that is used in radar tracking problems. In CMKF, a linear dynamical system is used with a nonlinear measurement model and the method tries to simplify or linearize the nonlinear measurement model.

The dynamical system can be modeled in continuous-time or discrete-time as,

$$dx = f(x, t)dt + G(x, t)d\beta(t) \tag{2.1}$$

for continuos-time which is given in Chapter 1 as (1.1) or

$$x_{k+1} = F(t_k)\, x_k + G(t_k) w_k \tag{2.2}$$

for discrete-time. w is a zero-mean Gaussian white process noise with covariance Q_w. Here *f(x,t)* and *G(x,t)* are linear functions i.e $f(x,t) = f(t)x(t)$ and $G(x,t) = G(t)$

In 2D radar tracking problem state space can be in Cartesian coordinates with 4 dimensions

$$x = \begin{bmatrix} x & v_x & y & v_y \end{bmatrix} \tag{2.3}$$

where *x, y* are target positions and v_x, $v_y$are target velocities.

For the given state space the nonlinear measurement model can be chosen in discrete-time as the 2D radar measurement model in polar coordinates.

$$y_k = h(x_k, t_k) + v_k = h(x_k) + v_k = \begin{bmatrix} r_k \\ \beta_k \end{bmatrix} + v_k = \begin{bmatrix} \sqrt{x_k^2 + y_k^2} \\ \tan^{-1} \dfrac{y}{x} \end{bmatrix} + \begin{bmatrix} v_r(k) \\ v_\beta(k) \end{bmatrix} \tag{2.4}$$

where *r* and β are the range and azimuth and the measurement noise *v* are assumed to be independent Gaussian white noises. $v_r$is the range error and v_β is the azimuth error.

In CMKF, the measurement model is linearized by converting measurements from Polar coordinates to Cartesian coordinates. Also measurement noises are converted into Cartesian coordinates and the mean and covariance of the converted errors are given as close as to their true values. Whether the measurement noise is Gaussian or non-Gaussian, the mean and covariance of converted errors are assumed to belong to a Gaussian one.

After the conversion, both the dynamical system and the measurement model become linear which makes the filtering steps easy to solve. For the continuous-time dynamical system, the prediction and update steps are calculated as in the Continuous-Discrete Kalman Filter. For the discrete-time dynamical system, these steps are calculated using the standard Kalman Filter steps.

Now we can give how the conversion is done. There exist debiased or unbiased conversions for CMKF in literature[7][8]. We will give an improved version of unbiased conversion for CMKF[9] here. Using the CMKF with 3D radar measurements can be found in [8][9].

Conversion of 2D Radar Measurements

Let $\begin{bmatrix} r_k \\ \beta_k \end{bmatrix}$ are the measured range and azimuth values at k-th sampling time. Range and azimuth errors are $v_r(k)$ and $v_\beta(k)$ having zero mean and variances σ_r^2 and σ_β^2.

Define two compansation factors as

$$\lambda_\beta = E\left[\cos v_\beta\right]$$
$$\lambda_\beta^{'} = E\left[\cos 2v_\beta\right] \qquad (2.5)$$

Errors can be Gaussian or non-Gaussian but these compansation factors are valid for both cases. Only the value of the compansation factors are changed for Gaussian or non-Gaussian errors.

If the range and azimuth errors are Gaussian the result of expectations will be as follows

$$\lambda_\beta = E\left[\cos v_\beta\right] = e^{-\sigma_\beta^2/2}$$
$$\lambda_\beta^{'} = E\left[\cos 2v_\beta\right] = e^{-2\sigma_\beta^2} = \lambda_\beta^4 \tag{2.6}$$

Using these compensation factors converted measurements will be

$$x_{c_k} = \lambda_\beta^{-1} r_k \cos\beta_k \qquad y_{c_k} = \lambda_\beta^{-1} r_k \sin\beta_k \tag{2.7}$$

The mean of the converted measurements errors for x_k and y_k will be

$$\mu_{x_k} = (\lambda_\beta^{-1} - \lambda_\beta) r_k \cos\beta_k$$
$$\mu_{y_k} = (\lambda_\beta^{-1} - \lambda_\beta) r_k \sin\beta_k \tag{2.8}$$

The elements of the covariance matrix of the converted measurements errors for x_k and y_k will be

$$R_{xx_k} = -\lambda_\beta^2 r_k^2 \cos^2\beta_k + \frac{1}{2}\left(r_k^2 + \sigma_r^2\right)\left(1 + \lambda_\beta^{'}\cos 2\beta_k\right)$$
$$R_{yy_k} = -\lambda_\beta^2 r_k^2 \sin^2\beta_k + \frac{1}{2}\left(r_k^2 + \sigma_r^2\right)\left(1 + \lambda_\beta^{'}\cos 2\beta_k\right) \tag{2.9}$$
$$R_{xy_k} = R_{yx_k} = -\lambda_\beta^2 r_k^2 \sin\beta_k \cos\beta_k + \frac{1}{2}\left(r_k^2 + \sigma_r^2\right)\left(1 + \lambda_\beta^{'}\sin 2\beta_k\right)$$

Since the mean of the converted errors are not zero, the converted measurements are biased. To get the unbiased converted measurements we extract the mean of the converted errors form the converted measurements.

$$\begin{bmatrix} x_{u_k} \\ y_{u_k} \end{bmatrix} = \begin{bmatrix} x_{c_k} - \mu_{x_k} \\ y_{c_k} - \mu_{y_k} \end{bmatrix} \tag{2.10}$$

After the conversion the linear measurement model is given as

$$y_k = Hx_k + c_k \tag{2.11}$$

where $x = \begin{bmatrix} x_{u_k} & v_x & y_{u_k} & v_y \end{bmatrix}$, $H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$ and c_k is the converted measurement error which is assumed to be Gaussian with zero mean and covariance $R_{c_k} = \begin{bmatrix} R_{xx_k} & R_{xy_k} \\ R_{yx_k} & R_{yy_k} \end{bmatrix}$

2.2 Unscented Kalman Filter

The Unscented Kalman Filter (UKF) is an easy and popular method which is in the catagory of the methods that try to find a global approximation to the pdf. The UKF is firstly published by Julier and Uhlman in 1997[10]. In the prediction and update steps of filtering, UKF applies Unscented Transform to the nonlinear dynamical and measurement models. Unscented Transform allows to find the mean and covariance of a random variable after that variable enters to a nonlinear function. Also the found mean and covariance values captures the true mean and covariances accurately to the 2nd order of nonlinearity, with errors only introduced in the 3rd and higher orders. To use with Kalman Filter, an assumtion is done in Unscented Transform. It is assumed that the random variable after the transformation is Gaussian. This assumption allows to use Kalman Filter steps to calculate the *a priori* and *a posteriori* pdfs. By doing so the resulting *a priori* and *a posteriori* pdfs are approximated with a Gaussian pdf in UKF.

Firstly we will give the Unscented Transform. Then we will give how the prediction and update steps are done in UKF using Unscented Transform.

The Unscented Transform

Let x is a n-dimensional random variable with mean $m_x$and covariance P_x. Consider that random variable x, enters to a nonlinear function g: $R^n \rightarrow R^m$.

$$y=g(x) \tag{2.12}$$

To find mean and covariance of the random variable y we first get 2n+1 sigma points using $m_x$and $P_x$in a deterministic way. These sigma points are samples from the pdf of random variable x. Nonlinear function is applied to each of the sigma points and we get transformed sigma points. Using these transformed sigma points mean m_y and covariance P_y of the random variable y are estimated. The sigma points are selected such that the values of m_y and P_y can be estimated from these transformed sigma points as accurately as possible. The procedure of finding sigma points and estimating mean and covariance of y is as follows;

1. Find the 2n+1 sigma points using $m_x$and P_x

$$\begin{aligned} x_0 &= m_x \\ x_i &= m_x + \left(\sqrt{(n+\lambda)P_x}\right)_i \qquad i = 1.....n \\ x_{i+n} &= m_x - \left(\sqrt{(n+\lambda)P_x}\right)_i \qquad i = 1.....n \end{aligned} \tag{2.13}$$

where the index i means the ith column of the expression in brackets.

2. Find the weights of the sigma points

$$\begin{aligned} W_0^m &= \frac{\lambda}{n+\lambda} \\ W_0^c &= \frac{\lambda}{n+\lambda} + \left(1-\alpha^2+\beta\right) \\ W_i^m &= W_i^c = \frac{\lambda}{2(n+\lambda)} \qquad i = 1.....2n \end{aligned} \tag{2.14}$$

λ is a scaling parameter defined as $\lambda = \alpha^2\left(n+\kappa\right)-n$

The constants α, β and κ are used as parameters of the method.

3. Transform the sigma points using the nonlinear function

$$y_i=g(x_i) \qquad i=0.....2n \tag{2.15}$$

4. Estimate the mean and covariance of y using the transformed sigma points.

$$\begin{aligned} m_y &\approx \sum_{i=0}^{2n} W_i^m y_i \\ P_y &\approx \sum_{i=0}^{2n} W_i^c \left(y_i - m_y\right)\left(y_i - m_y\right)^T \end{aligned} \tag{2.16}$$

Filtering Steps for Unscented Kalman Filter

The dynamical system can be modeled as,

$$dx = f(x, t)dt + G(x, t)d\beta(t) \tag{2.17}$$

for continuos-time which is given in Chapter 1 as (1.1) or

$$x_{k+1} = f(x_k, t_k) + G(x_k, t_k)w_k \tag{2.18}$$

for discrete-time.

Measurements are obtained in discrete-time as:

$$y_k = h(x_k, t_k) + v_k \tag{2.19}$$

$d\beta/dt$ is a continuous zero-mean Gaussian white process noise with intensity $Q(t)=Q_\beta$

w is a zero-mean Gaussian white process noise with covariance Q_w

v is a zero-mean Gaussian white measurement noise with covariance R

The filtering steps in UKF at the k-th sampling time are as follows;

Prediction Step

1. Find the 2n+1 sigma points $x_i(k-1)$ using $m_x(k-1)$ and $P_x(k-1)$

2. For the discrete-time dynamical system transform the sigma points using the noise free nonlinear function

$$x_i^-(k) = f\left(x_i(k-1), t_{k-1}\right) \qquad i = 0.....2n \tag{2.20}$$

For the continuous-time dynamical system integrate each of the sigma point through the noise free dymanical system $\frac{dx}{dt} = f(x,t)$. The integration can be done using numerical methods(Euler i.e.)

Find the predicted mean and covariance using the integrated or transformed sigma points.

$$\begin{aligned}
&m_x^-(k) = \sum_{i=0}^{2n} W_i^m x_i^- \\
&\widetilde{P}_x(k) = \sum_{i=0}^{2n} W_i^c \left(x_i^-(k) - m_x^-(k)\right)\left(x_i^-(k) - m_x^-(k)\right)^T \\
&P_x^-(k) = \widetilde{P}_x(k) + G\left(m_x(k-1), t_{k-1}\right) Q_w G\left(m_x(k-1), t_{k-1}\right)^T \text{ for discrete - time dynamical model} \\
&P_x^-(k) = \widetilde{P}_x(k) + G\left(m_x(k-1), t_{k-1}\right) Q_\beta T G\left(m_x(k-1), t_{k-1}\right)^T \text{ for continuous - time dynamical model}
\end{aligned} \tag{2.21}$$

where T is the sampling period

Update Step

3. Find the new 2n+1 sigma points $x_i(k)$ using $m_x^-(k)$ and $P_x^-(k)$

4. Transform the sigma points using the nonlinear measurement model

$$y_i(k) = h\left(x_i(k), t_k\right) \qquad i = 0.....2n \tag{2.22}$$

5. Find the mean m_y and covariance S_y and cross covariance C_{xy} using the transformed sigma points.

$$\begin{aligned} m_y(k) &= \sum_{i=0}^{2n} W_i^m y_i(k) \\ \widetilde{S}_y(k) &= \sum_{i=0}^{2n} W_i^c \left(y_i(k) - m_y(k)\right)\left(y_i(k) - m_y(k)\right)^T \\ S_y(k) &= \widetilde{S}_y(k) + R \\ C_{xy}(k) &= \sum_{i=0}^{2n} W_i^c \left(x_i(k) - m_x^-(k)\right)\left(y_i(k) - m_y(k)\right)^T \end{aligned} \tag{2.23}$$

6. Compute the filter gain K_k, and estimate the mean and covariance of state x at sampling time k as in the Kalman Filter

$$\begin{aligned} K_k &= C_{xy}(k).S_y(k)^{-1} \\ m_x(k) &= m_x^-(k) + K_k\left[y_k - m_y(k)\right] \\ P_x(k) &= P_x^-(k) - K_k S_y(k) K_k^T \end{aligned} \tag{2.24}$$

The steps above summarize the filtering steps for UKF. Also a continuous-discrete UKF version that can be used with continuous-time dynamical systems with discrete-time measurement models is given in [11]. Only the prediction step

of continuous-discrete UKF differs from the one given here. But the update step is same as here.

2.3 Sequential Monte Carlo Methods

In Sequential Monte Carlo(SMC) method a probability density function is represented in discrete-time with random samples taken from that pdf. If we can directly sample from the pdf this becomes perfect sampling. Many times, especially as the pdf propagates through nonlinear systems, perfect sampling is not possible and an auxiliary known density (which is named as importance density) is aided for sampling from the original density. This is called as Importance Sampling. In sampling process also every sample has associated with a weight.

In the filtering steps of SMC method, samples are mainly time evaluated using the dynamical system to find the *a priori* pdf and their weight are updated using measurements to find the *a posteriori* pdf. This procedure is called as Sequential Importance Sampling. As the number of samples increases, better approximations for the pdfs can be obtained. But this situation increases the computational load. As time passes, some of the samples can be less important and their weights can be very small. As a result of this,the divergence problem arouses. In this case, a resampling procedure is applied and samples with small weights are discarded. This kind of filtering is called as Sequential Importance Sampling with Resampling. But resampling can also yield sample impoverishment problem.

The details of SMC Method is given in [3]. As mentioned there, some methodologies exist in literature to reduce the effects of divergence and sample impoverishment. But choosing a good importance density function can reduce the negative effects of divergence and sample impoverishment and is important for a good approximation. In its easiest case, transition probability density function is chosen as importance density which is known as Particle Filter. But more efficient

importance densities can be chosen. For example; a density calculated using an Unscented Kalman Filter is used as importance density in Unscented Particle Filter. Also other methods can be used for choosing importance density but we will give the details of Particle Filter and Unscented Particle Filter here. Particle Filters can be found in [3][12]. Unscented Particle Filter is given in [12].

Now we can give the details of SMC method along with the Particle Filter and Unscented Particle Filter based on the expressions given in [3][12]

2.3.1 Perfect Sampling

Let us have a known probability density function, for example the *a posteriori* pdf. Using the Markov property, we can define the *a posteriori* pdf as

$$p(x, t_k|y_k) = P(X_k|Y_k) \text{ where } X_k=\{ x_k,x_{k-1}\ldots\} \text{ and } Y_k=\{ y_k,y_{k-1}\ldots\} \tag{2.25}$$

As an objective ,let's say we want to make estimations on the form

$$I\big(g(X_k)\big) = E\big[g(X_k)\big] = \int g(X_k)P(X_k \mid Y_k)dX_k \tag{2.26}$$

Also let us assume that it is possible to easily sample from the *a posteriori* pdf. Using the samples (or particles), we can represent the *a posteriori* pdf in discrete-time as

$$\hat{P}(X_k \mid Y_k) = \frac{1}{N}\sum_{i=1}^{N}\delta(X_k - X_k^i) \tag{2.27}$$

where X_k^i are particles sampled from the *a posteriori* pdf, N is the number of particles and $1/N$ is the weight of each particle.

Using the representation of pdf with particles, we can easily obtain the estimation as

$$\hat{I}\big(g(X_k)\big) = \int g(X_k)\hat{P}(X_k \mid Y_k)dX_k = \frac{1}{N}\sum_{i=1}^{N} g(X_k^i) \tag{2.28}$$

This estimate is unbiased and $\hat{I}\big(g(X_k)\big) \to I\big(g(X_k)\big)$ as $N \to \infty$

The problem is that we cannot sample directly from the *a posteriori* density for nonlinear systems. One solution to this problem is Importance Sampling using an importance density function.

2.3.2 Importance Sampling

In Importance Sampling, particles are sampled from an importance density $\pi(X_k \mid Y_k)$ that is easy to sample instead of the more complex *a posteriori* pdf. Importance density is needed to be chosen such that its domain encapsulates the domain that is to be approximated.

Using the importance density, the estimation can be rearranged as

$$I\big(g(X_k)\big) = \int g(X_k)\frac{P(X_k \mid Y_k)}{\pi(X_k \mid Y_k)}\pi(X_k \mid Y_k)dX_k \tag{2.29}$$

Here importance weight is defined as

$$q(X_k) = \frac{P(X_k \mid Y_k)}{\pi(X_k \mid Y_k)} \tag{2.30}$$

Appliying the normalization condition, the estimation becomes

$$I(g(X_k)) = \frac{\int g(X_k) q(X_k) \pi(X_k \mid Y_k) dX_k}{\int q(X_k) \pi(X_k \mid Y_k) dX_k} \tag{2.31}$$

Using the particles sampled from importance density, the estimation can be written as

$$\hat{I}(g(X_k)) = \frac{\frac{1}{N}\sum_{i=1}^{N} q(X_k^i) g(X_k^i)}{\frac{1}{N}\sum_{i=1}^{N} q(X_k^i)} = \sum_{i=1}^{N} \tilde{q}(X_k^i) g(X_k^i) \tag{2.32}$$

where the normalized importance weights are defined as

$$\tilde{q}(X_k^i) = \frac{q(X_k^i)}{\sum_{i=1}^{N} q(X_k^i)} \tag{2.33}$$

Importance sampling makes the representation of the *a posteriori* pdf as

$$\hat{P}(X_k \mid Y_k) = \sum_{i=1}^{N} \tilde{q}(X_k^i) \delta(X_k - X_k^i) \tag{2.34}$$

If we could sample from the *a posteriori* pdf i.e. $\pi(X_k \mid Y_k) = P(X_k \mid Y_k)$ then $\tilde{q}(X_k^i) = \frac{1}{N}$ and we can get the perfect sampling framework.

2.3.3 Sequential Importance Sampling

In the filtering problem the aim is to find the $P(X_k \mid Y_k)$ using the previous estimate $P(X_{k-1} \mid Y_{k-1})$ and the measurements. If we can find the importance

weights of the $P(X_k \mid Y_k)$ using the importance weights of $P(X_{k-1} \mid Y_{k-1})$ and measurements, then we can make estimations recursively. To achive this, we can arrange the *a posteriori* pdf as

$$\begin{aligned} P(X_k \mid Y_k) = P(X_k \mid Y_{k-1}, y_k) &= \frac{P(X_k, y_k \mid Y_{k-1})}{P(y_k \mid Y_{k-1})} \\ &\propto P(y_k \mid X_k, Y_{k-1}).P(X_k \mid Y_{k-1}) \\ &= P(y_k \mid x_k)P(x_k \mid X_{k-1}, Y_{k-1}).P(X_{k-1} \mid Y_{k-1}) \\ &= P(y_k \mid x_k)P(x_k \mid x_{k-1}).P(X_{k-1} \mid Y_{k-1}) \end{aligned} \tag{2.35}$$

Also importance density can be written as

$$\pi(X_k \mid Y_k) = \pi(x_k \mid X_{k-1}, Y_k)\pi(X_{k-1} \mid Y_k) = \pi(x_k \mid X_{k-1}, Y_k)\pi(X_{k-1} \mid Y_{k-1}) \tag{2.36}$$

Using (2.35),(2.36) importance weights can be updated recursively as

$$\begin{aligned} q(X_k^i) = \frac{P(X_k^i \mid Y_k)}{\pi(X_k^i \mid Y_k)} &= \frac{P(y_k \mid x_k^i)P(x_k^i \mid x_{k-1}^i).P(X_{k-1}^i \mid Y_{k-1})}{\pi(x_k^i \mid X_{k-1}^i, Y_k)\pi(X_{k-1}^i \mid Y_{k-1})} \\ &= \frac{P(y_k \mid x_k^i)P(x_k^i \mid x_{k-1}^i)}{\pi(x_k^i \mid x_{k-1}^i, y_k)} q(X_{k-1}^i) \end{aligned} \tag{2.37}$$

Using (2.37) we can recursively update importance weights using measurements.

While the importance weights are being updated, some of the weights can be less important and their weights can be very small as time passes. In this case, the variance of the weights will increases, more particles will be less important with time and the estimations will diverge. This is known as divergence problem. As a solution to this problem, resampling is done and particles having small importance weights are discarded and particles having large importance weights are multiplied. But resampling introduces another problem which is known as sample impoverishment. This is because particles with large weights will be selected

many times which makes dependence and loss of diversity between particles. To cope with this problem, some methodologies are offered(i.e jittering). Some of the methodologies are mentioned in [3]. To reduce the negative effects, resampling can be done as necessary. In order to do this, we need a measure of the degeneracy of the particles. As given in [3] a measure can be effective sample size which is given as

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} \left(\tilde{q}\left(X_k^i\right)\right)^2} \tag{2.38}$$

where $\left\{\tilde{q}\left(X_k^i\right)\right\}_{i=1}^{N}$ are the normalized importance weights. If the variance of the particles become large, N_{eff}will be small. As the value of N_{eff}becomes lower than a threshold,the resampling step can be applied. Several resampling methods have been proposed in litarature. Some of them can be found in [12].

2.3.4 Particle Filter

In the Particle Filter the importance density $\pi(x_k^i \mid x_{k-1}^i, y_k)$ used in the weight update (2.37) is chosen as the transition probability $P(x_k^i \mid x_{k-1}^i)$ and the weight update becomes as

$$q(X_k^i) = P(y_k \mid x_k^i) q(X_{k-1}^i) \tag{2.39}$$

For a dynamical system in continuous-time given as (2.17) or in discrete-ime given as (2.18) with measurements in discrete-time given as (2.19) the following steps summarizes the filtering steps for Particle Filter.

1. Initialize the particles by sampling from the an initial pdf $P(x_0)$ $\left\{x_0^i\right\}_{i=1}^{N} \sim P(x_0)$ The initial pdf is assumed to known and be easily sampled.

Assign the weight of each particle as $1/N$. This step is done once in the beginning of recursive filtering.

Prediction Step

2. Let us have particles $\{x_{k-1}^i\}_{i=1}^N$ with weights $q(x_{k-1}^i)$. Evaluate the particles in time using the dynamical system. For continuous-time dynamical system, two ways of evaluation can be done.

   a) Integrate each particle using noise free dynamical system $\frac{dx}{dt} = f(x,t)$. The integration can be done using numerical methods(Euler i.e.). After integration, add random noise to each particle. Random noise can be created using $N(0, TQ_\beta)$.

   b) Or at every discretization step of numerical integration, add random noise to integrated particles. Assume we are using Euler method and we have L Euler steps. At every Euler step, add random noise to integrated particle. Random noise can be created using $N(0,(T/L)Q_\beta)$ at every Euler Step. The pseudo code for numerical integration using L Euler steps can be given as

```
for i=1:L
        x(i)=x(i-1) + h*f(x(i-1), (i-1))
        x(i) = x(i) + random noise
end
```

   where $h=(T/L)$ and *random noise* $\sim N(0,(T/L)Q_\beta)$.

For discrete-time dynamical system, evaluate each particle using noise free dynamical system and then add random noise to each particle. Random noise can be created using $N(0, Q_w)$.

In the dynamical systems given in (2.17) and (2.18), we used Gaussian noise. But there can be non-Gaussian noise in dynamical systems. In this case, random noise can be created using non-Gaussian pdf.

After this step, we have the predicted particles $\{x_{k|k-1}^i\}_{i=1}^N$ with weights $q(x_{k-1}^i)$. The weights of the particles do not change in this step. The *a priori* pdf will be related to

$$P(x_k \mid y_{k-1}) = \sum_{i=1}^{N} q(x_{k-1}^i)\delta(x_{k|k-1} - x_{k|k-1}^i) \tag{2.40}$$

Update Step

3. Set new particles as predicted particles $x_k^i = x_{k|k-1}^i$. Update the weights of the new particles using likelihood and old weights as

$$q(x_k^i) = P(y_k \mid x_k^i)q(x_{k-1}^i) \tag{2.41}$$

4. Normalize the updated weights

$$\tilde{q}\left(x_k^i\right) = \frac{q\left(x_k^i\right)}{\sum_{i=1}^{N} q\left(x_k^i\right)} \tag{2.42}$$

5. Calculate the efficient sample size N_{eff}. If $N_{eff} < N_{thereshold}$ resample the particles and reset the importance weights of each particle to $1/N$.

6. Set the sampling time to $k+1$ and go to step 2

By using the particles, mean and covariance of the state variable can be estimated as

$$
\begin{aligned}
m_{x_k} &= \sum_{i=1}^{N} \tilde{q}(x_k^i) x_k^i \\
P_{x_k} &= \sum_{i=1}^{N} \tilde{q}(x_k^i)(x_k^i - m_{x_k})(x_k^i - m_{x_k})^T
\end{aligned}
\tag{2.43}
$$

2.3.5 Unscented Particle Filter

The Importance density used in Particle Filter does not use the information collected from the measurements. The quality of the particles can be improved by choosing an importance density incorporating the information collected as a result of the measurements. This also helps to reduce the divergence and sample impoverishment problems. In the Unscented Particle Filter, this incorparation is done by choosing an importance density that is calculated using an auxiliary UKF. In this case, the weight update becomes as

$$
q(X_k^i) = \frac{P(y_k \mid x_k^i) P(x_k^i \mid x_{k-1}^i)}{\pi(x_k^i \mid x_{k-1}^i, y_k)} q(X_{k-1}^i) \tag{2.44}
$$

where the importance density $\pi(x_k^i \mid x_{k-1}^i, y_k)$ is a Gaussian found using the UKF.

For a given system as in the Particle Filter, the following steps summarize the filtering steps for Unscented Particle Filter.

1. Initialize the particles by sampling from the initial pdf $\{x_0^i\}_{i=1}^N \sim P(x_0)$ The initial pdf is assumed to known and be easily sampled. Assign the weight of each particle as 1/*N*. Also assign an inital covariance value for each particle. This covariance value is uqual for every particle and can be the covariance value of the initial pdf.

Importance Density Calculation

2. Let us have particles $\{x_{k-1}^i\}_{i=1}^N$ with weights $q(x_{k-1}^i)$ and covariances $\{P_{x(k-1)}^i\}_{i=1}^N$. Calculate an importance density for each of the particle by inserting every particle to an UKF. After this step, we have new particles $\{\bar{x}_k^i\}_{i=1}^N$ which are updated using UKF. Here we add noise to this UKF updated particles to find new particles

$$x_k^i = \bar{x}_k^i + random\_noise$$
$$P_k^i = \bar{P}_k^i$$

$random\_noise \sim N(0, \bar{P}_k^i)$ where $\bar{P}_k^i$ is the updated covariance for the ith particle.

For every particle, the value of the importance density will be

$$\pi(x_k^i \mid x_{k-1}^i, y_k) = N(x_k^i - \bar{x}_k^i, P_k^i) \tag{2.45}$$

The weights of new particles $\{x_k^i\}_{i=1}^N$ do not change in this step $q(x_k^i) = q(x_{k-1}^i)$

Prediction Step

3. We also predict the particles $\{x_{k-1}^i\}_{i=1}^N$ as in Particle Filter case. This process is essential in order to calculate the transition probability. After this step, we have the predicted particles $\{x_{k|k-1}^i\}_{i=1}^N$ with weights $q(x_{k-1}^i)$. The *a priori* pdf will be as in (2.40)

Update Step

4. Update the weights of the new particles $\{x_k^i\}_{i=1}^N$ using likelihood, transition probability, importance density and old weights as

$$q(x_k^i) = \frac{P(y_k \mid x_k^i)P(x_k^i \mid x_{k-1}^i)}{\pi(x_k^i \mid x_{k-1}^i, y_k)} q(x_{k-1}^i) \tag{2.46}$$

Here we know the likelihood, importance density and old weights. Also the transition probability can be found easily. Since the noise is chosen as Gaussian for the given dynamical system, the value of transition probability will be $P(x_k^i \mid x_{k-1}^i) = N(x_k^i - x_{k|k-1}^i, TQ_\beta)$ for continuos-time dyamical system and $P(x_k^i \mid x_{k-1}^i) = N(x_k^i - x_{k|k-1}^i, Q_w)$ for discrete-time dynamical system.

5. Normalize the updated weights

$$\tilde{q}(x_k^i) = \frac{q(x_k^i)}{\sum_{i=1}^{N} q(x_k^i)} \tag{2.47}$$

6. Calculate the efficient sample size N_{eff}. If $N_{eff} < N_{thereshold}$ resample the particles and reset the importance weights of each particle to $1/N$.

Set the sampling time to $k+1$ and go to step 2

2.4 Particle Flow Filter

Although Particle Filter gives result near to optimal when used with sufficiently large particles, for problems where the dimension of the state variable is large

(d>3 i.e), it requires too much particles to achieve low error levels. This makes it not practically applicable to many problems.

A few years ago, Daum proposed a new filter which is called as Particle Flow Filter[13]. The first steps of the Particle Flow Filter is similar to Particle Filters. It samples many points from an initial density at first. Then evolve this particles in time using the dynamical system. But after this step, Particle Flow Filter uses a flow function to flow particles to the correct region of state space. Flow function is found using likelihood. Flow of particles is achieved by solving an ODE which uses the flow function.

Required number of particles is not too high as in the Particle Filter case. In [14], it is shown that in a radar tracking problem with 6 dimensional state space, between 1000 and 10000 particles are enough for a low estimation error while using with measurements having low measurement error. There is no particle degeneracy or resampling in Particle Flow Filter. But another issue similar to particle degeneracy exists. This is due to infinite speed particles. In [15] a list of methods is offered in a table to cope with this problem. But none of them removes this problem absolutely. In fact, infinite speed particles may require a methodology similar to resampling to eliminate them.

Now we can give the details of Particle Flow Filter based on the expressions given in [13].

Derivation of Flow Function using Poisson Equation

Suppose that we apply the prediction step of Particle Filter and we have the predicted particles. The fundamental idea of particle flow is to create a differential equation to implement Bayes' rule for update step. This differential equation is chosen as an ordinary differential equation (ODE). A homotopy is used to create this ODE. A scalar valued parameter, called as λ is introduced, that plays the role

of time, and which varies from 0 to 1. You can think of λ as a little loop of synthetic time, which we insert at each sampling time. By using the λ parameter, the Bayes' rule for the *a posteriori* pdf can be rearranged as

$$p(x_k|y_k) = p(x,\lambda) = \frac{g(x).h(x)^{\lambda}}{K(\lambda)} \tag{2.48}$$

where $g(x) = p(x_k|y_{k-1})$ is the *a priori* pdf and $h(x) = p(y_k|x_k)$ is the likelihood. $K(\lambda)$ normalizes the density $p(x,\lambda)$.

$\lambda = 0$ is the initial condition before the measurement arrives and $\lambda = 1$ is the uptaded conditional density that we want to reach. A log homotopy is defined using (2.48) as

$$\log\left(p(y_k|x_k)\right) = \log\left(g(x)\right) + \lambda \log\left(h(x)\right) - \log\left(K(\lambda)\right) \tag{2.49}$$

Next, we suppose that there exists a continuous flow of particles induced by the flow of probability density from $\lambda = 0$ to $\lambda = 1$. This flow of particles can be defined with following dynamics.

$$\frac{dx}{d\lambda} = f(x,\lambda) \tag{2.50}$$

The Fokker-Planck equation related to this dynamics or ODE is given as

$$\frac{dp}{d\lambda} = -Tr\left(\frac{\partial fp}{\partial x}\right) \tag{2.51}$$

If we can find a solution of Fokker-Planck Equation for the flow function $f(x,\lambda)$, then we can flow the particles using (2.50). Now we can show how this can be.

By derivating the equation (2.49) according to λ, we can get

$$\frac{\partial \log p(x,\lambda)}{\partial \lambda} = \log h(x) - \frac{\partial \log K(\lambda)}{\partial \lambda} \tag{2.52}$$

For nowhere vanishing densities, we can arrange the Fokker Planck equation (2.51) by using (2.52) as follows

$$Tr\left(\frac{\partial f(x,\lambda)p(x,\lambda)}{\partial x}\right) = -p(x,\lambda)\left(\log h(x) - \frac{\partial \log K(\lambda)}{\partial \lambda}\right) \tag{2.53}$$

If we define

$$q = f(x,\lambda)p(x,\lambda) \text{ and } \eta = -p(x,\lambda)\left(\log h(x) - \frac{\partial \log K(\lambda)}{\partial \lambda}\right) \tag{2.54}$$

We can get

$$div(q) = \eta \tag{2.55}$$

The above is the divergence form of the Fokker Planck equation

Here we can find the $\frac{\partial \log K(\lambda)}{\partial \lambda}$ as

$$K(\lambda) = \int g(x).h(x)^{\lambda} dx$$
$$\frac{\partial \log K(\lambda)}{\partial \lambda} = \frac{\partial K(\lambda)}{\partial \lambda} / K(\lambda)$$

$$\frac{\partial \log K(\lambda)}{\partial \lambda} = \frac{\int g(x).h(x)^{\lambda} \log h(x) dx}{\int g(x).h(x)^{\lambda} dx}$$
$$\frac{\partial \log K(\lambda)}{\partial \lambda} = \frac{\int p(x,\lambda) \log h(x) dx}{\int p(x,\lambda) dx} \tag{2.56}$$

we can now recognize this as the expected value of *logh(x)* with respect to the density *p(x,λ):*

$$\frac{\partial \log K(\lambda)}{\partial \lambda} = E[\log h(x)] \tag{2.57}$$

Using the divergence form, there is obviously no unique solution for q, because there are too few equations and too many unknowns. If an assumption is made that vector field q is the gradient of a scalar-valued potential, called $V(x,\lambda)$*:*

$$q = \frac{\partial \log V(x,\lambda)}{\partial x} \tag{2.58}$$

And we put it into Fokker-Planck equation then we get the Poisson's equation as

$$\frac{dp}{d\lambda} = -Tr\left(\frac{\partial^2 \log V(x,\lambda)}{\partial x^2}\right) \tag{2.59}$$

For state variables with dimension $d \geq 3$ $V(x,\lambda)$ can be computed using a convolution integral as follows:

$$V(x,\lambda) = -\int \eta(x,\lambda) \frac{c}{\|x-y\|^{d-2}} dy \quad \text{where} \quad c = \Gamma\left(\frac{d}{2}-1\right) / 4\pi^{d/2} \tag{2.60}$$

Using the definition of η and assuming that we can differentiate under the integral with respect to x, we get:

$$\frac{\partial V(x,\lambda)}{\partial x} = \int \left(\log h(y) - \frac{\partial \log K(\lambda)}{\partial \lambda} \right) p(x,\lambda) \frac{c(2-d)(x-y)^T}{\|x-y\|^{d-2}} dy \qquad (2.61)$$

we recognize that this integral is an expected value with respect to the probability

$$\frac{\partial V(x,\lambda)}{\partial x} = E\left[\left(\log h(y) - \frac{\partial \log K(\lambda)}{\partial \lambda} \right) \frac{c(2-d)(x-y)^T}{\|x-y\|^{d-2}} \right] \qquad (2.62)$$

Finally, using the predicted particles, we can apply the Monte Carlo approximation for integrals (or expectation):

$$\frac{\partial V(x_i,\lambda)}{\partial x} \approx \frac{1}{N} \sum_{j=1, j\neq i}^{N} \left(\log h(x_j) - \frac{\partial \log K(\lambda)}{\partial \lambda} \right) \frac{c(2-d)(x_i-x_j)^T}{\|x_i-x_j\|^{d-2}} \qquad (2.63)$$

Also we can find (2.57) as

$$\frac{\partial \log K(\lambda)}{\partial \lambda} = E[\log h(x)] \approx \frac{1}{N} \sum_{j=1}^{N} \log h(x_j) \qquad (2.64)$$

In which $x_i$denotes the ith particle. The summation can be over the subset of k, nearest neighbors of a given particle, rather than over all particles. If we summed over all particles, then the computational complexity would be quadratic in N.

If singularity problems occur in (2.63) then we can add a small positive variable (α) to the denominator, as follows

$$\frac{\partial V(x_i,\lambda)}{\partial x} \approx \frac{1}{N}\sum_{j=1, j\neq i}^{N}\left(\log h(x_j) - \frac{\partial \log K(\lambda)}{\partial \lambda}\right)\frac{c(2-d)(x_i - x_j)^T}{\left\|x_i - x_j\right\|^{d-2} + \alpha} \qquad (2.65)$$

where $\alpha = Tr(C)^{d/2} / \beta$

Here C is the conditional covariance of particles which can be found using particles, the parameter β can be on the order of 10 to 100.

After this step, the flow function and the ODE will be like that

$$\frac{dx}{d\lambda} = f(x,\lambda) \qquad \text{where} \quad f(x,\lambda) = \frac{\partial V(x,\lambda)}{\partial x} / \, p(x,\lambda) \qquad (2.66)$$

By solving this ODE from $\lambda = 0$ to $\lambda = 1$ for each particle, the particals flow to their correct place.

In practice there is a problem with that flow function. As mentioned in [13], there will be some particles that are very far from the peak of the likelihood, h(x), and this will cause p(x, λ) to have very small numerical values, and hence the particle speed will be extremely large. These infinite speed particles cause depletion of particles, which is analogous to particle degeneracy in Particle Filters. In [13] a list of methods offered to reduce this problem. In fact these infinite speed particles are needed to be eliminated and a methodology similar to resampling can eliminate the infinite speed particles and fix the particle size at every sampling time.

Altough the details of filtering steps for Particle Flow Filter has not exist in literature yet, we can apply the following filtering steps for Particle Flow Filter. Here the system model is assumed to be as in the Particle Filter case.

1. Initialize the particles by sampling from the an initial pdf $P(x_0)$ $\{x_0^i\}_{i=1}^N \sim P(x_0)$ The initial pdf is assumed to known and be easily sampled. Assign the weight of each particle as $1/N$.

In Particle Flow Filter, the weights of particles are not needed to be updated. So their value can stay as $1/N$.

Prediction Step

2. Let us have particles $\{x_{k-1}^i\}_{i=1}^N$. Predict the particles as in Particle Filter case.

Flow of Particles(Update Step)

3. Calculate the flow function for each of the particles using (2.63) or (2.65) and (2.64).

4. Flow the particles using the ODE (2.66).

5. Set λ=1 and calculate $\frac{p(x_k^i, \lambda)}{\sum_{i=1}^{N} p(x_k^i, \lambda)}$ values using the definitions (2.48) for each of the particle. Here the unnormalized $p(x, \lambda) = g(x).h(x)^{\lambda}$ can be used. Eliminate the particles having small values than a threshold value. These can be infinite speed particles.

6. Mean and covariance values can be estimated as in (2.43) using the survived particles. The weight of each particle can be taken as $1/N$.

7. To fix the particle size, eliminated particles can be replaced with the ones having big values.

8. Set the sampling time to *k+1* and go to step 2

Exact Flow for Gaussian Densities

In the previous section, we found the flow function by solving the Poisson Equation. But as briefly given in [16], Fokker-Planck equation can be solved using some other methods. Especially when the *a priori* and *likelihood* are Gaussian, a stable flow function can be easily found using the separation of variables method. Now we can give how this flow function can be found.

Using (2.49), Fokker-Planck equation (2.51) can be arranged as follows

$$\frac{d\log(p)}{dx}f+\log h-\frac{\partial \log K(\lambda)}{d\lambda}=-Tr\left(\frac{\partial f}{\partial x}\right) \tag{2.67}$$

When the *a priori* (g) and *likelihood* (h) are Gaussian as

$$g=\frac{1}{\sqrt{2\pi^{d}\det(P)}}e^{-\frac{1}{2}(x-\bar{x})^{T}P^{-1}(x-\bar{x})} \qquad h=\frac{1}{\sqrt{2\pi^{d_z}\det(R)}}e^{-\frac{1}{2}(z-Hx)^{T}R^{-1}(z-Hx)}$$

If the form of flow function is chosen as $f=Ax+b$, equation (2.67) can be solved exactly by equating like coefficients on the RHS & LHS. Substituting *a priori, likelihood* and f into equation (2.67) and equating like coefficients of x and terms that are quadratic in x results in the following exact solution for the particle flow;

$$\frac{dx}{d\lambda}=f(x,\lambda)=A(\lambda)x+b(\lambda) \tag{2.68}$$

where

$$A = -\frac{1}{2} PH^T (\lambda HPH^T + R)^{-1} H$$

$$b = (I + 2\lambda A)\left[(I + \lambda A) PH^T R^{-1} z + A\bar{x}\right] \tag{2.69}$$

The method used to solve the equation (2.67) to find A and b of f is called as "separation of variables". It is essentially the same as the exact solution of the Fokker-Planck equation for the Kalman filter problem for linear systems.

CHAPTER 3

CONTINUOUS-TIME NONLINEAR ESTIMATION FILTERS USING UKF-AIDED GAUSSIAN SUM REPRESENTATIONS

In the previous chapters, the filtering problem and some of the known filters that are applied for the solution of the problem are given. Basicly, there are two types of solution methodologies for the approximate solution of the filtering problem. One tries to simplify the models and the other tries to find a global approximation for the pdfs. We gave sample filters for both of the methodologies in Chapter 2.

A possible approach for pdf approximation methodology is based on the representation of the *a priori* and *a posteriori* pdfs by using a mixture (or, sum) of weighted Gaussians. For this purpose, we developed the UKF-aided Gaussian Sum Filter. Also as a second filter, we integrated the Particle Flow to UKF-aided Gaussian Filter. Now we can give the details of both filters.

3.1 UKF-aided Gaussian Sum Filter

In literature, there are filters that use the weighted Gaussian sums. The Gaussian Sum Filter (GSF) [17] has been developed for this purpose. In GSF, the pdfs are approximated using weighted Gaussian sums. However, the weights of the Gaussians are only updated while approximating the *a posteriori* pdf. In the Adaptive Gaussian Sum Filter (AGSF) [18], whichis an extension of the GSF, the weights of Gaussians are updated while approximating both the *a priori* and *a posteriori* pdfs.

The filter developed here also uses weighted Gaussian sums and weights of Gaussians are updated while both finding the *a priori* and *a posteriori* pdfs. But the means, covariances and weights of Gaussians are calculated in a different way

When we look at the differences, firstly valid search regions are determined for the *a priori* and *a posteriori* pdfs with the help of UKF in the developed filter. Number of used Gaussians and their mean and covariance values are deterministically found in these regions later. Also weights of Gaussians are calculated using least squares method in the manner of curve fitting with Gaussian functions which differs than the optimization problem defined in AGSF. The developed filter has ability to approximate the whole parts of the pdfs or some parts of the pdfs close to the mean by adjusting the size of search regions.

In the UKF-aided Gaussian Sum Filter,the above procedure for approximation of the *a pri*ori and the *a posteriori* pdfs are done mainly in two steps for the given system in Chapter 1. These steps are as follows;

- Gaussian Sum Approximation of the *a priori* pdf by means of the numerical solution of the Fokker-Planck equation using weighted Gaussian sums.

- Gaussian Sum approximation of the *a posteriori* pdf by using the approximated *a priori* pdf and *likelihood* in Bayes' rule.

The required steps for the developed filter can be seen from the diagram given in Figure 1

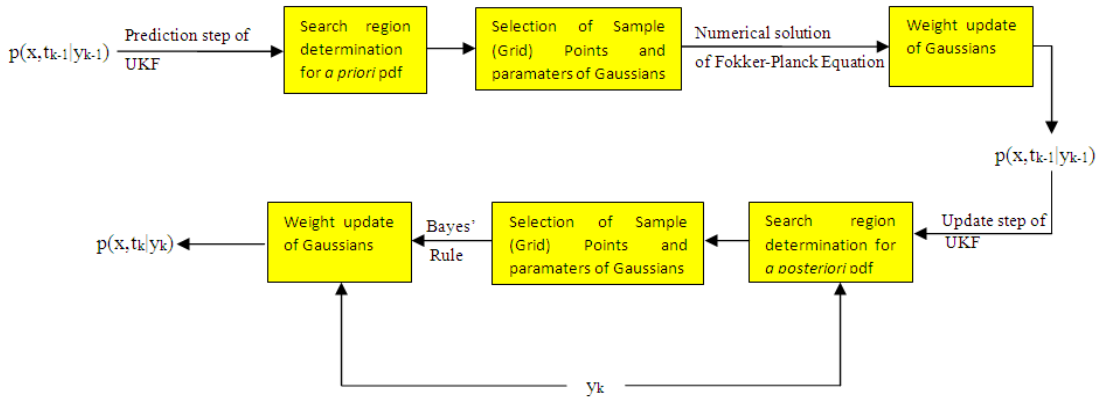

Figure 1 The Filtering Steps for UKF-aided Gaussian Sum Filter

Now we can give the details of these steps.

Gaussian Sum Approximation of the a priori pdf (Prediction Step)

For the given continuous-time dynamical system, the Fokker-Planck equation, which yields the time evolution of the *a priori* pdf, is given as follows

$$\frac{\partial p(x,t)}{\partial t} = g(p,t) \tag{3.1}$$

where $g(p,t) = -\frac{\partial p(x,t)}{\partial x} f(x,t) - p(x,t) Tr\left(\frac{\partial f(x,t)}{\partial x}\right) + \frac{1}{2} Tr\left(\frac{\partial^2 G Q_\beta G^T p(x,t)}{\partial x^2}\right)$

Since the analytical solution of the Fokker-Planck equation is not available in general, it has to be solved approximately by means of a numerical approach. For any *t,* the function *p(x,t)* is approximated in terms of weighted sum of Gaussians and the partial differential equation (3.1) reduces to an ordinary differential equation. In the steps given below, it is explained how these Gaussian functions are chosen, and how they evolve in time.

The evolution of $p(x,t)$is approximated over a "moving" grid (i.e set of sample points) by using Euler's method. The relationship between these sample points and the Gaussian functions used in the approximation of $p(x,t)$is important and is explained below. Beginning from an initial pdf $p_0(x,t) = p(x, t_0 \mid y_0)$ at t_0, n=$T/\Delta T$ Euler steps can be used to find the $p_1(x,t) = p(x, t_1 \mid y_0)$ (*a priori* pdf) at $t_1 = t_0 + T$. The following steps summarize the calculation of the mean, covariance and weights of the Gaussians used in the *a priori* pdf approximation in a single Euler step from t_0 to $t_0 + \Delta T$. The remaining Euler steps are similarly defined.

1. Predicted mean $\hat{x}_{t_0+\Delta T}$ and covariance $\hat{P}_{t_0+\Delta T}$ values are found by applying the prediction step of UKF to the mean x_{t_0} and covariance P_{t_0} values of initial

pdf (at later Euler steps, initial pdf is the *a priori* pdf calculated at the previous Euler step). For this purpose the unscented transform is applied using x_{t_0} , P_{t_0} . The nonlinear function used in the transformation is $x_{t_0} + \Delta T f(x_{t_0}, t_0)$

2. A uniform grid (search region) is chosen in the neighborhood of $\hat{x}_{t_0+\Delta T}$. The extent of the grid is defined as the matrix $c\left(\hat{P}_{t_0+\Delta T} + ldiag\left((x_{t_0} - \hat{x}_{t_0+\Delta T})^T (x_{t_0} - \hat{x}_{t_0+\Delta T})\right)\right)$ where $l \geq 0$ and c is the scaling factor. The number of sample points are determined according to the extent of the grid and their value is uniformly chosen by means of the diagonal entries of the extent of the grid. The Gaussian sum approximation of the *a priori* pdf in the grid is obtained by choosing the means of the Gaussians ($\mu^i_{t_0+\Delta T}$) as the sample points and for covariances the following holds.

   For 0<c<1, the covariances of Gaussians can be taken as the same value as $\nu\hat{P}_{t_0+\Delta T}$ where $\nu \geq 1$.

   For c≥1, *l* can be chosen as zero and the covariances of Gaussians can be taken as the same value as $\nu\hat{P}_{t_0+\Delta T}$ where 0<ν<1.

3. A second set of sample points is generated around x_{t_0} . The extent of this grid and the number of sample points are determined similar to $\hat{x}_{t_0+\Delta T}$ case. The reason for constructing the second grid is due to the simultaneous appearance of $p(x,t_0)$ and $p(x,t_{0+\Delta T})$ in Euler discretized Fokker Planck equation $p(x, t_0 + \Delta T \mid y_0) = p(x, t_0 \mid y_0) + \Delta T g\left(p(x, t_0 \mid y_0), t_0\right)$. These two sets of sample points can be combined as:

$$s = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} \qquad \text{where} \quad s_1 = \begin{bmatrix} s_1^1 \\ \vdots \\ s_1^N \end{bmatrix} = \begin{bmatrix} \mu_{t_0+\Delta T}^1 \\ \vdots \\ \mu_{t_0+\Delta T}^N \end{bmatrix} \quad s_2 = \begin{bmatrix} s_2^1 \\ \vdots \\ s_2^M \end{bmatrix} = \begin{bmatrix} x_{t_0}^1 \\ \vdots \\ x_{t_0}^M \end{bmatrix} \tag{3.2}$$

4. Using the values in step 2, the weighted Gaussian sum approximation of unnormalized *a priori* pdf for the current step can be modeled as:

$$\hat{p}(x, t_0 + \Delta T \mid y_0) \approx \sum_{i=1}^{N} w_{t_0+\Delta T}^{i} \hat{p}_i(x) \tag{3.3}$$

where $\hat{p}_i(x) = e^{-\frac{1}{2}(x-\mu_{t_0+\Delta T}^i)^T (P_{t_0+\Delta T}^i)^{-1}(x-\mu_{t_0+\Delta T}^i)}$ and N>1. The weights of Gaussians are calculated in step 5.

5. The value of the unnormalized *a priori* pdf can be found by using the Euler discretized Fokker-Planck Equation as follows:

$$p(x, t_0 + \Delta T \mid y_0) = p(x, t_0 \mid y_0) + \Delta T g\big(p(x, t_0 \mid y_0), t_0\big) \tag{3.4}$$

where g is $g(p,t) = -\dfrac{\partial p(x,t)}{\partial x} f(x,t) - p(x,t) Tr\left(\dfrac{\partial f(x,t)}{\partial x}\right) + \dfrac{1}{2} Tr\left(\dfrac{\partial^2 G Q_\beta G^T p(x,t)}{\partial x^2}\right)$

By equating (3.3) and (3.4) at the sample points we have the following linear equation systemfor the unknown weights:

$$\underbrace{\begin{bmatrix} \begin{bmatrix} \hat{p}_1(s_1^1) & \cdots & \hat{p}_N(s_1^1) \\ \vdots & \ddots & \vdots \\ \hat{p}_1(s_1^N) & \cdots & \hat{p}_N(s_1^N) \end{bmatrix} \\ \begin{bmatrix} \hat{p}_1(s_2^1) & \cdots & \hat{p}_N(s_2^1) \\ \vdots & \ddots & \vdots \\ \hat{p}_1(s_2^M) & \cdots & \hat{p}_N(s_2^M) \end{bmatrix} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} w_{t_0+\Delta T}^1 \\ \vdots \\ w_{t_0+\Delta T}^N \end{bmatrix}}_{w_{t_0+\Delta T}} = \underbrace{\begin{bmatrix} \begin{bmatrix} p(s_1^1, t_0+\Delta T \mid y_0) \\ \vdots \\ p(s_1^N, t_0+\Delta T \mid y_0) \end{bmatrix} \\ \begin{bmatrix} p(s_2^1, t_0+\Delta T \mid y_0) \\ \vdots \\ p(s_2^M, t_0+\Delta T \mid y_0) \end{bmatrix} \end{bmatrix}}_{B} \tag{3.5}$$

$$A.w_{t_0+\Delta T} = B \tag{3.6}$$

The least squares solution of this system for the weightscan be given as

$$w_{t_0+\Delta T} = (A^T A)^{-1} A^T B \tag{3.7}$$

If $A^T A$ is nearly singular, maximum values along each row of *A*matrix may be used. In this case,*NxN* part of the $A_{(M+N)xN}$ matrix will be diagonal for distinct values, avoiding the inversion problems

The above procedure is repeated at every Euler stepfor the approximation of the *a priori* pdf by means of weighted Gaussian sums.

Gaussian Sum Approximation of the a posteriori pdf (Update Step)

After n Euler steps, the approximate*a priori* pdf is obtained as $\hat{p}_1(x,t) = \hat{p}(x, t_1 \mid y_0)$ for $t_1 = t_0 + n\Delta T$. By using the likelihood $p(y_1 \mid x, t_1)$ and the approximate *a priori* pdf in Bayes' rule, the unnormalized *a posteriori* pdf at t_1 can be found as;

$$p(x, t_1 \mid y_1) = p(y_1 \mid x, t_1)\hat{p}(x, t_1 \mid y_0) \tag{3.8}$$

As a result of the multiplication operation with $p(y_1 \mid x, t_1)$, normalization of the *a posteriori* pdf is difficult . For this reason, the unnormalized *a posteriori* pdf is

also approximated with a weighted Gaussian sum. The update step of the UKF is used for finding the mean and covariance values of Gaussians for the *a posteriori* pdf approximation and the weights of the Gaussians are again found by solving the matrix equation obtained by evaluating the *a posteriori* pdf at the sample points.

The steps for the weighted Gaussian sum approximation of the *a posteriori* pdf at time t_1 are given below.

1. Updated mean $\tilde{x}_{t_1}$ and covariance $\tilde{P}_{t_1}$ values are found by applying the update step of UKF to the mean $\hat{x}_{t_1}$ and covariance $\hat{P}_{t_1}$ values of the approximate *a priori* pdf at time t_1.

2. A uniform grid is chosen around the updated mean $\tilde{x}_{t_1}$ and the extent of the grid and the number of sample points are determined using the diagonal entries of $c\left(\tilde{P}_{t_1} + ldiag\left((\hat{x}_{t_1} - \tilde{x}_{t_1})^T (\hat{x}_{t_1} - \tilde{x}_{t_1})\right)\right)$. Means of Gaussians ($\tilde{\mu}_{t_1}^i$) are chosen as the sample points in the uniform gird. Covariances of the Gaussians ($\tilde{P}_{t_1}^i$) are assigned to the same value chosen similar to *a priori* case

3. In this case, there is a single set of sample points which involves the means of the Gaussians given in step 2. This set is shown as;

$$g = \begin{bmatrix} \tilde{\mu}_{t_1}^1 \\ \vdots \\ \tilde{\mu}_{t_1}^K \end{bmatrix} \tag{3.9}$$

4. Using the values in step 2, the weighted Gaussian sum approximation of unnormalized *a posteriori* pdf at time t_1 can be modeled as;

$$\tilde{p}(x, t_1 \mid y_1) \approx \sum_{i=1}^{K} \tilde{w}_i \tilde{p}_i(x) = \sum_{i=1}^{K} \tilde{w}_i e^{-\frac{1}{2}(x-\tilde{\mu}_{t_1}^i)^T (\tilde{P}_{t_1}^i)^{-1} (x-\tilde{\mu}_{t_1}^i)} \tag{3.10}$$

where K>1. The weights of these Gaussians are calculated in step 5.

5. Equating (3.8) and (3.10) at the sample points, we have the following linear equation systemfor the unknown weights:

$$\underbrace{\begin{bmatrix} \widetilde{p}_1(\widetilde{\mu}_{t_1}^1) & \cdots & \widetilde{p}_K(\widetilde{\mu}_{t_1}^1) \\ \vdots & \ddots & \vdots \\ \widetilde{p}_1(\widetilde{\mu}_{t_1}^K) & \cdots & \widetilde{p}_K(\widetilde{\mu}_{t_1}^K) \end{bmatrix}}_{\widetilde{A}} \underbrace{\begin{bmatrix} \widetilde{w}_1 \\ \vdots \\ \widetilde{w}_K \end{bmatrix}}_{\widetilde{w}} = \underbrace{\begin{bmatrix} p(\widetilde{\mu}_{t_1}^1, t_1 \mid y_1) \\ \vdots \\ p(\widetilde{\mu}_{t_1}^K, t_1 \mid y_1) \end{bmatrix}}_{\widetilde{B}} \tag{3.11}$$

The solution can be obtained as $\widetilde{w} = (\widetilde{A}^T \widetilde{A})^{-1} \widetilde{A}^T \widetilde{B}$ (3.12)

The $\widetilde{A}^T \widetilde{A}$ matrix here is an KxK symmetric positive definite matrix given as in [19].

For both *a priori* and *a posteriori* pdfs, approximate expressions are obtained in terms of weighted Gaussian sum expressions defined over “moving” grids. The location and extent of these grids are obtained by the aid of UKF, which acts as a one step look ahead mechanism to determine where *a priori* and *a posteriori* pdfs are expected to reside. The number of Gaussians and their covariance values are determined in terms of these grids, which “move” as the pdfs evolve in time. In general, this approach provides a local approximation for the pdfs in regions defined in the close neighborhood of the values provided by the UKF results. In order to obtain “almost global” approximations, the grid extent (and, therefore the number of Gaussians) must be increased. Of course, the choice of the covariance values is also critical for a successful approximation. By using sufficient number of Gaussians and sample points, important parts or details of the pdfs can be captured and approximated. Currently, we do not propose a theoretical approach for the choice of the number of Gaussians, number of sample points or covariance values. They are chosen empirically according to the problem.

3.2 UKF-aided Gaussian Sum Filter with Particle Flow

Assuming that the *a priori* pdf is known and we have particles belonging to the *a priori* pdf, we see that a differential equation to implement Bayes' rule for update step is created in Particle Flow Filter. This differential equation uses flow function and flow function is calculated using *a priori* pdf and *likelihood*.

In the UKF-aided Gaussian Sum Filter with Particle Flow, *a priori* pdf is calculated same as in the UKF-aided Gaussian Sum Filter. Then a flow function is calculated using the Gaussian sum *a priori* pdf and *likelihood* using the separation of variables method. Using this flow function particles which are chosen from the*a priori* pdf are flowed. Also a Gaussian sum *a posteriori* pdf is found to eliminate the infinite speed particles. In order to find the *a posteriori* pdf, first likelihood is approximated as a Gaussian and every Gaussian of *a priori* pdf are updated using the Gaussian *likelihood* with standard Kalman Filter.

The below procedure summarizes the filtering steps of UKF-aided Gaussian Sum Filter with Particle Flow.

Filtering steps for UKF-aided Gaussian Sum Filter with Particle Flow

1. Beginning from an initial pdf, calculate the Gaussian sum *a priori* pdf as in the UKF-aided Gaussian Sum Filter. Initial pdf can be *a posteriori* pdf foundin the previous sampling time.

2. Calculate the flow function using the *a priori* pdf and *likelihood.* Choose particles from the *a priori* pdf and flow the particles using the flow function.

3. Approximate the *likelihood* with a Gaussian. Then update each Gaussian component of the *a priori* pdf with approximated*likelihood* using standard Kalman Filter to find the Gaussian sum *a posteriori* pdf.

4. To eliminate the infinite speed particles, calculate the mean of the *a posteriori* pdf. Take the flowed particles that are close to the mean. By using these particles, estimations about the state can be done. (such as mean of the state)

We give the details of step 1 in UKF-aided Gaussian Sum Filter. Estimation of *likelihood* with a Gaussian can be done using Unscented Transform or by conversion if the measurement model is a radar measurement model. Also Matlab "quad" function can be used if the measurement model is one dimensional. By using the "quad" function, mean and covariance of *likelihood* can be calculated for systems having one dimensional state variableand the mean and covariance can be assumed to belong to a Gaussian one. Also if the *likelihood* has more than one components (such as sum of two densities), then every component of *likelihood* can be approxiamated as a Gaussian and can be updated with each Gaussian component of *a priori* pdf. Now we can express how the flow function can be found and the particles are eliminated.

Calculating the Flow Function using a priori pdf and likelihood

Remember from the Particle Flow Filter that we define an ODE for the flow of particles and its corresponding Fokker-Planck equation as

$$\frac{dx}{d\lambda} = f(x,\lambda) \quad \frac{dp}{d\lambda} = -Tr\left(\frac{\partial fp}{\partial x}\right) \tag{3.13}$$

where

$$p = p(x,\lambda) = \frac{g(x).h(x)^{\lambda}}{K(\lambda)} \tag{3.14}$$

Here *g(x)* is the Gaussian sum *a priori* pdf coming from the prediction step of the UKF-aided Gaussian Sum Filter. We can define it as

$$g(x) = \sum_{i=1}^{N} w_i e^{-\frac{1}{2}(x-xcenter_i)Pcenter_i^{-1}(x-xcenter_i)^T} \tag{3.15}$$

h(x) is the *likelihood* and we also known its form.

As given in (2.67), Fokker-Planck equation can be arranged as

$$\frac{d\log(p)}{dx} f + \log h - \frac{\partial \log K(\lambda)}{d\lambda} = -Tr\left(\frac{\partial f}{\partial x}\right) \tag{3.16}$$

Also we can write it as

$$\left(\frac{d\log(g)}{dx} + \lambda \frac{d\log(h)}{dx}\right) f + \log h - \frac{\partial \log K(\lambda)}{d\lambda} = -Tr\left(\frac{\partial f}{\partial x}\right) \tag{3.17}$$

In the above equation, if we take the Taylor series approximation of $\frac{d\log(g)}{dx}$, $\frac{d\log(h)}{dx}$ and $\log(h)$ around each particle, we can define a flow function $f_i = A_i x + b_i$ for ith particle similar to Gaussian case and we can use the *separation of variables* method to find the $A_i s$ and $b_i s$. For this purpose first we need particles. In the developed filter mean of the each Gaussian component of *a priori* pdf ($xcenter_i$ values) can be taken as particles.

Following steps give the calculation of flow fuction for each particle.

1. Define the first order Taylor series approximation of $\frac{d\log(g)}{dx}$ around ith particle.

$$\frac{d\log(g)}{dx} \approx G1 + \left[G2\,(x - xcenter_i)\right] \tag{3.18}$$

For the given *a priori* pdf *G1* and *G2* are found as

$$G1 = \left.\frac{d\log(g)}{dx}\right|_{xcenter_i} = T3/T1$$

$$G1 = jacoboian\left(\left.\frac{d\log(g)}{dx}\right)\right|_{xcenter_i} = \left(T0\,T1 - (T2\,T3)\right)/\left(T1^2\right) \tag{3.19}$$

where

$$T0 = \sum_{j=1}^{N} w_j \left[\left(Pcenter_j^{-1}\right)\left(xcenter_i - xcenter_j\right)\left(xcenter_i - xcenter_j\right)^T\right]$$
$$e^{-\frac{1}{2}(xcenter_i - xcenter_j)Pcenter_i^{-1}(xcenter_i - xcenter_j)^T}$$

$$T1 = g(xcenter_i) \tag{3.20}$$

$$T2 = -\sum_{j=1}^{N} w_j \left[\left(Pcenter_j^{-1}\right)\left(xcenter_i - xcenter_j\right)\right] e^{-\frac{1}{2}(xcenter_i - xcenter_j)Pcenter_i^{-1}(xcenter_i - xcenter_j)^T}$$

$$T3 = \sum_{j=1}^{N} w_j \left[\left(xcenter_i - xcenter_j\right)^T\left(Pcenter_j^{-1}\right)\right] e^{-\frac{1}{2}(xcenter_i - xcenter_j)Pcenter_i^{-1}(xcenter_i - xcenter_j)^T}$$

2. Define the first order Taylor series approximation of $\frac{d\log(h)}{dx}$ around ith particle.

$$\frac{d\log(h)}{dx} \approx H1 + \left[H2\,(x - xcenter_i)\right] \tag{3.21}$$

$$H1 = \left.\frac{d\log(h)}{dx}\right|_{xcenter_i} \quad H2 = jacoboian\left(\left.\frac{d\log(h)}{dx}\right)\right|_{xcenter_i} \tag{3.22}$$

One can calculate *H1* and *H2* using the known form of *likelihood*

3. Define the second order Taylor series approximation of $\log(h)$ around ith particle.

$$\log(h) \approx H0 + \left[H1\,(x - xcenter_i)\right] + \left[(x - xcenter_i)^T\ H2\,(x - xcenter_i)\right] \tag{3.23}$$

where $H0 = \log(h)\big|_{xcenter_i}$ and *H1* and *H2* are given in step 2.

If we put these Taylor series approximations and flow function $f_i = A_i x + b_i$ for the ith particle we get the following result

$$\begin{aligned}&\left(G1 + \left[G2\,(x - xcenter_i)\right]^T + \lambda \times H1 + \left[H2\,(x - xcenter_i)\right]^T\right)\left(A_i x + b_i\right)\\&+ H0 + \left[H1\,(x - xcenter_i)\right] + \left[(x - xcenter_i)^T\ H2\,(x - xcenter_i)\right] - \frac{\partial \log K(\lambda)}{\partial \lambda}\\&= -Tr\left(A_i\right)\end{aligned} \tag{3.24}$$

We can use themethod of*separation of variables*to find A_i and b_i by equating like coefficients of x and terms that are quadratic in x in the above formula. By doing so, we get the following expressions for A_i and b_i

$$\begin{aligned}A_i &= -\frac{1}{2}\left(G2 + \lambda\, H2^T\right)H2\\b_i &= -(G2 + \lambda\, H2^T)\ ((G1{*}A_i\text{-}xcenter_i^T\ G2\ A_i + \lambda\, H1\ A_i\\&- \lambda\ xcenter_i^T\ H2^T\ A_i + H1 - \frac{1}{2}\,xcenter_i^T\ H2)^T - \frac{1}{2}\,H2\,xcenter_i)\end{aligned} \tag{3.25}$$

After that step, each particle is flowed or uptaded using the following ODE

$$\frac{dx}{d\lambda} = f_i(x,\lambda) = A_i x + b_i \text{ with intial condition } x_0 = xcenter_i$$

This ODE can be solved using Euler method.

After all particles are flowed, we found the Gaussian sum *a posteriori* pdf which is explained in the 3rd step of filtering steps. To eliminate the infinite speed particles we define a threshold value and we found the mean of the *a posteriori* pdf. Then we calculate the euclidian distances between this mean value and all flowed particles as

$$e_i = \left\| xflowed_i - xmean \right\| \tag{3.26}$$

where $xflowed_i$ is the ith flowed particle and $xmean$ is the mean of the *a posteriori* pdf. We choose the particles where their euclidian distances are smaller than the threshold (TH) value

$$e_i < TH \tag{3.27}$$

Estimated mean value of the state can be found by using the survived particles

$$xest = \frac{1}{L}\sum_{i=1}^{L} xflowed_i \tag{3.28}$$

where L is the number of survived particles.

CHAPTER 4

SIMULATION RESULTS

In order to assess the performance of the developed filters, firstly we implemented them in a nonlinear system with a single state variable. The performance for higher dimensional nonlinear systems incorporating non-Gaussian measurement noise are also investigated by using two radar tracking scenarios.Developed filters are compared with the Particle Filter (PF), UKF and CMKF in these simulations. The simulations are carried out on a computer with Intel Core i3 CPU and 1.8 GB RAM using MATLAB version 7.9.0.529.

A. Nonlinear system with a single state variable

Consider the following continuous-time nonlinear system with a discrete-time measurement model.

$$dx = \left[-\frac{x}{2} + \frac{25x}{(1+x^2)} + 8\cos(1.2t) \right] dt + dw(t)$$
$$y_k = \frac{x_k^3}{120} + v_k \tag{4.1}$$

w is a Wiener process with Gaussian increments whose intensity is taken as 10. Sampling period for measurements is taken as T=1 s. v_k is the Gaussian measurement noise ~N(0,1). Simulated data are generated using the Euler-Maruyama method [20] with 1000 steps per second and the simulation lasts for 100s. Two configurations are used while implementing the UKF-aided Gaussian Sum Filter. The first one uses 4 Gaussians while approximating the *a priori* pdf and 5 Gaussians while approximating the *a posteriori* pdf. The second one uses 40

Gaussians while approximating the *a priori* pdf and 50 Gaussians while approximating the *a posteriori* pdf. Totally 12 grid points are used in the first configuration and 190 grid points are used in the second configuration. The parameters of the filter is as follows; In the first configuration c=1, l=0, v=1/7 for *a posteriori* pdf. For *a priori* pdf c=1/5, l=1, v=1 for n=10 Euler discretization steps, c=1/8 l=1, v=1 for n=20 Euler discretization steps and c=1/10 l=1, v=1 for n=5 Euler discretization steps For different Euler discretization steps, parameters for *a priori* pdf are chosen such that the width of the grids are very close to each other. In the second configuration c=100, l=0, v=1/10 for *a priori* pdf and c=50, l=0, v=1/20 for *a posteriori* pdf and n=10 Euler discretization steps are used. While implementing UKF-aided Gaussian Sum Filter with Particle Flow 4 Gaussians are used while approximating the *a priori* pdf and the parameters of the filter are c=1/30, l=1, v=1 for *a priori* pdf and . n=10 Euler discretization steps are used. Matlab "quad" function is used to approximate the *likelihood* as a Gaussian. Also UKF and Sequential Importance Sampling (SIR) PF are implemented on the given system for comparison. n=10 Euler discretization steps. Estimation errors are given in Table 1 in terms of the Mean of Mean Absolute Error (MMAE) together with Computation Loads (normalized with respect to UKF) after 300 Monte Carlo runs.

$$MMAE = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T}\left[\sum_{t=1}^{T}\left|x_{true}^{i}(t) - x_{est}^{i}(t)\right|\right] \tag{4.2}$$

where T=100 s and N=300 runs.

Figure 2 gives the Mean of Absolute Error (MAE) after 300 Monte Carlo runs.

$$MAE(t) = \frac{1}{N}\sum_{i=1}^{N}\left|x_{true}^{i}(t) - x_{est}^{i}(t)\right| \tag{4.3}$$

Table 1 Comparison of MMAE and computation load

Filter No	Filters	Mean of Mean Absolute Error (MMAE)	Average Computation Load for One Period (Unitless)
1	UKF	1.68	1
2	SIR Particle Filter with 1000 particles	0.95	276.8
3	SIR Particle Filter with 200 particles	0.96	55.7
4	UKF-aided Gaussian Sum Filter with 40+50 Gaussians	0.94	78.9
5	UKF-aided Gaussian Sum Filter with 4+5 Gaussians n=20 discretization steps	1.14	15.2
6	UKF-aided Gaussian Sum Filter with 4+5 Gaussians n=10 discretization steps	1.16	8.9
7	UKF-aided Gaussian Sum Filter with 4+5 Gaussians n=5 discretization steps	1.22	5.1
8	UKF-aided Gaussian Sum Filter with Particle Flow with 4 Gaussians	1.12	20.8

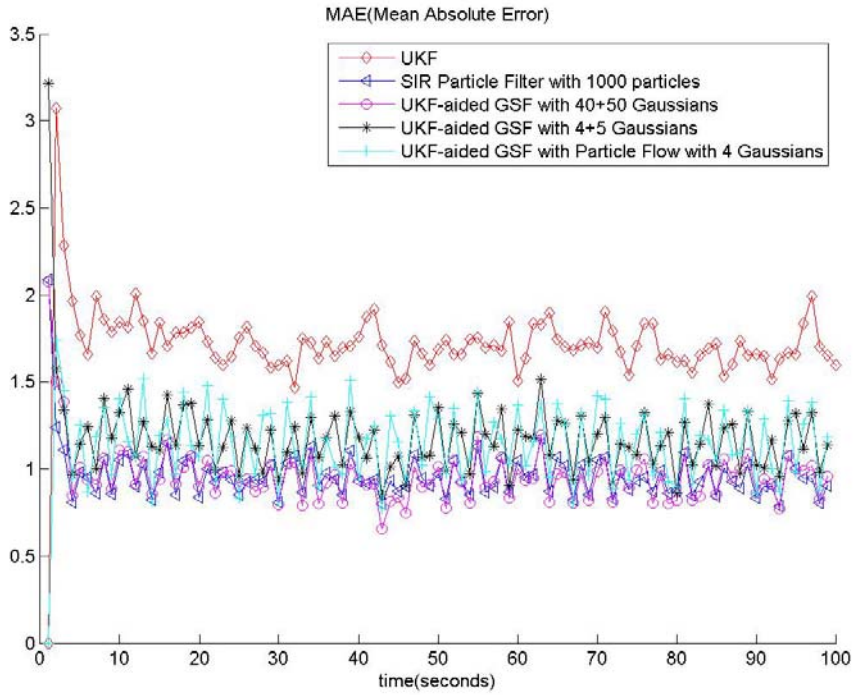

Figure 2 MAE after 300 Monte Carlo runs

For the system with a single state variable, reasonable accuracy can be achieved with about one hundred particles in the particle filter [21]. However, for obtaining comparable accuracy in higher dimensions, the number of particles increases extensively.

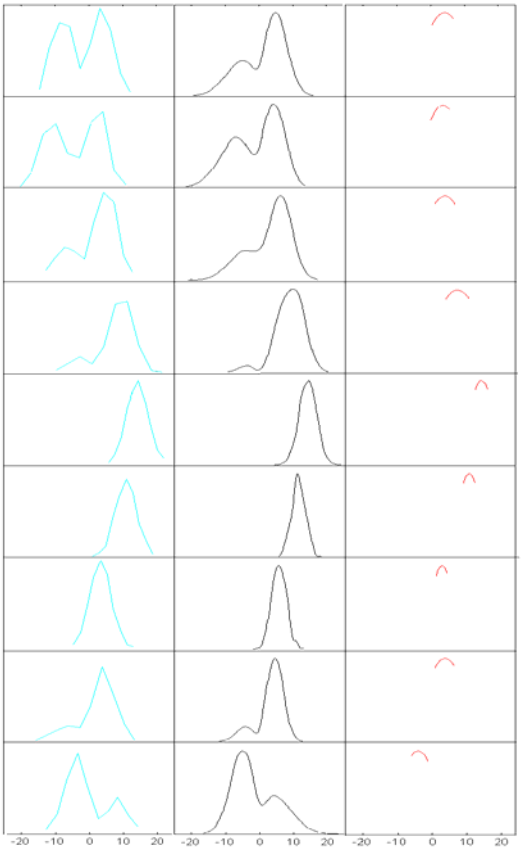

Figure 3.Unnormalized approximations for the *a priori* pdfs during the first 10 seconds of a sample Monte Carlo run: (a) Almost exact pdfs; (b) Approximate pdfs with UKF-aided Gaussian Sum Filter with 40+50 Gaussians; (c) Approximate pdfs with UKF-aided Gaussian Sum Filter with 4+5 Gaussians.

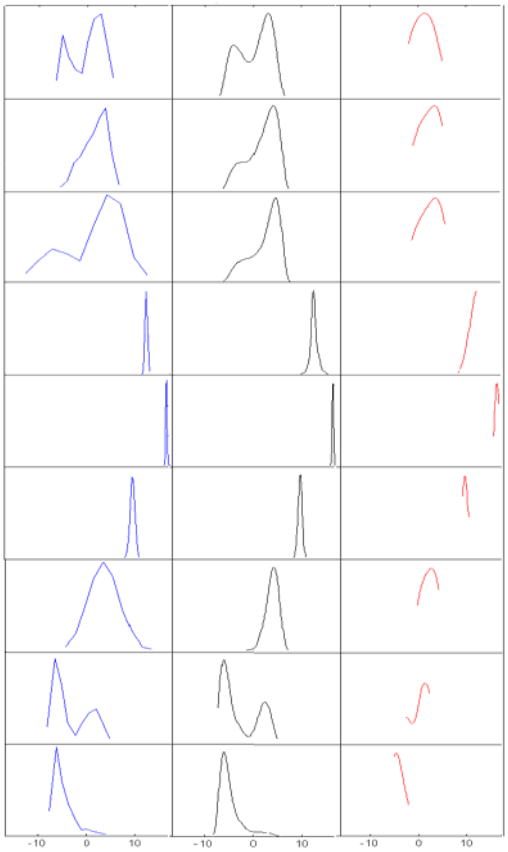

Figure 4. Unnormalized approximations for the *a posteriori* during the first 10 seconds of a sample Monte Carlo run: (a) Almost exact pdfs; (b) Approximate pdfs with UKF-aided Gaussian Sum Filter with 40+50 Gaussians; (c) Approximate pdfs with UKF-aided Gaussian Sum Filter with 4+5 Gaussians.

In Fig.s. 3 and 4, the graphs in the first columns are the almost exact unnormalized *a priori* and *a posteriori* pdfs obtained by SIR PF simulations

involving 1000 particles. The second and third columns contain the graphs of the local approximations of the unnormalized *a priori* and *a posteriori* pdfs over different grids which use UKF-aided Gaussian Sum Filter. As it can be seen from Fig.s3 and 4, UKF-aided Gaussian Sum Filter can successfully approximate the pdfs in the specified areas. For the configuration corresponding to approximation with a few Gaussians (Columns (c)), only a limited portion of the pdfs close to the mean are estimated, with low computational load. Since approximations are carried out in informationally meaningful areas predicted by the UKF, it turns out that the error levels are acceptable.

B Radar Tracking Scenario 1

A maneuvering target moving according to Curvilinear Motion Model (CMM) is simulated. The simulated paths are generated using a continuous time polar CMM. The differential equation for the continuous-time polar CMM is given as follows;

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{V} \\ \dot{\phi} \\ \dot{a}_t \\ \dot{a}_n \end{bmatrix} = \begin{bmatrix} V\cos\phi \\ V\sin\phi \\ a_t \\ \frac{a_n}{V} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} w(t) \tag{4.4}$$

where (x,y),V,φ are the target position in Cartesian coordinates, speed, and heading angle, and a_t, a_n denote the target tangential (along-track) and normal (cross track) accelerations in the horizontal plane. This is the generic model [22] for maneuvering targets where

1. $a_n=0$, $a_t=0$ — rectilinear, constant velocity (CV) motion;

2. $a_n=0$, $a_t\neq 0$ — rectilinear, accelerated motion (constant acceleration (CA) motion if a_t = constant);

3. $a_n\neq 0$, $a_t=0$— circular, constant speed motion (coordinated turn (CT) motion if a_n=constant).

are generated. The target motion starts from the location (200m, 200m) with 0.1 m/s speed, 4.30 radians heading angle and a_t=0.01 m/s^2 tangential and a_n=0.01 m/s^2 normal accelerations w(t) is zero mean Gaussian noise with Standard deviation σ_w=3.2x10^{-4} m/s^2. The motion lasts for 500s. During each time period (T=1s), we simulate the motion using 1000 steps of Euler-Maruyama method. The radar measurements obtained by the following 2D radar measurement model by sampling the states every second.

$$y(k)=\begin{bmatrix} r_k \\ \beta_k \end{bmatrix}=h(x_k)+v_k=\begin{bmatrix} \sqrt{(x_k-x_{obs})^2+(y_k-y_{obs})^2} \\ \tan^{-1}\dfrac{(y_k-y_{obs})}{(x_k-x_{obs})} \end{bmatrix}+\begin{bmatrix} v_r(k) \\ v_\beta(k) \end{bmatrix} \quad (4.5)$$

where v_r and v_β are independent zero mean Gaussian white noises $\sigma_{v_r}=0.24m$ and $\sigma_{v_\beta}=0.0084rad$. The stationary observer (radar) is located at (10 km,10 km). Target tracking is carried out by using the UKF-aided Gaussian Sum Filter and UKF-aided Gaussian Sum Filter with Particle Flow and the results are compared to those obtained by UKF. The UKF used here has two versions. One uses the continuous-time polar CMM and the other uses continuous-time Cartesian CMM as dynamical models. The continuous-time Cartesian CMM is given as follows

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{y} \\ \ddot{y} \\ \dot{a}_t \\ \dot{a}_n \end{bmatrix}=\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \dfrac{\dot{x}}{\sqrt{\dot{x}^2+\dot{y}^2}} & \dfrac{\dot{y}}{\sqrt{\dot{x}^2+\dot{y}^2}} \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \dfrac{\dot{y}}{\sqrt{\dot{x}^2+\dot{y}^2}} & -\dfrac{\dot{x}}{\sqrt{\dot{x}^2+\dot{y}^2}} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ a_t \\ a_n \end{bmatrix}+\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}w(t) \quad (4.6)$$

Where $x, y, \dot{x}, \dot{y}$ are the target positions and velocities in Cartesian coordinates. a_t and a_n are as the acceleration components given in the continuous-time polar CMM.

Continuous-time Polar and Cartesian CMM are used with UKF-aided Gaussian Sum Filter and UKF-aided Gaussian Sum Filter with Particle Flow. The results are compared with those obtained by UKF using continuous-time Polar and Cartesian CMM.

While using continuous-time Polar CMM, initial state and covariance values are given as follows.

$$x_0 = \begin{bmatrix} x_{obs} + r_0 \cos(\beta_0) \\ y_{obs} + r_0 \sin(\beta_0) \\ 0.001 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad P_0 = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1\times10^{-10} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1\times10^{-10} \end{bmatrix} \tag{4.7}$$

While using continuous-time Cartesian CMM, initial state and covariance values are given as follows.

$$x_0 = \begin{bmatrix} x_{obs} + r_0 \cos(\beta_0) \\ 0.001 \\ y_{obs} + r_0 \sin(\beta_0) \\ 0.001 \\ 1\times10^{-10} \\ 1\times10^{-10} \end{bmatrix} \quad P_0 = \begin{bmatrix} 100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1\times10^{-10} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1\times10^{-10} \end{bmatrix} \tag{4.8}$$

Small initial values are assigned to a_t and a_n. If $a_t = a_n = 0$, UKF fails to work properly with continuous-time Cartesian CMM.

n=10 Euler discretization steps (per second) are used in all filters.

While using continuous-time polar CMM, 6 Gaussians are used to approximate the *a priori* pdf and 10 Gaussians are used to approximate the *a posteriori* pdf in UKF-aided Gaussian Sum Filter. Totally 26 grid points are used. The parameters of the filter are as follows; c=1/800, l=8000, v=1 for *a priori* pdf and c=1/100, l=0, v=1 for *a posteriori* pdf. Also 6 Gaussians are used to approximate the *a priori* pdf in UKF-aided Gaussian Sum Filter with Particle Flow and unbiased conversion is used while approximating the likelihood as a Gaussian. The parameters of the filter are c=1/800, l=8000, v=1 for *a priori* pdf.

While using continuous-time Cartesian CMM, same number of Gaussians as in the continuous-time Polar CMM case are used in UKF-aided Gaussian Sum Filter and UKF-aided Gaussian Sum Filter with Particle Flow. Only the parameters are changed. In the UKF-aided Gaussian Sum Filter c=1/500, l=5000, v=1 for *a priori* pdf and c=1/100, l=0, v=1 for *a posteriori* pdf are used. In the UKF-aided Gaussian Sum Filter with Particle Flow c=1/500, l=5000, v=1 for *a priori* pdf are used.

After 300 Monte Carlo runs, the results for the Root Means Square (RMS) Position Error during 500 s are shown in Fig.s. 5, 6. Root Means Square (RMS) Position Error at a specific time is given with the following formula.

$$RMS \text{ Position Error}(t) = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\frac{1}{2}\left(x_{true}^{i}(t)-x_{est}^{i}(t)\right)^2+\left(y_{true}^{i}(t)-y_{est}^{i}(t)\right)^2} \qquad (4.9)$$

where N=300 runs.

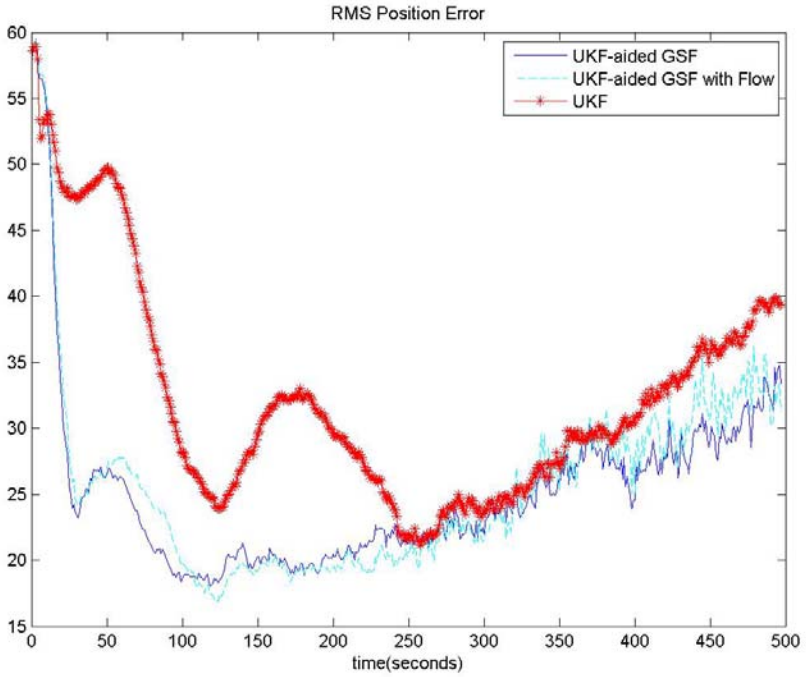

Figure.5. RMS position error while using continuous-time polar CMM as dynamical model

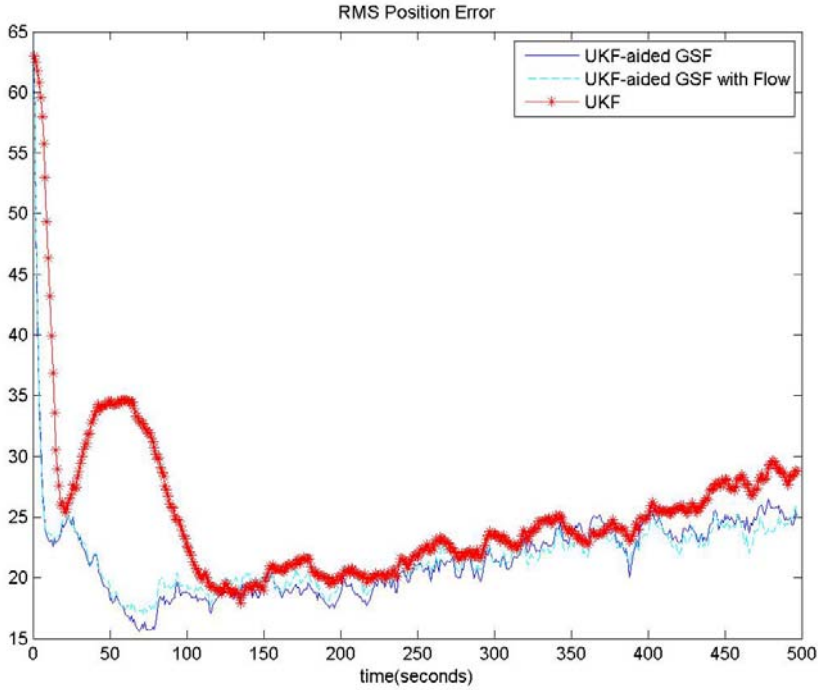

Figure.6. RMS position error while using continuous-time Cartesian CMM as dynamical model

Using continuous-time Cartesian CMM gives better results than the continuous-time polar CMM. A possible reason for using continuous-time Cartesian CMM, which gives better results, is that continuous-time Cartesian CMM has nonlinear terms only in two state equations, whereas continuous-time polar CMM has nonlinear terms in three state equations. Computational loads (normalized with respect to UKF) of the compared filters are given in Table 2.

Table 2 Average Computational Load for Radar Tracking Scenario 1

Filter No	**Filters**	**Average Computation Load for One Period (Unitless)**
1	UKF using continuous-time polar CMM	1
2	UKF using continuous-time Cartesian CMM	1
3	UKF aided Gaussian Sum Filter using continuous-time polar CMM	28.25
4	UKF aided Gaussian Sum Filter using continuous-time Cartesian CMM	26.24
5	UKF aided Gaussian Sum Filter with Particle Flow using continuous-time polar CMM	27.6
6	UKF aided Gaussian Sum Filter with Particle Flow using continuous-time Cartesian CMM	25.3

As it can be seen from Table 2, the computational load of the developed filters are reasonable.

C Radar Tracking Scenario 2

In this scenario the performance of the developed filter is investigated for target tracking with glint noise in its measurement model. A target moving according to Constant Velocity (CV) model is simulated. The simulated paths are generated using discretized version of the continuous-time CV model. Continuous-time CV model is given as follows;

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} w(t) \tag{4.10}$$

where $x, y, \dot{x}, \dot{y}$ are the target position and velocities in Cartesian coordinates. w(t) is a Gaussian white noise with w(t)~N(0, σ_w^2) Discrete-time version of continuous-time CV model is given as follows;

$$x_{k+1} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} x_k + w_k$$

$$\text{cov}(w_k) = \begin{bmatrix} Q & 0 \\ 0 & Q \end{bmatrix} Q = \begin{bmatrix} \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ \frac{1}{2}T^2 & T \end{bmatrix} \sigma_w^2 \tag{4.11}$$

Sampling period is chosen as T=1 s. The target starts moving from (200 m, 200 m) with 0.01 m/s velocity components. The uncertainty in velocities is modeled by Gaussian zero mean noise with standard deviation $\sigma_w = 0.1\,\text{m/s}^2$. The motion lasts for 500s. The radar measurements are obtained as:

$$y(k) = \begin{bmatrix} r_k \\ \beta_k \end{bmatrix} = h(x_k) + v_k = \begin{bmatrix} \sqrt{(x_k - x_{obs})^2 + (y_k - y_{obs})^2} \\ \tan^{-1} \dfrac{(y_k - y_{obs})}{(x_k - x_{obs})} \end{bmatrix} + \begin{bmatrix} v_r(k) \\ v_\beta(k) \end{bmatrix} \tag{4.12}$$

where v_r is zero mean Gaussian noise with $\sigma_{v_r} = 0.24$ m

v_β is chosen as glint noise, which is modeled via the mixture of two Gaussian distributions and its unnormalized pdf is given as

$$p_{v_\beta}(v_\beta) = 0.25e^{-\frac{1}{2}\frac{(v_\beta-\mu 1_\beta)^2}{\sigma 1_\beta^2}} + 0.75e^{-\frac{1}{2}\frac{(v_\beta-\mu 2_\beta)^2}{\sigma 2_\beta^2}} \quad (4.13)$$

In the simulations, the glint noise is used with two different value sets

Value set1 has values $\mu 1_\beta = \mu 2_\beta = 0$ and $\sigma 1_\beta = 0.01$ rad. and $\sigma 2_\beta = 0.0071$ rad. Value set2 has values $\mu 1_\beta = \mu 2_\beta = 0$ and $\sigma 1_\beta = 0.0114$ rad. and $\sigma 2_\beta = 0.0063$ rad.

The glint noise has zero mean and standard deviation $\sigma_{v_\beta} = 0.0079$ rad. for both of the value sets. The stationary observer is located at (10 km,10 km). The developed filters are implemented to track the target and the results are compared with those obtained by a CMKF. While using the measurement model in CMKF, we assume that v_β is a zero-mean Gaussian noise with $\sigma_{v_\beta} = 0.0079$ rad.

Continuous-time CV model is used in the developed filters. Glint noise is used as *likelihood* in the calculations. n=10 steps of Euler discretization(per second) is used while solving Fokker-Planck equation. In the UKF-aided Gaussian Sum Filter 10 Gaussians are used while approximating the *a priori* and *a posteriori* pdfs and totally 40 grid points are used. The parameters of the filter are as follows; c=1/800, l=8000, v=1 for *a priori* pdf and c=1/100, l=0, v=1 for *a posteriori* pdf. In the UKF-aided Gaussian Sum Filter with Particle Flow 10 Gaussians are used while approximating the *a priori* pdf. The parameters of the filter are c=1/800, l=8000, v=1 for *a priori* pdf. Since *likelihood* has 2 components because of glint noise, unbiased conversion is applied to every component of *likelihood* in UKF-aided Gaussian Sum Filter with Particle Flow. Developed filters and CMKF are started with the following initial state and covariance values

$$x_0 = \begin{bmatrix} x_{obs} + r_0 \cos(\beta_0) \\ 0 \\ y_{obs} + r_0 \sin(\beta_0) \\ 0 \end{bmatrix}$$

$$P_0 = 100 * \sigma_w^2 * \begin{bmatrix} \frac{1}{3}T^3 & \frac{1}{2}T^2 & 0 & 0 \\ \frac{1}{2}T^2 & T & 0 & 0 \\ 0 & 0 & \frac{1}{3}T^3 & \frac{1}{2}T^2 \\ 0 & 0 & \frac{1}{2}T^2 & T \end{bmatrix} = \begin{bmatrix} .33 & .5 & 0 & 0 \\ .5 & 1 & 0 & 0 \\ 0 & 0 & .33 & .5 \\ 0 & 0 & 50 & 1 \end{bmatrix} \quad (4.14)$$

The developed filter and CMKF are compared with the generated data using both value sets of glint noise.

After 300 Monte Carlo runs, the results for the RMS position error during 500 s are shown in Fig.s. 7, 8.

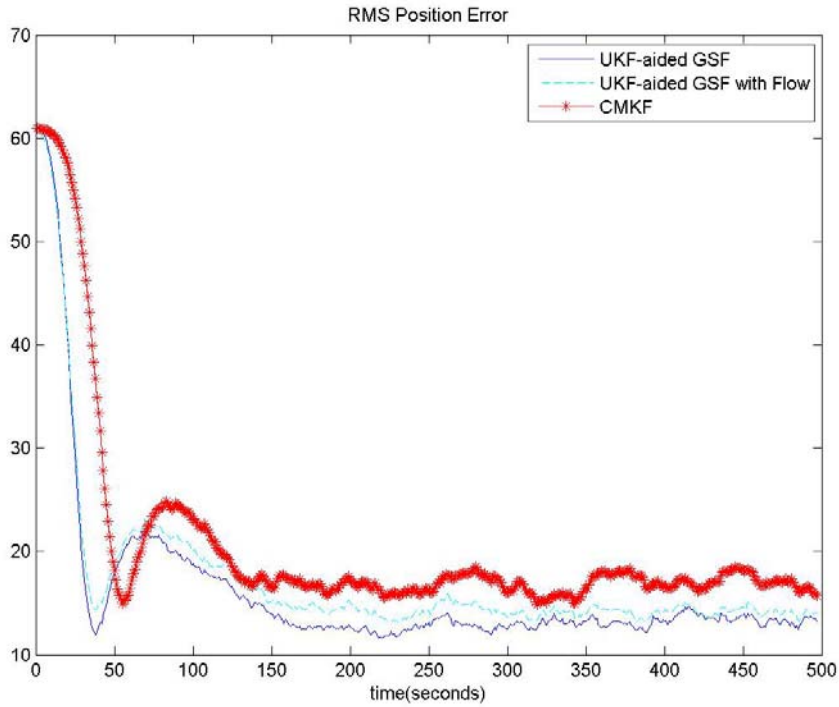

Figure.7. RMS position error while using value set 1

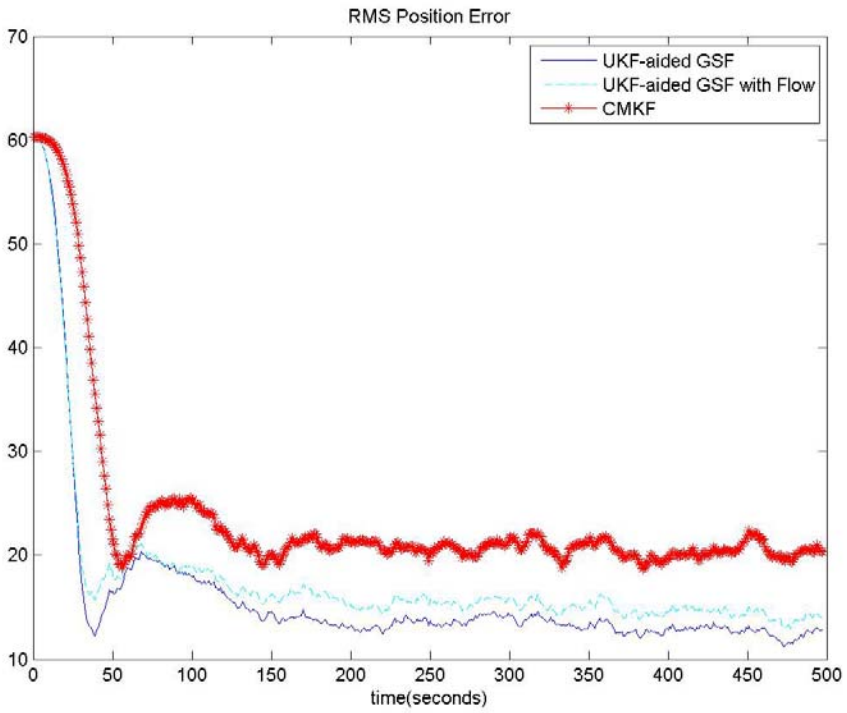

Figure.8 RMS position error while using value set 2

As it can be seen from the figures, the developed filters take the advantage of using directly the non-Gaussian glint noise in its calculations and make less error than the CMKF. Also as the variance difference of Gaussians used in glint noise gets larger, the error level of CMKF increases. The error levels of the developed filters are almost same when applied to data generated by different value sets.

Computational loads (normalized with respect to CMKF) of the compared filters are given in Table 3.

Table 3 Average Computational Load for Radar Tracking Scenario 2

Filter No	**Filters**	**Average Computation Load for One Period (Unitless)**
1	CMKF using discrete-time version of continuous-time CV model	1
2	UKF aided Gaussian Sum Filter using continuous-time CV Model	492.6
3	UKF aided Gaussian Sum Filter with Particle Flow using continuous-time CV Model	493.5

Since CMKF only involves conversion and standard Kalman Filter steps without discretization, it has very small computational load. Also computational load of the developed filters are reasonable. Since a linear dynamical model (CV model) is used, the prediction step of UKF is not needed and the prediction step of Kalman Filter is used while approximating the *a priori* pdf.

CHAPTER 5

CONCLUSIONS

An approximate estimation method which is named as UKF-aided Gaussian Sum Filter is proposed for nonlinear systems represented in terms of a continuous-time dynamical model and discrete-time measurement model. The approach tries to approximate the *a priori* and *a posteriori* pdfs in the filtering steps using weighted Gaussian sum representations. The method has the ability to approximate the pdfs in a given search area. As it is shown in the one dimensional example, almost exact pdfs can be captured using the developed filter by adjusting the grid used in the search area. Also as a second approximate estimation method Particle Flow is integrated with the UKF-aided Gaussian Sum Filter and a new flow function based on Taylor series approximation is used. Both filters can also be used in higher dimensional systems and under non-Gaussian noises. The simulation results for these cases indicate that developed filters gives smaller error values with a reasonable computational load when compared with some of the existing nonlinear filtering methods.

REFERENCES

[1] Kovacevic B., Durovic Z.: 'Optimum recursive linear estimation:Kalman Filtering': 'Fundamentals of Stochastic Signals, Systems and Estimation Theory:With Examples' Springer Press 2nd edn., pp. 266, 2008

[2] Bagnaschi L.L., "A Comparative Study of Nonlinear Tracking Algorithms" A dissertation submitted to the Swiss Federal Institute of Technology Zurich, 1993

[3] Schön T. "On Computational Methods for Nonlinear Estimation" A dissertation submitted to the Department of Electrical Engineering of Linköping University, 2003

[4]Daum F. "Exact Finite-Dimensional Nonlinear Filters", IEEE Transactions on Automatic Control, Vol. AC-31, no. 7, pp.616-622, July 1984

[5] Daum F. "Nonlinear Filters: Beyond the Kalman Filter" IEEE Aerospace and Electronic Systems Magazine, Vol. 20, pp. 57–69, Aug. 2005

[6] Cubkow A. and Spoglona B., "Generalize Wiener Process and Kolmogorov's Equation for diffusion induced by non-Gaussian noise source" Fluctiation and Noise Letters Vol. 5, 2005

[7] Lerro D. and Bar-Shalom Y., "Tracking with debiased consistent converted measurements versus EKF," IEEE Trans. on Aeros. and Elec. Systems, Vol. 29, pp. 1015-1022, July 1993

[8] Mo Longbin, Song Xiaoquan, Zhou Yiyu, Sun Zhong Kang, Bar-Shalom Y."Unbiased Converted Measurements for Tracking"IEEE Transactions onAerospace and Electronic Systems Vol. 53 Page(s): 1023–1027, 1998

[9] Zhansheng Duan, Chongzhao Han, Rong Li X. "Comments on "Unbiased Converted Measurementsfor Tracking"IEEE Transactions on Aerospace and Electronic Systems, Vol: 40 , Issue: 4, 2004

[10] Julier S.J., Uhlmann J.K. "A New Extension of the Kalman Filter to Nonlinear Systems" InProc. of AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls, 1997

[11]SimoS. "On Unscented Kalman Filtering for State Estimation of Continuous-Time Nonlinear Systems" IEEE Transactions on Automatic Control Vol. 53 Page(s): 1631 – 1641, 2007

[12] Van der Merwe R., Doucet A., de Freitas J. F. G., and Wan E., "The Unscented Particle Filter", Adv. Neural Inform. Process. Syst., 2000

[13] Daum F., Huang J.,Noushin A. “Coulomb’s law Particle Flow for Nonlinear Filters” Proc. SPIE 8137, Signal and Data Processing of Small Targets, 2011

[14] Daum F., Huang J. “Numerical experiments for nonlinear filters with exact particle flow induced by log-homotopy” Proc. SPIE 7698, Signal and Data Processing of Small Targets, 2010

[15] Daum F., Huang J., Noushin A. “Numerical experiments for Coulomb's law particle flow for nonlinear filters ” Proc. SPIE 8137, Signal and Data Processing of Small Targets, 2011

[16]Daum F., Huang J. "Exact particle flow for nonlinear filters: seventeen dubious solutions to a first order linear underdeteremined PDE," Proceedings of IEEE Conference on Signals, Systems & Computers, Asilomar California, Nov. 2010

[17] Chen R. and Liu J. “Mixture Kalman filters,” J. Royal Stat. Soc. B,Vol. 62, pp. 493–508, 2000

[18] Singla P., Singh T., Scott P.D. “Adaptive Gaussian Sum Filter for Nonlinear Bayesian Estimation”, IEEE Transactions on Automatic Control,Vol. 56, pp.2151-2156, Sept. 2011

[19] Mongillo M. “Choosing basis functions and shape parameters for radial basis function methods,” SIAM Undergraduate Research Online, Sept 2011

[20] Desmond J. Higham, “An Algorithmic Introduction to Numerical Simulation of Stochastic Differential Equations”, SIAM REVIEW, Vol. 43,No. 3,pp. 525–546, 2001

[21] Daum F. and Huang J., “Curse of Dimensionality and Particle Filters,” Proceedings of the IEEE Aerospace Conference, Vol. 4, pp. 1979–1993, March 2003

[22] Rong Li X. and Jilkov V.P. “A Survey of Maneuvering Target Tracking: Dynamic Models” IEEE Transactions on Aerospace and Electronic Systems, Vol: 40,pp: 1333 – 1364, 2003

CURRICULUM VITAE

Murat GÖKCE

Address : 1335 Str. No:47/20 Öveçler Ankara

Mobile Phone : +90 533 647 65 67

E-mail : mmgokce@yahoo.com

Date of birth : 10 August 1978, Çankırı, Turkey

Marital Status : Married

EDUCATION

Master of Science: Istanbul Technical University
Electronics & Communication (2001-2004)
Thesis: Modelling Propagation of EM waves in Chiral Waveguides(Simulation using MATLAB)

Graduate: Istanbul Technical University
Electronics Communication (1996-2001)
Thesis: Modelling Propagation of EM waves in Planar Waveguides(Simulation using MATLAB)

WORK EXPERIENCE

COMPANY : TUBİTAK-TEYDEB. / Ankara (February 2011 - Today)

POSITION : Scientific Programs Expert

PROJECTS: Working in BILTEG(Information Technologies Group), Responsible from the R&D projects that are applied to support programs by industy companies.

COMPANY: HAVELSAN AŞ. / Ankara (July 2005 - February 2011)

POSITION : Specialist Software Engineer

PROJECTS: Worked in Desktop Simulation Software Group of the Peace Eagle Project and made the design, implementation and testing of the ESM(Electronic Support Measures) Subsystem simulation.

Worked in Build Team as a Build master.

Worked in Multi Sensor Fusion Team. Developed

algorithms in MATLAB about target tracking and classification. Wrote the related white papers(technical documents) about the algorithms.

Developed Multi Sensor Integration(MSI) simulation in C++ working incorparative with the multiple simulated sensors data (Radar/IFF, ESM, Datalink) and Mission Computing Subsytem(MCS) of the AEW&C(AWACS)

COMPANY : BELBİM AŞ./İSTANBUL(October 2001-November 2004)

POSITION : Software Engineer

PROJECTS: Worked in MIS department of ISKI(Istanbul Water and Sewage Administration)

- Experience in Oracle Developer tools (Forms 6i and Reports 6i)
- ISKI PACS(personel attandance check system) and wastewater system using Oracle Developer 6i
- ISKI Workflow automation using Oracle

Web based document archive system using Oracle Jdeveloper 10g

COMPUTER SKILLS

OS: Windows, Sun Solaris (User Level).

Languages, Modelling and Software Life Cycle Tools: C,C++, Java,MATLAB, UML, PL/SQL, Rational Rose UML CASE Tool, Clearcase, DOORs, Purify, Parasoft, Quantify

COURSES&TRAININGS

Sensor Fusion and Target Tracking (Prof.Dr. Mübeccel Demirekler, Prof.Dr. Mustafa Kuzuoğlu), Havelsan A.Ş(2009-2010)

Corba Development Using ORB Express for C++ - Havelsan A.Ş(2006)

ADAPTIVE Communications Environment(ACE)–Havelsan A.Ş(2006)

Java and J2EE using Oracle Jdeveloper – Hitit Computer Sys.(2weeks) (2004)

C/C++ Programming Language– C&System Prog.Asso.(150+150 hours)(2002)

Oracle8i Developer– Oracle Turkey (2weeks)(2002)

Oracle8i SQL- PL/SQL– Oracle Turkey(1 week)(2002)

Interconnecting Cisco Network Devices – Netron (1 week)(2001)

PUBLICATONS

Gökce M., Kuzuoglu M., "Continuous-time Nonlinear Estimation Filters Using UKF-aided Gaussian Sum Representations", FUSION 2013 Conference, İstanbul (2013).

Vuran N., Gökce M., "HİK Sistemi Görev Bilgisayarı için Çoklu Sensör Füzyonu Yazılımı", SAVTEK , Ankara (2010).

Gökce M., Kuzuoglu M., "UKF-aided Gaussian Sum Filter", IET Radar Sonar and Navigation (2013)(in review).