Improved Progressive Gaussian Filtering Using LRKF Priors

Gerhard Kurz[1] and Uwe D. Hanebeck[1]

***Abstract*— In this paper, we present a new extension to the Progressive Gaussian Filter (PGF) to improve its performance in certain situations. The PGF is a nonlinear Gaussian-assumed filter. Its basic idea is to gradually take the likelihood of the measurement into account when performing the measurement update step. Because the filter does not assume the prior and the measurement to be jointly Gaussian distributed, it outperforms commonly used LRKFs in strongly nonlinear scenarios. However, it suffers from suboptimality in cases where the model is close to linear, as the PGF does not contain the Kalman filter as a special case. To remedy this issue, we propose an extension where the PGF is combined with an LRKF.**

I. Introduction

The problem of nonlinear filtering consists in estimation of the state of a system whose internal dynamics or whose measure model is nonlinear. This issue has been considered by many authors over the past decades. So-called linear regression Kalman filters (LRKFs) [1] have emerged as a popular class of filters, for example the unscented Kalman filter (UKF) [2], the Gaussian Hermite Kalman filter [3], [4], the cubature Kalman filter (CKF) [5], [6], the smart sampling Kalman filter (S^{2}KF) [7] and the hyperspherical Kalman filter (HSKF) [8]. They rely on statistical linearization of the nonlinear system and/or measurement model using a set of samples. These approaches typically work well for scenarios that are at least locally (in relation to the current uncertainty) close to linear. For exactly linear systems, they reduce to the Kalman filter [9] and thus, are optimal with respect to the mean squared error (MSE). However, they suffer from low accuracy and even possible divergence in the presence of strong nonlinearities.

The more general class of Gaussian-assumed filters consists of approaches that merely assume the state after each prediction step and each measurement update step to be Gaussian distributed. In contrast to LRKFs, some of these approaches avoid linearization of the system and measurement function, for example the Gaussian Particle filter (GPF) [10]. This typically leads to better results in the presence of strong nonlinearities provided the Gaussian assumption is at least approximately fulfilled, especially during the measurement update step. Other examples are the progressive filters such as the Progressive Gaussian Filter (PGF) [11], [12], which only consider the difficult update step and rely on an LRKF for the prediction step. The downside is that these approaches are suboptimal on (almost) linear systems because they do not include the Kalman filter as a special case.

[1] The authors are with the Intelligent Sensor-Actuator-Systems Laboratory (ISAS), Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology (KIT), Germany. E-mail: gerhard.kurz@kit.edu, uwe.hanebeck@ieee.org

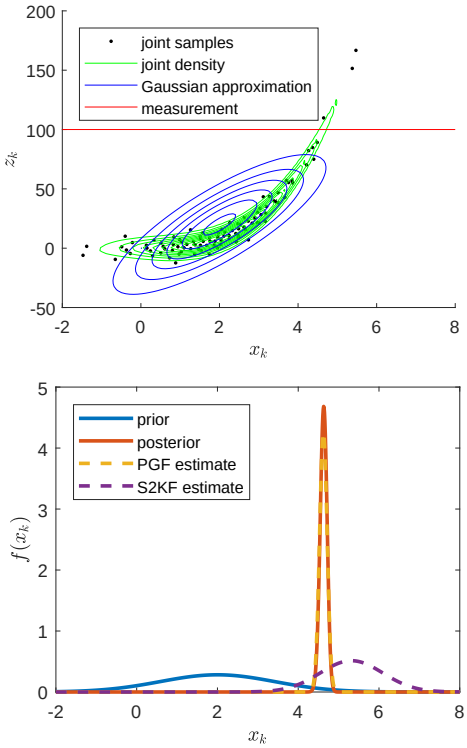


Fig. 1: Example for the measurement equation $z_k = x_k^3 + v_k$. Observe that the joint density of state and measurement is highly non-Gaussian.

As a result, we seek to combine LRKFs, which perform better for no or little nonlinearity, with general Gaussian-assumed filters, which perform better for strong nonlinearity. Note that the same system can exhibit varying degrees of nonlinearity depending on the current state and the current uncertainty, i.e., it is not sufficient to determine its nonlinearity once and choose the superior filter. Rather, we would like to combine the results from both approaches at runtime.

Example 1 (Limitations of LRKFs) *To illustrate the limitations of LRKFs, we consider the cubic measurement model $z_k = x_k^3 + v_k$. The prior density of the state x_k is given by a normal distribution with mean $\mu = 2$ and variance $\sigma^2 = 2$. The additive measurement noise v_k is distributed according to $v_k \sim \mathcal{N}(0, 30)$. Assume we obtain a measurement of $z_k = 100$.*

This scenario is depicted in Fig. 1. As can be seen, the true joint density $f(x_k, z_k)$ is highly non-Gaussian even though the prior density and the noise are Gaussian. However, all LRKFs approximate the joint density with a Gaussian, no matter how many samples are used. As a result, even LRKFs with many samples (such as the S^{2}KF) or LRKFs that match

higher moments (such as the higher order CKF) will not obtain a good result.

Despite the highly non-Gaussian joint density, the true posterior is approximately Gaussian in the considered example. As a result, Gaussian-assumed filters such as the GPF and the PGF can still obtain a very good approximation of the posterior density.

II. The Progressive Gaussian Filter (PGF)

The PGF was originally proposed in [11], [12]. Further development can be found in [13], [14], [15]. A similar approach has also been applied in the context of directional estimation [16], [17]. The concept of progressive filtering dates back to the work by Oudjane and Musso [18], who applied similar concepts to regularized particle filters.

The basic approach of the PGF is similar to the GPF [10]. Both filters draw samples from the prior density and reweigh them with the likelihood function according to Bayes' theorem. The GPF, however, has two problems. First, it draws the samples at random, and hence, is nondeterministic. Thus, the results are not reproducible and many samples are required just to properly capture the prior distribution. Second, it suffers from particle degeneration, in particular if the likelihood is very narrow or the state space is high-dimensional, i.e., a lot of particles get assigned a weight that is close or identical to zero. To address these two issues, the PGF relies on deterministic sampling and a progressive measurement update.

The idea of deterministic sampling is to approximate the prior density using a set of samples $\underline{s}_1, \ldots, \underline{s}_L$ that are obtained in a deterministic way rather than drawing samples at random. This can be done by moment matching (as in the UKF [2]), quadrature rules (as in the Gaussian Hermite KF [3]), or by approximating the shape of the continuous distribution using a suitable distance measure, usually with an additional constraint that mean and covariance are matched exactly. The PGF makes use of the approximation presented in [7], the same method as used by the S^{2}KF. Compared with random samples, a much smaller number of deterministic samples is typically sufficient to provide a good approximation of the true continuous density. Just as for the S^{2}KF, the samples can be precomputed for a standard Gaussian and transformed to a Gaussian with different mean and covariance using the Mahalanobis transformation [19, Sec. 3].

However, the problem of particle degeneration remains. The idea to solve this problem is to use a progressive measurement update, where the information contained in the measurement is fused gradually with the prior $f(\underline{x}_k)$. To achieve this, the PGF carries out the measurement update in several steps by applying Bayes' theorem and factoring the likelihood $f(\underline{z}_k|\underline{x}_k)$ according to

$$\begin{aligned} f(\underline{x}_k|\underline{z}_k) &= \frac{f(\underline{z}_k|\underline{x}_k) f(\underline{x}_k)}{f(\underline{z}_k)} \\ &= \frac{f(\underline{z}_k|\underline{x}_k)^{\lambda_1} \cdot \ldots \cdot f(\underline{z}_k|\underline{x}_k)^{\lambda_N} \cdot f(\underline{x}_k)}{f(\underline{z}_k)} \ , \end{aligned}$$

with step sizes and $\lambda_1, \ldots, \lambda_N > 0$, where

$$\sum\nolimits_{i=1}^{N} \lambda_i = 1 \ .$$

Each step reweighs the samples according to the corresponding factor $f(\underline{z}_k|\underline{x}_k)^{\lambda_i}$ of the likelihood, reapproximates the reweighted samples using a Gaussian density, and use deterministic sampling to obtain new equally weighted samples for this Gaussian. As the new samples are equally weighted, particle degeneration is avoided.

This poses the question of how to choose the step sizes $\lambda_1, \ldots, \lambda_N$. For this purpose, we introduce a predefined lower bound $R \in (0, 1)$ for the ratio between the smallest and the largest sample weight after reweighting

$$\frac{\min_j f(\underline{z}_k|\underline{s}_j)^{\lambda_i}}{\max_j f(\underline{z}_k|\underline{s}_j)^{\lambda_i}} \geq R \ ,$$

where the sample positions are $\underline{s}_1, \ldots, \underline{s}_L$. For

$$\min_j f(\underline{z}_k|\underline{s}_j) \neq \max_j f(\underline{z}_k|\underline{s}_j) \ ,$$

this yields an upper bound for the step size

$$\lambda_i \leq \frac{\log(R)}{\log(\min_j f(\underline{z}_k|\underline{s}_j)) - \log(\max_j f(\underline{z}_k|\underline{s}_j))} \ .$$

Otherwise, any step size will fulfill the condition. Note that this result is only valid if the prior density has uniform weights. The case of a prior density with weighted samples is considered in [16, Sec. VI-B].

Based on the upper bound for the step size, we always choose the largest admissible step size that also guarantees that $\sum_{i=1}^{N} \lambda_i \leq 1$. As soon as $\sum_{i=1}^{N} \lambda_i = 1$, the algorithm terminates. Pseudo code for the entire procedure can be found in [11, Algorithm 1].

III. Extension of the PGF using LRKF Priors

The PGF performs quite well in strongly nonlinear situations and often outperforms commonly used LRKFs. However, it is suboptimal for linear systems, and thus, is inferior to the LRKFs for problems that are close to linear. In order to improve the performance of the PGF in such scenarios, we seek to combine an LRKF and the PGF.

The basic idea of the proposed approach is to obtain the LRKF result $f^{\text{LRKF}}(\underline{x}_k|\underline{z}_k)$ first and to use it as a prior in the PGF. The standard PGF performs a progression from the prior $f(\underline{x}_k)$ to the (unnormalized) posterior $f(\underline{z}_k|\underline{x}_k) \cdot f(\underline{x}_k)$. Instead of $f(\underline{x}_k)$ we want to use $f^{\text{LRKF}}(\underline{x}_k|\underline{z}_k)$ as a starting point for the progression, as it tends to be closer to the true posterior than $f(\underline{x}_k)$. In the linear case, $f^{\text{LRKF}}(\underline{x}_k|\underline{z}_k)$ actually exactly matches the true posterior. The proposed approach is similar to the Unscented Particle Filter (UPF) [20], [21], where a UKF is used to generate a better prior for a particle filter.

To achieve this, we rewrite the filtering step as follows

$$\begin{aligned} f(\underline{x}_k|\underline{z}_k) \propto & f(\underline{z}_k|\underline{x}_k) \cdot f(\underline{x}_k) \\ &= f(\underline{z}_k|\underline{x}_k) \cdot f(\underline{x}_k) \cdot \overbrace{\frac{f^{\text{LRKF}}(x_k|\underline{z}_k)}{f^{\text{LRKF}}(\underline{x}_k|\underline{z}_k)}}^{1} \end{aligned}$$

$$= \underbrace{\left(\frac{f(\underline{z}_k|\underline{x}_k) \cdot f(\underline{x}_k)}{f^{\text{LRKF}}(\underline{x}_k|\underline{z}_k)} \right)}_{:=g(\underline{x}_k)} \cdot f^{\text{LRKF}}(\underline{x}_k|\underline{z}_k) \ .$$

By doing so, we can use the LRKF result as a prior, which is multiplied by a function $g(\underline{x}_k)$ to obtain the true posterior. Similar to the PGF, we can perform this multiplication gradually by factoring the modified likelihood $g(\underline{x}_k)$ according to

$$g(\underline{x}_k) = g(\underline{x}_k)^{\lambda_1} \cdot \ldots \cdot g(\underline{x}_k)^{\lambda_N} \ .$$

To compute the step size λ_i, we use the same approach as in Sec. II by considering $g(\underline{x}_k)$ instead of $f(\underline{z}_k|\underline{x}_k)$. For

$$\min_j g(\underline{s}_j) \neq \max_j g(\underline{s}_j) \ ,$$

this results in the condition

$$\lambda_i \leq \frac{\log(R)}{\log(\min_j g(\underline{s}_j)) - \log(\max_j g(\underline{s}_j))} \ ,$$

where $\underline{s}_1, \ldots, \underline{s}_L$ are the samples of the current progression step. Pseudo code of the novel method is given in Algorithm 1.

Algorithm 1: Proposed Measurement Update

Input: prediction $\underline{\mu}_k^p, \mathbf{C}_k^p$, measurement $\underline{z}_k$, threshold R

Output: estimate $\underline{\mu}_k^e, \mathbf{C}_k^e$

1 $\underline{\mu}^{\text{LRKF}}, \mathbf{C}^{\text{LRKF}} \leftarrow$S^{2}KFMeasUpdate$(\underline{\mu}_k^p, \mathbf{C}_k^e, \underline{z}_k)$;
2 $\underline{\mu} \leftarrow \underline{\mu}^{\text{LRKF}}, \mathbf{C} \leftarrow \mathbf{C}^{\text{LRKF}}$;
3 $N \leftarrow 0$;
/* Define function $g(\underline{x})$ */
4 $g \leftarrow \left(\underline{x} \mapsto \frac{f(\underline{z}_k|\underline{x}) \cdot \mathcal{N}(\underline{x}; \underline{x}_k^p, \mathbf{C}_k^e)}{\mathcal{N}(\underline{x}; \underline{\mu}^{\text{LRKF}}, \mathbf{C}^{\text{LRKF}})} \right)$;
5 **do**
6 $N \leftarrow N + 1$;
/* Compute samples */
7 $\underline{s}_1, \ldots, \underline{s}_L \leftarrow$sampleDeterministic$(\underline{\mu}, \mathbf{C})$;
/* Compute step size */
8 **if** $\min_j g(\underline{s}_j) \neq \max_j g(\underline{s}_j)$ **then**
9 $\lambda_N \leftarrow \frac{\log(R)}{\log(\min_j g(\underline{s}_j)) - \log(\max_j g(\underline{s}_j))}$;
10 **else**
11 $\lambda_N \leftarrow 1$;
12 **end**
13 **if** $\sum_{i=1}^N \lambda_i > 1$ **then**
14 $\lambda_N \leftarrow 1 - \sum_{i=1}^{N-1} \lambda_i$;
15 **end**
/* Reweigh samples */
16 $(w_1, \ldots, w_L) \leftarrow (g(\underline{s}_1)^{\lambda_N}, \ldots, g(\underline{s}_L)^{\lambda_N})$;
17 $W \leftarrow \sum_{j=1}^L w_j$;
/* Compute mean and covariance */
18 $\underline{\mu} = \frac{1}{W} \sum_{j=1}^L w_j \underline{s}_j$;
19 $\mathbf{C} \leftarrow \frac{1}{W} \sum_{j=1}^L w_j (\underline{s}_j - \underline{\mu})(\underline{s}_j - \underline{\mu})^T$;
20 **while** $\sum_{i=1}^N \lambda_i < 1$;
21 $\underline{\mu}_k^e = \underline{\mu}$, $\mathbf{C}_k^e = \mathbf{C}$;

Theorem 1 (Linear Case) *For a linear measurement equation with additive Gaussian noise, the result of the proposed filter is equal to that of the Kalman filter.*

Proof: In the case of a linear measurement equation, the S^{2}KF reduces to the Kalman filter. Hence, $\mathcal{N}(\underline{\mu}^{\text{LRKF}}, \mathbf{C}^{\text{LRKF}})$ is identical to the Kalman filter result and to the exact Bayesian estimator. It follows that $g(\underline{x}_k) = c$ for all $\underline{x}_k$ and some constant $c > 0$. Thus, the step size is $\lambda_1 = 1$ and the weights are $w_1 = \cdots = w_L = c$. As a result, $\underline{\mu}_k^e$ and $\mathbf{C}_k^e$ from Algorithm 1 correspond to the sample mean and sample covariance of $\underline{s}_1, \ldots, \underline{s}_L$, which in turn correspond to $\underline{\mu}^{\text{LRKF}}$ and $\mathbf{C}^{\text{LRKF}}$, because the deterministic sampling scheme maintains mean and covariance. ■

Remark 1 (Implementation Details) *There are some further details that should be considered when implementing the proposed algorithm. First of all, the following errors can occur and need to be handled.*

1) *Due to numerical inaccuracy or a pathological likelihood function, it can happen that $W \leq 0$ in line 17. In this case, we propose to abort the progression and to return the S^{2}KF estimate instead, which should usually perform better than the PGF's strategy of simply ignoring the measurement.*
2) *As a result of numerical issues, $\mathbf{C}$ may not be positive definite in line 19. In this case, we also propose to fall back to the S^{2}KF's estimate.*

To increase numerical stability, it is advisable to consider $\log(g(\cdot))$ instead of $g(\cdot)$ whenever possible. However, for the computation of the weights in line 16, we need the non-logarithmic version of the weights. At this point, we can compute the weights as

$$w_i = \exp\left(\lambda_N \cdot \left(\log(g(\underline{s}_i)) - \max_j \log(g(\underline{s}_j)) \right) \right) \ ,$$

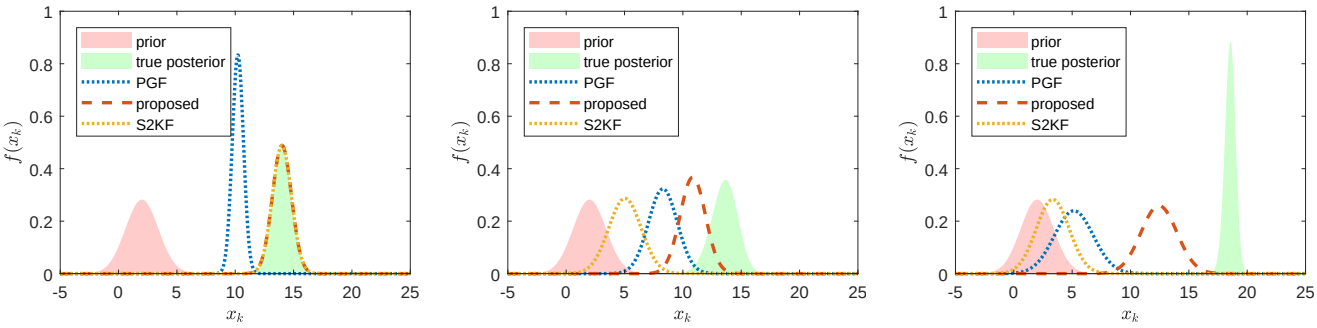
which differs up to a constant factor from the true weight. This constant factor does not change the results because of the normalization using $1/W$.

Remark 2 (Runtime) *The runtime of the proposed approach is essentially the sum of the runtime of the LRKF and the PGF. However, the number of progression steps necessary in the progressive part is typically lower than in the PGF.*

In the case of additive Gaussian noise, a d-dimensional state vector and L samples, the runtime of the S^{2}KF is $\mathcal{O}(Ld^2)$ plus $\mathcal{O}(L)$ evaluations of the measurement function. In the same setting, the PGF has a runtime $\mathcal{O}(NLd^2)$ plus $\mathcal{O}(NL)$ evaluations of the likelihood, where N is the number of progression steps. As a result, the runtime fo the proposed method is dominated by the cost of the PGF part.

IV. Evaluation

In this section, we evaluate the novel approach in several scenarios and compare it with multiple state-of-the-art methods. All evaluations were based on the implementations available in the nonlinear estimation toolbox [22]. We plan to include the novel filter in the toolbox as well.



$h_1(x) = x$
$z = 20$
$\sigma_v^2 = 1$

$h_2(x) = x^2$
$z = 20^2$
$\sigma_v^2 = 1000$

$h_3(x) = x^3$
$z = 20^3$
$\sigma_v^2 = 20000$

Fig. 2: Estimate with polynomial measurement function.

A. *Polynomial Measurement Example*

In the following, we consider a single measurement update step of a scalar system to illustrate the behavior of the proposed filter. The measurement models are given by

$$z_k = h_\alpha(x_k) + v_k \ ,$$

where $h_\alpha(x_k) = x_k^\alpha$ for a constant $\alpha \in \{1, 2, 3\}$ and measurement noise $v_k \sim \mathcal{N}(\mu = 0, \sigma_v^2)$. The prior distribution $f^p(x_k)$ is given by $x_k \sim \mathcal{N}(\mu = 2, \sigma^2 = 2)$. As an illustrative example, we perform a single measurement update with the measurement $z_k = 20^\alpha$. Note that this measurement is quite unlikely considering the prior density, which makes it difficult to process for the PGF. We compare the true posterior density to the estimate of the PGF, the proposed filter, and an LRKF, the S^{2}KF. The results are depicted in Fig. 1. All methods use the default of $L = 11$ samples and the progressive methods use the threshold $R = 1/L \approx 0.0909$ as suggested in [11, Sec. IV-C].

It can be seen that for $\alpha = 1$, i.e., the linear case, the LRKF is optimal and matches the true posterior exactly. As the proposed method is initialized with the LRKF's estimate, it also obtains the optimal result. The PGF is clearly inferior in this case, as it is generally not optimal for linear systems. For the quadratic measurement model ($\alpha = 2$), it can be seen that the PGF outperforms the LRKF significantly, but is inferior to the proposed filter. A similar result can be seen in the cubic case ($\alpha = 3$), but the advantage of the proposed approach is even more pronounced. The reason why the proposed approach outperforms the PGF in this case is that the proposed approach starts out with the LRKF's estimate, which is closer to the true posterior than the PGF's initial density, i.e., the prior density.

B. *Extended Object Tracking Scenario*

In the following, we consider a simple extended object tracking application. The exact same scenario was considered in [11, Sec. V], where the PGF, PGF42, GPF and a particle filter were compared. The PGF was shown to be superior to the other considered approaches. In this paper, we compare the novel PGF with LRKF priors, the S^{2}KF, and the classical PGF from [11].

The scenario considers a stick whose length l_k and one-dimensional position p_k are to be estimated. Thus, the state $\underline{x}_k = [l_k, p_k]^T$ is two-dimensional. Measurements can be obtained from any point of a stick and are subject to additive noise. The extent of the stick can be modeled using multiplicative noise [23], which leads to the measurement model

$$z_k = l_k \cdot v_k + p_k + r_k \ ,$$

where $r_k \sim \mathcal{N}(0, 0.15^2)$ is additive Gaussian noise and $v_k \sim \mathcal{U}(-1, 1)$ is uniform noise. In [23], it has been shown that LRKFs are unable to estimate the length of the stick in this scenario. Intuitively, this problem arises because both very large and very small measurements indicate a long stick, whereas measurements in the middle indicate a short stick, which causes the measurements and the length entry of the state vector to be uncorrelated (but obviously not independent).

The likelihood function is given by

$$f(z_k|\underline{x}_k) = \int_{-1}^{1} f_k^r(z_k - (l_k \cdot v_k + p_k)) \cdot f_k^v(v_k) \, \mathrm{d}v_k \ ,$$

where $f_k^v(\cdot)$ and $f_k^r(\cdot)$ refer to the densities of v_k and r_k, respectively. Numerical methods have to be used to evaluate the integral with respect to v_k. The system model used by the filters is given by a random walk model

$$\underline{x}_{k+1} = \underline{x}_k + \underline{w}_k \ ,$$

where the system noise is distributed according to

$$\underline{w}_k \sim \mathcal{N}\left(\underline{0}, \begin{bmatrix} 0.5 & 0 \\ 0 & 0.1 \end{bmatrix}\right) \ .$$

The initial estimate is given by

$$\underline{x}_0^e \sim \mathcal{N}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \ .$$

We assume that we obtain 50 measurements per time step. The PGF and the proposed approach use $L = 10$ samples, and the S^{2}KF uses its default number of 521 samples because

$$(\dim \underline{x}_k + 50 \cdot \dim \underline{v}_k) \cdot 10 + 1 = 521$$

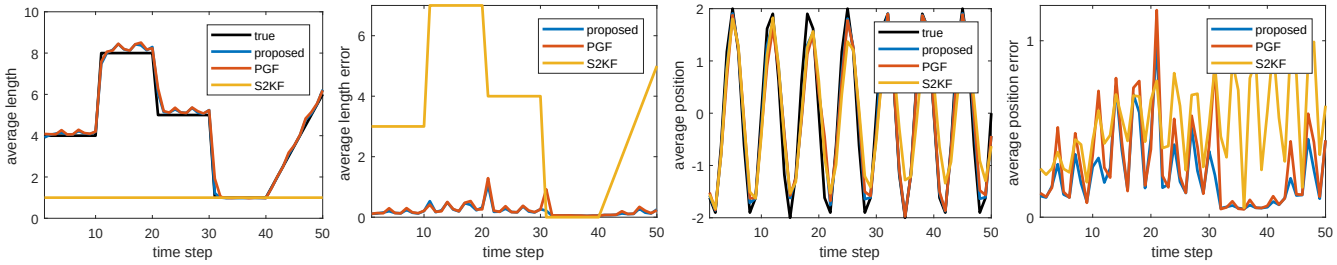


Fig. 3: Evaluation results from the extended object tracking scenario.

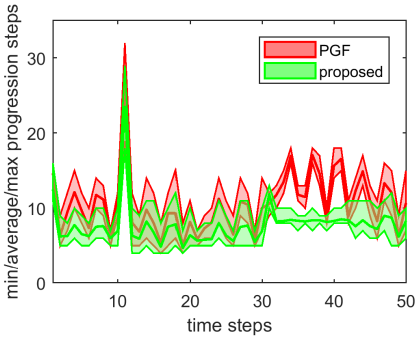


Fig. 4: Number of progression steps required in the extended object tracking scenario.

where 50 is the number of measurements and 10 is the number of samples per dimension. The progression threshold is chosen as $R = \frac{1}{L} = 0.1$ for both progressive methods as suggested in [11, Sec. IV-C].

The true target extent is defined as

$$l_k^{\text{true}} = \begin{cases} 4, & k \leq 10 \\ 8, & 10 < k \leq 20 \\ 5, & 20 < k \leq 30 \\ 1, & 30 < k \leq 40 \\ 1 + (k-40)/2, & 40 < k \end{cases}$$

and the true position is given by

$$p_k^{\text{true}} = -2\sin(2\pi k \cdot 0.15) \ .$$

We performed 100 Monte Carlo runs, each of which lasts 50 time steps. The resulting estimates and the errors in the estimates are shown in Fig. 3. It can be seen that the proposed filter and the PGF both perform very well, with a very slight advantage for the proposed approach. As expected, the S^{2}KF is unable to estimate the length. Its estimated position is also significantly less accurate than that of the PGF and the proposed method.

In Fig. 4, we show the number of progression steps required for the PGF (average over all Monte Carlo runs 10.57) and the proposed approach (average 7.78). It can be seen that the proposed method typically needs fewer progression steps, but one additional S^{2}KF step.

Filter	proposed	PGF	SIRPF	GPF	UKF	S^{2}KF
RMSE	0.630	0.642	0.680	0.751	1.306	1.453

TABLE I: Results of the vehicle tracking scenario over all time steps.

C. Vehicle Tracking Scenario

In the following, we consider the scenario of tracking a moving vehicle based on distance-only measurements. The state $\underline{x}_k \in \mathbb{R}^2$ corresponds to the 2D position the vehicle. The system model is given by a planar rotation around the origin

$$\underline{x}_{k+1} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \underline{x}_k + \underline{w}_k \ ,$$

where $\phi = 0.05\,\text{rad}$, i.e., the vehicle is slowly driving around the origin on a circular trajectory. The system noise $\underline{w}_k$ is distributed according to $\mathcal{N}(\underline{0}, 10^{-2} \cdot \mathbf{I})$.

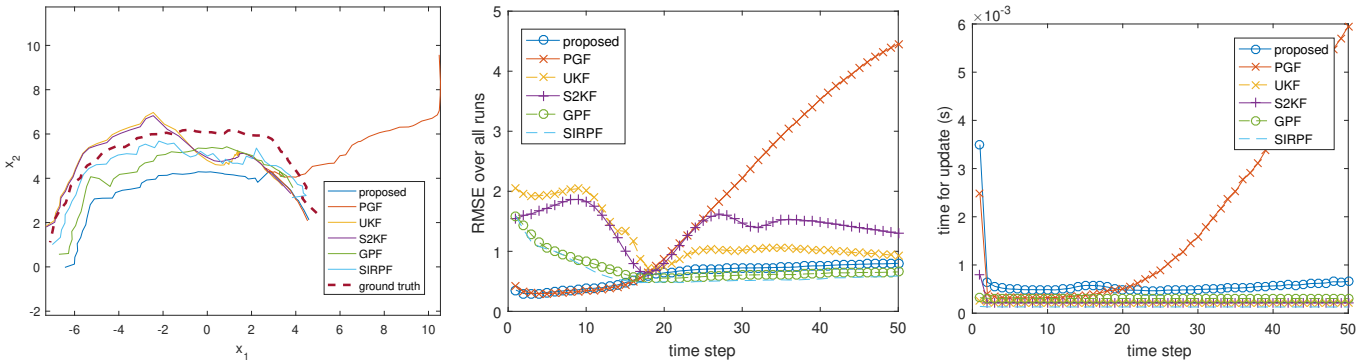
The measurements consist in the Euclidean distance to the point $\underline{p} = [2, 5]^T$, i.e., the measurement equation is

$$z_k = \|\underline{p} - \underline{x}_k\|_2 + v_k \ ,$$

where $v_k \sim \mathcal{N}(\underline{\mu} = \underline{0}, \sigma_v^2 = 0.1)$ is the measurement noise. The initial estimate is given by $\mathcal{N}([5, 2], 10 \cdot \mathbf{I})$ and the true initial state is $[5, 2]^T$.

We compare the proposed approach, the PGF, the UKF, the S^{2}KF, the GPF, and the SIR particle filter. For every method, we performed 1000 Monte Carlo runs with 50 time steps each. The UKF uses $L = 5$ samples, the S^{2}KF, the PGF, and the proposed filter use $L = 21$ samples, and the GPF as well as the SIRPF use $L = 1000$ samples. The progression threshold is chosen as $R = 1/L$.

The results are shown in Fig. 5. The root mean square error (RMSE) over all time steps and all runs is given in Table I. It can be seen that the LRKFs (UKF and S^{2}KF) have a lot of difficulties with this scenario. The other approaches work much better and it can be seen that the proposed filter and the PGF exhibit faster convergence. We also show the computation time on an Intel Core i7-4770 with 16 GB RAM and MATLAB 2016a. It can be seen that the proposed approach is somewhat slower than the classical methods, but much faster than the PGF. This is due to the fact that the PGF has a worse estimate and, as a result, requires more progression steps.



(a) Example trajectory of a single run. (b) RMSE averaged over all runs. (c) Computation time.

Fig. 5: Evaluation results of the vehicle tracking scenario.

V. Conclusion

In this paper, we have presented a novel extension to the PGF in order to improve its performance in situations that are nearly (or exactly) linear. The extension is based on computing a prior for the PGF using an LRKF. This way, the prior is already close to optimal in a nearly linear setting and the progression is only used to slightly refine the result.

We have performed multiple evaluations in different scenarios. The proposed filter generally performs as good as or better than both the PGF and the S^2KF.

As future work, it might be possible to implement the proposed approach on the GPU similar to the GPU implementation of the PGF [14], as the evaluation of the likelihood can be efficiently parallelized. It may also be possible to combine the new algorithm with the particle flow approach from [15]. Furthermore, we are interested in a more general investigation of the combination of multiple filters in order to combine their strengths and alleviate their weaknesses.

Acknowledgment

This work was partially supported by the German Research Foundation (DFG) under grant HA 3789/13-1.

References

[1] T. Lefebvre, H. Bruyninckx, and J. D. Schuller, "Comment on "A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators" [with Authors' Reply]," *IEEE Transactions on Automatic Control*, vol. 47, no. 8, pp. 1406–1409, Aug. 2002.

[2] S. J. Julier and J. K. Uhlmann, "Unscented Filtering and Nonlinear Estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.

[3] K. Ito and K. Xiong, "Gaussian Filters for Nonlinear Filtering Problems," *IEEE Transactions on Automatic Control*, vol. 45, no. 5, pp. 910–927, May 2000.

[4] Z. Wang and Y. Li, "Cross Approximation-based Quadrature Filter," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, Las Vegas, NV, USA, 2016.

[5] I. Arasaratnam and S. Haykin, "Cubature Kalman Filters," *IEEE Transactions on Automatic Control*, vol. 54, no. 6, pp. 1254–1269, 2009.

[6] B. Jia, M. Xin, and Y. Cheng, "High-degree Cubature Kalman Filter," *Automatica*, vol. 49, no. 2, pp. 510–518, 2013.

[7] J. Steinbring, M. Pander, and U. D. Hanebeck, "The Smart Sampling Kalman Filter with Symmetric Samples," *Journal of Advances in Information Fusion*, vol. 11, no. 1, pp. 71–90, Jun. 2016.

[8] G. Kurz and U. D. Hanebeck, "Linear Regression Kalman Filtering Based on Hyperspherical Deterministic Sampling," in *Proceedings of the 56th IEEE Conference on Decision and Control (CDC 2017)*, Melbourne, Australia, Dec. 2017.

[9] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[10] J. H. Kotecha and P. Djuric, "Gaussian Particle Filtering," *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2592–2601, 2003.

[11] J. Steinbring and U. D. Hanebeck, "Progressive Gaussian Filtering Using Explicit Likelihoods," in *Proceedings of the 17th International Conference on Information Fusion (Fusion 2014)*, Salamanca, Spain, Jul. 2014.

[12] U. D. Hanebeck, "PGF 42: Progressive Gaussian Filtering with a Twist," in *Proceedings of the 16th International Conference on Information Fusion (Fusion 2013)*, Istanbul, Turkey, Jul. 2013.

[13] Y. Huang, Y. Zhang, N. Li, and L. Zhao, "Gaussian Approximate Filter with Progressive Measurement Update," in *2015 54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 4344–4349.

[14] J. Steinbring and U. D. Hanebeck, "GPU-Accelerated Progressive Gaussian Filtering with Applications to Extended Object Tracking," in *Proceedings of the 18th International Conference on Information Fusion (Fusion 2015)*, Washington D. C., USA, Jul. 2015.

[15] C. Chlebek, J. Steinbring, and U. D. Hanebeck, "Progressive Gaussian Filter Using Importance Sampling and Particle Flow," in *Proceedings of the 19th International Conference on Information Fusion (Fusion 2016)*, Heidelberg, Germany, Jul. 2016.

[16] G. Kurz, I. Gilitschenski, and U. D. Hanebeck, "Recursive Bayesian Filtering in Circular State Spaces," *IEEE Aerospace and Electronic Systems Magazine*, vol. 31, no. 3, pp. 70–87, Mar. 2016.

[17] G. Kurz, F. Pfaff, and U. D. Hanebeck, "Nonlinear Toroidal Filtering Based on Bivariate Wrapped Normal Distributions," in *Proceedings of the 20th International Conference on Information Fusion (Fusion 2017)*, Xi'an, China, Jul. 2017.

[18] N. Oudjane and C. Musso, "Progressive Correction for Regularized Particle Filters," in *Proceedings of the Third International Conference on Information Fusion*, vol. 2. IEEE, 2000.

[19] W. Härdle and L. Simar, *Applied Multivariate Statistical Analysis*, 2nd ed. Berlin, Germany: Springer, 2007.

[20] R. van der Merwe, A. Doucet, N. de Freitas, and E. Wan, "The Unscented Particle Filter," Cambridge University Engineering Department, Cambridge CB2 1PZ, England, Tech. Rep. 380, Aug. 2000.

[21] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. A. Wan, "The Unscented Particle Filter," in *Advances in Neural Information Processing Systems*. MIT Press, 2001, pp. 584–590.

[22] J. Steinbring, "Nonlinear Estimation Toolbox," 2015. [Online]. Available: https://bitbucket.org/nonlinearestimation/toolbox

[23] M. Baum, F. Faion, and U. D. Hanebeck, "Modeling the Target Extent with Multiplicative Noise," in *Proceedings of the 15th International Conference on Information Fusion (Fusion 2012)*, Singapore, Jul. 2012.