Graph Laplacian Distributed Particle Filtering

Michael Rabbat and Mark Coates
Dept. Electrical and Computer Engineering
McGill University, Montréal, QC, Canada
michael.rabbat@mcgill.ca, mark.coates@mcgill.ca

Stephane Blouin
DRDC Atlantic Research Centre
Dartmouth, NS, Canada
stephane.blouin@drdc-rddc.gc.ca.

***Abstract*—We address the problem of designing a distributed particle filter for tracking one or more targets using a sensor network. We propose a novel approach for reducing the communication overhead involved in the data fusion step. The approach uses graph-based signal processing to construct a transform of the joint log likelihood values of the particles. This transform is adaptive to particle locations and in many cases leads to a parsimonious representation, so that the the joint likelihood values of all particles can be accurately approximated using only a few transform coefficients. The proposed particle filter uses gossip to perform distributed, approximate computation of the transform coefficients. Numerical experiments highlight the potential of the proposed approach to provide accurate tracks with reduced communication overhead.**

I. Introduction

Particle filters have proven to be highly effective for tracking scenarios with non-linear and/or non-Gaussian dynamic and/or measurement models. In distributed tracking systems, where multiple nodes obtain measurements and collaborate to achieve improved tracking accuracy, developing efficient and accurate distributed particle filters is an important challenge. An important design goal is to reduce the communication overhead by transmitting fewer and/or smaller messages.

This paper focuses on distributed particle filtering algorithms, wherein a network of nodes collaborate to track one or more targets. Within the class of distributed particle filtering algorithms, we concentrate on gossip-based methods, where each node maintains its own local particle filter and nodes communicate to share information obtained through their local measurements. There is a trade-off between the communication overhead and the tracking accuracy.

Related Work: Several distributed particle filters have been proposed over the past decade; [1] provides a valuable survey of the methods. The approaches involve distributed, approximate computation of the particle weights. This is achieved by gossiping on the particles with the highest joint likelihood [2], [3], or forming approximations of the joint likelihood function or posterior and gossiping on sufficient statistics [4]–[7]. Although these filters can perform very effectively in some settings, the former family of filters can often incur a high communication overhead, and the latter can struggle if the assumptions underpinning the approximations are not justified.

Contribution: We propose a novel approach for reducing the communication overhead involved in the data fusion step of a distributed particle filter. The reduction is obtained by approximating the joint log likelihood values. The approach uses graph-based signal processing to construct a transform of the joint log likelihood values for each particle which is adapted to the particle locations. In many cases, the transform is such that the joint likelihood values at all particle can be accurately approximated using only a few transform coefficients. Numerical experiments highlight the potential of the proposed approach to provide accurate tracks with reduced communication overhead. In comparison to other state-of-the-art distributed particle filtering approaches, we find that the proposed approach can significantly reduce the communication overhead while suffering only a modest decrease in tracking accuracy. Among the spectrum of distributed tracking algorithms, this makes it possible to achieve performance in a different regime than was previously possible.

II. Problem Formulation and Background

A. Target Dynamics and Measurement Model

In this paper we focus on the problem of tracking the state $\boldsymbol{x}_t \in \mathbb{R}^{n_x}$ of a target over times $t \geq 0$. The state is assumed to evolve according to the discrete-time dynamics

$$\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}) + \boldsymbol{\xi}_t \ , \tag{1}$$

where $f(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ is a possibly non-linear mapping, and $\boldsymbol{\xi}_t \in \mathbb{R}^{n_x}$ is time-varying process noise.

Our aim is to track $\boldsymbol{x}_t$ from noisy observations made at a collection of n_v sensor nodes. We denote the set of nodes by $\mathcal{V} = \{1, \ldots, n_v\}$. Let $\boldsymbol{z}_{t,v} \in \mathbb{R}^{n_{z,v}}$ denote the measurement taken at node $v \in \mathcal{V}$ at time t. We assume that the measurements follow the model

$$\boldsymbol{z}_{t,v} = h_{t,v}(\boldsymbol{x}_t) + \boldsymbol{\zeta}_{t,v} \ , \tag{2}$$

where $h_{t,v}(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_{z,v}}$ is a node-dependent, possibly non-linear mapping, and $\boldsymbol{\zeta}_{t,v}$ is the measurement noise at sensor v at time t. We allow the measurement function $h_{t,v}(\cdot)$ to vary with time.

Let $\boldsymbol{z}_t \stackrel{\text{def}}{=} \{\boldsymbol{z}_{t,v} : v \in \mathcal{V}\}$ denote the collection of all measurements made at all nodes at time t, and $\boldsymbol{z}_{1:t} \stackrel{\text{def}}{=} \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_t\}$ denote the collection of all measurements from times 1 up to t. We assume that the measurements taken at different nodes at time t are conditionally independent given the state $\boldsymbol{x}_t$. Under this assumption, the joint likelihood of the measurements $\boldsymbol{z}_t$ factorizes as

$$p(\boldsymbol{z}_t|\boldsymbol{x}_t) = \prod_{v \in \mathcal{V}} p(\boldsymbol{z}_{t,v}|\boldsymbol{x}_t) \ . \tag{3}$$

978-0-9928-6265-7/16/$31.00 ©2016 IEEE

We assume that the model $f(\cdot)$ for the target dynamics and the distribution of the process noise $\boldsymbol{\xi}_t$ are known to all nodes. In addition, we assume that node v knows the local measurement functions $\{h_{t,v}(\cdot)\}_{t\geq 0}$, as well as the distribution of the local measurement noise $\boldsymbol{\zeta}_{t,v}$.

Although each node could individually track $\boldsymbol{x}_t$ using its own observations, in general more accurate estimates of $\boldsymbol{x}_t$ are obtained by fusing measurements from multiple nodes. To this end, nodes may share information with each other by communicating over a network. We assume that if two nodes can communicate in the network, then bidirectional communication is achievable, and that these undirected links form a connected topology.

We focus on algorithms where each node maintains its own estimate of the state $\boldsymbol{x}_t$, and nodes communicate using gossip protocols (simple message-passing methods) in order to keep their state estimates synchronized. The use of gossip protocols is desirable because they do not require central coordination and they are robust to typical challenges arising in wireless distributed systems, such as time-varying network topologies and dropped packets [8], [9].

The particle filtering algorithms makes use of two types of gossip. The first reaches consensus, asymptotically, on the *average* of a set of scalar values at different nodes [10]; the second reaches consensus on the *maximum* of the scalar values [11]. The fusion in the particle filtering algorithm involves averaging, so we need to employ average gossip. The convergence of average gossip is asymptotic, but we need all nodes to have exactly the same estimates, so we subsequently apply the maximum gossip procedure, which can be shown to converge almost surely to the state where all nodes know the maximum value after a finite number of iterations [11].

B. Particle Filters

Particle filters approximate the posterior density $p(\boldsymbol{x}_t|\boldsymbol{z}_{1:t})$ using a set of N weighted particles,

$$\mathcal{X}_t \stackrel{\text{def}}{=} \{\widehat{\boldsymbol{x}}_t^{(i)}, w_t^{(i)}\}_{i=1}^N \ ,$$

where $\widehat{\boldsymbol{x}}_t^{(i)} \in \mathbb{R}^{n_x}$ and $w_t^{(i)} \in \mathbb{R}$; specifically, the approximation is given by

$$\widehat{p}(\boldsymbol{x}_t|\boldsymbol{z}_{1:t}) = \sum_{i=1}^N w_t^{(i)} \delta(\boldsymbol{x}_t - \widehat{\boldsymbol{x}}_t^{(i)}) \ , \tag{4}$$

where $\delta(\cdot)$ is the Dirac delta function.

Given an initial target detection at location $\widehat{\boldsymbol{x}}^0$ and time $t = 0$, the particle locations $\widehat{\boldsymbol{x}}_0^{(i)}$ are initialized by sampling from a density $p_0(\cdot|\widehat{\boldsymbol{x}}^0)$ and the weights are initialized to $w_0^{(i)} = 1/N$ for all i.

Each particle is updated from one time step to the next by first sampling a new location $\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}$ from a proposal density $q(\boldsymbol{x}|\widehat{\boldsymbol{x}}_{t-1}^{(i)}, \boldsymbol{z}_{1:t})$. Then the weight associated with $\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}$ is

$$w_{t|t}^{(i)} \propto w_{t-1}^{(i)} \frac{p(\boldsymbol{z}_t|\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}) p(\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}|\widehat{\boldsymbol{x}}_{t-1}^{(i)})}{q(\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}|\widehat{\boldsymbol{x}}_{t-1}^{(i)}, \boldsymbol{z}_{1:t})} \ , \tag{5}$$

where the proportionality constant is such that $\sum_{i=1}^N w_{t|t}^{(i)} = 1$. Finally, a new set of particles $\{\widehat{\boldsymbol{x}}_t^{(i)}, w_t^{(i)}\}_{i=1}^N$ is obtained by sampling N particles with replacement from the set $\{\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}\}_{i=1}^N$, where $\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}$ is sampled with probability $w_{t|t}^{(i)}$, and setting the corresponding weights to $w_t^{(i)} = 1/N$. This last step is referred to as resampling in the literature. For more background on particle filters see, e.g., [12].

In the *bootstrap particle filter*, $q(\cdot|\widehat{\boldsymbol{x}}_{t-1}^{(i)}, \boldsymbol{z}_{1:t})$ is taken to be $p(\cdot|\widehat{\boldsymbol{x}}_{t-1}^{(i)})$, the density of $\boldsymbol{x}_t$ given $\boldsymbol{x}_{t-1}$ defined by the dynamic model (1) with $\boldsymbol{x}_{t-1} = \widehat{\boldsymbol{x}}_{t-1}^{(i)}$. For this choice, the propagation of $\widehat{\boldsymbol{x}}_t^{(i)}$ only depends on $\widehat{\boldsymbol{x}}_{t-1}^{(i)}$; in particular, it does not depend on the measurement $\boldsymbol{z}_t$.

C. Gossip-Based Distributed Particle Filtering

We focus on a distributed framework where each node maintains and updates its own copy of particles $\mathcal{X}_{t,v}$, and gossip protocols are used to homogenize the particles across nodes over time. We assume that nodes have a shared source of randomness which can be used to sample from the distributions $p_0(\cdot|\widehat{\boldsymbol{x}}^0)$ and $q(\cdot|\widehat{\boldsymbol{x}}_{t-1}^{(i)}, \boldsymbol{z}_{1:t})$.[1]

Suppose that the weighted particle sets at all nodes are exactly the same at time $t-1$, and also suppose that the network implements the bootstrap particle filter. Then the proposal distribution $q(\cdot)$ only depends on the current particle locations. Based on the assumption of shared randomness, the newly sampled particle locations $\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}$ will be identical at all nodes. For the bootstrap particle filter, the weight update (5) thus simplifies to

$$w_{t|t-1}^{(i)} \propto w_{t-1}^{(i)} p(\boldsymbol{z}_t|\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}) \ ,$$

and so new weight values are a function of the measurements taken at all nodes, $\boldsymbol{z}_t = \{\boldsymbol{z}_{t,v} : v \in \mathcal{V}\}$.

Under the conditional independence assumption (3), the joint likelihood can be expressed as

$$p(\boldsymbol{z}_t|\boldsymbol{x}_t) = \exp\left\{\sum_{v\in\mathcal{V}} \log p(\boldsymbol{z}_{t,v}|\boldsymbol{x}_t)\right\} \ .$$

Let $\gamma_{t,v}^{(i)} \stackrel{\text{def}}{=} \log p(\boldsymbol{z}_{t,v}|\widehat{\boldsymbol{x}}_{t|t-1}^{(i)})$ denote the log-likelihood of the measurement at node v and time t. Then the weight updates can be expressed as

$$w_{t|t-1}^{(i)} = \frac{w_{t-1}^{(i)} \exp\{\bar{\gamma}_t^{(i)}\}}{\sum_{j=1}^N w_{t-1}^{(j)} \exp\{\bar{\gamma}_t^{(j)}\}} \ ,$$

where $\bar{\gamma}_t^{(i)} \stackrel{\text{def}}{=} \sum_{v\in\mathcal{V}} \gamma_{t,v}^{(i)}$. Because $\bar{\gamma}_t^{(i)}$ is a linear function of values $\gamma_{t,v}^{(i)}$ at each node, an approximation of the values $\{\bar{\gamma}_t^{(i)}\}_{i=1}^N$ can be computed at all nodes in a distributed manner using gossip protocols.

[1] In practice, sampling is implemented using a pseudo-random number generator, and shared randomness can be achieved by having the nodes initially agree on the seed. If the nodes are initially not synchronized, each node can draw a random number. Then they can use the max-gossip protocol to determine the maximum value and use it as the common seed to the pseudo-random number generator.

III. Graph Laplacian Approximation for Distributed Particle Filtering

This section proposes an approach to updating particle weights in a distributed manner. The technique constructs a transform that is adapted to the particle locations. Since particles near each other generally have more similar joint likelihood values than particles that are far apart, we build a graph over the particles by placing edges between pairs of particles that are close to each other. Then the Laplacian matrix of this graph is used to construct a transform such that the joint log-likelihood values at all particles can be succinctly approximated using only a few coefficients in the transform domain.

Particles with similar locations in the state space typically have similar weights; i.e., $\|\widehat{\boldsymbol{x}}_t^{(i)} - \widehat{\boldsymbol{x}}_t^{(j)}\|$ being small typically implies that $|w_t^{(i)} - w_t^{(j)}|$ is also small. This suggests that some form of transform coding may be effective. We propose to use a transform adapted to the propagated locations $\{\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}\}_{i=1}^N$ in order to obtain a reduced-order approximation to the weights $\{w_{t|t}^{(i)}\}_{i=1}^N$. Specifically, we propose to use the eigenvectors of a graph constructed from the propagated locations $\{\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}\}_{i=1}^N$ as a transform basis.

Let $\boldsymbol{A}$ denote the $N \times N$ adjacency matrix of a graph with one vertex for each particle in the set $\{\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}\}_{i=1}^N$. Fix a positive integer $\kappa \ll N$. We place an edge between particles i and j (equivalently, we set $A_{i,j} = A_{j,i} = 1$) if particle i is one of the κ nearest neighbours of j, or vice versa, where the distance between two particles is $\|\widehat{\boldsymbol{x}}_t^{(i)} - \widehat{\boldsymbol{x}}_t^{(j)}\|_2$; this is sometimes referred to as the symmetrized κ-nearest neighbour graph. Let $\boldsymbol{D}$ denote a diagonal $N \times N$ matrix with diagonal entries $D_{i,i} = \sum_{j=1}^N A_{i,j}$ containing the number of neighbours of node i. Note that $\boldsymbol{A}$ is symmetric and that not all particles necessarily have the same number of neighbours since the κ-nearest neighbour relationship is not necessarily symmetric.

Let $\bar{\boldsymbol{\gamma}}_t \in \mathbb{R}^N$ denote a vector obtained by stacking the particle log-likelihoods $\{\bar{\gamma}_t^{(i)}\}_{i=1}^N$ using the same indexing used when forming the graph corresponding to $\boldsymbol{A}$. We view $\bar{\boldsymbol{\gamma}}_t$ as a signal supported on the graph and seek a transform that concentrates the energy of $\bar{\boldsymbol{\gamma}}_t$ into a small number of coefficients.

The Laplacian matrix $\boldsymbol{L}$ of the graph with adjacency matrix $\boldsymbol{A}$ is $\boldsymbol{L} \stackrel{\text{def}}{=} \boldsymbol{D} - \boldsymbol{A}$. Because $\boldsymbol{L}$ is a symmetric real matrix it has an eigendecomposition of the form $\boldsymbol{L} = \boldsymbol{F}\boldsymbol{\Lambda}\boldsymbol{F}^T$, where $\boldsymbol{F}$ is an $N \times N$ orthonormal matrix of eigenvectors and $\boldsymbol{\Lambda}$ is a diagonal matrix of eigenvalues. Let

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$$

denote the eigenvalues (the diagonal elements of $\boldsymbol{\Lambda}$) sorted in ascending order, and let $\boldsymbol{f}_j$ denote the j-th column of $\boldsymbol{F}$. The eigenvalues and eigenvectors of the graph Laplacian have a number of interesting properties for processing signals supported on the graph [13]–[15]. In particular, the columns of $\boldsymbol{F}$ can be interpreted as a Fourier basis for signals supported on the graph. Eigenvectors corresponding to larger eigenvalues have the interpretation of being higher frequency basis vectors. Typical log-likelihood functions $\log p(\boldsymbol{z}_t|\widehat{\boldsymbol{x}}_{t|t-1}^{(i)})$ vary smoothly as a function of $\widehat{\boldsymbol{x}}_{t|t-1}^{(i)}$, and so one may hope that most of the energy of $\bar{\boldsymbol{\gamma}}$ concentrates in a few low frequency components.

Given $\{\widehat{\boldsymbol{x}}_{t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^N$, execute at all nodes $v \in \mathcal{V}$ in parallel:

1: **for** $i = 1, \ldots, N$ **do**
2: Sample $\widehat{\boldsymbol{x}}_t^{(i)} \sim q(\cdot|\widehat{\boldsymbol{x}}_{t-1}^{(i)}, \boldsymbol{z}_{1:t})$
3: **end for**
4: Form the κ nearest neighbour graph for particles $\{\widehat{\boldsymbol{x}}_t^{(i)}\}_{i=1}^N$.
5: Calculate the Laplacian $\boldsymbol{L}$ and perform the eigendecomposition $\boldsymbol{L} = \boldsymbol{F}\boldsymbol{\Lambda}\boldsymbol{F}^T$.
6: Calculate Laplacian transform coefficients $\boldsymbol{\alpha}_{t,v} \stackrel{\text{def}}{=} \boldsymbol{F}_m^T\bar{\boldsymbol{\gamma}}_t$ for $\boldsymbol{F}_m$ defined in (6).
7: $\{\widehat{\alpha}_t^{(j)}\}_{j=1}^m = \textsc{Gossip}_v(\{n_v\alpha_{t,v}^{(j)}\}_{j=1}^m)$.
8: Set $\widehat{\boldsymbol{\gamma}}_t = \boldsymbol{F}_m\widehat{\boldsymbol{\alpha}}_t$.
9: **for** $i = 1, \ldots, N$ **do**
10: $w_{t|t}^{(i)} = \frac{w_{t-1}^{(i)} \exp\{\widehat{\gamma}_t^{(i)}\}}{\sum_{j=1}^N w_{t-1}^{(j)} \exp\{\widehat{\gamma}_t^{(j)}\}}$
11: **end for**
12: **for** $i = 1, \ldots, N$ **do**
13: Sample $\widehat{\boldsymbol{x}}_t^{(i)}$ from $\{\widehat{\boldsymbol{x}}_{t|t-1}^{(j)}\}_{j=1}^N$ with replacement, with $\Pr(\widehat{\boldsymbol{x}}_t^{(i)} = \widehat{\boldsymbol{x}}_{t|t-1}^{(j)}) = w_{t|t}^{(j)}$. Set $w_t^{(i)} = 1/N$.
14: **end for**

Fig. 1. Pseudo-code for one step of the graph Laplacian approximation distributed particle filter. Line 7 involves an application of average and max-consensus.

Given a signal $\bar{\boldsymbol{\gamma}}_t \in \mathbb{R}^N$ on the graph, we refer to $\boldsymbol{\alpha}_t \stackrel{\text{def}}{=} \boldsymbol{F}^T\bar{\boldsymbol{\gamma}}_t$ as the vector of Laplacian transform coefficients. Intuitively, smooth signals on the graph—those for which the values at neighbouring nodes are similar—should have most of their energy concentrated in the coefficients corresponding to lower frequency basis vectors. For a positive integer $m \leq N$, an m-term approximation to $\bar{\boldsymbol{\gamma}}_t$ can be obtained by first transforming into the Laplacian eigenvalue domain, thresholding the magnitudes corresponding to the largest eigenvalues to zero, and taking the inverse transform. Equivalently, we take

$$\widehat{\boldsymbol{\gamma}}_t = \sum_{j=1}^m (\boldsymbol{f}_j^T\bar{\boldsymbol{\gamma}})\boldsymbol{f}_j \ ,$$

where $\{\boldsymbol{f}_j\}_{j=1}^m$ are the graph Laplacian eigenvectors corresponding to the m smallest Laplacian eigenvalues. Since $\boldsymbol{F}$ is an orthonormal matrix, when $m = N$ there is no approximation error and $\widehat{\boldsymbol{\gamma}}_t$ is identical to $\bar{\boldsymbol{\gamma}}_t$.

In the setting of distributed particle filtering, let us again assume that the value m is fixed in advance and known to all nodes. All nodes also know the particle locations $\{\widehat{\boldsymbol{x}}_t^{(i)}\}_{i=1}^N$, and so each node can locally compute the adjacency matrix $\boldsymbol{A}$, the corresponding Laplacian matrix $\boldsymbol{L}$, and a matrix

$$\boldsymbol{F}_m \stackrel{\text{def}}{=} [\boldsymbol{f}_1, \ldots, \boldsymbol{f}_m] \tag{6}$$

composed of the m eigenvectors corresponding to the m smallest eigenvalues of $\boldsymbol{L}$. Node v then locally computes the m-dimensional vector of coefficients

$$\boldsymbol{\alpha}_{t,v} = \boldsymbol{F}_m^T\boldsymbol{\gamma}_{t,v} \ ,$$

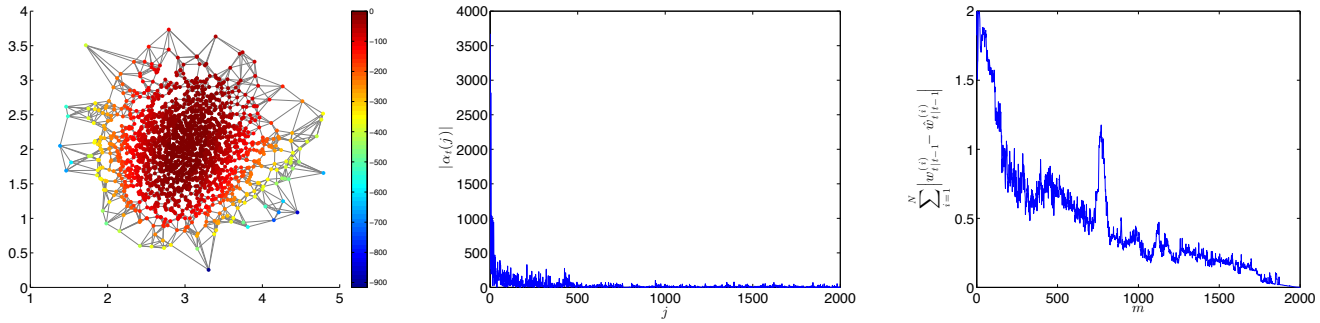

Fig. 2. (a) An example particle cloud shown with a κ-nearest neighbour graph ($\kappa = 7$). (b) Magnitude of the coefficients $|\alpha_t^{(j)}|$ of the joint log-likelihood expressed in terms of the graph Laplacian eigenvectors. (c) Typical particle weight approximation error for this example using only the coefficients corresponding to the m first Laplacian eigenvectors for different values of m.

where $\boldsymbol{\gamma}_{t,v}$ is the vector of particle log-likelihood values computed using only the measurements at node v. The nodes then perform K iterations of distributed averaging using the gossip algorithm, followed by the application of maximum gossip. Let $\widehat{\boldsymbol{\alpha}}_t$ denote the resulting m-dimensional vector of coefficients. To obtain the log-likelihood estimates, each node sets $\widehat{\boldsymbol{\gamma}}_t = \boldsymbol{F}_m \widehat{\boldsymbol{\alpha}}_t$ and proceeds with the weight update. Figure 1 provides pseudocode that summarizes the proposed approach.

Figure 2 illustrates the graph Laplacian approach to reducing communications when computing the weights associated with each particle. We take the particle cloud for the example shown in Fig. 2(a) and construct a nearest neighbour graph with $\kappa = 7$. Fig. 2(b) depicts the magnitude of the elements of the vector $\boldsymbol{\alpha}_t$; i.e., $|\alpha_t^{(j)}| = |\boldsymbol{f}_j^T \bar{\boldsymbol{\gamma}}_t|$, where $\boldsymbol{f}_j$ is the j'th column of $\boldsymbol{F}$. Observe that the bulk of the energy is concentrated in the first few coefficients. Finally, Fig. 2(c) shows the absolute error in particle weights when only the first m graph Laplacian coefficients are used to obtain an approximation $\hat{\boldsymbol{\gamma}}_t$.

IV. Numerical Evaluation

We evaluate the performance of the proposed approach and compare it with state-of-the-art methods using a simulated data set. The scenario is a bearings-only tracking problem with two targets and $n_v = 25$ sensor nodes arranged in a perturbed grid over a 10km by 10km area. The two targets make a counter-clockwise maneuver over a period of $T = 50$ time steps. At each time step, the second target senses the first and moves in its direction with a constant velocity. Because the dynamics of the two targets are coupled, we take the state $\boldsymbol{x}_t$ to be an 8-dimensional vector, with the first four coordinates representing the position and velocity of the first target, and the second four coordinates representing the position and velocity of the second target.

At each time step, a random subset of the nodes gather bearings measurements. On average five measurements are made across the network per target per time step. Each measurement is of the bearings from the target to the sensor making the measurement, and the measurements are corrupted by additive zero-mean Gaussian noise with a standard deviation of 5 degrees. To make a fair comparison among the algorithms, and to avoid issues related to data association, we assume that each measurement is associated with the corresponding target by an oracle and these same associations are used by all of the algorithms. The sensors communicate over a grid topology, with each node able to communicate with up to four neighbours.

Below we evaluate the performance of the proposed algorithms. To quantify accuracy, we report the *time-averaged root mean square position error* (ARMSE), $\sqrt{\frac{1}{T}\sum_{t=1}^{T} \|\boldsymbol{x}_t - \widehat{\boldsymbol{x}}_t\|^2}$. All results reported are drawn from 100 Monte Carlo trials. To study the performance of the proposed algorithm, detailed in the pseudocode of Figure 1, we fix the number of particles to $N = 2000$ and examine the algorithm performance as we vary m and the number of gossip iterations. At each step, a graph is formed over the particle cloud using $\kappa = 20$ neighbours. Figure 3 shows the total ARMSE as a function of the average number of scalars transmitted per node per step for different values of m. Each curve corresponds to fixing a value of m and varying the number of gossip iterations from 500 to 2000 in increments of 500. The figure illustrates that a filter using only 100 or 500 coefficients can achieve accuracy approaching that of a centralized filter if there are sufficient gossip iterations (at least 1000 in this case).

Next, we compare the performance of top-m selective gossip and the graph Laplacian approximation approach to other distributed particle filtering algorithms from the literature. Specifically, we compare with the distributed bootstrap filter running gossip iterations directly on the particle weights, with the Gaussian approximation distributed particle filter [5], with the set-membership constrained (SMC) distributed particle filter [2], and with the top-m selective gossip particle filter [3]. All filters use 2000 particles. The communication overhead of the distributed bootstrap, Gaussian approximation, and SMC particle filters can be adjusted by varying the number of gossip iterations used by each filter.

The tradeoff between total mean ARMSE (for both targets)

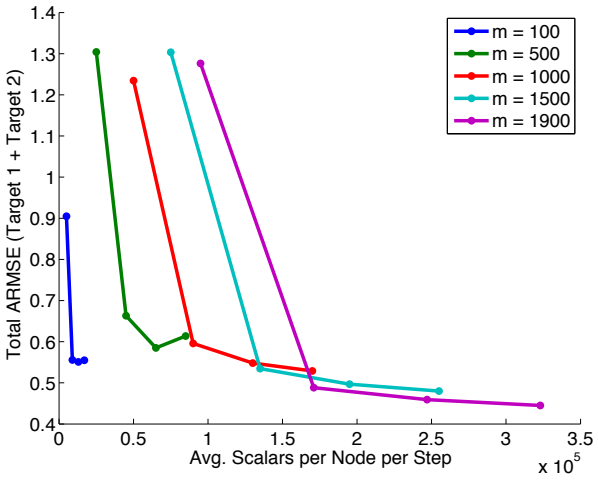

Fig. 3. The sum of the mean ARMSE for both targets as a function of the average number of scalar values transmitted per node per time step of the simulated scenario, running the distributed particle filter with graph Laplacian approximation. The points on each curve are obtained by fixing the value of m and varying the number of gossip iterations per tracking time step from 500 to 2500 in increments of 500.

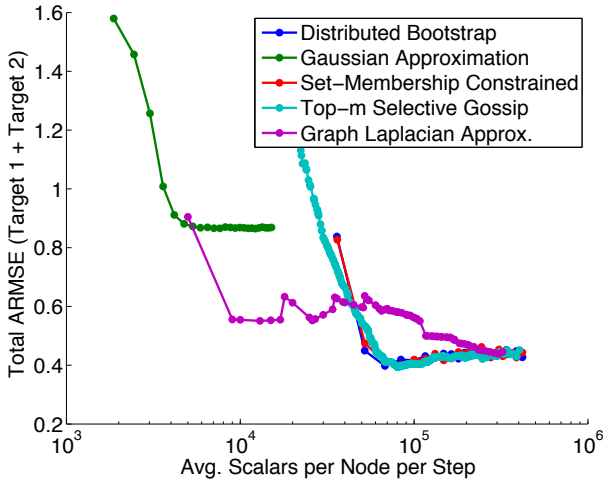

Fig. 4. The sum of the mean ARMSE (km) for both targets in the simulated scenario as a function of the average number of scalar values transmitted per node per time step.

and the communication overhead, as measured in terms of the average number of scalars transmitted per sensor per step of the distributed filter, is shown in Figure 4. The Gaussian approximation filter has much less communication overhead than the other approaches, but the total ARMSE is much higher, probably because the Gaussian approximation does not adequately capture the shape of the posterior distribution for the tracking scenario considered. The distributed bootstrap, set-membership constrained, and top-m selective gossip filters all exhibit a similar tradeoff. The graph Laplacian approximation offers interesting performance for communication overheads in the range 5×10^3 to 5×10^4 scalars transmitted per node per filtering step. In this range, the communication overhead is reduced by nearly one order of magnitude, as compared to the set-membership constrained filter or top-m selective gossip filters, and the total ARMSE only increases by a little over 0.1 km.

V. Conclusion

We have presented a novel distributed particle filter for tracking one or more targets using a sensor network. A graph signal processing approach is used to approximate the log-likelihoood values of the particles. Numerical experiments highlight the potential of the proposed approach to provide accurate tracks with reduced communication overhead. In future work, we will explore how to reduce the computational overhead of constructing a nearest neighbour graph and extracting the eigenvectors from the Laplacian.

References

[1] O. Hlinka, F. Hlawatsch, and P. Djurić, "Distributed particle filtering in agent networks: A survey, classification, and comparison," *IEEE Signal Processing Mag.*, vol. 30, no. 1, pp. 61–81, Jan 2013.

[2] S. Farahmand, S. Roumeliotis, and G. Giannakis, "Set-membership constrained particle filter: Distributed adaptation for sensor networks," *IEEE Trans. Signal Processing*, vol. 59, no. 9, pp. 4122 –4138, Sept. 2011.

[3] D. Üstebay, M. J. Coates, and M. G. Rabbat, "Distributed auxiliary particle filters using selective gossip," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Proc. (ICASSP)*, Prague, Czech Republic, May 2011, pp. 3296–3299.

[4] D. Gu, J. Sun, Z. Hu, and H. Li, "Consensus based distributed particle filter in sensor networks," in *Proc. IEEE ICIA*, Zhangjiajie, China, Jun 2008, pp. 302–307.

[5] B. Oreshkin and M. Coates, "Asynchronous distributed particle filter via decentralized evaluation of Gaussian products," in *Proc. ISIF Int. Conf. Information Fusion*, Edinburgh,UK, July 2010.

[6] O. Hlinka, O. Slučiak, F. Hlawatsch, P. Djurić, and M. Rupp, "Likelihood consensus and its application to distributed particle filtering," *IEEE Trans. on Signal Processing*, vol. 60, no. 8, pp. 4334–4349, Aug 2012.

[7] A. Mohammadi and A. Asif, "A constraint sufficient statistics based distributed particle filter for bearing only tracking," in *IEEE Intl. Conf. on Communications*, June 2012, pp. 3670–3675.

[8] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[9] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1847 – 1864, November 2010.

[10] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Information Theory*, vol. 52, no. 6, pp. 2508–2530, Jun 2006.

[11] F. Iutzeler, P. Ciblat, and J. Jakubowicz, "Analysis of max-consensus algorithms in wireless channels," *IEEE Trans. on Signal Processing*, vol. 60, no. 11, pp. 6103–6107, Nov. 2012.

[12] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," in *Handbook of Nonlinear Filtering*. Oxford, UK: Oxford University Press, 2009, ch. 12, pp. 656–704.

[13] F. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[14] X. Zhu and M. Rabbat, "Approximating signals supported on graphs," in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Kyoto, Japan, March 2012.

[15] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.