Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models

Rudolph van der Merwe & Eric Wan
OGI School of Science & Engineering
Oregon Health & Science University
Beaverton, Oregon, 97006, USA
{*rvdmerwe,ericwan*}*@ece.ogi.edu*

Abstract

Probabilistic inference is the problem of estimating the hidden states of a system in an optimal and consistent fashion given a set of noisy or incomplete observations. The optimal solution to this problem is given by the recursive Bayesian estimation algorithm which recursively updates the posterior density of the system state as new observations arrive online. This posterior density constitutes the complete solution to the probabilistic inference problem, and allows us to calculate any "optimal" estimate of the state. Unfortunately, for most real-world problems, the optimal Bayesian recursion is intractable and approximate solutions must be used. Within the space of approximate solutions, the extended Kalman filter (EKF) has become one of the most widely used algorithms with applications in state, parameter and dual estimation. Unfortunately, the EKF is based on a sub-optimal implementation of the recursive Bayesian estimation framework applied to Gaussian random variables. This can seriously affect the accuracy or even lead to divergence of any inference system that is based on the EKF or that uses the EKF as a component part. Recently a number of related novel, more accurate and theoretically better motivated algorithmic alternatives to the EKF have surfaced in the literature, with specific application to state estimation for automatic control. We have since generalized these algorithms, all based on derivativeless *statistical linearization*, to a family of filters called *Sigma-Point Kalman Filters* (SPKF) and successfully expanded their use within the general field of probabilistic inference, both as stand-alone filters and as subcomponents of more powerful sequential Monte Carlo filters (particle filters). We have consistently shown that there are large performance benefits to be gained by applying Sigma-Point Kalman filters to areas where EKFs have been used as the de facto standard in the past, as well as in new areas where the use of the EKF is impossible. This paper is a summary of that work that has appeared in a number of separate publications as well as a presentation of some new results that has not been published yet.

1 Introduction

Sequential probabilistic inference (SPI) is the problem of estimating the hidden states of a system in an optimal and consistent fashion as set of noisy or incomplete observations becomes available online. Examples of this include vehicle navigation and tracking, time-series estimation and system identification or parameter estimation, to name but a few. This paper focuses specifically on discrete-time nonlinear dynamic systems that can be described by a *dynamic state-space model* (DSSM) as depicted in Figure 1. The hidden system state $\mathbf{x}_k$, with initial distribution $p(\mathbf{x}_0)$, evolves over time (k is the discrete time index) as an indirect or partially observed first order Markov process according to the conditional probability density $p(\mathbf{x}_k|\mathbf{x}_{k-1})$. The observations $\mathbf{y}_k$ are conditionally independent given the state and are generated according to the probability density $p(\mathbf{y}_k|\mathbf{x}_k)$. The DSSM can also be written as a set of nonlinear system equations

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}, \mathbf{u}_k; \mathbf{W}) \quad \text{(process equation)} \tag{1}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{n}_k; \mathbf{W}) \quad \text{(observation equation)} \tag{2}$$

where $\mathbf{v}_k$ is the process noise that drives the dynamic system through the nonlinear state transition function $\mathbf{f}$, and $\mathbf{n}_k$ is the observation or measurement noise corrupting the observation of the state through the nonlinear observation function $\mathbf{h}$. The state transition density $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is fully specified by $\mathbf{f}$ and the process noise distribution $p(\mathbf{v}_k)$, whereas $\mathbf{h}$ and the observation noise distribution $p(\mathbf{n}_k)$ fully specify the observation likelihood $p(\mathbf{y}_k|\mathbf{x}_k)$. The exogenous input to the system, $\mathbf{u}_k$is assumed known. Both $\mathbf{f}$ and/or $\mathbf{h}$ are parameterized via the parameter vector $\mathbf{W}$**.**

In a Bayesian framework, the posterior filtering density $p(\mathbf{x}_k|\mathbf{Y}_k)$ of the state given all the observations $\mathbf{Y}_k = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k\}$ constitutes the complete solution to the sequential probabilistic inference problem, and allows us to calculate any "optimal" estimate of the state, such as the *conditional mean* $\hat{\mathbf{x}}_k = E\left[\mathbf{x}_k|\mathbf{Y}_k\right] = \int \mathbf{x}_k p(\mathbf{x}_k|\mathbf{Y}_k) d\mathbf{x}_k$. The optimal method to recursively update the posterior density as new observations arrive is given by the *recursive Bayesian estimation* algorithm. This approach first projects the previous posterior $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$ forward in time using the probabilistic process model, i.e.

$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) d\mathbf{x}_{k-1} \tag{3}$$

and then incorporates the latest noisy measurement using the observation likelihood to generate the updated posterior

$$p(\mathbf{x}_k|\mathbf{Y}_k) = C p(\mathbf{y}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{Y}_{k-1}). \tag{4}$$

The normalizing factor is given by

$$C = \left(\int p(\mathbf{y}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{Y}_{k-1}) d\mathbf{x}_k \right)^{-1}. \tag{5}$$

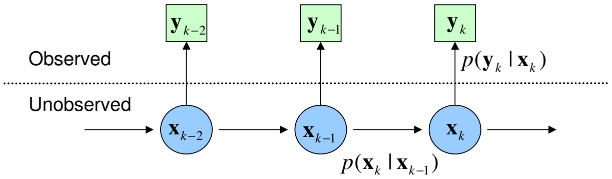


Figure 1: Dynamic state space model.

Although this is the optimal recursive solution, it is usually only tractable for *linear, Gaussian* systems in which case the closed-form recursive solution of the Bayesian integral equations is the well known *Kalman filter* [20]. For most general real-world (nonlinear, non-Gaussian) systems however, the multi-dimensional integrals are intractable and approximate solutions must be used. These include methods such as *Gaussian approximations* (extended Kalman filter [15, 10]), *hybrid Gaussian methods* (score function EKF [26], Gaussian sum filters [1]), *direct and adaptive numerical integration* (grid-based filters [33]), *sequential Monte Carlo methods* (particle filters [6, 24, 7]) and *variational methods* (Bayesian mixture of factor analyzers [11]) to name but a few.

Off all these methods, the *extended Kalman filter* (EKF) has probably had the most widespread use in nonlinear estimation and inference over the last 20 years. It has been applied successfully to problems in all areas of probabilistic inference, including state estimation, parameter estimation (machine learning) and dual estimation [29, 2, 15, 12, 10]. Even newer, more powerful inference frameworks such as *sequential Monte Carlo methods* sometimes makes use of extended Kalman filters as subcomponents [6]. Unfortunately, the EKF is based on a suboptimal implementation of the recursive Bayesian estimation framework applied to Gaussian random variables. This can seriously affect the accuracy or even lead to divergence of any inference system that is based on the EKF or that uses the EKF as a component part.

2 Gaussian Approximations

If we make the basic assumption that the state, observation and noise terms can be modeled as Gaussian random variables (GRVs), then the Bayesian recursion can be greatly simplified. In this case, only the conditional mean $\hat{\mathbf{x}}_k = E\left[\mathbf{x}_k|\mathbf{Y}_k\right]$ and covariance $\mathbf{P}_{\mathbf{x}_k}$ need to be maintained in order to recursively calculate the posterior density $p(\mathbf{x}_k|\mathbf{Y}_k)$, which, under these Gaussian assumptions, is itself a Gaussian distribution[1]. It is straightforward to show [2, 10, 23] that this leads to the recursive estimation

$$\begin{aligned}
\hat{\mathbf{x}}_k &= (\text{prediction of } \mathbf{x}_k) + \mathcal{K}_k \cdot \left[\mathbf{y}_k - (\text{prediction of } \mathbf{y}_k)\right] && (6)\\
&= \hat{\mathbf{x}}_k^- + \mathcal{K}_k \cdot \left(\mathbf{y}_k - \hat{\mathbf{y}}_k^-\right) && (7)\\
\mathbf{P}_{\mathbf{x}_k} &= \mathbf{P}_{\mathbf{x}_k}^- - \mathcal{K}_k \mathbf{P}_{\mathbf{y}_k} \mathcal{K}_k^T . && (8)
\end{aligned}$$

While this is a *linear* recursion, we have not assumed linearity of the model. The optimal terms in this recursion are given by

$$\begin{aligned}
\hat{\mathbf{x}}_k^- &= E\left[\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}, \mathbf{u}_k)\right] && (9)\\
\hat{\mathbf{y}}_k^- &= E\left[\mathbf{h}(\mathbf{x}_k^-, \mathbf{n}_k)\right] && (10)\\
\mathcal{K}_k &= E\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)^T\right] E\left[(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)^T\right]^{-1} && (11)\\
&= \mathbf{P}_{\mathbf{x}_k\mathbf{y}_k} \mathbf{P}_{\mathbf{y}_k}^{-1} && (12)
\end{aligned}$$

where the optimal prediction (i.e., prior mean at time k) of $\mathbf{x}_k$ is written as $\hat{\mathbf{x}}_k^-$, and corresponds to the expectation (taken over the posterior distribution of the state at time $k-1$) of a nonlinear function of the random variables $\mathbf{x}_{k-1}$ and $\mathbf{v}_{k-1}$ (similar interpretation for the optimal prediction $\hat{\mathbf{y}}_k^-$, except the expectation is taken over the prior distribution of the state at time k). The optimal gain term $\mathcal{K}_k$ is expressed as a function of the expected cross-correlation matrix (covariance matrix) of the state prediction error and the observation prediction error, and the expected auto-correlation matrix of the observation prediction

[1] Given a Gaussian posterior, all optimal estimates of the system state such as MMSE, ML and MAP estimates are equivalent to the mean of the Gaussian density modeling the posterior.

error[2]. Note that evaluation of the covariance terms also require taking expectations of a nonlinear function of the prior state variable.

2.1 The EKF and its flaws

The celebrated Kalman filter [20] calculates all terms in these equations exactly in the linear case, and can be viewed as an efficient method for analytically propagating a GRV through linear system dynamics. For nonlinear models, however, the *extended Kalman filter* (EKF) must be used that first linearizes the system equations through a Taylor-series expansion around the mean of the relevant Gaussian RV, i.e.

$$\begin{aligned} \mathbf{y} &= \mathbf{g}(\mathbf{x}) = \mathbf{g}(\bar{\mathbf{x}} + \boldsymbol{\delta}_{\mathbf{x}}) \\ &= \mathbf{g}(\bar{\mathbf{x}}) + \boldsymbol{\nabla}\mathbf{g}\boldsymbol{\delta}_{\mathbf{x}} + \frac{1}{2}\boldsymbol{\nabla}^2\mathbf{g}\boldsymbol{\delta}_{\mathbf{x}}^2 + \frac{1}{3!}\boldsymbol{\nabla}^3\mathbf{g}\boldsymbol{\delta}_{\mathbf{x}}^3 + \ldots \qquad (13) \end{aligned}$$

where the zero mean random variable $\boldsymbol{\delta}_{\mathbf{x}}$ has the same covariance, $\mathbf{P}_{\mathbf{x}}$, as $\mathbf{x}$. The mean and covariance used in the EKF is thus obtained by taking the first-order truncated expected value of Equation 13 (for the mean) and its outer-product (for the covariance), i.e. $\bar{\mathbf{y}} \approx \mathbf{g}(\bar{\mathbf{x}})$, and $\mathbf{P}_{\mathbf{y}}^{LIN} = \boldsymbol{\nabla}\mathbf{g}\mathbf{P}_{\mathbf{x}}(\boldsymbol{\nabla}\mathbf{g})^T$. Applying this result to Equations 6 and 8, we obtain,

$$\hat{\mathbf{x}}_k^- \approx \mathbf{f}(\hat{\mathbf{x}}_{k-1}, \bar{\mathbf{v}}, \mathbf{u}_k) \qquad (14)$$

$$\hat{\mathbf{y}}_k^- \approx \mathbf{h}(\hat{\mathbf{x}}_k^-, \bar{\mathbf{n}}) \qquad (15)$$

$$\mathcal{K}_k \approx \hat{\mathbf{P}}_{\mathbf{x}_k\mathbf{y}_k}^{LIN}\left(\hat{\mathbf{P}}_{\mathbf{y}_k}^{LIN}\right)^{-1}. \qquad (16)$$

In other words, the EKF approximates the state distribution by a GRV, which is then propagated analytically through the "first-order" linearization of the nonlinear system. For the explicit equations for the EKF see [15]. As such, the EKF can be viewed as providing "first-order" approximations to the optimal terms[3]. Furthermore, the EKF does not take into account the "uncertainty" in the underlying random variable when the relevant system equations are linearized. This is due to the nature of the first-order Taylor series linearization that expands the nonlinear equations around a *single point* only, disregarding the "spread" (uncertainty) of the prior RV. These approximations and can introduce large errors in the true posterior mean and covariance of the transformed (Gaussian) random variable, which may lead to suboptimal performance and sometimes divergence of the filter [45, 43, 44].

3 Sigma-Point Kalman Filters (SPKF)

In order to improve the accuracy, consistency and efficiency[4] of Gaussian approximate inference algorithms applied to general nonlinear DSSMs, the two major shortcomings of the EKF need to be addressed. These are: 1) Disregard for the "probabilistic spread" of the underlying system state and noise RVs during the linearization of the system equations, and 2) Limited *first-order accuracy* of propagated means and covariances resulting from a first-order truncated Taylor-series linearization method. A number of related algorithms [16, 31, 14] have recently been proposed that address these issues by making use of novel

[2] The error between the true observation and the predicted observation, $\tilde{\mathbf{y}}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k^-$ is called the *innovation*.

[3] While "second-order" versions of the EKF exist, their increased implementation and computational complexity tend to prohibit their use.

[4] The *accuracy* of an estimator is an indication of the average magnitude (taken over the distribution of the data) of the estimation error. An estimator is *consistent* if $trace[\hat{\mathbf{P}}_{\mathbf{x}}] \geq trace[\mathbf{P}_{\mathbf{x}}]$ and it is *efficient* if that lower bound on the state error covariance is tight.

deterministic sampling approaches that circumvent the need to calculate analytical derivatives (such as the Jacobians of the system equations as needed by the EKF). It turns out that these algorithms, collectively called **Sigma-Point Kalman Filters** (SPKFs) are related through the implicit use of a technique called *weighted statistical linear regression*[5] [22] to calculate the optimal terms in the Kalman update rule (Equation 6).

3.1 Weighted statistical linear regression of a nonlinear function

Weighted statistical linear regression (WSLR) provides a solution to the problem of linearizing a nonlinear function of a random variable (RV), which takes into account the actual uncertainty (probabilistic spread) of that RV. In stead of linearizing the nonlinear function through a truncated Taylor-series expansion at a *single* point (usually taken to be the mean value of the RV), we rather linearize the function through a linear regression between r points drawn from the prior distribution of the RV, and the *true* nonlinear functional evaluations of those points. Since this statistical approximation technique takes into account the statistical properties of the prior RV the resulting *expected* linearization error tends to be smaller than that of a truncated Taylor-series linearization.

In other words, consider a nonlinear function $\mathbf{y} = \mathbf{g}(\mathbf{x})$ which is evaluated in r points $(\boldsymbol{\mathcal{X}}_i, \boldsymbol{\mathcal{Y}}_i)$ where $\boldsymbol{\mathcal{Y}}_i = \mathbf{g}(\boldsymbol{\mathcal{X}}_i)$. Define

$$\begin{aligned}
\bar{\mathbf{x}} &= \sum_{i=1}^{r} w_i \boldsymbol{\mathcal{X}}_i && (17)\\
\hat{\mathbf{P}}_{\mathbf{xx}} &= \sum_{i=1}^{r} w_i (\boldsymbol{\mathcal{X}}_i - \bar{\mathbf{x}})(\boldsymbol{\mathcal{X}}_i - \bar{\mathbf{x}})^T && (18)\\
\bar{\mathbf{y}} &= \sum_{i=1}^{r} w_i \boldsymbol{\mathcal{Y}}_i && (19)\\
\hat{\mathbf{P}}_{\mathbf{yy}} &= \sum_{i=1}^{r} w_i (\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}})(\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}})^T && (20)\\
\hat{\mathbf{P}}_{\mathbf{xy}} &= \sum_{i=1}^{r} w_i (\boldsymbol{\mathcal{X}}_i - \bar{\mathbf{x}})(\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}})^T && (21)
\end{aligned}$$

where w_i is a set of r scalar regression weights such that $\sum_{i=1}^{r} w_i = 1$. Now, the aim is to find the linear regression $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ that minimizes a statistical cost-function of the form $J = E\left[\phi(\mathbf{e}_i)\right]$, where the expectation is taken over i and hence the distribution of the $\mathbf{x}$. The point-wise linearization error is given by $\mathbf{e}_i = \boldsymbol{\mathcal{Y}}_i - (\mathbf{A}\boldsymbol{\mathcal{X}}_\mathbf{i} + \mathbf{b})$, with covariance $\mathbf{P}_{\mathbf{ee}} = \hat{\mathbf{P}}_{\mathbf{yy}} - \mathbf{A}\hat{\mathbf{P}}_{\mathbf{xx}}\mathbf{A}^T$, and the error functional ϕ is usually taken to be the vector dot product. This reduces the cost function to the well known sum-of-squared-errors and the resulting minimization to

$$\{\mathbf{A}, \mathbf{b}\} = \text{argmin} \sum_{i=1}^{r} (w_i \mathbf{e}_i \mathbf{e}_i^T) \qquad (22)$$

which has the following solution [10]

$$\mathbf{A} = \hat{\mathbf{P}}_{\mathbf{xy}}^T \hat{\mathbf{P}}_{\mathbf{xx}}^{-1} \qquad \mathbf{b} = \bar{\mathbf{y}} - \mathbf{A}\bar{\mathbf{x}}. \qquad (23)$$

Given this statistically better motivated "linearization" method for a nonlinear function of a RV, how do we actually choose the number of and the specific location of these regression

[5] This technique is also known as *statistical linearization* [10].

points $\boldsymbol{\mathcal{X}}_i$, called *sigma-points*, as well as their corresponding regression weights, w_i? It is the answer to this question that differentiate the different SPKF variants from each other. The next section gives a concise summary of these different approaches.

A simple example showing the superiority of the sigma-point approach is shown in Figure 2 for a 2-dimensional system: the left plot shows the true mean and covariance propagation using Monte-Carlo sampling; the center plots show the results using a linearization approach as would be done in the EKF; the right plots show the performance of the sigma-point approach (note only 5 sigma points are required). The superior performance of the sigma-point approach is clear. The sigma-point approach results in approximations that are accurate to the third order for Gaussian inputs for all nonlinearities and at least to the second order for non-Gaussian inputs. Another benefit of the sigma-point approach over the EKF is that no analytical derivative need to be calculated. Only point-wise evaluations of the system equations are needed which makes the SPKF ideally suited for estimation in "black box" systems.

3.2 SPKF Family : UKF, CDKF, SR-UKF & SR-CDKF

The Unscented Kalman Filter (UKF)

The UKF [16] is a SPKF that derives the location of the sigma-points as well as their corresponding weights according to the following rationale [17]: *The sigma-points should be chosen so that they capture the most important statistical properties of the prior random variable* $\mathbf{x}$. This is achieved by choosing the points according to a constraint equation of the form $\xi\left(\langle\boldsymbol{\mathcal{X}}, w\rangle, r, p(\mathbf{x})\right) = 0$, where $\langle\boldsymbol{\mathcal{X}}, w\rangle$ is the set of all sigma-points $\boldsymbol{\mathcal{X}}_i$ and weights w_i for $i = 1, \ldots r$. It is possible to satisfy such a constraint condition and still have some degree of freedom in the choice of the point locations, by minimizing a further penalty or cost-function of the form $c\left(\langle\boldsymbol{\mathcal{X}}, w\rangle, r, p(\mathbf{x})\right)$. The purpose of this cost-function is to incorporate statistical features of $\mathbf{x}$ which are desirable, but do not necessarily have to be met. In other words, the sigma-points are given by the solution to the following optimization problem

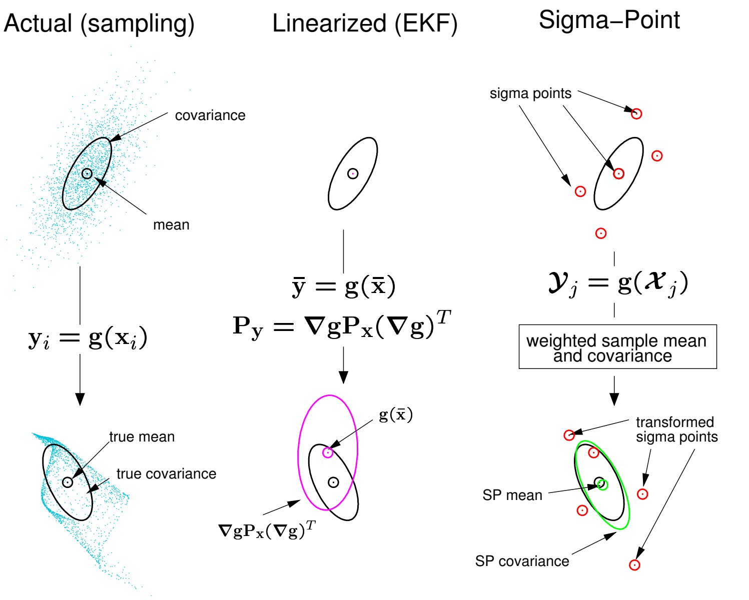
$$\min_{\langle\boldsymbol{\mathcal{X}}, w\rangle} c\left(\langle\boldsymbol{\mathcal{X}}, w\rangle, r, p(\mathbf{x})\right) \quad \text{subject to} \quad \xi\left(\langle\boldsymbol{\mathcal{X}}, \mathrm{w}\rangle, \mathrm{r}, \mathrm{p}(\mathbf{x})\right) = 0\,. \tag{24}$$

The *necessary* statistical information captured by the UKF is the first and second order moments of $p(\mathbf{x})$. The number[6] of sigma-points needed to do this $r = 2L + 1$ where L is the dimension of $\mathbf{x}$. It can be shown [18] that matching the moments of $\mathbf{x}$ accurately up to the nth order means that Equations 19 and 20 capture $\bar{\mathbf{y}}$ and $\hat{\mathbf{P}}_{\mathbf{yy}}$ accurately up the nth order as well. See [18, 44, 45] for more detail on how the sigma-points are calculated as a solution to Equation 24). The resulting set of sigma-points and weights utilized by the UKF is,

$$\begin{array}{llll}
\boldsymbol{\mathcal{X}}_0 = \bar{\mathbf{x}} & & w_0^{(m)} = \frac{\lambda}{L+\lambda} & \\
\boldsymbol{\mathcal{X}}_i = \bar{\mathbf{x}} + \left(\sqrt{(L+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_i & i=1,\ldots,L & w_0^{(c)} = \frac{\lambda}{L+\lambda} + (1-\alpha^2+\beta) & \\
\boldsymbol{\mathcal{X}}_i = \bar{\mathbf{x}} - \left(\sqrt{(L+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_i & i=L+1,\ldots,2L & w_i^{(m)} = w_i^{(c)} = \frac{1}{2(L+\lambda)} & i=1,\ldots,2L
\end{array} \tag{25}$$

where $\lambda = \alpha^2(L+\kappa) - L$ is a scaling parameter. α determines the spread of the sigma points around $\bar{\mathbf{x}}$ and is usually set to a small positive value (e.g. $1e - 2 \leq \alpha \leq 1$). κ is a secondary scaling parameter which is usually set to either 0 or $3 - L$ (see [45] for details), and β is an extra degree of freedom scalar parameter used to incorporate any extra prior knowledge of the distribution of $\mathbf{x}$ (for Gaussian distributions, $\beta = 2$ is optimal).

[6] By using a larger number of sigma-points higher order moments such as the *skew* or *kurtosis* can also be captured. See [17] for detail.



Actual (sampling)
Linearized (EKF)
Sigma−Point
covariance
mean
sigma points
$\mathbf{y}_i = \mathbf{g}(\mathbf{x}_i)$
$\bar{\mathbf{y}} = \mathbf{g}(\bar{\mathbf{x}})$
$\mathbf{P}_\mathbf{y} = \boldsymbol{\nabla}\mathbf{g}\mathbf{P}_\mathbf{x}(\boldsymbol{\nabla}\mathbf{g})^T$
$\boldsymbol{\mathcal{Y}}_j = \mathbf{g}(\boldsymbol{\mathcal{X}}_j)$
weighted sample mean and covariance
true mean
true covariance
$\mathbf{g}(\bar{\mathbf{x}})$
$\boldsymbol{\nabla}\mathbf{g}\mathbf{P}_\mathbf{x}(\boldsymbol{\nabla}\mathbf{g})^T$
transformed sigma points
SP mean
SP covariance

Figure 2: Example of the sigma-point approach for mean and covariance propagation. a) actual, b) first-order linearization (EKF), c) sigma-point approach.

$\left(\sqrt{(L+\lambda)\mathbf{P}_\mathbf{x}}\right)_i$ is the *i*th row (or column) of the weighted matrix square root[7] of the covariance $\mathbf{P}_\mathbf{x}$. See [44, 45] for precise implementational detail for the UKF.

The Central Difference Kalman Filter (CDKF)

Separately from the development of the UKF, two different groups [34, 14] proposed another SPKF implementation based on *Sterling's interpolation formula.* This formulation was derived by replacing the analytically derived 1st and 2nd order derivatives in the Taylor series expansion (Equation 13) by numerically evaluated *central divided differences*, i.e.

$$\boldsymbol{\nabla}\mathbf{g} \approx \frac{\mathbf{g}(\mathbf{x}+h\boldsymbol{\delta}_\mathbf{x}) - \mathbf{g}(\mathbf{x}-h\boldsymbol{\delta}_\mathbf{x})}{2h} \tag{26}$$

$$\boldsymbol{\nabla}^2\mathbf{g} \approx \frac{\mathbf{g}(\mathbf{x}+h\boldsymbol{\delta}_\mathbf{x}) + \mathbf{g}(\mathbf{x}-h\boldsymbol{\delta}_\mathbf{x}) - 2\mathbf{g}(\mathbf{x})}{h^2} \ . \tag{27}$$

Even though this approach is not explicitly derived starting from the statistical linear regression rationale, it can be shown [22] that the resulting Kalman filter again implicitly employs WSLR based linearization. The resulting set of sigma-points for the CDKF is once again $2L+1$points deterministically drawn from the prior statistics of $\mathbf{x}$, i.e.

$$\begin{array}{llll}
\boldsymbol{\mathcal{X}}_0 = \bar{\mathbf{x}} & & w_0 = \frac{h^2-L}{h^2} & \\
\boldsymbol{\mathcal{X}}_i = \bar{\mathbf{x}} + \left(\sqrt{h^2\mathbf{P}_\mathbf{x}}\right)_i & {\scriptstyle i=1,\dots,L} & w_i = \frac{1}{2h^2} & {\scriptstyle i=1,\dots,2L} \\
\boldsymbol{\mathcal{X}}_i = \bar{\mathbf{x}} - \left(\sqrt{h^2\mathbf{P}_\mathbf{x}}\right)_i & {\scriptstyle i=L+1,\dots,2L} & &
\end{array}$$

For exact implementation details of the CDKF algorithm, see [30, 39, 45]. It is shown in [31] that the CDKF has marginally higher theoretical accuracy than the UKF in the

[7] We typically use the numerically efficient *Cholesky factorization* method to calculate the matrix square root.

higher order terms of the Taylor series expansion, but we've found in practice that all SPKFs (CDKF and UKF) perform equally well with negligible difference in estimation accuracy. Both generate estimates however that are clearly superior to those calculated by an EKF. One advantage of the CDKF over the UKF is that it only uses a single scalar scaling parameter, the central difference half-step size h, as opposed to the three (α, κ, β)that the UKF uses. Once again this parameter determines the *spread* of the sigma-points around the prior mean. For Gaussian priors, its optimal value is $\sqrt{3}$ [31].

Square-root Forms (SR-UKF, SR-CDKF)

One of the most costly operations in the SPKF is the calculation of the matrix square-root of the state covariance at each time step in order to form the sigma-point set. Due to this and the need for more numerical stability (especially during the state covariance update), we derived numerically efficient *square-root* forms of both the UKF and the CDKF [39, 40]. These forms propagate and update the square-root of the state covariance directly in Cholesky factored form, using the sigma-point approach the following three powerful linear algebra techniques: *QR decomposition*, *Cholesky factor updating* and *efficient pivot-based least squares*. For implementation details, see [45, 39, 40]. The square-root SPKFs (SR-UKF and SR-CDKF) has equal (or marginally better) estimation accuracy when compared to the standard SPKF, but at the added benefit of reduced computational cost for certain DSSMs and a consistently increased numerical stability. For this reason, they are our preferred form for SPKF use in stand-alone estimators as well as in SMC/SPKF hybrids (see Section 5).

4 Experimental Results : SPKF applied to state, parameter and dual estimation

4.1 State Estimation

The UKF was originally designed for state estimation applied to nonlinear control applications requiring full-state feedback [16, 18, 19]. We provide a new application example corresponding to noisy time-series estimation with neural networks. For further examples, see [45].

Noisy time-series estimation : In this example, a SPKF (UKF) is used to estimate an underlying clean time-series corrupted by additive Gaussian white noise. The time-series used is the Mackey-Glass-30 chaotic series [25, 21]. The clean times-series is first modeled as a nonlinear autoregression

$$x_k = f(x_{k-1}, x_{k-2}, \ldots, x_{k-M}; \mathbf{w}) + v_k \tag{28}$$

where the model f (parameterized by $\mathbf{w}$) was approximated by training a feedforward neural network on the clean sequence. The residual error after convergence was taken to be the process noise variance. Next, white Gaussian noise was added to the clean Mackey-Glass series to generate a noisy time-series $y_k = x_k + n_k$. In the estimation problem, the noisy-time series y_k is the only observed input to either the EKF or SPKF algorithms (both utilize the known neural network model). Note that for time-series estimation, both the EKF and the SPKF are$\mathcal{O}(L^2)$ complexity. Figure 3 shows a sub-segment of the estimates generated by both the EKF and the SPKF (the original noisy time-series has a 3dB SNR). The superior performance of the SPKF is clearly visible.

4.2 Parameter Estimation

The classic machine learning problem involves determining a nonlinear mapping

$$\mathbf{y}_k = \mathbf{g}(\mathbf{x}_k, \mathbf{w}) \tag{29}$$

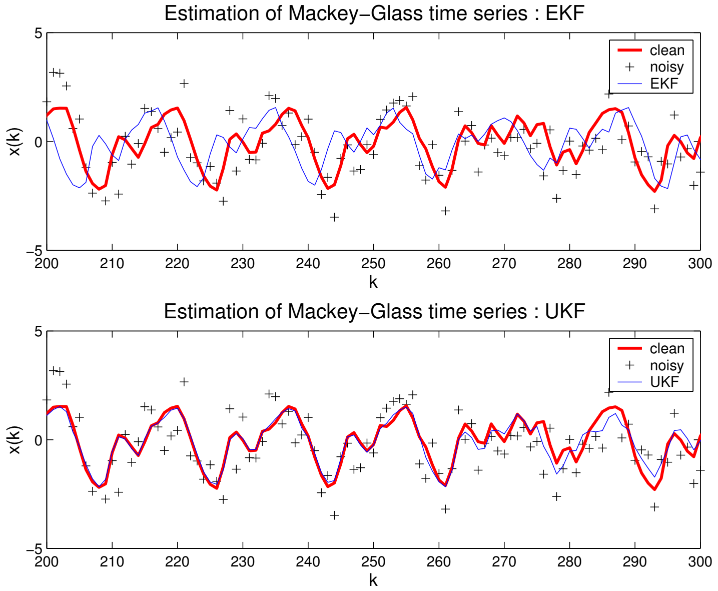


Figure 3: Estimation of Mackey-Glass time-series with the EKF and SPKF using a known model. Bottom graph shows comparison of estimation errors for complete sequence.

where $\mathbf{x}_k$ is the input, $\mathbf{y}_k$is the output, and the nonlinear map $\mathbf{g}$ is parameterized by the vector $\mathbf{w}$. The nonlinear map, for example, may be a feed-forward or recurrent neural network ($\mathbf{w}$ are the weights), with numerous applications in regression, classification, and dynamic modeling. It can also be a general nonlinear parametric model, such as the dynamic/kinematic vehicle model, parameterized by a finite set of coefficients . Learning corresponds to estimating the parameters $\mathbf{w}$. Typically, a training set is provided with sample pairs consisting of known input and desired outputs, $\{\mathbf{x}_k, \mathbf{d}_k\}$. The error of the machine is defined as $\mathbf{e}_k = \mathbf{d}_k - \mathbf{g}(\mathbf{x}_k, \mathbf{w})$, and the goal of learning involves solving for the parameters $\mathbf{w}$ in order to minimize the expectation of some function of this error (typically the expected square error). Note that in contrast to state (and dual) estimation, the input is usually observed noise-free and does not require estimation. While a number of optimization approaches exist (e.g., gradient descent), the SPKF may be used to estimate the parameters by writing a new state-space representation

$$\begin{aligned} \mathbf{w}_k &= \mathbf{w}_{k-1} + \mathbf{v}_k \\ \mathbf{y}_k &= \mathbf{g}(\mathbf{x}_k, \mathbf{w}_k) + \mathbf{e}_k \end{aligned} \tag{30}$$

where the parameters $\mathbf{w}_k$ correspond to a stationary process with identity state transition matrix, driven by process noise $\mathbf{v}_k$ (the choice of variance determines tracking performance). The output $\mathbf{y}_k$ corresponds to a nonlinear observation on $\mathbf{w}_k$. The SPKF can then be applied directly as an efficient “second-order” technique for learning the parameters.

Four Regions Classification : In the parameter estimation example, we consider a benchmark pattern classification problem having four interlocking regions [37]. A three-layer feedforward network (MLP) with 2-10-10-4 nodes is trained using inputs randomly drawn within the pattern space, $S = [-1, 1] \times [1, 1]$, with the desired output value of $+0.8$ if the pattern fell within the assigned region and -0.8 otherwise. Figure 4 illustrates the classification task, learning curves for the SPKF (UKF) and EKF, and the final classification regions. For the learning curve, each epoch represents 100 randomly drawn input samples. The test set evaluated on each epoch corresponds to a uniform grid of 10,000 points. Again,

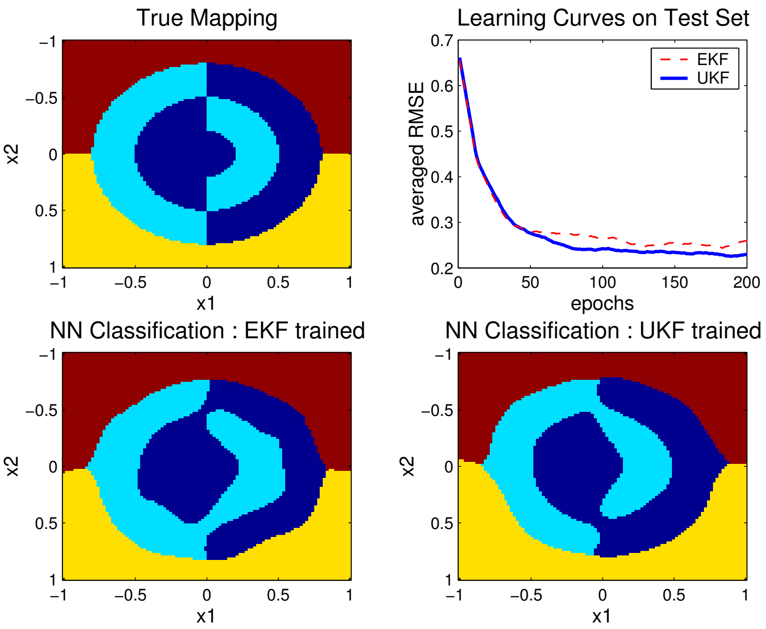


Figure 4: Singhal and Wu's Four Region classification problem.

we see the superior performance for the SPKF.

4.3 Dual Estimation

This example brings together both state estimation and parameter estimation in an application of the SPKF to dual estimation for a *inverted double pendulum* control system. An inverted double pendulum (See Figure 5) has states corresponding to cart position and velocity, and top and bottom pendulum angle and angular velocity, $\mathbf{x} = [x, \dot{x}, \theta_1, \dot{\theta_1}, \theta_2, \dot{\theta_2}]$. The system parameters correspond the length and mass of each pendulum, and the cart mass, $\mathbf{w} = [l_1, l_2, m_1, m_2, M]$. The system dynamic equations are also given in (see Figure 5 for equations). These continuous-time dynamics are discretized with a sampling period of 0.02 seconds. The double pendulum is stabilized by applying a horizontal control force u to the cart. In this case we use a state dependent Ricatti Equation (SDRE) [3] controller to stabilize the system. The SDRE controller needs accurate estimates of the system state as well as system parameters in order to accurately and robustly balance the pendulum. In our experiment, only partial and noisy observations of the system states were available and the system parameters were also initialized to incorrect values. A SPKF (a joint-UKF in this case) is used to estimate both the underlying system states and the true system parameters using only the noisy observations at each time step, which is then fed back to the SDRE controller for closed-loop control. Figure **??** illustrates the performance of this adaptive control system by showing the evolution of the estimated and actual states (top) as well as the system parameter estimates (bottom). At the start of the simulation both the states and parameters are unknown (the control system is unstable at this point). However, within one trial, the SPKF enables convergence and stabilization of the pendulum without a single crash! This is in stark contrast to standard system identification and adaptive control approaches, where numerous off-line trials, large amounts of training data and painstaking "hand-tuning" are often needed.

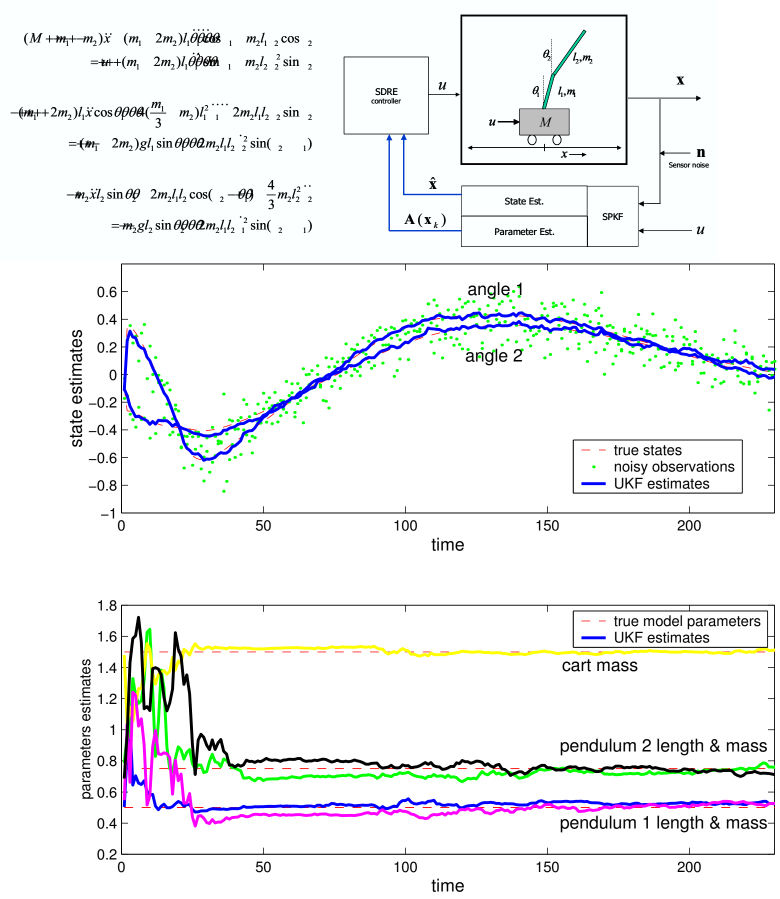


Figure 5: Inverted double pendulum dual estimation and control experiment. Dynamic model and schematic of control system (top plot). Estimation results are shown in the bottom plot: state estimates (top) and parameter estimates(bottom).

5 Sequential Monte Carlo / SPKF Hybrids

The SPKF, like the EKF, still assumes a Gaussian posterior which can fail in certain nonlinear non-Gaussian problems with multi-modal and/or heavy tailed posterior distributions. The Gaussian sum filter (GSF) addresses this issue by approximating the posterior density with a finite Gaussian mixture and can be interpreted as a parallel bank of EKFs. Unfortunately, due to the use of the EKF as a subcomponent, it also suffers from similar shortcomings as the EKF. Recently, particle based sampling filters have been proposed and used successfully to recursively update the posterior distribution using *sequential importance sampling and resampling* [7]. These methods (collectively called *particle filters*) approximate the posterior by a set of weighted samples without making any explicit assumption about its form and can thus be used in general nonlinear, non-Gaussian systems. In this section, we present hybrid methods that utilizes the SPKF to augment and improve the standard particle filter, specifically through generation of the *importance proposal distributions*. We will briefly review only the background fundamentals necessary to introduce particle filtering and then discuss the two extensions based on the SPKF, the *Sigma-Point Particle Filter* (SPPF) and the *Gaussian-Mixture Sigma-Point Particle Filter* (GMSPPF).

5.1 Particle Filters

The *particle filter* is a sequential Monte Carlo (SMC) based method that allows for a complete representation of the state distribution using sequential importance sampling and resampling [6, 24]. Whereas the standard EKF and SPKF make a Gaussian assumption to simplify the optimal recursive Bayesian estimation, particle filters make no assumptions on the form of the probability densities in question, i.e, full nonlinear, non-Gaussian estimation.

Monte Carlo simulation and sequential importance sampling

Particle filtering is based on Monte Carlo simulation with sequential importance sampling (SIS). The overall goal is to directly implement optimal Bayesian estimation by recursively approximating the complete posterior state density. Importance sampling is a Monte-Carlo method that represents a distribution $p(\mathbf{x})$ by an empirical approximation based on a set of weighted samples (particles), i.e. $p(\mathbf{x}) \approx \widehat{p}(\mathbf{x}) = \sum_{i=1}^{N} w^{(i)}\delta(\mathbf{x} - \boldsymbol{x}^{(i)})$, where $\delta(.)$ is the Dirac delta function, and the weighted sample set, $\{w^{(l)}, \boldsymbol{x}^{(i)}; l = 1 \ldots N\}$ are drawn from some related, easy-to-sample-from *proposal* distribution $\pi(\mathbf{x})$. The weights are given by $w^{(i)} = \frac{p(\boldsymbol{x}^{(i)})/\pi(\boldsymbol{x}^{(i)})}{\sum_{i=1}^{N} p(\boldsymbol{x}^{(i)})/\pi(\boldsymbol{x}^{(i)})}$. Given this, any estimate of the system such as $E_p[g(\mathbf{x})] = \int g(\mathbf{x})p(\mathbf{x})d\mathbf{x}$ can be approximated by $\hat{E}[g(\mathbf{x})] = \sum_{i=1}^{N} w^{(i)} g(\mathcal{X}^{(i)})$ [7]. Using the first order Markov nature of our DSSM and the conditional independence of the observations given the state, a recursive update formula (implicitly a nonlinear measurement update) for the importance weights can be derived [7]. This is given by

$$w_k^{(i)} = w_{k-1}^{(i)} p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})/\pi(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k) \text{ for } \mathbf{x}_k = \boldsymbol{x}^{(i)}. \tag{31}$$

Equation (31) provides a mechanism to sequentially update the importance weights given an appropriate choice of proposal distribution, $\pi(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k)$. Since we can sample from the proposal distribution and evaluate the likelihood $p(\mathbf{y}_k|\mathbf{x}_k)$ and transition probabilities $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, all we need to do is generate a prior set of samples and iteratively compute the importance weights. This procedure then allows us to evaluate the expectations of interest by the following estimate

$$E\left[g(\mathbf{x}_k)\right] \approx \frac{1/N \sum_{i=1}^{N} w_k^{(i)} \mathbf{g}(\boldsymbol{x}_k^{(i)})}{1/N \sum_{i=1}^{N} w_k^{(i)}} = \sum_{i=1}^{N} \tilde{w}_k^{(i)} \mathbf{g}(\boldsymbol{x}_k^{(i)}) \tag{32}$$

where the normalized importance weights $\tilde{w}_k^{(i)} = w_k^{(i)} / \sum_{j=1}^{N} w_k^{(j)}$. This estimate asymptotically converges if the expectation and variance of $\mathbf{g}(\mathbf{x}_k)$ and w_k exist and are bounded, and if the support of the proposal distribution includes the support of the posterior distribution. Thus, as N tends to infinity, the posterior filtering density function can be approximated arbitrarily well by the point-mass estimate

$$\hat{p}(\mathbf{x}_k|\mathbf{Y}_k) = \sum_{i=1}^{N} \tilde{w}_k^{(i)} \delta(\mathbf{x}_k - \boldsymbol{x}_k^{(i)})$$

These point-mass estimates can approximate any general distribution arbitrarily well, limited only by the number of particles used and how well the above mentioned importance sampling conditions are met.

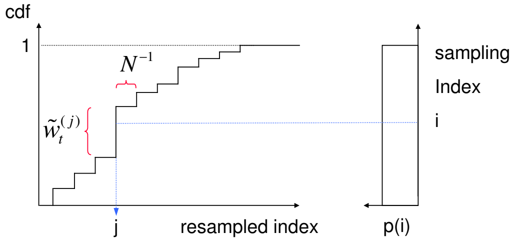


Figure 6: Resampling process, whereby a random measure {$\boldsymbol{x}_k^{(i)}, w_k^{(i)}$} is mapped into an equally weighted random measure {$\boldsymbol{x}_k^{(j)}, N^{-1}$}. The index i is drawn from a uniform distribution.

Resampling and Sample Depletion

The sequential importance sampling (SIS) algorithm discussed so far has a serious limitation: the variance of the importance weights increases stochastically over time. Typically, after a few iterations, one of the normalized importance weights tends to 1, while the remaining weights tend to zero. A large number of samples are thus effectively removed from the sample set because their importance weights become numerically insignificant. To avoid this degeneracy, a resampling or selection stage may be used to eliminate samples with low importance weights and multiply samples with high importance weights. We typically make use of either *sampling-importance resampling* (SIR) or *residual resampling*. See [7] for more theoretical and implementational detail on resampling.

After the selection/resampling step at time k, we obtain N particles distributed marginally approximately according to the posterior distribution. Since the selection step favors the creation of multiple copies of the "fittest" particles, many particles may end up having no children ($N_i = 0$), whereas others might end up having a large number of children, the extreme case being $N_i = N$ for a particular value i. In this case, there is a severe depletion of samples. Therefore, and additional procedure, such as a single MCMC step, is often required to introduce sample variety after the selection step without affecting the validity of the approximation they infer. See [42] for more detail.

The Particle Filter Algorithm

The pseudo-code of a generic particle filter is presented in Algorithm 1. In implementing this algorithm, the choice of the proposal distribution $\pi(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k)$ is the most critical design issue. The optimal proposal distribution (which minimizes the variance on the importance weights) is given by [6, 24],

$$\pi(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k) = p(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k)$$

i.e., the true conditional state density given the previous state history and all observations. Sampling from this is, of course, impractical for arbitrary densities (recall the motivation for using importance sampling in the first place). Consequently the transition prior is the most popular choice of proposal distribution [7][8],

$$\pi(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k) \overset{\circ}{=} p(\mathbf{x}_k|\mathbf{x}_{k-1}).$$

For example, if an additive Gaussian process noise model is used, the transition prior is simply,

$$p(\mathbf{x}_k|\mathbf{x}_{k-1}) = \mathcal{N}\left(\mathbf{x}_k; \mathbf{f}(\bar{\mathbf{x}}_{k-1}), \mathbf{R}^{\mathbf{v}}_{k-1}\right) \quad . \tag{33}$$

[8] The notation $\overset{\circ}{=}$ denotes *"chosen as"*, to indicate a subtle difference versus *"approximation"*.

Algorithm 1 Algorithm for the generic particle filter

```
1. Initialization: k=0
    • For $i = 1, \ldots, N$, draw (sample) particle $\boldsymbol{x}_0^{(i)}$ from the prior $p(\mathbf{x}_0)$.
2. For $k = 1, 2, \ldots$
    (a) Importance sampling step
        • For $i = 1, \ldots, N$, sample $\boldsymbol{x}_k^{(i)} \sim \pi(\mathbf{x}_k | \boldsymbol{x}_{k-1}^{(i)}, \mathbf{Y}_k)$.
        • For $i = 1, \ldots, N$, evaluate the importance weights up to a normalizing constant:
              $w_k^{(i)} = w_{k-1}^{(i)} \frac{p(\mathbf{y}_k | \boldsymbol{x}_k^{(i)}) p(\boldsymbol{x}_k^{(i)} |}{\pi(\boldsymbol{x}_k^{(i)} | \boldsymbol{x}_{k-1}^{(i)}, \mathbf{Y}_k)}$    (34)
        • For $i = 1, \ldots, N$, normalize the importance weights: $\tilde{w}_k^{(i)} = w_k^{(i)} / \sum_{j=1}^{N} w_k^{(j)}$.
    (b) Selection step (resampling)
        • Multiply/suppress samples $\boldsymbol{x}_k^{(i)}$ with high/low importance weights $\tilde{w}_k^{(i)}$,
          respectively, to obtain $N$ random samples approximately distributed according to $p(\mathbf{x}_k | \mathbf{Y}_k)$.
        • For $i = 1, \ldots, N$, set $w_k^{(i)} = \tilde{w}_k^{(i)} = N^{-1}$.
        • (optional) Do a single MCMC (Markov chain Monte Carlo) move step to
          add further 'variety' to the particle set without changing their distribution.
    (c) Output: The output of the algorithm is a set of samples that can be used to approximate
        the posterior distribution as follows: $\hat{p}(\mathbf{x}_k | \mathbf{Y}_k) = \frac{1}{N} \sum_{i=1}^{N} \delta(\mathbf{x}_k - \boldsymbol{x}_k^{(i)})$.
        From these samples, any estimate of the system state can be calculated, such as the MMSE estimate,
              $\hat{\mathbf{x}}_k = E\left[\mathbf{x}_k | \mathbf{Y}_k\right] \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_k^{(i)}.$
        Similar expectations of the function $\mathbf{g}(\mathbf{x}_k)$ (such as MAP estimate, covariance,
        skewness, etc.) can be calculated as a sample average.
```

The effectiveness of this approximation depends on how close the proposal distribution is to the true posterior distribution. If there is not sufficient overlap, only a few particles will have significant importance weights when their likelihood are evaluated.

5.2 The Sigma-Point Particle Filter

An improvement in the choice of proposal distribution over the simple transition prior, which also address the problem of sample depletion, can be accomplished by moving the particles towards the regions of high likelihood, based on the most recent observation $\mathbf{y}_k$ (See Figure 7).

An effective approach to accomplish this, is to use an EKF generated Gaussian approximation to the optimal proposal, i.e,

$$\pi(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{Y}_k) \overset{\circ}{=} q_{\mathcal{N}}(\mathbf{x}_k | \mathbf{Y}_k),$$

which is accomplished by using a separate EKF to generate and propagate a Gaussian

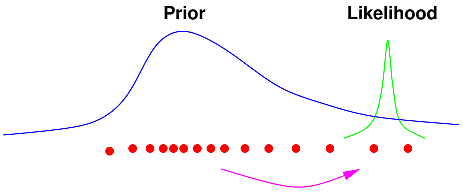


Figure 7: Including the most current observation into the proposal distribution, allows us to move the samples in the prior to regions of high likelihood. This is of paramount importance if the likelihood happens to lie in one of the tails of the prior distribution, or if it is too narrow (low measurement error).

proposal distribution for each particle, i.e.,

$$q_{\mathcal{N}}(\mathbf{x}_k|\mathbf{Y}_k) = \mathcal{N}\left(\mathbf{x}_k; \boldsymbol{x}_k^{(i)}, \mathbf{P}_k^{(i)}\right) \qquad i = 1, 2, \ldots N \tag{35}$$

That is, at time k one uses the EKF equations, with the new data, to compute the mean and covariance of the importance distribution for each particle from the previous time step $k-1$. Next, we redraw the i-th particle (at time k) from this new updated distribution. While still making a Gaussian assumption, the approach provides a better approximation to the optimal conditional proposal distribution and has been shown to improve performance on a number of applications [5].

By replacing the EKF with a SPKF[9], we can more accurately propagate the mean and covariance of the Gaussian approximation to the state distribution. Distributions generated by the SPKF will have a greater support overlap with the true posterior distribution than the overlap achieved by the EKF estimates. In addition, scaling parameters used for sigma point selection can be optimized to capture certain characteristic of the prior distribution if known, i.e., the algorithm can be modified to work with distributions that have heavier tails than Gaussian distributions such as Cauchy or Student-t distributions. The new filter that results from using a SPKF for proposal distribution generation within a particle filter framework is called the *Sigma-Point Particle Filter* (SPPF). Referring to Algorithm 1 for the generic particle filter, the first item in the importance sampling step:

- For $i = 1, \ldots, N$, sample $\boldsymbol{x}_k^{(i)} \sim \pi(\mathbf{x}_k|\boldsymbol{x}_{k-1}^{(i)}, \mathbf{Y}_k)$.

is replaced with the following SPKF update:

- For $i = 1, \ldots, N$:
  - Update the Gaussian prior distribution for each particle with the SPKF :
    - $*$ Calculate sigma-points for particle, $\boldsymbol{x}_{k-1}^{a,(i)} = [\boldsymbol{x}_{k-1}^{(i)} \;\; \bar{\mathbf{v}}_{k-1} \;\; \bar{\mathbf{n}}_{k-1}]^T$:

$$\boldsymbol{\mathcal{X}}_{k-1,(0\ldots 2L)}^{a,(i)} = \left[\boldsymbol{x}_{k-1}^{a,(i)} \qquad \boldsymbol{x}_{k-1}^{a,(i)} + \gamma\sqrt{\mathbf{P}_{k-1}^{a,(i)}} \qquad \boldsymbol{x}_{k-1}^{a,(i)} - \gamma\sqrt{\mathbf{P}_{k-1}^{a,(i)}}\right]$$

[9] Specifically we make use of either a *Square-Root Unscented Kalman filter* (SR-UKF) or a *Square-Root Central-Difference Kalman* Filter (SR-CDKF).

∗ Propagate sigma-points into future (time update):

$$\begin{aligned}
\boldsymbol{\mathcal{X}}^{x,(i)}_{k|k-1,(0\dots 2L)} &= \mathbf{f}\left(\boldsymbol{\mathcal{X}}^{x,(i)}_{k-1,(0\dots 2L)}, \boldsymbol{\mathcal{X}}^{v,(i)}_{k-1,(0\dots 2L)}, \mathbf{u}_k\right) \\
\bar{\boldsymbol{x}}^{(i)}_{k|k-1} &= \sum_{j=0}^{2L} w_j^{(m)} \boldsymbol{\mathcal{X}}^{x,(i)}_{k|k-1,j} \\
\mathbf{P}^{(i)}_{k|k-1} &= \sum_{j=0}^{2L} w_j^{(c)} (\boldsymbol{\mathcal{X}}^{x,(i)}_{k|k-1,j} - \bar{\boldsymbol{x}}^{(i)}_{k|k-1})(\boldsymbol{\mathcal{X}}^{x,(i)}_{k|k-1,j} - \bar{\boldsymbol{x}}^{(i)}_{k|k-1})^T \\
\boldsymbol{\mathcal{Y}}^{(i)}_{k|k-1,(0\dots 2L)} &= \mathbf{h}\left(\boldsymbol{\mathcal{X}}^{x,(i)}_{k|k-1,(0\dots 2L)}, \boldsymbol{\mathcal{X}}^{n,(i)}_{k-1,(0\dots 2L)}\right) \\
\bar{\mathbf{y}}^{(i)}_{k|k-1} &= \sum_{j=0}^{2L} W_j^{(m)} \mathcal{Y}^{(i)}_{j,k|k-1}
\end{aligned}$$

∗ Incorporate new observation (measurement update):

$$\begin{aligned}
\mathbf{P}_{\mathbf{y}_k\mathbf{y}_k} &= \sum_{j=0}^{2L} w_j^{(c)} [\boldsymbol{\mathcal{Y}}^{(i)}_{k|k-1,j} - \bar{\mathbf{y}}^{(i)}_{k|k-1}][\boldsymbol{\mathcal{Y}}^{(i)}_{k|k-1,j} - \bar{\mathbf{y}}^{(i)}_{k|k-1}]^T \\
\mathbf{P}_{\mathbf{x}_k\mathbf{y}_k} &= \sum_{j=0}^{2L} w_j^{(c)} [\mathcal{X}^{(i)}_{k|k-1,j} - \bar{\boldsymbol{x}}^{(i)}_{k|k-1}][\boldsymbol{\mathcal{Y}}^{(i)}_{k|k-1,j} - \bar{\mathbf{y}}^{(i)}_{k|k-1}]^T \\
\mathbf{K}_k &= \mathbf{P}_{\mathbf{x}_k\mathbf{y}_k}\mathbf{P}^{-1}_{\mathbf{x}_k\mathbf{y}_k} \qquad \bar{\boldsymbol{x}}^{(i)}_k = \bar{\boldsymbol{x}}^{(i)}_{k|k-1} + \mathbf{K}_k(\mathbf{y}_k - \bar{\mathbf{y}}^{(i)}_{k|k-1}) \\
\mathbf{P}^{(i)}_k &= \mathbf{P}^{(i)}_{k|k-1} - \mathbf{K}_k\mathbf{P}_{\mathbf{y}_k\mathbf{y}_k}\mathbf{K}_k^T
\end{aligned}$$

– Sample $\boldsymbol{x}^{(i)}_k \sim q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{Y}_k) \approx \boldsymbol{\mathcal{N}}\left(\mathbf{x}_k; \bar{\boldsymbol{x}}^{(i)}_k, \mathbf{P}^{(i)}_k\right)$

All other steps in the particle filter formulation remain unchanged. The SPPF presented above makes use of a UKF for proposal generation. Our preferred form however, is a SPPF based around the square-root CDKF (SR-CDKF) which has the numerical efficiency and stability benefits laid out in Section 3.2. The UKF was used in above in order to simplify the presentation of the algorithm. For experimental verification of the superior estimation performance of the SPPF over a standard particle filter, see Section 6. For more detail on the SPPF, see [42, 38][10] .

5.3 Gaussian Mixture Sigma-Point Particle Filters

Particle filters need to use a large number of particles for accurate and robust operation, which often make their use computationally expensive. Furthermore, as pointed out earlier, they suffer from an ailment called "sample depletion" that can cause the sample based posterior approximation to collapse over time to a few samples. In the SPPF section we showed how this problem can be addressed by moving particles to areas of high likelihood through the use of a SPKF generated proposal distribution. Although the SPPF has large estimation performance benefits over the standard PF, it still incurs a heavy computational burden since it has to run an $\mathcal{O}(m_y m_x^2)$ SPKF for each particle in the posterior state distribution.

In this section we present a further refinement of the SPPF called the ***Gaussian Mixture Sigma-Point Particle Filter*** (GMSPPF) [41]. This filter has equal or better estimation performance when compared to standard particle filters and the SPPF, at a largely reduced

[10] The original name of the SPPF algorithm was the *unscented particle filter* (UPF).

computational cost. The GMSPPF combines an *importance sampling* (IS) based measurement update step with a *SPKF based Gaussian sum filter* for the time-update and proposal density generation. The GMSPPF uses a finite Gaussian mixture model (GMM) representation of the posterior filtering density, which is recovered from the weighted posterior particle set of the IS based measurement update stage, by means of a *Expectation-Maximization (EM)* step. The EM step either follows directly after the resampling stage of the particle filter, or it can completely replace that stage if a *weighted EM* algorithm is used. The EM or WEM recovered GMM posterior further mitigates the "sample depletion" problem through its inherent "kernel smoothing" nature. The three main algorithmic components used in the GMSPPF are briefly discussed below to provide some background on their use. Then we show how these three components are combined to form the GMSPPF algorithm.

- SPKF based Gaussian mixture approximation

It can be shown [2] than any probability density $p(\mathbf{x})$ can be approximated as closely as desired by a Gaussian mixture model (GMM) of the following form, $p(\mathbf{x}) \approx p_{\mathcal{G}}(\mathbf{x}) = \sum_{g=1}^{G} \alpha^{(g)} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}^{(g)}, \mathbf{P}^{(g)})$, where G is the number of mixing components, $\alpha^{(g)}$ are the mixing weights and $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{P})$ is a normal distribution with mean vector $\boldsymbol{\mu}$ and positive definite covariance matrix $\mathbf{P}$. Given Equations 1 and 2, and assuming that the prior density $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$ and system noise densities $p(\mathbf{v}_{k-1})$ and $p(\mathbf{n}_k)$ are expressed by GMMs, the following densities can also be approximated by GMMs: 1) the *predictive prior density*, $p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \approx p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \sum_{g'=1}^{G'} \alpha^{(g')} \mathcal{N}(\mathbf{x}; \widetilde{\boldsymbol{\mu}}_k^{(g')}, \widetilde{\mathbf{P}}_k^{(g')})$, and 2) the *updated posterior density*, $p(\mathbf{x}_k|\mathbf{Y}_k) \approx p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_k) = \sum_{g''=1}^{G''} \alpha^{(g'')} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k^{(g'')}, \mathbf{P}_k^{(g'')})$, where $G' = GI$ and $G'' = G'J = GIJ$ (G, I and J are the number of components in the state, process noise, and observation noise GMMs respectively). The predicted and updated Gaussian component means and covariances of $p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_{k-1})$ and $p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_k)$ are calculated using the Kalman filter Equations 6 and 8 (In the GMSPPF we use a bank of SPKFs). The mixing weight update procedure are shown below in the algorithm pseudo-code. It is clear that the number of mixing components in the GMM representation grows from G to G' in the predictive (time update) step and from G' to G'' in the subsequent measurement update step. Over time, this will lead to an exponential increase in the total number of mixing components and must be addressed by a mixing-component reduction scheme. See below for how this is achieved.

- Importance sampling (IS) based measurement update

In the GMSPPF we use the GMM approximate $p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_k)$ from the bank of SPKFs (see above) as the proposal distribution $\pi(\mathbf{x}_k)$. In Section 5.2 and [38] we showed that sampling from such a proposal (which incorporates the latest observation), moves particles to areas of high likelihood which in turn reduces the "sample depletion" problem. Furthermore we use the predictive prior distribution $p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_{k-1})$ (see Sec.5.3) as a *smoothed* (over prior distribution of $\mathbf{x}_{k-1}$) evaluation of the $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ term in the weight update equation. This is needed since the GMSPPF represents the posterior (which becomes the prior at the next time step) by a GMM, which effectively smoothes the posterior particle set by a set of Gaussian kernels.

- EM/WEM for GMM recovery

The output of the IS-based measurement update stage is a set of N weighted particles, which in the standard particle filter is *resampled* in order to discard particles with insignificant weights and multiply particles with large weights. The GMSPPF represents the posterior by a GMM which is recovered from the resampled, equally weighted particle set using a standard *Expectation-Maximization* (EM) algorithm, or directly from the *weighted* particles using a *weighted-EM* (WEM) [27] step. Whether a *resample-then-EM* or a *direct-WEM* GMM recovery step is used depends on the particular nature of the inference problem at hand. It is shown in [7] that resampling is needed to keep the variance of the particle set from growing too rapidly . Unfortunately, resampling can also contribute to the “particle depletion” problem in cases where the measurement likelihood is very peaked, causing the particle set to collapse to multiple copies of the same particle. In such a case, the *direct-WEM* approach might be preferred. On the other hand, we have found that for certain problems where the disparity (as measured by the KL-divergence) between the true posterior and the GMM-proposal is large, the *resample-then-EM* approach performs better. This issue is still being investigated.

This GMM recovery step implicitly smoothes over the posterior set of samples, avoiding the “particle depletion” problem, and at the same time the number of mixing components in the posterior is reduced to G, avoiding the exponential growth problem alluded to above. Alternatively, one can use a more powerful “clustering” approach that automatically tries to optimize the model order (i.e. number of Gaussian components) through the use of some probabilistic cost function such as AIC or BIC [28, 32]. This adaptive approach allows for the complexity of the posterior to change over time to better model the true nature of the underlying process, although at an increased computational cost.

5.3.1 *The GMSPPF Algorithm*

The full GMSPPF algorithm will now be presented based on the component parts discussed above.

A) Time update and proposal distribution generation

At time *k-1*, assume the posterior state density is approximated by the G-component GMM

$$p_{\mathcal{G}}(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) = \sum_{g=1}^{G} \alpha_{k-1}^{(g)} \mathcal{N}\left(\mathbf{x}_{k-1};\boldsymbol{\mu}_{k-1}^{(g)}, \mathbf{P}_{k-1}^{(g)}\right), \tag{36}$$

and the process and observation noise densities are approximated by the following I and J component GMMs respectively

$$p_{\mathcal{G}}(\mathbf{v}_{k-1}) = \sum_{i=1}^{I} \beta_{k-1}^{(i)} \mathcal{N}\left(\mathbf{v}_{k-1};\boldsymbol{\mu}_{\boldsymbol{v}}{}_{k-1}^{(i)}, \mathbf{Q}_{k-1}^{(i)}\right) \tag{37}$$

$$p_{\mathcal{G}}(\mathbf{n}_{k}) = \sum_{j=1}^{J} \gamma_{k}^{(j)} \mathcal{N}\left(\mathbf{n}_{k};\boldsymbol{\mu}_{\boldsymbol{n}}{}_{k}^{(j)}, \mathbf{R}_{k}^{(j)}\right) \tag{38}$$

Following the GSF approach of [1], but replacing the EKF with a SPKF, the output of a bank of $G'' = GIJ$ parallel SPKFs are used to calculate GMM approximations of $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$ and $p(\mathbf{x}_k|\mathbf{Y}_k)$ according to the pseudo-code given below. For clarity of notation define $g' = g + (i-1)G$ and note that references to g' implies references to the respective g and i, since they are uniquely mapped. Similarly define $g'' = g' + (j-1)GI$ with the same implied unique index mapping. Now,

1. For $j{=}1 \ldots J$, set $\widetilde{p}_j(\mathbf{n}_k) = \mathcal{N}(\mathbf{n}_k; \boldsymbol{\mu}_{n,k}^{(j)}, \mathbf{R}_k^{(j)})$. For $i{=}1 \ldots I$, set $\widetilde{p}_i(\mathbf{v}_{k-1}){=}\mathcal{N}(\mathbf{v}_{k-1}; \boldsymbol{\mu}_{v,k-1}^{(i)}, \mathbf{Q}_{k-1}^{(i)})$ and for $g{=}1 \ldots G$, set $\widetilde{p}_g(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}){=}\mathcal{N}(\mathbf{x}_{k-1}; \boldsymbol{\mu}_{k-1}^{(g)}, \mathbf{P}_{k-1}^{(g)})$.
2. For $g'{=}1 \ldots G'$ use the time update step of a SPKF[11] (employing the system process equation (1) and densities $\widetilde{p}_g(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$ and $\widetilde{p}_i(\mathbf{v}_{k-1})$ from above) to

[11] The SPKF algorithm pseudo-code will not be given here explicitly. See [45, 39] for implementation details.

calculate a Gaussian approximate $\widetilde{p}_{g'}(\mathbf{x}_k|\mathbf{Y}_{k-1})=\mathcal{N}(\mathbf{x}_k;\widetilde{\boldsymbol{\mu}}_k^{(g')},\widetilde{\mathbf{P}}_k^{(g')})$ and update the mixing weights, $\alpha_k^{(g')}=\alpha_{k-1}^{(g)}\beta_{k-1}^{(i)}/(\sum_{g=1}^{G}\sum_{i=1}^{I}\alpha_{k-1}^{(g)}\beta_{k-1}^{(i)})$.

(a) For $g''=1\ldots G''$, complete the measurement update step of each SPKF (employing the system observation equation (2), current observation $\mathbf{y}_k$, and densities $\widetilde{p}_{g'}(\mathbf{x}_k|\mathbf{Y}_{k-1})$ and $\widetilde{p}_j(\mathbf{n}_k)$ from above) to calculate a Gaussian approximate $\widetilde{p}_{g''}(\mathbf{x}_k|\mathbf{Y}_k)=\mathcal{N}(\mathbf{x}_k;\boldsymbol{\mu}_k^{(g'')},\mathbf{P}_k^{(g'')})$. Also update the GMM mixing weights, $\alpha_k^{(g'')}=\alpha_k^{(g')}\gamma_k^{(j)}S_k^{(j)}/(\sum_{g'=1}^{G'}\sum_{j=1}^{J}\alpha_k^{(g')}\gamma_k^{(j)}S_k^{(j)})$, where $S_k^{(j)}=p_j(\mathbf{y}_k|\mathbf{x}_k)$ evaluated at $\mathbf{x}_k=\widetilde{\boldsymbol{\mu}}_k^{(g')}$ for the current observation, $\mathbf{y}_k$.

The predictive state density is now approximated by the GMM

$$p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_{k-1})=\sum_{g'=1}^{G'}\alpha_k^{(g')}\mathcal{N}\left(\mathbf{x}_k;\widetilde{\boldsymbol{\mu}}_k^{(g')},\widetilde{\mathbf{P}}_k^{(g')}\right) \tag{39}$$

and the posterior state density (which will only be used as the proposal distribution in the IS-based measurement update step) is approximated by the GMM

$$p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_k)=\sum_{g''=1}^{G''}\alpha_k^{(g'')}\mathcal{N}\left(\mathbf{x}_k;\boldsymbol{\mu}_k^{(g'')},\mathbf{P}_k^{(g'')}\right) \tag{40}$$

B) Measurement update

1. Draw N samples $\{\boldsymbol{x}_k^{(i)};l=1\ldots N\}$ from the proposal distribution $p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_k)$ (Equation 40) and calculate their corresponding importance weights

$$\tilde{w}_k^{(i)}=\frac{p(\mathbf{y}_k|\boldsymbol{x}_k^{(i)})p_{\mathcal{G}}(\boldsymbol{x}_k^{(i)}|\mathbf{Y}_{k-1})}{p_{\mathcal{G}}(\boldsymbol{x}_k^{(i)}|\mathbf{Y}_k)} \tag{41}$$

2. Normalize the weights: $w_k^{(l)}=\tilde{w}_k^{(l)}/\sum_{l=1}^{N}\tilde{w}_k^{(l)}$
3. Use a EM or WEM algorithm to fit a G-component GMM to the set of weighted particles $\{w_k^{(l)},\mathcal{X}_k^{(l)};l=1\ldots N\}$, representing the updated GMM approximate state posterior distribution at time k, i.e.

$$p_{\mathcal{G}}(\mathbf{x}_k|\mathbf{Y}_k)=\sum_{g=1}^{G}\alpha_k^{(g)}\mathcal{N}\left(\mathbf{x}_k;\boldsymbol{\mu}_k^{(g)},\mathbf{P}_k^{(g)}\right). \tag{42}$$

   The EM algorithm is 'seeded' by the G means, covariances and mixing weights of the prior state GMM, $p_{\mathcal{G}}(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$, and iterated until a certain convergence criteria (such as relative dataset likelihood increase) is met. Convergence usually occur within 4-6 iterations. Alternatively, an adaptive-model-order EM/WEM approach can be used to recover the posterior as discussed earlier.

C) Inference

The conditional mean state estimate $\hat{\mathbf{x}}_k=E[\mathbf{x}_k|\mathbf{Y}_k]$ and the corresponding error covariance $\hat{\mathbf{P}}_k=E[(\mathbf{x}_k-\hat{\mathbf{x}}_k)(\mathbf{x}_k-\hat{\mathbf{x}}_k)^T]$ can be calculated in one of two ways: The estimates can be calculated before the WEM smoothing stage by a direct weighted sum of the particle set,

$$\hat{\mathbf{x}}_k=\sum_{l=1}^{N}w_k^{(l)}\mathcal{X}_k^{(l)} \quad \text{and} \quad \hat{\mathbf{P}}_k=\sum_{l=1}^{N}w_k^{(l)}(\mathcal{X}_k^{(l)}-\hat{\mathbf{x}}_k)(\mathcal{X}_k^{(l)}-\hat{\mathbf{x}}_k)^T \tag{43}$$

or after the posterior GMM has been fitted,

$$
\begin{aligned}
\hat{\mathbf{x}}_k &= \sum_{g=1}^{G} \alpha_k^{(g)} \boldsymbol{\mu}_k^{(g)} \\
\hat{\mathbf{P}}_k &= \sum_{g=1}^{G} \alpha_k^{(g)} \left(\mathbf{P}_k^{(g)} + (\boldsymbol{\mu}_k^{(g)} - \hat{\mathbf{x}}_k)(\boldsymbol{\mu}_k^{(g)} - \hat{\mathbf{x}}_k)^T \right)
\end{aligned} \tag{44}
$$

Since $N \gg G$, the first approach is computationally more expensive than the second, but possibly generates better (lower variance) estimates, since it calculates the estimates before the implicit resampling of the WEM step. The choice of which method to use will depend on the specifics of the inference problem.

6 Experimental Results : SPPF & GMSPPF

6.1 Comparison of PF, SPPF & GMSPPF on nonlinear, non-Gaussian state estimation problem

In this section the estimation performance and computational cost of the SPPF and GMSPPF are evaluated on a state estimation problem and compared to that of the standard sampling importance-resampling particle filter (SIR-PF) [7]. The experiment was done using Matlab and the *ReBEL Toolkit*[12]. A scalar time series was generated by the following process model:

$$x_k = \phi_1 x_{k-1} + 1 + sin(\omega\pi(k-1)) + v_k \ \ , \tag{45}$$

where v_k is a Gamma $\mathcal{G}a(3,2)$ random variable modeling the process noise, and $\omega = 0.04$ and $\phi_1 = 0.5$ are scalar parameters. A nonstationary observation model,

$$y_k = \begin{cases} \phi_2 x_k^2 + n_k & k \leq 30 \\ \phi_3 x_k - 2 + n_k & k > 30 \end{cases} \tag{46}$$

is used, with $\phi_2 = 0.2$ and $\phi_3 = 0.5$. The observation noise, n_k, is drawn from a Gaussian distribution $\mathcal{N}(n_k; 0, 10^{-5})$. Figure 8 shows a plot of the hidden state and noisy observations of the time series. Given only the noisy observations y_k, the different filters were used to estimate the underlying clean state sequence x_k for $k = 1 \ldots 60$. The experiment was repeated 100 times with random re-initialization for each run in order to calculate Monte-Carlo performance estimates for each filter. All the particle filters used 500 particles and residual resampling where applicable (SIR-PF and SPPF only). Two different GMSPPF filters were compared: The first, GMSPPF (5-1-1), use a 5-component GMM for the state posterior, and 1-component GMMs for the both the process and observation noise densities. The second, GMSPPF (5-3-1), use a 5-component GMM for the state posterior, a 3-component GMM to approximate the “heavy tailed” Gamma distributed process noise and a 1-component GMM for the observation noise density. The process noise GMM was fitted to simulated Gamma noise samples with an EM algorithm. Both GMSPPFs use Inference Method 1 (Equation 43) to calculate the state estimate. Table 1 summarizes the performance of the different filters. The table shows the means and variances of the mean-square-error of the state estimates as well as the average processing time in seconds of each filter. The reason why the standard PF performs so badly on this problem is due to the highly peaked likelihood function of the observations (arising from the small observation noise variance) combined with the spurious jumps in the state due to the heavy tailed

[12] **ReBEL** is a **Matlab** toolkit for sequential Bayesian inference in general DSSMs. It contains a number of estimation algorithms including all those discussed in this paper as well as the presented GMSPPF. ReBEL is developed by the *MLSP Group* at OGI and can be freely downloaded from http://cslu.ece.ogi.edu/mlsp/rebel for academic and/or non-commercial use.

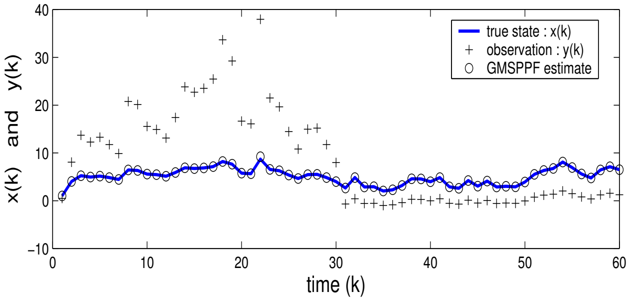


Figure 8: Nonstationary nonlinear time series estimation experiment.

Algorithm	MSE (mean)	MSE (var)	Time (s)
SIR-PF	0.2517	4.9e-2	1.70
SPPF	0.0049	8.0e-5	35.6
GMSPPF (5-1-1)	0.0036	5.0e-5	1.04
GMSPPF (5-3-1)	0.0004	3.0e-6	2.03

Table 1: Estimation results averaged over 100 Monte Carlo runs.

process noise. This causes severe "sample depletion" in the standard PF that uses the transition prior $p(x_k|x_{k-1})$ as proposal distribution. As reported in [38], the SPPF addresses this problem by moving the particles to areas of high likelihood through the use of a SPKF derived proposal distribution, resulting in a drastic improvement in performance. Unfortunately this comes at a highly increased computational cost. The GMSPPFs clearly have the same or better estimation performance (with reduced variance) as the SPPF but at a highly reduced computational cost. The best performance is achieved by the 5-3-1 GMSPPF that better models the non-Gaussian nature of the process noise.

6.2 Human face tracking using the SPPF

This section reports on the use of the SPPF for a video based human face tracking system. The system was developed my Yong Rui and his group at Microsoft Research [35] and directly utilizes our SPPF algorithm in an efficient C++ implementation. They found that the SPPF based face tracker is more robust than the baseline system based on Isard and Blake's CONDENSATION[13] [13] algorithm. Figure 9 shows the comparative results. The superior performance of the SPPF is evident. Sample video files of the tracking performance of both methods can be downloaded at: http://varsha.ece.ogi.edu/files/cond_tracking.mpg (CONDENSATION) and http://varsha.ece.ogi.edu/files/sppf_tracking.mpg (SPPF). For more experimental SPPF results, see [42, 38].

[13] CONDENSATION is the application of the generic likelihood based SIR particle filter to visual contour tracking.

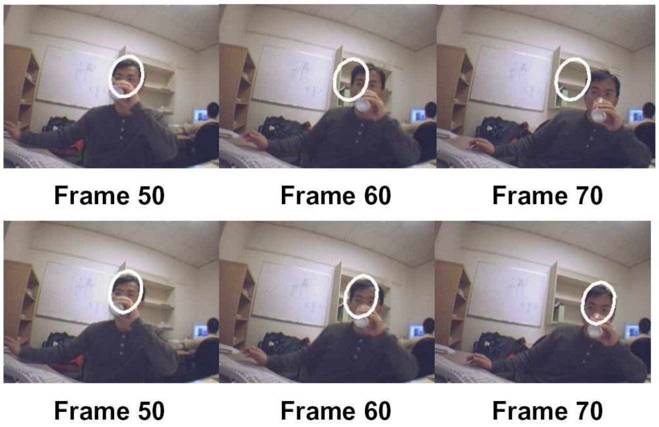


Figure 9: SPPF based human face tracking. The top plot shows the tracking results of a standard (CONDENSATION [13]) particle filter and the bottom plot shows the superior tracking performance of the SPPF.

6.3 Robot localization[14]

Mobile robot localization (MRL) is the problem of estimating a robot's *pose* (2D position and heading) relative to a map of its environment. This problem has been recognized as one of the fundamental problems in mobile robotics [4]. The mobile robot localization problem comes in two flavors. The simpler of these is *position tracking* where the robots *initial pose* is known, and localization seeks to correct small, incremental errors in the robots odometry. More challenging is the *global localization* problem, also known as the *hijacked robot problem*, where the robot is does not known its initial pose, but instead has to determine it from scratch based only only on noisy sensor measurements and a map of its environment. Note, the robot does not initially know *where* on the map it is. See Dieter Fox's publications a

In this inference problem there are often ambiguous or equally viable solutions active at any given moment (due to symmetries in the map and sensor data) which rules out simple single Gaussian representations of the posterior state distribution. Particle filter solutions are the algorithm of choice to tackle this problem [9] due to their ability to present the non-Gaussian multi-modal posterior state distribution related to the multiple *hypotheses* for the underlying robot pose. One counter intuitive problem often arising in mobile robot localization using standard particle filters is that very accurate sensors often results in worse estimation (localization) performance than using inaccurate sensors. This is due to the fact that accurate sensors have very peaked likelihood functions, which in turn results in severe particle depletion during the SIR step of the sequential Monte Carlo measurement update. When the effective size of the particle set become to small due to particle depletion, the particle filter can no longer accurate represent the true shape of the posterior state distribution resulting in inaccurate estimates, or even filter divergence when the correct hypothesis are no longer maintained (the correct mode in the posterior disappears). In order to address this problem one typically need to use a very large number (≥ 20000)of particles, resulting

[14] This work was done in collaboration with Dieter Fox and Simon Julier who provided the environment maps as well as the the observation likelihood model for the laser-based range finder sensors. The preliminary results of this collaboration is reported here, but will be presented more thoroughly in a future publication. See Dieter Fox's publications at http://www.cs.washington.edu/homes/fox/ for a wealth of information on mobile robot localization.

in a high computational cost. The irony is that the number of particles needed later on in the localization process, when much of the uncertainty or ambiguity of the robots pose has been reduced, is much less than what is initially needed. So, a large number of the particles eventually becomes superfluous but still incurs a computational burden. Recent work on *adaptive particle filters* using KLD sampling [8] attempts to address this by adapting the number of particles used in real time. We present here some preliminary results in using the GMSPPF to address the same problem. Take note, that due to the large number of particles needed to accurate represent the posterior (at least initially), the use of the SPPF, although highly accurate, would incur a prohibitively high computational cost.

The GMSPPF has two appealing features which make it an attractive solution to the MLR problem. As discussed in Section 5, the GMSPPF (and SPPF) moves particles to areas of high likelihood by using a proposal distribution that incorporates the latest measurement. This addresses the particle depletion problem. Furthermore, by using an adaptive clustering EM algorithm such as *x-means* initialized *FastEM* [9, 28] in the GMM recovery step, the model order (number of Gaussian components) is automatically adapted over time to more efficiently model the true nature of the posterior. We typically find that the model order which is initially set to a large number (to model the almost uniform uncertainty over the state space) slowly decreases over time as the posterior becomes more peaked in only a few ambiguous areas. The advantage of this is that the computational cost also reduces over time in relation to the model order, since we use a fixed number of particles per Gaussian component in the posterior GMM.

Figure 10 gives a graphical comparison of the localization results using a SIR particle filter (SIR-PF) on the left and the GMSPPF on the right. The observation model used in this example is based on a laser-range finder that measures distances to the closest walls in a $\pi/2$ radian fan centered around the robots current heading. Due to the high accuracy of such a sensor, the measurement noise variance was set to a low value ($\sigma = 10cm$). The SIR-PF uses 20,000 particles (shown in blue) to represent the posterior and uses the standard transition prior (movement model of the robot) as proposal distribution. Residual resampling is used. The initial particle set was uniformly distributed in the free-space of the map. The GMSPPF uses 500 samples per Gaussian component, the number of which are determined automatically using an x-means initialized EM clustering stage. The initial number of Gaussian components (shown in red) in the posterior was set to 42, uniformly distributed across the free-space of the map. The SIR-PF quickly suffers from severe particle depletion even though 20000 particles are used to represent the posterior. At $k = 10$ (third plot from the top on the left) the SIR-PF has already lost track of the true position of the robot. The GMSPPF in contrast accurately represent all possible robot location hypotheses through multiple Gaussian components in its GMM posterior. As the ambiguity and uncertainty of the robots location is reduced, the number of modes in the posterior decreases as well as the number of Gaussian component densities needed to model it. The GMSPPF accurately tracks the true position of the robot for the whole movement trajectory. The superior performance of the GMSPPF over that of the SIR-PF is clearly evident. Sample video files of the localization performance of both filters can be downloaded at: http://varsha.ece.ogi.edu/files/sirpf_mrl.mpg (SIR-PF) and http://varsha.ece.ogi.edu/files/gmsppf_mrl.mpg (GMSPPF).

7 Conclusions

Over the last 20 years the extended Kalman filter has become a standard technique within the field of probabilistic inference, used in numerous diverse estimation algorithms and related applications. One of the reasons for its wide spread use has been the clear understanding of the underlying theory of the EKF and how that relates to different aspects of the inference problem. This insight has been enhanced by unifying studies towards the rela-

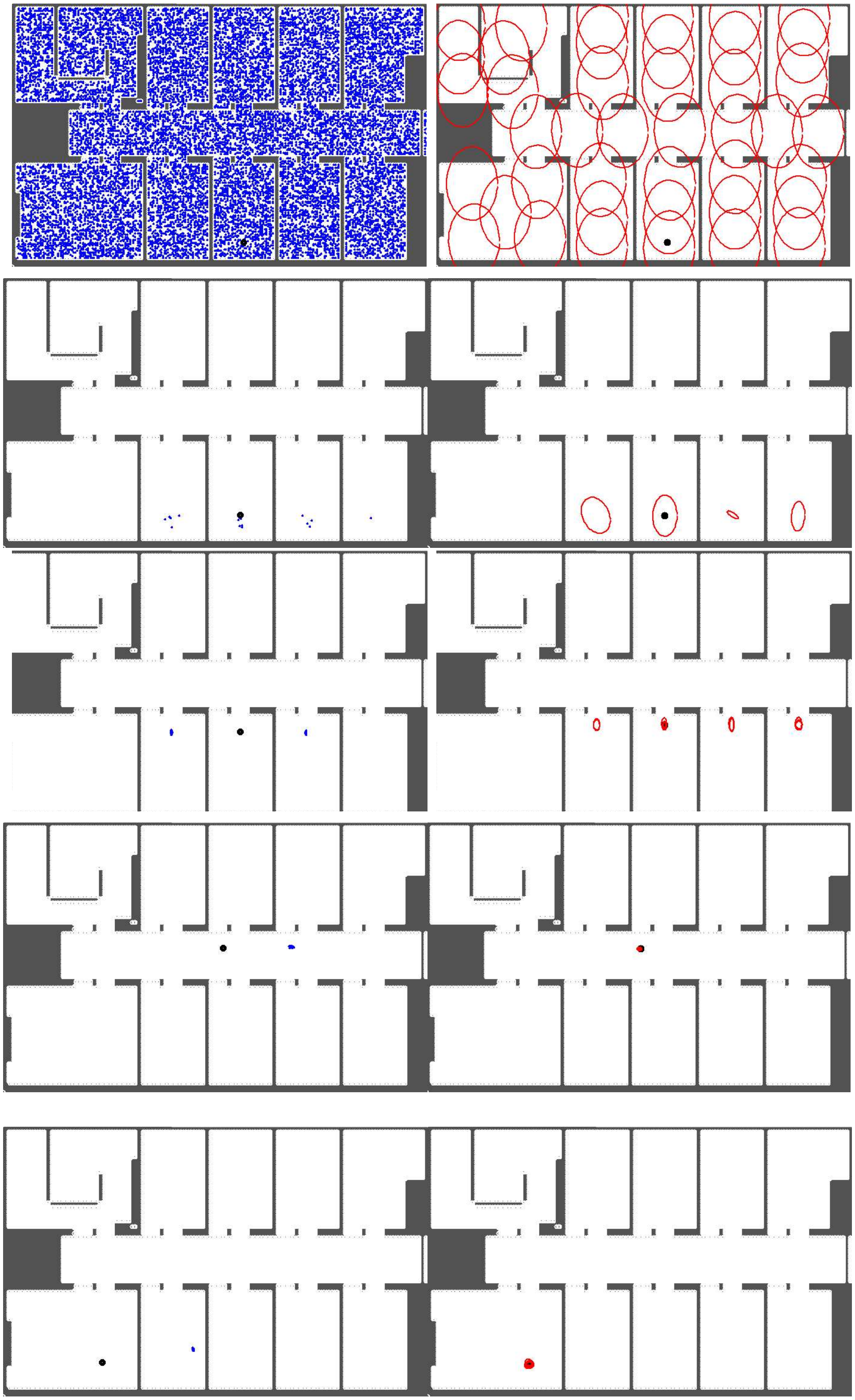


Figure 10: Mobile robot localization: Results using a SIR particle filter (left) and GMSPPF (right). The true robot position (black dot) and filter generated posterior is shown for $k = \{1, 2, 10, 20, 50\}$(from top to bottom) as the robot moves from one room up into the hallway, turn left, move down the hallway and down again into the large room. SIR-PF particles are shown in blue (on the left) and the GMSPPF's GMM posterior's Guassian component densities are shown in red on the right.

tionship between the EKF and other related algorithms [36], allowing for the improvement of numerous existing algorithms and the development of new ones.

Recently, the unscented Kalman filter (UKF) and the central difference Kalman filter (CDKF) have been introduced as viable and more accurate alternatives to the EKF within the framework of state estimation. Like most new algorithms, these methods were not widely known or understood and their application has been limited. We have attempted to unify these differently motivated and derived algorithms under a common family called sigma-point Kalman filters, and extended their use to other areas of probabilistic inference, such as parameter and dual estimation as well as sequential Monte Carlo methods. In doing so we have also extended the theoretical understanding of SPKF based techniques and developed new novel algorithmic structures based on the SPKF. The SPKF and its derivatives are clearly set to become an invaluable standard tool within the "machine learning toolkit".

References

[1] D. L. Alspach and H. W. Sorenson. Nonlinear Bayesian Estimation using Gaussian Sum Approximation. *IEEE Transactions on Automatic Control*, 17(4):439–448, 1972.

[2] B. Anderson and J. Moore. *Optimal Filtering*. Prentice-Hall, 1979.

[3] J. R. Cloutier, C. N. D'Souza, and C. P. Mracek. Nonlinear regulation and nonlinear H-infinity control via the state-dependent Riccati equation technique: Part1, Theory. In *Proceedings of the International Conference on Nonlinear Problems in Aviation and Aerospace*, Daytona Beach, FL, May 1996.

[4] I. J. Cox and G. T. Wilfong, editors. *Autonomous Robot Vehicles*. Springer Verlag, 1990.

[5] J. F. G. de Freitas. *Bayesian Methods for Neural Networks*. PhD thesis, Cambridge University Engineering Department, 1999.

[6] A. Doucet. On Sequential Simulation-Based Methods for Bayesian Filtering. Technical Report CUED/F-INFENG/TR 310, Department of Engineering, University of Cambridge, 1998.

[7] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte-Carlo Methods in Practice*. Springer-Verlag, April 2001.

[8] D. Fox. KLD-Sampling: Adaptive Particle Filters. In *Advances in Neural Information Processing Systems 14*, 2001.

[9] D. Fox, S. Thrun, F. Dellaert, and W. Burgard. *Sequential Monte Carlo Methods in Practice.*, chapter Particle filters for mobile robot localization. Springer Verlag, 2000.

[10] A. Gelb, editor. *Applied Optimal Estimation*. MIT Press, 1974.

[11] Z. Ghahramani and M. Beal. Variational Inference for Bayesian Mixture of Factor Analysers. In *Advances in Neural Information Processing Systems 12*, 1999.

[12] S. Haykin, editor. *Kalman Filtering and Neural Networks*. Wiley, 2001.

[13] M Isard and A Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, Cambridge, UK, 1996.

[14] Kazufumi Ito and Kaiqi Xiong. Gaussian Filters for Nonlinear Filtering Problems. *IEEE Transactions on Automatic Control*, 45(5):910–927, may 2000.

[15] A. Jazwinsky. *Stochastic Processes and Filtering Theory*. Academic Press, New York., 1970.

[16] S. Julier, J. Uhlmann, and H. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of the American Control Conference*, pages 1628–1632, 1995.

[17] S. J. Julier. A Skewed Approach to Filtering. In *SPIE Conference on Signal and Data Processing of Small Targets*, volume 3373, pages 271–282, Orlando, Florida, April 1998. SPIE.

[18] S. J. Julier and J. K. Uhlmann. A General Method for Approximating Nonlinear Transformations of Probability Distributions. Technical report, RRG, Dept. of Engineering Science, University of Oxford, Nov 1996. http://www.robots.ox.ac.uk/˜siju/work/publications/letter_size/Unscented.zip.

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

[20] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME Journal ofBasic Engineering*, pages 35–45, 1960.

[21] A. Lapedes and R. Farber. Nonlinear Signal Processing using Neural Networks: Prediction and System modelling. Technical Report LAUR872662, Los Alamos National Laboratory, 1987.

[22] T. Lefebvre, H. Bruyninckx, and J. De Schutter. Comment on 'A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators'. *IEEE Transactions on Automatic Control*, 47(8), Aug 2002.

[23] Frank L. Lewis. *Optimal Estimation*. John Wiley & Sons, Inc., New York, 1986.

[24] J. Liu and R. Chen. Sequential Monte Carlo Methods for Dynamic Systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[25] M. Mackey and L. Glass. Oscillation and chaos in a physiological control system. *Science*, 197(287), 1977.

[26] C. J. Masreliez. Approximate Non-Gaussian Filtering with Linear State and Observation Relations. *IEEE Transactions on Automatic Control*, AC-20:107–110, Feb 1975.

[27] G.J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley, 1997.

[28] Andrew Moore. Very fast em-based mixture model clustering using multiresolution kd-trees. In M. Kearns and D. Cohn, editors, *Advances in Neural Information Processing Systems*, pages 543–549, 340 Pine Street, 6th Fl., San Francisco, CA 94104, April 1999. Morgan Kaufman.

[29] Alex T. Nelson. *Nonlinear Estimation and Modeling of Noisy Time-Series by Dual Kalman Filtering Methods*. PhD thesis, Oregon Graduate Institute, 2000.

[30] M. Nørgaard, N. Poulsen, and O. Ravn. Advances in Derivative-Free State Estimation for Nonlinear Systems. Technical Report IMM-REP-1998-15, Dept. of Mathematical Modelling, Technical University of Denmark, 28 Lyngby, Denmark, April 2000.

[31] M. Norgaard, N. Poulsen, and O. Ravn. New Developments in State Estimation for Nonlinear Systems. *Automatica*, 36, 2000.

[32] Dan Pelleg and Andrew Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 727–734, San Francisco, 2000. Morgan Kaufmann.

[33] A. Pole, M. West, and P. J. Harrison. Non-normal and non-linear dynamic Bayesian modelling. In J. C. Spall, editor, *Bayesian Analysis of Time Series and Dynamic Models*. Marcel Dekker, New York, 1988.

[34] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C : The Art of Scientific Computing*. Cambridge University Press, 2 edition, 1992.

[35] Yong Rui and Yunqiang Chen. Better Proposal Distributions: Object Tracking Using Unscented Particle Filter. In *Proc. of IEEE CVPR*, volume II, pages 786–793, Kauai, Hawaii, Dec 2001.

[36] A. H. Sayed and T. Kailath. A State-Space Approach to Adaptive RLS Filtering. *IEEE Sig. Proc. Mag.*, July 1994.

[37] S. Singhal and L. Wu. Training multilayer perceptrons with the extended Kalman filter. In *Advances in Neural Information Processing Systems 1*, pages 133–140, San Mateo, CA, 1989. Morgan Kauffman.

[38] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan. The Unscented Particle Filter. In *Advances in Neural Information Processing Systems 13*, Nov 2001.

[39] R. van der Merwe and E. Wan. Efficient Derivative-Free Kalman Filters for Online Learning. In *Proc. of ESANN*, Bruges, April 2001.

[40] R. van der Merwe and E. Wan. The square-root unscented kalman filter for state and parameter-estimation. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Salt Lake City, Utah, May 2001.

[41] R. van der Merwe and E. A. Wan. Gaussian Mixture Sigma-Point Particle Filters for Sequential Probabilistic Inference in Dynamic State-Space Models. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Hong Kong, April 2003. IEEE.

[42] Rudolph van der Merwe, Nando de Freitas, Arnaud Doucet, and Eric Wan. The unscented particle filter. Technical Report CUED/F-INFENG/TR 380, Cambridge University Engineering Department, August 2000.

[43] E. Wan, R. van der Merwe, and A. T. Nelson. Dual Estimation and the Unscented Transformation. In *Neural Information Processing Systems 12*, pages 666–672. MIT Press, 2000.

[44] E. A. Wan and R. van der Merwe. The Unscented Kalman Filter for Nonlinear Estimation. In *Proc. of IEEE Symposium 2000 (AS-SPCC)*, Lake Louise, Alberta, Canada, October 2000.

[45] E. A. Wan and R. van der Merwe. *Kalman Filtering and Neural Networks, Ed. Simon Haykin.*, chapter 7 - The Unscented Kalman Filter. Wiley, 2001.