DATA-DRIVEN ONLINE VARIATIONAL FILTERING IN WIRELESS SENSOR NETWORKS

Hichem Snoussi [(1)], *Jean-Yves Tourneret* [(2)], *Petar M. Djurić* [(3)], *Cédric Richard* [(1)]

[(1)] ICD/LM2S, Université de Technologie de Troyes, 10010, Troyes, France
[(2)] Université de Toulouse, IRIT/ENSEEIHT/TéSA, 31071, Toulouse, France
[(3)] Dept. Elec. and Comp. Eng., Stony Brook University, Stony Brook, New York 11794-2350

ABSTRACT

In this paper, a data-driven extension of the variational algorithm is proposed. Based on a few selected sensors, target tracking is performed distributively without any information about the observation model. Tracking under such conditions is possible if one exploits the information collected from extra inter-sensor RSSI measurements.The target tracking problem is formulated as a kernel matrix completion problem. A probabilistic kernel regression is then proposed that yields a Gaussian likelihood function. The likelihood is used to derive an efficient and accelerated version of the variational filter without resorting to Monte Carlo integration. The proposed data-driven algorithm is, by construction, robust to observation model deviations and adapted to non-stationary environments.

Index Terms— Bayesian filtering, sensor networks, machine learning.

1. INTRODUCTION

In a Bayesian framework, tracking a target in a sensor network consists in estimating the posterior distribution of an n_x-dimensional state vector $\boldsymbol{x}_t$ (target position) given data measured by sensors densely deployed in the region under surveillance. In this work, we consider distributed filtering where the online update of the posterior is locally performed based on some selected sensors whose data are considered quite relevant for an accurate tracking. Previous works, related to this problem, have been devoted to the implementation of Bayesian filtering methods in a wireless sensor network. Most of the proposed solutions are based on sequential Monte Carlo methods [1], [2], [5]. The popularity of this kind of methods stems from their ability to deal with the nonlinear aspect of the state dynamics and also from their flexibility to deal with any nonlinear likelihood model. However, a crude implementation of distributed particle filters needs the exchange of large sample-based distribution representations between selected leaders. Message approximations should therefore be considered in order to reduce the consumption of communication energy [1]. However, successive message approximations may cause the propagation of an inference error. Recently, a collaborative distributed variational filter has been proposed in [3]. Based on an online updating of a free-form approximation of the filtering distribution, the variational approach allows a natural adaptive compression of the non-Gaussian filtering distribution. The approximating message can then be communicated between leader nodes without loss. In addition to the constraints on exchanging information between leader nodes, another energy concern in wireless sensor network is the measurement modality. Equipping nodes with high performance detectors may compromise the potential of the sensors to be densely deployed with a low cost. For tracking and localization purposes, the received signal strength indicator (RSSI) is a commonly used technique for measuring the proximity between two nodes [4]. However, the RSSI is based on a parametric model with parameters to be tuned according to the environment where the measurements are made. The tracking and localization performances are very sensitive to the relevance of the parametric model and also to the fixed values of its parameters. In fact, the RSSI model is the used likelihood model in Bayesian filtering techniques. Deviations from the true model may cause severe degradations in tracking performances. A way to circumvent this problem is the use of binary sensors as proposed in [5] with a centralized particle filter implementation and then extended in [6] with a variational filter implementation. Binary sensors are based on thresholding the RSSI signal to decide whether the target is in the range of the sensor or not. This minimizes the dependency on the RSSI model but exploits only part of the available information, leading to a less accurate target tracking.

In this paper, we propose an efficient data-driven variational tracking method without the use of an RSSI model. The essence of the proposed technique is the exploitation of extra RSSI (or any other similarity) measurements exchanged between selected sensors. By considering the extra inter-sensor data as learning data, powerful tools of machine learning can be employed. In particular, the tracking problem is recast into a kernel matrix completion problem. A probabilistic formulation of the kernel matrix regression solution proposed in [7] allows for a construction of a linear likelihood model. The variational filter is then efficiently implemented without resorting to Monte Carlo integration as was the case for general nonlinear likelihood models. As the likelihood model is locally constructed, the proposed distributed filter is particularly adapted to nonstationary environments.

In Section 2, the analogy between the tracking without using an RSSI model and the kernel matrix completion problem is established. Section 3 is devoted to the implementation of the variational filter exploiting the linear likelihood model. Finally, some simulation results corroborating the efficiency of the proposed data-driven technique are presented and discussed in Section 4.

2. KERNEL REGRESSION FORMULATION

2.1. Target tracking as a matrix completion problem

At each time step t, we assume that a set of n sensors $\{\boldsymbol{s}_1^{(t)}, ..., \boldsymbol{s}_n^{(t)}\}$ are selected to be activated for tracking the unknown target position $\boldsymbol{x}_t$. We further assume that all pairwise RSSI signals (or any other similarity measurement) between the sensors and the target and between the selected sensors themselves are available and collected in one selected node in charge of updating of the filtering distribution. The extra inter-sensor RSSI measurements will play here the role of learning data to be exploited for circumventing the absence of any information about the RSSI model. In order to establish a connection with the kernel matrix completion problem, we consider the RSSI measurements as pairwise similarities. Following the kernel trick, commonly used in the machine learning community, the similarity measurements are considered as scalar products in a reproducing kernel Hilbert space (RKHS). In other words, the RSSI between a sensor $\boldsymbol{s}_i^{(t)}$ and another sensor $\boldsymbol{s}_j^{(t)}$ is considered as the Euclidean scalar product of their features $\phi(\boldsymbol{s}_i^{(t)})$ and $\phi(\boldsymbol{s}_j^{(t)})$ in the RKHS: $RSSI(\boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)}) = k(\boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)}) =< \phi(\boldsymbol{s}_i^{(t)}), \phi(\boldsymbol{s}_j^{(t)}) >$.

According to this formulation, the RSSI $(N \times N)$-matrix (with $N = n + 1$) corresponds to the fully available kernel matrix $\boldsymbol{K}$ whose elements can be defined as follows:

$$\begin{cases} (K)_{i,j} = RSSI(\boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)}) & 1 \leq i \neq j \leq n, \\ (K)_{i,n+1} = RSSI(\boldsymbol{s}_i^{(t)}, \boldsymbol{x}_t) & 1 \leq i \leq n, \\ (K)_{l,l} = c = const. & 1 \leq l \leq n+1. \end{cases}$$

As the target position is unknown, the $(N \times N)$-matrix $\boldsymbol{G}$ formed by the pairwise Euclidean scalar products of elements belonging to the set $\{\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)}, ..., \boldsymbol{s}_n^{(t)}, \boldsymbol{x}_t\}$ has missing entries corresponding to the scalar products between the sensors and the target. The objective of matrix completion is then the estimation of the missing entries of the matrix $\boldsymbol{G}$ exploiting a form of correlation with the complete kernel matrix $\boldsymbol{K}$. By splitting the matrix $\boldsymbol{G}$ into 4 blocks $\boldsymbol{G}_{tt}$, $\boldsymbol{G}_{tp}$, $\boldsymbol{G}_{pt}$ and $\boldsymbol{G}_{pp}$ corresponding respectively to sensor set versus itself, sensor set versus target, target versus sensor set and target versus itself, the completion problem can be illustrated by the following diagram:

$$\underbrace{\begin{array}{|c|c|}\hline \boldsymbol{K}_{tt} & \boldsymbol{K}_{tp} \\ \hline \boldsymbol{K}_{pt} & \boldsymbol{K}_{pp} \\ \hline \end{array}}_{\boldsymbol{K}} \longrightarrow \underbrace{\begin{array}{|c|c|}\hline \boldsymbol{G}_{tt} & \color{gray}{\boldsymbol{G}_{tp}} \\ \hline \color{gray}{\boldsymbol{G}_{pt}} & \color{gray}{\boldsymbol{G}_{pp}} \\ \hline \end{array}}_{\boldsymbol{G}} \tag{1}$$

where the objective is the prediction of the unknown blocks (in gray) $\boldsymbol{G}_{tp}$, $\boldsymbol{G}_{pt}$ and $\boldsymbol{G}_{pp}$ by learning a mapping between $\boldsymbol{K}_{tt}$ and $\boldsymbol{G}_{tt}$. As we have only one missing object, the block $\boldsymbol{G}_{tp}$ (resp. $\boldsymbol{K}_{tp}$) is a column, the block $\boldsymbol{G}_{pt}$ (resp. $\boldsymbol{K}_{pt}$) is a row and $\boldsymbol{G}_{pp}$ (resp. $\boldsymbol{K}_{pp}$) is a scalar. More precisely, the elements of $\boldsymbol{G}_{tt}$ are the scalar products between the sensor position vectors $\langle \boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)} \rangle$, for $i, j = 1, ..., n$, $\boldsymbol{G}_{tp}$ is a vector containing the scalar products between the sensors and the target:

$$\boldsymbol{G}_{tp} = [(\boldsymbol{s}_1^{(t)})^T \boldsymbol{x}_t, \ldots, (\boldsymbol{s}_i^{(t)})^T \boldsymbol{x}_t, \ldots, (\boldsymbol{s}_n^{(t)})^T \boldsymbol{x}_t]^T, \tag{2}$$

with $\boldsymbol{G}_{pt} = \boldsymbol{G}_{tp}^T$ and $\boldsymbol{G}_{pp} = \|\boldsymbol{x}_t\|^2$. Note that $\boldsymbol{G}_{tt}$ is a known matrix and that the unknown matrix $\boldsymbol{G}_{tp}$ is linear with respect to the target position $\boldsymbol{x}_t$. This property will be exploited in designing the likelihood function and in efficiently implementing the variational filter, both shown in the next section.

2.2. Probabilistic matrix regression

In order to solve the kernel matrix completion, a matrix regression method has been proposed in [7]. This method assumes that the elements K_{ij} are obtained by applying a kernel k on a set of of N explanatory random variables $\{\boldsymbol{e}_i \in \mathbb{R}^d\}_{i=1}^N$. Similarly, the elements G_{ij} are obtained by applying a kernel g on a set of N response random variables $\{\boldsymbol{r}_i \in \mathbb{R}^l\}_{i=1}^N$. Both data sets can be considered as two different representations of the same objects. Solving the matrix completion problem is essentially based on modifying the features of the explanatory variables so that their similarities match the similarities of the response variables. We follow here the same idea but rather than predicting the missing block $\boldsymbol{G}_{tp}$, we compute its probability distribution.

Let $\boldsymbol{u}(\boldsymbol{e})$ refer to the new feature of the explanatory variable $\boldsymbol{e}$. The new feature lies in the RHKS defined by the kernel k. In order to compute the Euclidean scalar product between two features $\boldsymbol{u}(\boldsymbol{e})$ and $\boldsymbol{u}(\boldsymbol{e}')$, it is sufficient to define their coordinates with respect to an orthonormal basis. A common approach in kernel methods, based on the representer theorem, is to define the coordinates as a combination of learning data kernels as follows:

$$u_l(\boldsymbol{e}) = \sum_{j=1}^{n} w_{j,l} k(\boldsymbol{e}_j, \boldsymbol{e}), \quad l = 1, \cdots, m,$$

where we have considered m coordinates. By defining the matrix $\boldsymbol{W} = (w_{j,l})_{j=1..n}^{l=1..m}$, the new feature $\boldsymbol{u}(\boldsymbol{e})$ may be written in a matrix form,

$$\boldsymbol{u}(\boldsymbol{e}) = \boldsymbol{W}^T [k(\boldsymbol{e}_j, \boldsymbol{e})]_{j=1..n}. \tag{3}$$

The regression problem consists in finding the coefficients $\boldsymbol{W}$ such that the scalar products of the new features fit the similarities of the response variables. The regression problem can be formulated as follows:

$$g(\boldsymbol{r}_i, \boldsymbol{r}_j) = \boldsymbol{u}(\boldsymbol{e}_i)^T \boldsymbol{u}(\boldsymbol{e}_j) + \epsilon_{i,j}, \tag{4}$$

where $\epsilon_{i,j}$ is a zero-mean normally distributed noise with variance σ_{ij}^2. Using the vector form (3), equations (4) can be put in a compact matrix form as according to

$$\begin{aligned} \boldsymbol{G}_{tt} &= \boldsymbol{K}_{tt} \boldsymbol{A} \boldsymbol{K}_{tt} + \boldsymbol{\Psi}_{tt} && (5) \\ \boldsymbol{G}_{tp} &= \boldsymbol{K}_{tt} \boldsymbol{A} \boldsymbol{K}_{tp} + \boldsymbol{\Psi}_{tp} && (6) \\ \boldsymbol{G}_{pp} &= \boldsymbol{K}_{pt} \boldsymbol{A} \boldsymbol{K}_{tp} + \boldsymbol{\Psi}_{pp} && (7) \end{aligned}$$

where $\boldsymbol{A}$ is the unknown matrix $\boldsymbol{W}\boldsymbol{W}^T$ and $\boldsymbol{\Psi} = (\epsilon_{i,j})_{i,j=1..N}$ is an $(N \times N)$-matrix of Gaussian random variables. For simplicity, we assume that the variables $\epsilon_{i,j}$ are independent and identically distributed (i.i.d) with $\sigma_{ij}^2 = \boldsymbol{\sigma}^2$.

According to the above statistical formulation of the matrix regression problem, it is straightforward to show that, given the matrices $\boldsymbol{G}_{tt}$, $\boldsymbol{K}_{tt}$ and $\boldsymbol{K}_{tp}$, the matrix $\boldsymbol{G}_{tp}$ is normally distributed with the following expressions for its mean and covariance:

$$\left\{ \begin{array}{lcl} \boldsymbol{\mu}_g &=& \boldsymbol{G}_{tt} \boldsymbol{K}_{tt}^{-1} \boldsymbol{K}_{tp} \\ \boldsymbol{\Sigma}_g &=& \sigma^2 (\boldsymbol{K}_{pt} \boldsymbol{K}_{tt}^{-2} \boldsymbol{K}_{tp} + 1) \boldsymbol{I}_n \end{array} \right. \tag{8}$$

where $\boldsymbol{I}_n$ is the $(n \times n)$ identity matrix.

The Gaussian distribution of the vector $\boldsymbol{G}_{tp}$ is the key point for the remainder of the paper. By denoting $\boldsymbol{S} = [\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)}, ..., \boldsymbol{s}_n^{(t)}]^T$ the $(n \times 2)$ matrix of sensor locations and taking into account that the kernel g is the Euclidean scalar product, the Gaussianity of the vector $\boldsymbol{G}_{tp}$ could be rewritten as

$$\boldsymbol{G}_{tp} = \boldsymbol{S} \boldsymbol{x}_t = \boldsymbol{G}_{tt} \boldsymbol{K}_{tt}^{-1} \boldsymbol{K}_{tp} + \boldsymbol{\gamma}_t \tag{9}$$

where $\boldsymbol{\gamma}_t$ is a zero-mean Gaussian noise with a diagonal covariance $\boldsymbol{\Sigma}_g$ defined in (8). Expression (9) can be considered as the resulting statistical model linking the measured data and the unknown target position $\boldsymbol{x}_t$ and plays thus the role of the likelihood function when tracking the target in a Bayesian framework. The quantity

$\boldsymbol{G}_{tt}\boldsymbol{K}_{tt}^{-1}\boldsymbol{K}_{tp}$ on the right hand side of (9) can be interpreted as a sufficient statistic obtained from the available data based on a kernel matrix regression formulation.

3. ONLINE VARIATIONAL FILTERING

3.1. State-space model

In the remainder of the paper, the likelihood function is based on the linear model (9). Concerning the transition dynamics $p_x(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$, we adopt a mean-scale mixture model. According to this model, introduced in [8], the hidden state $\boldsymbol{x}_t \in \mathbb{R}^{n_x}$ has a Gaussian distribution with a random mean $\boldsymbol{\mu}_t$ and a random precision matrix $\boldsymbol{\lambda}_t$. The mean vector dynamics are described by a Gaussian random walk reflecting the time correlation of the system trajectory, and the precision matrix is distributed according to a Wishart distribution, that is,

$$\left\{ \begin{array}{lcl} \boldsymbol{\mu}_t & \sim & \mathcal{N}(\boldsymbol{\mu}_t \mid \boldsymbol{\mu}_{t-1}, \bar{\boldsymbol{\lambda}}) \\ \\ \boldsymbol{\lambda}_t & \sim & \mathcal{W}_{\bar{n}}(\boldsymbol{\lambda}_t \mid \bar{\boldsymbol{S}}) \\ \\ \boldsymbol{x}_t & \sim & \mathcal{N}(\boldsymbol{x}_t \mid \boldsymbol{\mu}_t, \boldsymbol{\lambda}_t) \end{array} \right. \tag{10}$$

where the fixed hyperparameters $\bar{\boldsymbol{\lambda}}$, $\bar{n}$ and $\bar{\boldsymbol{S}}$ are respectively the random walk precision matrix, the degrees of freedom and the precision of the Wishart distribution. Note that assuming a random mean and a random covariance for the state $\boldsymbol{x}_t$ leads to a prior probability distribution covering a wide range of tail behaviors allowing discrete jumps in the target trajectory.

3.2. Updating free form approximate distributions

According to the model (10), the augmented hidden state is now $\boldsymbol{\alpha}_t = (\boldsymbol{x}_t, \boldsymbol{\mu}_t, \boldsymbol{\lambda}_t)$. At each time step t, the observed data y_t consist of the matrices $\{\boldsymbol{K}_{tt}, \boldsymbol{K}_{tp}, \boldsymbol{G}_{tt}\}$. Instead of approximating the filtering distribution $p(\boldsymbol{\alpha}_t \mid \boldsymbol{y}_{1:t})$ (with the standard notation $\boldsymbol{y}_{1:t} = (y_1, ..., y_t)$) by a point-mass distribution (particle filtering), the variational approach [3] consists in approximating the filtering distribution by a more tractable posterior distribution $q(\boldsymbol{\alpha}_t)$. The variational approach determines the approximate filtering distribution by minimizing the the Kullback-Leibler divergence between the true filtering distribution and the approximate distribution,

$$D_{\text{KL}}(q||p) = \int q(\boldsymbol{\alpha}_t) \log \frac{q(\boldsymbol{\alpha}_t)}{p(\boldsymbol{\alpha}_t \mid \boldsymbol{y}_{1:t})} d\boldsymbol{\alpha}_t, \tag{11}$$

to obtain the optimal approximate distribution. In order to ensure that the best model is automatically chosen, we assume a free form (non parametric) approximate distribution. Choosing a separable distribution $q(\boldsymbol{\alpha}_t) = q(\boldsymbol{x}_t)q(\boldsymbol{\mu}_t)q(\boldsymbol{\lambda}_t)$ and minimizing the Kullback-Leibler divergence (11) with variational calculus, one obtains the following approximate distributions:

$$\left\{ \begin{array}{lcl} q(\boldsymbol{x}_t) & \propto & \exp \langle \log p(\boldsymbol{y}_{1:t}, \boldsymbol{\alpha}_t) \rangle_{q(\boldsymbol{\mu}_t)q(\boldsymbol{\lambda}_t)} \\ q(\boldsymbol{\mu}_t) & \propto & \exp \langle \log p(\boldsymbol{y}_{1:t}, \boldsymbol{\alpha}_t) \rangle_{q(\boldsymbol{x}_t)q(\boldsymbol{\lambda}_t)} \\ q(\boldsymbol{\lambda}_t) & \propto & \exp \langle \log p(\boldsymbol{y}_{1:t}, \boldsymbol{\alpha}_t) \rangle_{q(\boldsymbol{x}_t)q(\boldsymbol{\mu}_t)} \end{array} \right. \tag{12}$$

The update of the approximate distribution $q(\boldsymbol{\alpha}_t)$ can be sequentially implemented given only the approximate distribution $q(\boldsymbol{\mu}_{t-1})$. Using the separable approximate distribution at time $t-1$, the filtering distribution is written as

$$p(\boldsymbol{\alpha}_t|\boldsymbol{y}_{1:t}) \propto p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t, \boldsymbol{\lambda}_t|\boldsymbol{\mu}_t) \int p(\boldsymbol{\mu}_t|\boldsymbol{\mu}_{t-1})q(\boldsymbol{\mu}_{t-1})d\boldsymbol{\mu}_{t-1} \tag{13}$$

where only integration with respect to $\boldsymbol{\mu}_{t-1}$ remains due to the separable form of the approximate distribution $q(\boldsymbol{\alpha}_{t-1})$. The temporal dependence on the past is hence limited to only one of the component approximate distributions. Communication between two successive sets of activated nodes is then limited to sending $q(\boldsymbol{\mu}_{t-1})$ which is the sufficient statistic for updating the filtering distribution. As it will be shown in the following, it turns out that $q(\boldsymbol{\mu}_{t-1})$ is a Gaussian distribution and thus it can be communicated by sending only a mean and a covariance. The variational algorithm is then implemented in a collaborative sensor network without lossy compression.

After substituting the filtering distribution (13) into (12) and taking into account the prior mean-scale mixture transition model (10), one obtains the updated separable distribution $q(\boldsymbol{\alpha}_t)$ in the following form:

$$\begin{array}{lcl} q(\boldsymbol{x}_t) & \propto & p(\boldsymbol{y}_t \mid \boldsymbol{x}_t)\mathcal{N}(\boldsymbol{x}_t \mid \langle \boldsymbol{\mu}_t \rangle, \langle \boldsymbol{\lambda}_t \rangle) \propto \mathcal{N}(\boldsymbol{x}_t \mid \boldsymbol{x}_t^*, \boldsymbol{\Gamma}_t^*) \\ q(\boldsymbol{\mu}_t) & \propto & \mathcal{N}(\boldsymbol{\mu}_t \mid \boldsymbol{\mu}_t^*, \boldsymbol{\lambda}_t^*) \\ q(\boldsymbol{\lambda}_t) & \propto & \mathcal{W}_{n^*}(\boldsymbol{\lambda}_t \mid \boldsymbol{S}_t^*) \end{array}$$

where the parameters are iteratively updated according to the following scheme:

$$\begin{array}{l} \boldsymbol{x}_t^* = \boldsymbol{\Gamma}_t^{*-1}(\boldsymbol{S}^T\boldsymbol{\Sigma}_g^{-1}\boldsymbol{G}_{tt}\boldsymbol{K}_{tt}^{-1}\boldsymbol{K}_{tp} + \langle \boldsymbol{\lambda}_t \rangle \langle \boldsymbol{\mu}_t \rangle) \\ \boldsymbol{\Gamma}_t^* = \boldsymbol{S}^T\boldsymbol{\Sigma}_g^{-1}\boldsymbol{S} + \langle \boldsymbol{\lambda}_t \rangle \\ \boldsymbol{\mu}_t^* = \boldsymbol{\lambda}_t^{*-1}(\langle \boldsymbol{\lambda}_t \rangle \langle \boldsymbol{x}_t \rangle + \boldsymbol{\lambda}_t^p \boldsymbol{\mu}_t^p) \\ \boldsymbol{\lambda}_t^* = \langle \boldsymbol{\lambda}_t \rangle + \boldsymbol{\lambda}_t^p \\ n^* = \bar{n} + 1 \\ \boldsymbol{S}_t^* = (\langle \boldsymbol{x}_t \boldsymbol{x}_t^T \rangle - \langle \boldsymbol{x}_t \rangle \langle \boldsymbol{\mu}_t \rangle^T - \langle \boldsymbol{\mu}_t \rangle \langle \boldsymbol{x}_t \rangle^T + \langle \boldsymbol{\mu}_t \boldsymbol{\mu}_t^T \rangle + \bar{\boldsymbol{S}}^{-1})^{-1} \\ \boldsymbol{\mu}_t^p = \boldsymbol{\mu}_{t-1}^* \\ \boldsymbol{\lambda}_t^p = (\boldsymbol{\lambda}_{t-1}^{*-1} + \bar{\boldsymbol{\lambda}}^{-1})^{-1} \end{array}$$

In the above expressions, all the variable expectations have closed forms, i.e.,

$$\left\{ \begin{array}{l} \langle \boldsymbol{x}_t \rangle = \boldsymbol{x}_t^*, \;\; \langle \boldsymbol{x}_t \boldsymbol{x}_t^T \rangle = \boldsymbol{\Gamma}_t^{*-1} + \boldsymbol{x}_t^* \boldsymbol{x}_t^{*T}, \\ \langle \boldsymbol{\mu}_t \rangle = \boldsymbol{\mu}_t^*, \;\; \langle \boldsymbol{\mu}_t \boldsymbol{\mu}_t^T \rangle = \boldsymbol{\lambda}_t^{*-1} + \boldsymbol{\mu}_t^* \boldsymbol{\mu}_t^{*T}, \langle \boldsymbol{\lambda}_t \rangle = n^* \boldsymbol{S}_t^* \end{array} \right.$$

4. NUMERICAL RESULTS

In this section, we illustrate the effectiveness of the proposed data-driven variational filter (DDVF) for target tracking in a wireless sensor network. We also compare the DDVF algorithm with a classical variational filter algorithm (VF) where the observation model is known. We considered the tracking of a target moving according to a trajectory composed of two sinusoids in a two-dimensional field (figure 1), for a duration of $T = 200$ time slots. An abrupt change was simulated at time $t_a = 100$ in order to test the ability of the algorithm to track the target in a difficult discontinuous situation. A set of 500 nodes were randomly deployed in a $120m \times 120m$ area. Each node has a sensing range set to 20m. At each time step t, the known matrices $\boldsymbol{K}_{tt}$ and $\boldsymbol{K}_{tp}$ (input of the algorithm) were simulated according to the following stationary model:

$$\left\{ \begin{array}{l} K_{tt}(i,j) = \exp\{-\|\boldsymbol{s}_i^{(t)} - \boldsymbol{s}_j^{(t)}\|/2\sigma^2\} + \epsilon_{ij}^{(t)}, 1 \leq i,j \leq n \\ K_{tp}(j) = \exp\{-\|\boldsymbol{s}_j^{(t)} - \boldsymbol{x}_t^*\|/2\sigma^2\} + \epsilon_j^{(t)}, 1 \leq j \leq n \end{array} \right. \tag{14}$$

where $\boldsymbol{s}_m^{(t)} = (s_1^m, s_2^m)$ and $\boldsymbol{x}_t^* = (x_1, x_2)$ were the activated node and the true target positions at time t, σ was set to 10 and $\epsilon_{ij}^{(t)}$ was the corrupting noise due to modeling error, instrumental noise and background additive interfering signals. The noise variance depended on the inter-sensor distance. The elements of the matrix $\boldsymbol{G}_{tt}$ were obtained as the scalar products between the known sensor position vectors $\langle \boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)} \rangle$, for $i, j = 1, ..., n$.

The number of selected sensors was fixed to 10. The selection protocol was based on the Gaussian predictive distribution. More details about the selection protocol are reported in [6].

The hyperparameters of the transition dynamical state model were set to the following values:

$$\bar{\boldsymbol{\lambda}} = 10^{-2}\boldsymbol{I},\ \bar{n} = 1,\ \bar{\boldsymbol{S}} = 10^{2}\boldsymbol{I},$$

where the hyperparameters values allow a general non informative prior. It is worth noting that in target tracking applications, an informative prior, involving the target velocity and acceleration, is usually assumed. Here, the transition prior has a more general form which can be used in other sensor network applications.

The proposed DDVF algorithm was applied to simulated data, without using the observation model (14). Figure 1 depicts the estimated trajectory superimposed with the true simulated trajectory. The target position was estimated by its expectation ($\hat{\boldsymbol{x}}_t = \langle \boldsymbol{x}_t \rangle$) according to the approximate filtering distribution $q(\boldsymbol{x}_t)$. Note the accuracy of the tracking with a mean square error $mse = \sum_t \|\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t\|^2/T = 0.29$. In the same figure, the 10 selected sensors are plotted in circles, for four chosen instants: $t = 40, t = 80, t = 160$ and $t = 190$. The algorithm is able to select the relevant nodes based on a compact (Gaussian) approximation of the predictive distribution.

For comparison purposes, the classical variational filter (VF) [3] was applied to track the target in the same configuration as above, with 10 selected nodes at each time slot. In one set of experiments, the classical VF algorithm was applied assuming the exact knowledge of the observation model (14). Figure 2 depicts the tracking performances of the classical VF filtering with a mean square error $mse = 1.3$. Note that the classical VF algorithm was less accurate than the proposed data-driven DDVF algorithm. It should be kept in mind that, although the classical VF filter is based on the true observation model, the DDVF algorithm exploits more data obtained by the extra inter-sensor RSSI signals. This is a very interesting result. The difference between the DDVF algorithm and the classical VF should be even more contrasted when the observation model is not known exactly or when the environment is non-stationary. Indeed, the DDVF is not sensitive to the observation model and its parameters contrary to the classical VF algorithm.

5. CONCLUSION

The paper studied a data-driven target tracking algorithm, based on extra inter-sensor similarity measures. The key point of the proposed algorithm was to formulate the tracking problem as a kernel matrix completion problem. The kernel matrix completion problem was solved by using training data coming from intra-sensor measurements. The Gaussian likelihood function, obtained by a probabilistic formulation of the regression problem, was used to implement a fast variational filter. Preliminary results showed the efficiency of the proposed algorithm. Perspectives include the study of more extensive tests on simulated and real data.

6. REFERENCES

[1] A. Ihler, J. Fisher III, and A. Willsky, "Particle filtering under communications constraints", in *Proc. Statistical Signal Processing (SSP) 2005*, 2005.

[2] X. Sheng, Y. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking ion wireless sensor network", in *Information Processing in Sensor Networks, 2005. IPSN 2005*, April 2005, pp. 181–188.

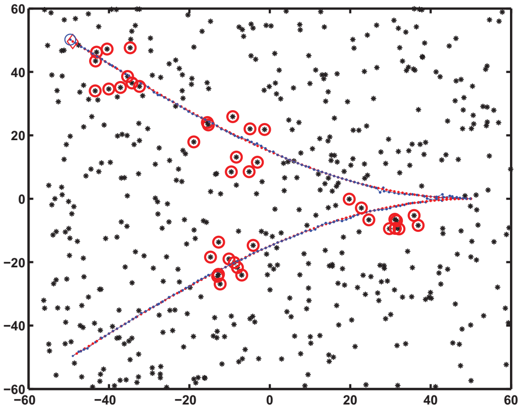

Fig. 1. Data-driven variational filtering in a collaborative sensor network: Estimated positions in blue and true positions in red. The 10 selected sensors are plotted in circles, for four chosen instants, $t = 40, t = 80, t = 160$ and $t = 190$.

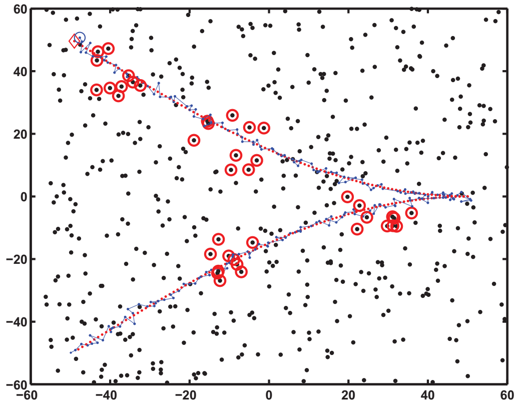

Fig. 2. Classical variational filtering in a collaborative sensor network: Estimated positions in blue and true positions in red. The 10 selected sensors are plotted in circles, for four chosen instants: $t = 40, t = 80, t = 160$ and $t = 190$

[3] H. Snoussi and C. Richard, "Ensemble learning online filtering in wireless sensor networks", in *IEEE ICCS International Conference on Communications Systems*, 2006.

[4] N. Patwari and A. O. Hero, "Manifold learning algorithms for localization in wireless sensor networks", in *IEEE ICASSP*, May 2004, vol. 3, pp. 857–860.

[5] P. Djurić, M. Vemula, and M. Bugallo, "Tracking with particle filtering in tertiary wireless sensor networks", in *ICASSP*, Philadelphia, PA, USA, March 18-23 2005.

[6] J. Teng, H. Snoussi, and C. Richard, "Prediction-based proactive cluster target tracking protocol for binary sensor networks", in *The 7th IEEE International Symposium on Signal Processing and Information Technology*, 2007.

[7] Y. Yamanishi and J.-P. Vert, "Kernel matrix regression", Tech. Rep. http://arxiv.org/abs/q-bio/0702054v1, 2007.

[8] J. Vermaak, N. Lawrence, and P. Perez, "Variational inference for visual tracking", in *Conf. Computer Vision and Pattern Recog, CVPR'03*, June 2003.