Bootstrapping Particle Filters using Kernel Recursive Least Squares

Boris Oreshkin and Mark Coates
Department of Electrical and Computer Engineering
McGill University
Montreal, QC, Canada
Email: boris.oreshkin@mail.mcgill.ca, coates@ece.mcgill.ca

Abstract—Although particle filters are extremely effective algorithms for object tracking, one of their limitations is a reliance on an accurate model for the object dynamics and observation mechanism. The limitation is circumvented to some extent by the incorporation of parameterized models in the filter, with simultaneous on-line learning of model parameters, but frequently, identification of an appropriate parametric model is extremely difficult. This paper addresses this problem, describing an algorithm that combines Kernel Recursive Least Squares and particle filtering to learn a functional approximation for the measurement mechanism whilst generating state estimates. The paper focuses on the specific scenario when a training period exists during which supplementary measurements are available from a source that can be accurately modelled. Simulation results indicate that the proposed algorithm, which requires very little information about the true measurement mechanism, can approach the performance of a particle filter equipped with the correct observation model.

1. INTRODUCTION

Particle filters have proved to be an extremely effective methodology in tracking applications, particularly when the dynamics of the tracked object are highly non-linear or the observations are corrupted by non-Gaussian noise. One major shortcoming, however, is the need for a reasonably accurate model both of object dynamics and observation mechanisms. Performance can deteriorate dramatically if an incorrect model is applied. One approach to this issue is to incorporate a parametric model, thereby expanding the applicability of the filter, and to learn the model parameters online [1–7]. Successful application of these approaches requires that the true model is a member, or is at least closely approximated by a member, of the parameterized class of models.

When there is little or no *a priori* knowledge about the object dynamics or measurement mechanism, practitioners generally must resort to alternative methods that construct a model during their operation. A powerful such method that has been recently proposed is kernel recursive least squares (KRLS), which migrates the familiar recursive least squares algorithm into a kernel-space, allowing the method to capture non-linear dynamics [8]. KRLS uses the data to learn a dictionary of support vectors; the model dynamics are described as a functional mapping of the current state and these support vectors. Although KRLS can be applied to non-linear tracking problems, it is not as effective as particle filtering, particularly in noisy environments. Moreover, standard implementations of KRLS do not account for unobserved state parameters.

In this paper we combine kernel recursive least squares and particle filtering, addressing the "supervised learning" scenario, in which we assume that there is an initial training period during which additional measurements are available from a measurement device with known characteristics. An example of such a scenario arises in sensor network tracking, when accurate, but energy-expensive, range-finding or global-positioning devices can be activated for a short bursts of time to calibrate sensors. During these bursts, we apply KRLS to generate a dictionary and learn an observation model for secondary sensors. The secondary sensors are much less accurate but also consume much less energy. Subsequently, we use a particle filter based on this model that we have learned. This approach captures the best aspects of both algorithms — the learning power of KRLS and the robustness of particle filters to noise and missing observations.

We demonstrate through simulations of tracking scenarios how the combination of KRLS and particle filtering results in superior performance to a straightforward application of KRLS and approaches that of particle filtering when the model is known beforehand. For concreteness, we focus on the application of bearings-only tracking.

Related Work

The idea of online estimation of model parameters has received much attention since the introduction by Gordon of the sample roughening technique [9]. However, it was recognized in [6, 7] that the application of sample roughening to static parameters of the model may (and often does) lead to inadequate diffusion of posterior distributions and divergence. Kernel smoothing of parameters, as introduced by West [5], is used to address this problem in [7]. Recently, a more elegant and mathematically sound solution based on marginalization of the particle space with respect to unknown system parameters was proposed in [6]. In an alternative approach, Roweis and Ghahramani use the Expectation-Maximization algorithm to address uncertainty in a paramet-

1-4244-0525-4/07/$20.00 ©2007 IEEE
IEEEAC paper # 1329, Version 3, Updated January 14, 2007

ric model [1]. In their two-step algorithm they first use an Extended Kalman Smoother to estimate the approximate state distribution given the previous set of model parameters and then utilize a Radial Basis Network as a nonlinear regressor of model parameters given the approximate state estimate. Miguez at al. [2–4] recently introduced an alternative parametric technique for dealing with uncertainties in the state model. Their "Cost-Reference Particle Filter" uses a cost-reference function to address uncertainty in the system noise distribution. The function governing the system equation is parameterized with respect to a set of basis functions and the parameters of this basis representation are learned online via the Recursive Least Squares (RLS) algorithm. In contrast to these approaches, we strive in this work to avoid the introduction of a parametric model, and attempt to learn a non-parametric kernel representation of the unknown measurement function.

Outline of Paper

This paper is organized as follows. Section 2 reviews the nonlinear state-space model we adopt and discusses particle filtering as an efficient numerical approach to recursive Bayesian estimation. It also outlines the KRLS algorithm, which we apply to the problem of particle filtering with unknown observation mechanisms. Section 3 provides a concrete statement of the particular problem considered in this paper, and Section 4 outlines the KRLS-particle filter, our proposed algorithm for jointly performing function learning and estimation (tracking). Section 5 addresses the bearings-only tracking problem and describes how the KRLS-particle filter can be applied in this setting. Section 6 describes simulation experiments and discusses the results. Section 7 provides concluding remarks and describes ongoing work.

2. Background

Nonlinear State-Space Model

Many problems in signal processing may be solved by the application of a state-space estimation framework [10]. The state-space estimation approach is based on the following signal model:

$$\begin{aligned} \mathbf{x}_t &= f_x(\mathbf{x}_{t-1}) + \mathbf{v}_t \quad (1)\\ \mathbf{y}_t &= f_y(\mathbf{x}_t) + \mathbf{u}_t \quad (2) \end{aligned}$$

where $t = 1, 2, \dots$ is the discrete time, $\mathbf{x}_t$ denotes the state vector, and $\mathbf{y}_t$ indicates the measurement obtained at time t. The state of the system evolves according to the stochastic difference equation (1) characterized by the nonlinear mapping $f_x : \mathbb{R}^x \to \mathbb{R}^x$ and excited by white random noise $\mathbf{v}_t$. The state vector $\mathbf{x}_t$ is observed through the measurement vector $\mathbf{y}_t$, which is functionally related to $\mathbf{x}_t$ via the function $f_y : \mathbb{R}^x \to \mathbb{R}^y$ and corrupted by white random noise $\mathbf{u}_t$.

Particle Filtering

As discussed in [11], a practical Bayesian approach to the estimation of unobservable state $\mathbf{x}_t$ from the collection of measurements $\mathbf{y}_{1:t} \triangleq \{\mathbf{y}_1, \dots, \mathbf{y}_t\}$ available up to time t consists in sequential construction of prediction densities

$$p(\mathbf{x}_{t+1}|\mathbf{y}_{1:t}) = \int p(\mathbf{x}_t|\mathbf{y}_{1:t}) p(\mathbf{x}_{t+1}|\mathbf{x}_t) \mathrm{d}\mathbf{x}_t \quad (3)$$

and filtering densities

$$p(\mathbf{x}_{t+1}|\mathbf{y}_{1:t+1}) = \frac{p(\mathbf{y}_{t+1}|\mathbf{x}_{t+1}) p(\mathbf{x}_{t+1}|\mathbf{y}_{1:t})}{p(\mathbf{y}_{t+1}|\mathbf{y}_{1:t})} \quad (4)$$

However, closed form solutions to these recursions can be found only for linear Gaussian models and finite state-space representations of the Markovian model (1). In all other cases approximate numerical methods must be used.

Monte Carlo *particle filters* belong to the class of sequential approximation algorithms capable of solving (approximately) the problem (3), (4) by direct numerical simulation [12]. In this class of algorithms, the filtering distribution (4) is represented by the empirical point mass function of the particle set

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \approx \sum_{i=1}^{N} \omega_t^i \delta(\mathbf{x}_t - \mathbf{x}_t^i) \quad , \quad \sum_{i=1}^{N} \omega_t^i = 1 \quad (5)$$

where $\delta(\cdot)$ is the Dirac delta function and ω_t^i denotes the weight of each particle $\mathbf{x}_t^i$. The particle filter is initialized with a random sample $\{\mathbf{x}_0^i\}_{i=1}^N$ drawn from a known prior distribution $p(\mathbf{x}_0)$. Subsequent propagation of this particle set using the equations (3), (4) yields at every time step the approximation of state vector by a discrete statistical measure of the form (5). Many practical approaches to the problem of effective particle propagation have been developed during recent years (see [13, 14]).

Kernel Recursive Least Squares

The Kernel Recursive Least Squares algorithm was introduced in [8] and has a conceptual foundation related to Principle Component Analysis and Support Vector Machines. In contrast to these methods, KRLS is a fully online algorithm designed to operate in real-time environments where data become available one sample at a time. The KRLS algorithm can be used in nonlinear regression problems, time series prediction, and nonlinear adaptive filtering, but in the present work we exploit its function learning capabilities.

In our setting, KRLS is presented with input-output pairs $(\mathbf{x}_i, y_i)$ arising from an unknown mapping $f_y : \mathbb{R}^x \to \mathbb{R}^y$. These form the sequence V_t of available data:

$$V_t = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_t, y_t)\}.$$

The first step in KRLS involves the choice of a positive definite kernel function $\mathrm{k}(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$, which is defined as an inner product $\langle \cdot, \cdot \rangle$ of a possibly infinite dimensional mapping $\phi : \mathbb{R}^x \to \mathcal{H}$ that transforms the input vector into a Hilbert feature space.

KRLS attempts to learn an approximation to the mapping $f_y(\mathbf{x})$ in the form of a weighted linear sum of the kernels $\mathrm{k}(\mathbf{x}, \mathbf{x}_i)$, where $\{\mathbf{x}_i\}_{i=1}^t$ are the training data points up to time t. This approximation,

$$\hat{f}_y(\mathbf{x}) = \sum_{i=1}^{t} \alpha_i \, \mathrm{k}(\mathbf{x}_i, \mathbf{x}) \ , \tag{6}$$

is trained using the standard recursive least squares algorithm to minimize the squared error

$$\mathcal{L}(\boldsymbol{\alpha}_t) = \sum_{i=1}^{t} \left(y_i - \hat{f}_y(\mathbf{x}_i) \right)^2 \ . \tag{7}$$

In order to reduce the number of adjustable parameters in (6), KRLS employs a form of *online constructive sparsification*. The sparsification methodology only permits the addition of a training sample into the approximation (6) if it is approximately linearly independent of the preceding training samples (in the feature space). The input vectors that are included in the approximation form a *dictionary* $\mathcal{D}_t = \{\tilde{\mathbf{x}}_j\}_{j=1}^{m_t}$, which is sequentially constructed only from the samples $\mathbf{x}_t$ that satisfy an *approximate linear dependence (ALD)* test

$$\delta_t = \min_{\mathbf{a}} \left\| \sum_{j=1}^{t-1} a_j \phi(\tilde{\mathbf{x}}_j) - \phi(\mathbf{x}_t) \right\|^2 \leq \nu \ . \tag{8}$$

Here ν is the dictionary inclusion threshold that determines the accuracy of approximation (6). The coefficients $\mathbf{a} = (a_1, \ldots, a_{m_t - 1})^\top$ can be determined by solving (8):

$$\mathbf{a}_t = \tilde{\mathbf{K}}_{t-1}^{-1} \tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t) \tag{9}$$

where $[\tilde{\mathbf{K}}_{t-1}]_{i,j} = \mathrm{k}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$ and $[\tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t)]_i = \mathrm{k}(\tilde{\mathbf{x}}_i, \mathbf{x}_t)$. The weights $\boldsymbol{\alpha}_t = (\alpha_1, \ldots, \alpha_{m_t - 1})^\top$ are learned by KRLS over time through successive minimization of the approximation errors (7) in the least-squares sense.

3. Problem Statement

In standard particle filtering settings it is assumed that the transition functions f_x and f_y in (1) and (2) are completely known. In some extensions the assumption is relaxed to allow the inclusion of models with unknown parameters [1–7]. In many practical settings, the measurement model is unknown and even identification of a parametric model is challenging.

We consider the case where there is an initial training period, $t = 1, \ldots, K$, during which measurements are available according to a known model f_z:

$$\mathbf{z}_t = f_z(\mathbf{x}_t) + \mathbf{u}'_t$$

where $\mathbf{u}'_t$ is additive noise of distribution $p_{u'}$. Throughout the learning period, but also subsequently, we have measurements derived from an unknown mapping function f_y:

$$\mathbf{y}_t = f_y(\mathbf{x}_t) + \mathbf{u}_t$$

where $\mathbf{u}_t$ is additive noise of *known* distribution p_u. Our goal is to use the measurements $\mathbf{z}_t$ to learn an approximation to f_y that can be used to process the $\mathbf{y}_t$ measurements after the training period is over and form estimates of the state $\mathbf{x}_t$.

As a special case, we can consider the scenario of bias or nonlinear distortion, in which the $\mathbf{y}_t$ measurements are corrupted versions of the $\mathbf{z}_t$ measurements. In this case, the unknown mapping function is a composition of the known mapping f_z and an unknown contaminating function $g_y : \mathbb{R}^z \rightarrow \mathbb{R}^y$:

$$f_y = g_y \circ f_z \ . \tag{10}$$

In general, we anticipate that the training measurements $\mathbf{z}_t$ are generated by a high-precision measurement device, so the variance $\sigma_{u'}^2$ of the associated noise is much smaller than σ_u^2.

4. Function Approximation and Filtering Algorithm

In this section we describe the KRLS-particle filter, which combines the function approximation capabilities of KRLS and the tracking power of particle filtering to address uncertainty in the measurement model. Algorithm 1 provides a high-level description of the algorithm. During the first K time steps, a standard particle filter is applied to the measurements $\mathbf{z}_t$ to form estimates $\widehat{\mathbf{x}}_t$. These estimates are passed to the KRLS algorithm, which constructs a dictionary to learn an approximation $\hat{f}_y$ to the function f_y. After time step K, the resulting dictionary $\mathcal{D}$ and expansion coefficients $\boldsymbol{\alpha}$ are delivered to the particle filter algorithm, which now processes the $\mathbf{y}_t$ measurements. In this second stage, the true likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$ is not available, so $p_u(\mathbf{y}_t - \hat{f}_y(x_t))$ is used as an approximation. We assume that the distribution p_u is known.

In the special case where (10) applies and the measurements $\mathbf{y}_t$ are corrupted versions of $\mathbf{z}_t$, KRLS is used to form an estimate of the corrupting function g_y. This approach is adopted because the structure of g_y is often simpler to learn than that of f_y. We then approximate the likelihood as $p_u(\mathbf{y}_t - \hat{g}_y(f_z(\mathbf{x}_t)))$.

5. Multiple Observer Bearings-Only Tracking

As a clarifying example, we focus on the problem of bearings-only tracking. We consider a scenario where eight stationary sensors measure the bearings of a target and assume that the position of each sensor is known (Figure 1). The target movement is modeled according to a standard model with random Gaussian acceleration:

$$\mathbf{x}_t = \mathbf{F}\mathbf{x}_{t-1} + \mathbf{G}\mathbf{v}_t, \tag{11}$$

where $\mathbf{F}$, $\mathbf{G}$ are transition matrices, $\mathbf{x}_t$ is a state vector of the target, $\mathbf{v}_t$ is a random acceleration vector. These matrices and

Algorithm 1: Outline of Function Approximation and Filtering Algorithm

```
1  Set KRLS threshold $\nu$;
/* Initialize particle filter */
2  $\mathbf{x}_0^i \sim p_{u'}(\mathbf{x}_0)$;
3  for $t = 1, 2, \ldots, K$ do
   /* High-precision measurements */
4      Data: $z_t$
   /* Low-precision measurements */
5      Data: $y_t$
   /* Sample from proposal */
6      $\mathbf{x}_t^i \sim q(\mathbf{x}_{t-1}^i, z_{1:t})$;
   /* Evaluate particle weights */
7      $\tilde{\omega}_t^i = p_{u'}(z_t - f_z(\mathbf{x}_t^i))\frac{p(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_{t-1}^i, z_{1:t})}\omega_{t-1}^i$;
8      $\omega_t^i = \frac{\tilde{\omega}_t^i}{\sum_{i=1}^N \tilde{\omega}_t^i}$;
   /* Calculate estimate of state */
9      $\widehat{\mathbf{x}}_t = \mathbf{x}_t^\top \omega_t$;
   /* Resample */
10     $\mathbf{x}_t^i = \text{Resample}(\mathbf{x}_t^i, \omega_t^i)$;
   /* Run KRLS algorithm */
11     $(\mathcal{D}_t, \alpha_t) = \text{KRLS}(\widehat{\mathbf{x}}_t, y_t, \mathcal{D}_{t-1}, \alpha_{t-1})$
12 endfor
/* KRLS output: */
13 Data: $(\mathcal{D}, \alpha)$
14 for $t = K+1, K+2, \ldots$ do
   /* Sample from proposal */
15     $\mathbf{x}_t^i \sim q(\mathbf{x}_{t-1}^i, y_{1:t})$;
   /* Evaluate particle weights */
16     $\tilde{\omega}_t^i = p_u(y_t - \hat{f}_y(\mathbf{x}_t^i))\frac{p(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_{t-1}^i, y_{1:t})}\omega_{t-1}^i$;
17     $\omega_t^i = \frac{\tilde{\omega}_t^i}{\sum_{i=1}^N \tilde{\omega}_t^i}$;
   /* Calculate estimate of state */
18     $\widehat{\mathbf{x}}_t = \mathbf{x}_t^\top \omega_t$;
   /* Resample */
19     $\mathbf{x}_t^i = \text{Resample}(\mathbf{x}_t^i, \omega_t^i)$;
20
21 endfor
```

vectors are of the following form:

$$\mathbf{x}_t = \begin{pmatrix} x_{1,t} \\ x_{2,t} \\ v_t^{x_1} \\ v_t^{x_2} \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (12)$$

$$\mathbf{v}_t = \begin{pmatrix} v_t^{x_1} \\ v_t^{x_2} \end{pmatrix}, \quad \mathbf{G} = \begin{pmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{pmatrix} \quad (13)$$

Observers are fixed with the state of the j-th observer being:

$$\mathbf{R}_j = \begin{pmatrix} r_j^{x_1} \\ r_j^{x_2} \\ 0 \\ 0 \end{pmatrix} \quad (14)$$

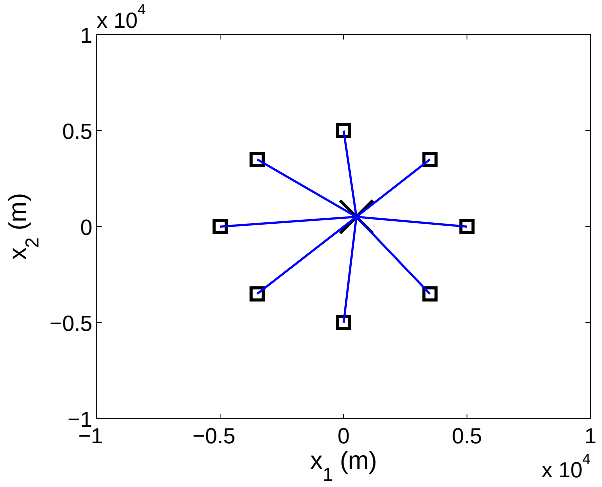


Figure 1. Outline of experimental setup: eight stationary observers measure the bearings of the target. Observers are marked as □ and target is marked as ×.

For the first K time steps t we record the bearings of the target. These measurements are corrupted by independent identically distributed additive white Gaussian noise $\mathbf{u}_t'$:

$$\mathbf{z}_t = f_z(\mathbf{x}_t) + \mathbf{u}_t', \quad (15)$$

$$f_z(\mathbf{x}_t) = \begin{pmatrix} \tan^{-1}\left(\frac{x_{2,t} - r_1^{x_2}}{x_{1,t} - r_1^{x_1}}\right) \\ \vdots \\ \tan^{-1}\left(\frac{x_{2,t} - r_j^{x_2}}{x_{1,t} - r_j^{x_1}}\right) \\ \vdots \end{pmatrix} \quad (16)$$

We also have low-precision measurements y_t derived from an unknown measurement function $f_y(\mathbf{x}_t)$. We impose the assumption that the measurements y_t are corrupted versions of the true bearings, so $f_y = g_y \circ f_z$. Note that this assumption implies that the influence of the actual coordinates on the measurement process is solely through the bearings. For example, according to this model, the distance from the sensor has no effect on the nature of the measurement. The algorithm described in the previous section is used to learn the measurement function g_y on-line. For this specific case, we utilize a Gaussian kernel with variance σ^2 within the KRLS algorithm. At each time step, KRLS takes as input pairs $\{(\mathbf{z}_t^j, \mathbf{y}_t^j); j = 1, \ldots, 8\}$ and updates its dictionary $\mathcal{D}_t$ and function approximation coefficients α_t. The algorithm places no restrictions on the nature of the particle filter (importance sampling function, resampling strategies, etc.).

6. Simulation Results

To evaluate the performance of proposed online function learning technique we conducted a set of simulation experiments based on the bearings-only tracking problem described in Section 5. Matlab code implementing the algorithm is available at [15].

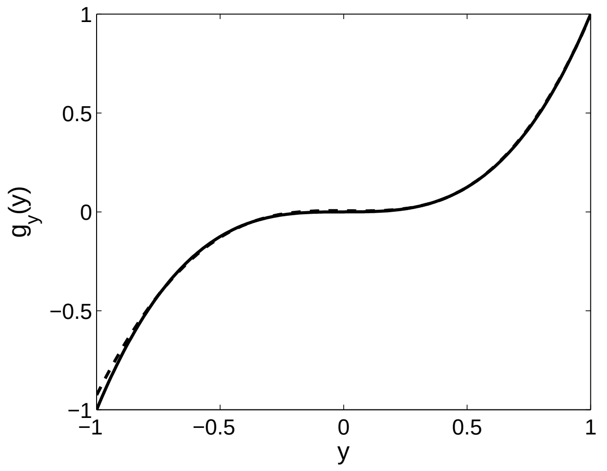


Figure 2. An example of the function approximation capability of KRLS. The Function $g_y(y) = y^3$ is learned from 50 time steps of noisy data with variance $\sigma^2_{u'} = 2.78 \times 10^{-4}$. The actual function is the solid line; the approximation is the dashed line.

We evaluate mean square error (MSE) performance of particle filtering algorithms with and without the KRLS function approximation step. Our comparison is based on the following absolute MSE measure:

$$\bar{\epsilon}^2 = \frac{1}{L(M-1-K)} \sum_{i=1}^{L} \sum_{t=K+1}^{M} (x_{t,i} - \hat{x}_{t,i})^2 + (y_{t,i} - \hat{y}_{t,i})^2 \tag{17}$$

where L is the number of independent trials and M is the length of target trajectory. Three variants of particle filter configuration are compared:

1. a particle filter with complete knowledge of both measurement and contaminating functions (PF-IDEAL)
2. a particle filter with knowledge of measurement function and KRLS approximation to contaminating function (PF-KRLS)
3. a particle filter with knowledge of measurement function and with no knowledge of contaminating function or KRLS approximation step (PF-UNAWARE)

We used $L = 50$ trials to generate the results, fixing the trajectory length to $M = 80$ samples and the learning period to $K = 50$ samples. In our experiments we utilized basic particle filter configuration with sampling from prior, residual resampling, and $N = 1000$ particles. We considered two values for the noise variance of the low-precision measurements $\mathbf{y}_t$: $\sigma^2_u = 2.78 \times 10^{-4}$ and $\sigma^2_u = 2.78 \times 10^{-3}$. The MSE performance results for these two cases are presented in Tables 1 and 2, respectively. We investigated the performance of the algorithms for three different contaminating functions $g(y)$: bias in measurements $g_y(y) = y + 0.01$, nonlinear distortion of polynomial type $g_y(y) = y^3$, and nonlinear distortion of exponential type $g_y(y) = e^y$. Figure 3 illustrates the function approximation capability of the KRLS algorithm. With 50 samples to learn from, KRLS generates an approximation to the actual function $g_y(y) = y^3$ that achieves very good accuracy almost everywhere in the region of interest.

Table 1. MSE performance of particle filter in three different settings: PF-IDEAL, PF-KRLS, and PF-UNAWARE. Variance of low-precision measurement device is $\sigma^2_u = 2.78 \times 10^{-4}$.

$g_y(y)$	$y + 0.01$	y^3	e^y
PF-IDEAL,	0.19	0.15	0.14
PF-KRLS	1.34	5.86	42.99
PF-UNAWARE	9120.90	51099.76	143510.54

Table 2. MSE performance of particle filter in three different settings: PF-IDEAL, PF-KRLS, and PF-UNAWARE. Variance of low-precision measurement device is $\sigma^2_u = 2.78 \times 10^{-3}$.

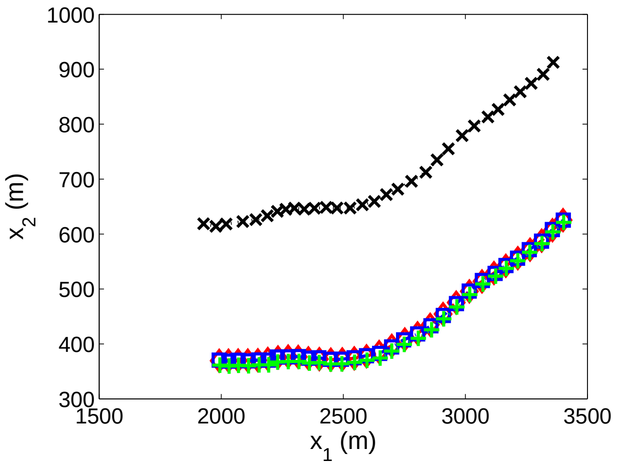
$g_y(y)$	$y + 0.01$	y^3	e^y
PF-IDEAL,	5.13	3.28	3.66
PF-KRLS	6.62	24.03	7.93
PF-UNAWARE	11.30	43766.17	17342.11

It is clear that the particle filter algorithm is very sensitive to variations in the measurement function. If the particle filter is unaware of even a comparatively small bias $g_y(y) = y + 0.01$ (which represents only 0.5% of the range of the measurement), then its performance is very poor. The KRLS-particle filter provides orders-of-magnitude improvement in MSE performance compared to the particle filter with no knowledge of the contaminating function. As expected, the KRLS-particle filter cannot compete in performance with the particle filter that has complete knowledge of both measurement and contaminating function. Small errors in the function approximation can induce large errors in state estimation.

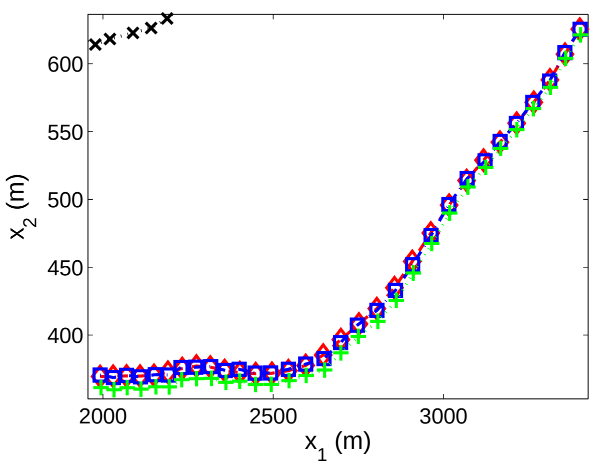
Figure 2 shows examples of tracking trajectories corresponding to the MSE results presented in Table 1. Despite its minimal assumptions and prior knowledge, the proposed online function learning approach using KRLS algorithm can offer reasonably good performance compared to particle filter with complete knowledge.

7. Discussion and Future Work

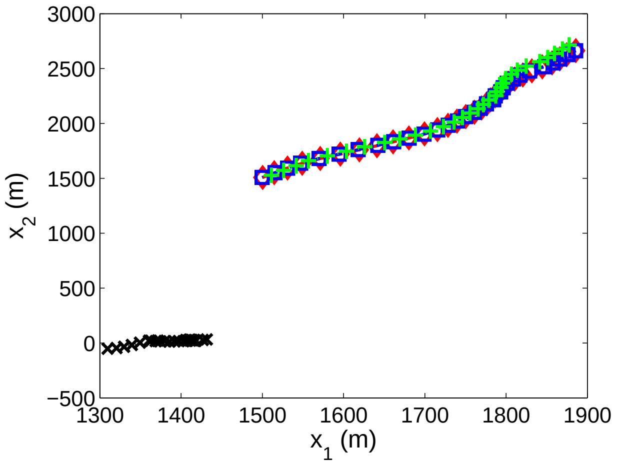
Although the KRLS-particle filter we have described does a good job in compensating for unknown distortions or biases in the measurement model, it has the major shortcoming that it is reliant on supervised learning, requiring a set of measurements from a well-modelled device. Our primary goal in



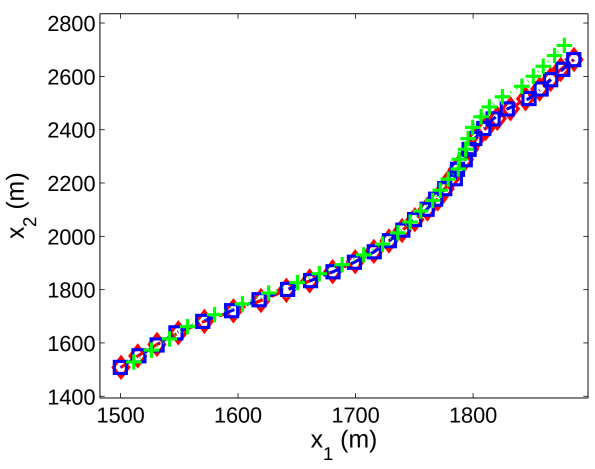
(a) Contaminating function $g_y(y) = y + 0.01$



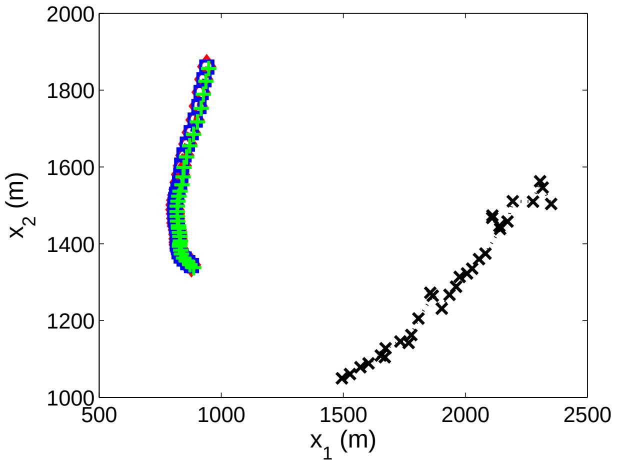
(b) Contaminating function $g_y(y) = y + 0.01$



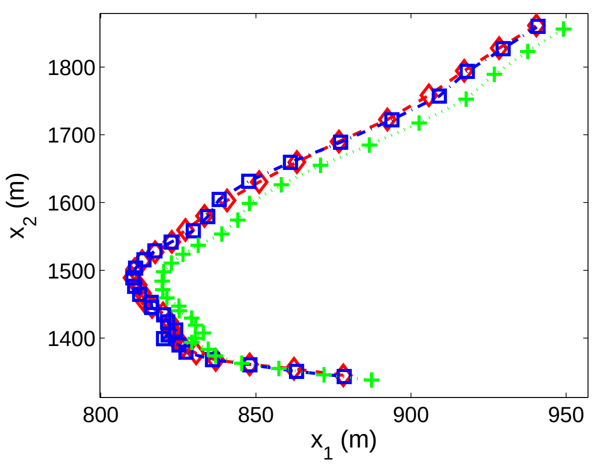
(c) Contaminating function $g_y(y) = y^3$



(d) Contaminating function $g_y(y) = y^3$



(e) Contaminating function $g_y(y) = e^y$



(f) Contaminating function $g_y(y) = e^y$

Figure 3. Examples of tracking trajectories corresponding to the MSE results presented in Table 1. Here $\square$ denotes result of tracking the trajectory in PF-IDEAL scenario, $\times$ corresponds to PF-UNAWARE scenario, + denotes PF-KRLS scenario, and $\diamond$ shows real trajectory of the target. Right column of figures shows zoomed versions of figures in the left column.

future work is to extend the technique to address the unsupervised learning situation, in which KRLS and the particle filter interact and use the same single set of measurements to learn the observation model and generate state estimates. Beyond this, it is highly desirable to consider the situation where the measurement model is well-understood but there is no available model for the object dynamics.

References

[1] S. Roweis and Z. Ghahramani, "Learning nonlinear dynamical systems using the expectation-maximization algorithm," in *Kalman Filtering and Neural Networks*, S. Haykin, Ed. New York, NY, USA: John Wiley & Sons, Inc., 2001, ch. 6.

[2] J. Miguez, S. Xu, W. F. Bugallo, and P. M. Djuric, "Joint estimation of states and transition functions of dynamic systems using cost-reference particle filtering," in *Proc. ICASSP*, Philadelphia, PA, Mar. 2005.

[3] J. Miguez, M. F. Bugallo, and P. M. Djuric, "A new class of particle filters for random dynamic systems with unknown statistics," *EURASIP J. Applied Signal Proc.*, no. 15, pp. 2278–2294, 2004.

[4] M. F. Bugallo, S. Xu, J. Miguez, and P. M. Djuric, "Maneuvering target tracking using cost reference particle filtering," in *Proc. ICASSP*, Montreal, QC, May 2004.

[5] M. West, "Approximating posterior distributions by mixtures," *Journal of the Royal Statistical Society. Series B.*, vol. 55, no. 2, pp. 409–422, 1993.

[6] R. Martinez-Cantin, N. de Freitas, and J. A. Castellanos, "Marginal-SLAM: A convergent particle method for simultaneous robot localization and mapping," 2006, unpublished.

[7] J. Liu and M. West, "Combined parameter and state estimation in simulation-based filtering," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds. Berlin: Springer-Verlag, 2001, pp. 197–223.

[8] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares algorithm," *IEEE Trans. Signal Proc.*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[9] N. J. Gordon, "Bayesian methods for tracking," Ph.D. dissertation, University of London, 1993.

[10] S. Haykin, *Kalman Filtering and Neural Networks*. New York, NY, USA: John Wiley & Sons, Inc., 2001.

[11] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *Radar and Signal Processing, IEE Proceedings F*, vol. 140, no. 2, Apr. 1993, pp. 107–113.

[12] A. Doucet, N. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Proc.*, vol. 49, no. 3, pp. 613–624, Mar. 2001.

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

[14] R. Douc, O. Cappe, and E. Moulines, "Comparison of resampling schemes for particle filtering," in *Proc. Int. Symp. on Image and Signal Proc. and Analysis*, Zagreb, Croatia, Sept. 2005.

[15] B. Oreshkin and M. Coates. Bootstrapping particle filters using kernel recursive least squares. Project Description. [Online]. Available: http://www.tsp.ece.mcgill.ca/Networks/projects/proj-particlefiltering.html

Mark Coates *received the B.E. degree (first class honours) in computer systems engineering from the University of Adelaide, Australia, in 1995, and a Ph.D. degree in information engineering from the University of Cambridge, U.K., in 1999. Currently, he is an Assistant Professor at McGill University, Montreal, Canada. He was awarded the Texas Instruments Postdoctoral Fellowship in 1999 and was a research associate and lecturer at Rice University, Texas, from 1999-2001. His research interests include communication and sensor/actuator networks, statistical signal processing, causal analysis, and Bayesian and Monte Carlo inference. His email address is coates@ece.mcgill.ca.*

Boris Oreshkin *received the B.E. degree with honours in electrical engineering from Ryazan State Academy of Radio Engineering, Russia, in 2004. From 2004 to 2006 he was doing graduate research in Moscow Aviation Institute. Also at this time he was involved in hardware and software design and Digital Signal Processing algorithms simulation and implementation in industry. He was certified by Analog Devices as Blackfin Field Application Engineer. Currently, he is pursuing his PhD degree in electrical engineering at McGill University, Montreal, Canada. In 2006 he was awarded Dean's Doctoral Student Research Recruitment Award by the Faculty of Engineering at McGill University. His research interests include statistical signal processing, detection and estimation theory, and Bayesian and Monte Carlo inference. His email address is boris.oreshkin@mail.mcgill.ca.*