Distributed Estimation for Sensor Networks with Arbitrary Topologies

Shukui Zhang and Anastasios I. Mourikis[1]

Abstract— **This paper addresses the problem of distributed recursive estimation of the state of a dynamical system, using measurements obtained by a network of locally-communicating sensors. Prior work on this problem has mostly focused on the development of methods suitable for very large networks, where multiple iterations of communication between neighboring sensors are used to obtain good asymptotic estimation accuracy. By contrast, we here focus on the case of medium-sized networks (consisting of up to a few tens of nodes), where limited communication is the primary constraint. We propose a distributed estimation method that uses knowledge of the network topology to explicitly model the accuracy of different nodes' estimates, as well as the correlations between them. As a result, the proposed method is able to optimally utilize the measurements and state estimates communicated between neighboring nodes. This method does not require iterations, imposes no assumptions on the structure of the communication graph, and can provide an accurate representation of the accuracy of the produced estimates. We provide Monte-Carlo simulation results that demonstrate that the proposed method outperforms existing algorithms, when all methods use the same amount of inter-node communication.**

I. Introduction

In this paper, we address the problem of estimating the state of a dynamical system using measurements from a network of sensors. Our work is motivated by a large number of application domains, where the ability to estimate quantities of interest using multiple, spatially distributed sensors is critical. Examples include traffic monitoring and management [1], [2], environmental monitoring [3], [4], health care systems [5], [6], and target tracking [7], [8], among others.

It is well known that a centralized system is vulnerable to a single point-of-failure, and can cause communication bottlenecks (e.g., if all nodes have to "flood" their measurements onto the network so that they reach the fusion center) [9]–[11]. Therefore, research efforts have concentrated on the problem of *distributed* estimation in sensor networks, where no central fusion center exists, and each node can only communicate with a small number of neighbors.

To date, the vast majority of research on distributed estimation has focused on developing efficient estimation methods suitable for very large networks. This is a challenging problem, due to the nodes' constrained computational capabilities and the availability of only local communication. Since the amount of information that can be exchanged among nodes is limited, it is difficult to model the interdependency (e.g., correlation) of different nodes' estimates, and this typically leads to suboptimal estimation results. To obtain estimation algorithms that are realizable in very large networks, proposed approaches (e.g., consensus-based methods) typically do not explicitly model these interdependencies, and instead rely on a large number of data exchanges between nodes to attain good *asymptotic* performance.

In our work, our objective is to perform distributed estimation in a different setting. We are interested in networks of medium size (e.g., consisting of a few tens of sensors), where *communication* is the primary challenge. Consensus-based methods are not well suited for this setting, as their main characteristic is a low computational cost at the expense of increased communication requirements (due to the multiple consensus iterations). Moreover, we seek to develop a method that can provide performance guarantees not asymptotically, but for a single round of data transmission at each estimation time step.

To achieve these goals, we propose a recursive distributed-estimation method that *explicitly models* the accuracy and the correlation between all the nodes' state estimates. At each time step, each node transmits to its neighbors an optimally-weighted sum of its current state estimate and its latest measurement. In turn, each node receives these quantities from its neighbors, and fuses them *optimally*, in the MMSE sense. The nodes employ prior knowledge of the network topology in order to obtain estimation accuracy higher than that attained by methods that do not use this information (e.g., consensus-type methods). It is important to point out that the proposed approach does *not* assume any specific structure for the communication graph (e.g., a hierarchical structure, a fully connected graph, or a balanced one), and is thus applicable with any network topology. In what follows, we present the details of our method after a discussion of related work.

II. Related Work

The existing literature on the subject of distributed estimation is vast, and any attempt at presenting a comprehensive overview will inevitably be incomplete. In what follows, we instead discuss a small number of representative methods, categorized into broad classes.

A. Consensus Fusion

Several methods for distributed estimation are based on the concept of consensus. The key characteristic of these methods is that each node shares its estimates with its neighbors, and corrects its own estimate using neighbors' information. These corrections are performed iteratively, and after a large (theoretically infinite) number of iterations, all

[1]The authors are with the Dept. of Electrical and Computer Engineering, University of California, Riverside, CA, 92521, USA. Emails: szhan023@ucr.edu, mourikis@ece.ucr.edu

nodes have the same estimate for the state, which can often be proven to be identical to the estimate attainable by a centralized estimator. An overview of several approaches in this class can be found in [12]–[14].

Specifically for the problem of recursive estimation of the state of a dynamical system, several methods are described in [13], [15], [16]. Among these, the Kalman Consensus Information Filter (KCIF) [16] offers a favorable tradeoff between accuracy and computational cost. It is based on the standard Kalman filter, but includes an additional "consensus term" in the state-update equation, so that all nodes' estimates asymptotically converge to a common value.

However, a limitation of all the above methods, including the KCIF, is that they are not able to provide a reliable measure of the accuracy of the computed estimates. As a result, the performance of these methods may be negatively impacted by the presence of "naive nodes" in the network [17]. "Naive node" here refers to a node that has little information about the state of the dynamical system from its own measurements and the data it has received from its neighbors. To address this issue, the Information-Weighted Consensus Filter (ICF) was proposed in [17]. The ICF performs average consensus on the information vector and information matrix of the state estimates. In this process, each estimate is weighted according to its corresponding information matrix, leading to improved performance over the KCIF, and quicker convergence.

The key advantage of all the aforementioned consensus-based methods is that they do not require detailed knowledge of the topology of the network (although the number of nodes must be known in certain methods). Moreover, given enough iterations, the estimates of all nodes can converge to the optimal estimate. However, convergence to the optimal requires a balanced graph [18], a requirement that may not be satisfied in all cases. Additionally, the performance guarantees (e.g., convergence to the optimal) hold when the number of iterations is large. When the estimate is computed using only a small number of iterations no guarantees typically exist for its accuracy, and since the reported error-covariance matrix does not provide a reliable description of the actual errors, it is difficult to evaluate the quality of the result.

B. Measurement/Information fusion

A different class of methods is termed measurement fusion or information fusion [19]. The key idea in measurement/information fusion is that each node simply transmits its raw measurements to its neighbors, and each node fuses the raw measurements it receives from its neighborhood, to obtain a local estimate of the dynamical system's state [20].

Since the data transmitted between nodes are the raw measurements only (perhaps in information-weighted form), measurement/information fusion can obtain an optimal estimate (equivalent to the centralized one) only when the network is fully connected. In the general case of a non-fully-connected topology, these approaches only utilize measurements from a small neighborhood of the network, and therefore obtain suboptimal state estimates.

C. State-Vector Fusion

In contrast to the measurement-sharing approaches described in the preceding section, in state-vector fusion methods *state estimates* are shared between nodes, and fused in a non-iterative fashion. The case of a fully-connected graph for state-vector fusion is explored in [21], [22], where solutions to the maximum-likelihood estimation problem are proposed. An LMMSE estimator for the non-Gaussian case is also studied in [23].

The approach we describe in this paper is effectively a combination of measurement fusion and state-vector fusion. Specifically, each node transmits to its neighbors an *optimally-weighted* sum of its current state estimate and its latest measurement. The weighting is optimal, in the sense that it allows for computing the optimal MMSE state estimate on each node, given its neighbors' estimates and measurements. In contrast to prior work, we do not impose any assumptions on the topology of the communication graph.

III. Distributed Estimator

A. Problem Formulation

To present our approach, we consider the case of a dynamical system whose state, $\mathbf{x}$, evolves in time as[1]:

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{w}_k \tag{1}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the system state vector at time-step k, $\mathbf{\Phi}_k \in \mathbb{R}^{n\times n}$ is the state-transition matrix, and $\mathbf{w}_k \in \mathbb{R}^n$ is the process noise, modelled as zero-mean, white, Gaussian, with covariance matrix $\mathbf{Q}_k$. This system could represent, for example, the position of a target moving in an area of interest. A network of N sensors obtains measurements of the system state, described by the equations:

$$\mathbf{z}_{i_k} = \mathbf{H}_{i_k}\mathbf{x}_k + \mathbf{n}_{i_k}, \quad i = 1, \ldots, N \tag{2}$$

where $\mathbf{H}_{i_k} \in \mathbb{R}^{m\times n}$ is the measurement matrix for the observation of sensor i at time step k, and $\mathbf{n}_{i_k} \in \mathbb{R}^m$ is the corresponding measurement noise, modelled as zero-mean, white, Gaussian, with covariance matrix $\mathbf{R}_{i_k}$. The prior estimate of the system's initial state, $\mathbf{x}_0$, is described by a Gaussian pdf:

$$\mathbf{x}_0 \sim \mathcal{N}(\hat{\mathbf{x}}_{0|0}, \mathbf{P}^{xx}_{0|0}) \tag{3}$$

and we assume that the process noise $\mathbf{w}_k$, measurement noise vectors $\mathbf{n}_{i_k}$ for $i = 1, \ldots, N$, and the prior $\mathbf{x}_0$ are mutually independent. Note that, while in this section we present the proposed estimation method using linear dynamics and measurement models, the extension to nonlinear models is immediate by employing linearization, as in the extended-Kalman-filter paradigm.

[1] *Notation*: $\mathbf{I}_p$ denotes the $p \times p$ identity matrix, while $\mathbf{1}_{p\times q}$ the $p \times q$ matrix of ones. $\hat{\mathbf{x}}_{k|j}$ is the estimate of $\mathbf{x}_k$ given measurements up to and including time step j. The error in this estimate is defined as $\tilde{\mathbf{x}}_{k|j} = \mathbf{x}_k - \hat{\mathbf{x}}_{k|j}$.

To facilitate the presentation, it will be useful to define an "augmented" state vector $\mathbf{y}_k$, which consists of N copies of the original system's state vector:

$$\mathbf{y}_k \triangleq \begin{bmatrix} \mathbf{x}_k \\ \vdots \\ \mathbf{x}_k \end{bmatrix} \tag{4}$$

The dynamics of $\mathbf{y}_k$ can be written as:

$$\mathbf{y}_{k+1} = \mathbf{\Phi}_{\mathbf{y}_k}\mathbf{y}_k + \mathbf{w}_{\mathbf{y}_k} \tag{5}$$

where the state-transition matrix for the augmented system is given by $\mathbf{\Phi}_{\mathbf{y}_k} = \text{kron}(\mathbf{I}_N, \mathbf{\Phi}_k)$, with kron denoting the Kronecker matrix product, and the process noise vector is given by $\mathbf{w}_{\mathbf{y}_k} = \text{kron}(\mathbf{1}_{N\times 1}, \mathbf{w}_k) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\mathbf{y}_k})$, with $\mathbf{Q}_{\mathbf{y}_k} = \text{kron}(\mathbf{1}_{N\times N}, \mathbf{Q}_k)$. The prior estimate for the initial state $\mathbf{y}_0$ is described by

$$\mathbf{y}_0 \sim \mathcal{N}(\text{kron}(\mathbf{1}_{N\times 1}, \hat{\mathbf{x}}_{0|0}), \mathbf{P}_{0|0}) \tag{6}$$

where $\mathbf{P}_{0|0} = \text{kron}(\mathbf{1}_{N\times N}, \mathbf{P}^{xx}_{0|0})$.

We stress that the system models described by (1) and (5) describe the same physical system. If a centralized Kalman-filter estimator were used to estimate the states $\mathbf{x}_k$ and $\mathbf{y}_k$ using the measurements in (2), then at any time the estimates will satisfy $\hat{\mathbf{y}}_{k|k} = \text{kron}(\mathbf{1}_{N\times 1}, \hat{\mathbf{x}}_{k|k})$, and in this sense the two system formulations are equivalent. However, the use of the "augmented" state, $\mathbf{y}_k$, will facilitate the representation of the state estimates of all N nodes during distributed estimation, where they are generally different from each other (see (7)). Using this representation, we are now ready to present our estimator formulation.

B. Distributed estimator formulation

The N sensors of the network estimate the state of the system by using their own measurements, as well as the state estimates and most recent measurements of their neighbors. To present our formulation, let us initially assume that each node transmits its state estimate and current measurement to all its neighbors at each time instant (we later show that, in fact, only a weighted sum of these will suffice). In general, each sensor will only be able to communicate with a small subset of the N sensors, and this induces a *communication graph*, whose nodes represent the sensors of the network, while its directed edges represent the information flow between them. Specifically, an edge from node i to node j is present in the graph if node i sends data to node j.

We here define by N_i the one-hop neighborhood of node i, i.e., the set of nodes from which node i receives information, and $J_i = N_i \cup \{i\}$. Furthermore, for each node, we define the "structure matrix" $\mathbf{\Pi}_i \in \mathbb{R}^{|J_i|n\times Nn}$, as $\mathbf{\Pi}_i = \text{kron}(\boldsymbol{\pi}_i, \mathbf{I}_n)$, with $\boldsymbol{\pi}_i \in \mathbb{R}^{|J_i|\times N}$ being the "selector matrix", which consists of the rows of the identity corresponding to the indices in J_i. For example, in Figure 1, $J_1 = \{1, 2, 3\}$, $J_3 = \{3, 4\}$, and the matrices $\mathbf{\Pi}_1$ and $\mathbf{\Pi}_3$ are given by:

$$\mathbf{\Pi}_1 = \begin{bmatrix} \mathbf{I}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_n & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_n & \mathbf{0} \end{bmatrix}, \quad \mathbf{\Pi}_3 = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_n \end{bmatrix}$$

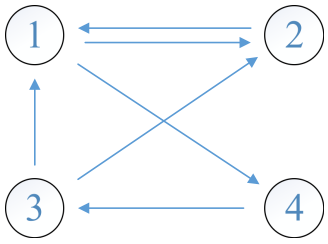

Fig. 1. An example network with four nodes.

The proposed estimator is a recursive one, similar in nature to a Kalman filter. At time step $k-1$, after processing all information available to it up to and including this time step, node i has an estimate, $\hat{\mathbf{x}}_{i_{k-1|k-1}}$, of the state of the dynamical system. Therefore the estimates of all nodes in the network form an estimate for $\mathbf{y}_{k-1}$, written as:

$$\hat{\mathbf{y}}_{k-1|k-1} = \begin{bmatrix} \hat{\mathbf{x}}_{1_{k-1|k-1}} \\ \vdots \\ \hat{\mathbf{x}}_{N_{k-1|k-1}} \end{bmatrix} \tag{7}$$

In addition, each node in the network computes the *joint* covariance matrix of the errors in the estimates of all the N nodes (i.e., the covariance matrix of the error $\tilde{\mathbf{y}}_{k-1|k-1}$), given by $\mathbf{P}_{k-1|k-1}$.

The "propagation" step of the estimator is implemented locally in each node. Specifically, each node computes an estimate of the state $\mathbf{x}_k$ by "propagating" its estimate from time step $k-1$:

$$\hat{\mathbf{x}}_{i_{k|k-1}} = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{i_{k-1|k-1}}, \quad i = 1, \ldots, N \tag{8}$$

while the joint covariance matrix of the estimates is propagated as:

$$\mathbf{P}_{k|k-1} = \mathbf{\Phi}_{\mathbf{y}_{k-1}}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}^T_{\mathbf{y}_{k-1}} + \mathbf{Q}_{\mathbf{y}_{k-1}} \tag{9}$$

Note that for the propagation step, no communication is required.

During the "update" at time step k, each node records a measurement of the system state, described by (2), and receives the state estimates and measurements from its neighbors. These are all fused with the state estimate maintained locally at the node. We now derive the equations for this update. To this end, we start by observing that node i has access to two qualitatively different sources of information: the measurements recorded by itself and its immediate neighbors:

$$\mathbf{z}_{j_k} = \mathbf{H}_{j_k}\mathbf{x}_k + \mathbf{n}_{j_k} \qquad j \in J_i \tag{10}$$

as well as the state estimates of itself and its immediate neighbors, which can be represented as:

$$\begin{aligned} \mathbf{v}_{i_k} &= \mathbf{\Pi}_i\hat{\mathbf{y}}_{k|k-1} && (11)\\ &= \mathbf{\Pi}_i\mathbf{y}_k + \mathbf{\Pi}_i\tilde{\mathbf{y}}_{k|k-1} && (12)\\ &= \mathbf{L}_i\mathbf{x}_k + \mathbf{\Pi}_i\tilde{\mathbf{y}}_{k|k-1} && (13) \end{aligned}$$

where $\mathbf{L}_i = \text{kron}(\mathbf{1}_{|J_i|\times 1}, \mathbf{I}_n)$.

Clearly, the errors in $\mathbf{v}_{i_k}$ and the measurements in (10) are mutually independent. Therefore the optimal (MMSE) estimate at node i for the state $\mathbf{x}_k$, can be computed as:

$$\hat{\mathbf{x}}_{i_{k|k}} = \mathbf{\Lambda}_i^{-1}\left(\sum_{j\in J_i}\mathbf{H}_{j_k}^T\mathbf{R}_{j_k}^{-1}\mathbf{z}_{j_k} + \mathbf{L}_i^T\mathbf{S}_{ii}^{-1}\mathbf{v}_{i_k}\right) \tag{14}$$

where we have defined

$$\mathbf{\Lambda}_i \triangleq \sum_{j\in J_i}\mathbf{H}_{j_k}^T\mathbf{R}_{j_k}^{-1}\mathbf{H}_{j_k} + \mathbf{L}_i^T\mathbf{S}_{ii}^{-1}\mathbf{L}_i \tag{15}$$

and

$$\mathbf{S}_{ij} = \mathbf{\Pi}_i\mathbf{P}_{k|k-1}{\mathbf{\Pi}_j}^T$$

We now note that if we partition the $|J_i|n \times |J_i|n$ matrix $\mathbf{S}_{ii}^{-1}$ into $|J_i|$ block columns as follows:

$$\mathbf{S}_{ii}^{-1} = \begin{bmatrix}\mathbf{A}_{ij_1} & \mathbf{A}_{ij_2} & \cdots & \mathbf{A}_{ij_{|J_i|}}\end{bmatrix} \tag{16}$$

then we can write $\mathbf{L}_i^T\mathbf{S}_{ii}^{-1}\mathbf{v}_{i_k} = \sum_{j\in J_i}\mathbf{L}_i^T\mathbf{A}_{ij}\hat{\mathbf{x}}_{j_{k|k-1}}$, and thus (14) can be written as:

$$\hat{\mathbf{x}}_{i_{k|k}} = \mathbf{\Lambda}_i^{-1}\sum_{j\in J_i}\mathbf{u}_{ij} \tag{17}$$

with

$$\mathbf{u}_{ij} = \mathbf{H}_{j_k}^T\mathbf{R}_{j_k}^{-1}\mathbf{z}_{j_k} + \mathbf{L}_i^T\mathbf{A}_{ij}\hat{\mathbf{x}}_{j_{k|k-1}} \tag{18}$$

The above equation shows that each node $j \in N_i$ can compute the weighted sum $\mathbf{u}_{ij}$ locally and transmit it to node i, instead of separately transmitting its state estimate and measurement. In this way, communication is reduced, while node i can still compute the optimal MMSE estimate.

In addition to computing the state estimate via (17), each node also updates the covariance matrix for the state estimate $\hat{\mathbf{y}}_{k|k}$. This can be performed in a block-by-block fashion. Specifically, the (i,j)-th $n \times n$ block of this matrix, with $i \neq j$, is updated as:

$$\mathbf{P}_{ij_{k|k}} = \mathbf{\Lambda}_i^{-1}\left(\mathbf{L}_i^T\mathbf{S}_{ii}^{-1}\mathbf{S}_{ij}\mathbf{S}_{jj}^{-1}\mathbf{L}_j + \sum_{\ell\in J_i\cap J_j}\mathbf{H}_{\ell_k}^T\mathbf{R}_{\ell_k}^{-1}\mathbf{H}_{\ell_k}\right)\mathbf{\Lambda}_j^{-1} \tag{19}$$

while for the i-th diagonal $n \times n$ block we obtain:

$$\mathbf{P}_{ii_{k|k}} = \mathbf{\Lambda}_i^{-1} \tag{20}$$

For clarity, the steps of the estimator are summarized in Algorithm 1.

Algorithm 1 Estimator steps for node i at time-step k

Input: $\hat{\mathbf{x}}_{i_{k-1|k-1}}, \mathbf{P}_{k-1|k-1}$

I. Propagation

1: Propagate state estimate using (8) and covariance matrix using (9)

II. Communication

2: Compute $\mathbf{u}_{li}$ via (18), for all indices l corresponding to the out-edge neighbors of node i, and transmit these to them.

3: Receive $\mathbf{u}_{ij}$ from all neighgbors $j \in N_i$

III. Update

4: Update the state estiumate using (17), and the covariance matrix using (19)-(20).

Output: $\hat{\mathbf{x}}_{i_{k|k}}$, $\mathbf{P}_{k|k}$

C. Discussion

The proposed approach involves no iteration (in contrast to consensus-type methods), and is applicable with arbitrary graph topologies, e.g., unbalanced graphs. Since the state estimates are included in the weighted sum that is communicated between nodes, the information of the measurements collected by *all* nodes is gradually propagated through the entire network. This is in contrast to measurement-fusion methods, which only exploit measurements from a node's neighbors. We also point out that the covariance matrix maintained by the nodes is an accurate description of estimation error, which is not the case in consensus-based methods. On the other hand, the proposed approach achieves these advantages by exploiting a *known* graph topology, which is not a requirement in consensus-based methods (as long as certain requirements, such as connectedness and balance, are met).

It is important to examine the computational and communication cost of the proposed method, as compared to the alternatives. To simplify this presentation, let us consider the case of a balanced graph with degree d. Moreover, we assume that the measurement matrices and noise covariance matrices for all nodes are known in advance, thus removing the need for communicating them. Computing the number of floating point operations (flops) that take place in each of the nodes of the network per update step [24], we find that this is given by approximately $\left((2N^2-N)(d+1)^3 + \frac{19}{6}N^2 + \frac{N}{6}\right)n^3$. We thus see that the computational cost per node scales quadratically with the number of nodes in the network. This may make the method unsuitable for very large networks, but as shown in the results of the next section, this is the penalty we pay for improved estimation accuracy. By contrast, consensus-based methods, such as the ICF or KCIF, have a per-node computational cost approximately equal to $\frac{13}{3}n^3 + Kdn^2$, where K is the number of iterations ($K = 1$ in the KCIF). Note that this cost is independent of the number of nodes in the network.

To quantify the communication cost, we count the total number of elements (scalars) that must be communicated per node. The per-node communication cost for the proposed method at each time step is dn, while for the KCIF it is $d(n+m)$ [16], and for the ICF it is Kdn. We thus see that, in order to attain the same communication cost as the proposed method, the ICF would have to perform only one iteration. As shown in the results of the next section, however, in this case the proposed method achieves higher performance than the ICF – as well as the KCIF. This makes our proposed approach better suited in environments where communication is severely limited (e.g., in underwater applications).

IV. Simulation results

In this section, we present Monte-Carlo simulation results that demonstrate the performance of the proposed method against alternative approaches, in terms of estimation accuracy. Moreover, we present results from the application of the method to a nonlinear estimation problem.

A. Reported standard deviation – Single trial

Before presenting the results of the Monte-Carlo simulations, we begin by presenting the results of a single simulated estimation experiment, using a randomly-generated network with $N = 4$ nodes, and a dynamical system model with $\mathbf{x} \in \mathbb{R}^4$ and $\mathbf{z}_i \in \mathbb{R}^2$. The state-transition matrix is a randomly-generated invertible matrix, while the measurement matrices are also random, and different for each node. The matrices $\mathbf{Q}_k$ and $\mathbf{R}_{i_k}$ are all randomly chosen positive definite matrices. A randomly generated adjacency matrix is used for the network, given by:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

In Figure 2 we plot the estimation error as well as the reported $\pm 3\sigma$ envelope[2] for the first element of the state vector, in node 1, for the following methods: (i) a centralized KF estimator, (ii) an estimator where node 1 only uses its own measurements in a KF, (iii) the KCIF, (iv) the ICF with one iteration to ensure equal communication cost with the proposed method, and (v) the proposed method.

We can see that the reported standard deviation of the single-node approach is the worst – as expected. On the other hand, the optimal method is the centralized one, and no estimator should report a smaller standard deviation. However, we observe that the ICF reports an accuracy *better* than the centralized KF. Clearly, this is incorrect, and results from the fact that the covariance reported from the ICF with a finite number of iterations does not represent the covariance of the actual estimation errors, as discussed earlier. By contrast, the proposed method reports a standard deviation that actually represents the true statistics of the estimation errors.

B. Monte-Carlo Accuracy Analysis

We now present the results of Monte-Carlo simulations conducted to compare the performance of our proposed method to consensus-based estimation. Our goal is to examine the relative performance of the methods for graphs with different topology characteristics. Therefore, we conduct two sets of simulations: in one case we maintain the number of nodes in the network, N, constant and vary the node in-degree, d, while in the second we fix the node in-degree and vary the number of nodes.

Since the reported covariance matrix of consensus-type methods is not a reliable measure of estimation accuracy, to evaluate precision for each selection of N and d we perform 100 Monte-Carlo simulations, and compute the average root mean square error (RMSE) for each method. In each Monte-Carlo trial a different randomly generated graph is used, as well as different randomly generated state-transition, measurement, and noise-covariance matrices. Estimation is performed with each of the algorithms processing the same measurements, and then the RMSE at each time step is computed by averaging over all Monte-Carlo trials. Finally, the mean RMSE over the entire duration of the estimation experiment for each algorithm is computed, and used as the measure of performance.

[2]The standard deviation σ is computed as the square root of the first diagonal element of the reported covariance matrix, for each method.

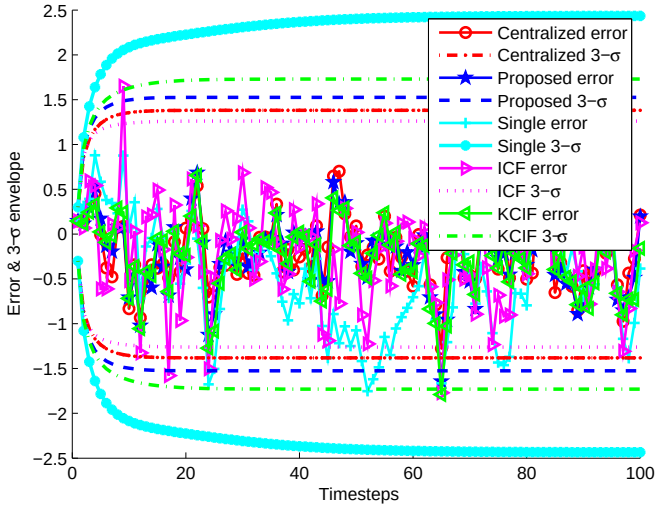

Fig. 2. Comparison of the proposed method to alternatives, on a randomly-generated simulation example.

Figure 3 shows the performance of the estimators for networks with a fixed number of nodes, $N = 30$, and varying node in-degree d. We can clearly observe that the proposed approach outperforms both the single-iteration ICF and the KCIF. Specifically, as compared to the ICF, our proposed method leads to at least a 30% reduction in mean RMSE. On the other hand, the improvement in performance against the KCIF is smaller (up to approximately 18% reduction in mean RMSE), and is most pronounced for sparser graphs. Intuitively, this observation makes sense, since in sparser graphs the proposed approach is able to benefit from having an accurate description of the relative accuracy of different nodes' estimates. As the graph becomes denser, the performance gap between the proposed approach and the KCIF becomes smaller, and both algorithms approach the performance of the centralized KF.

Figure 4 shows the performance of the methods for networks with a fixed node in-degree, $d = 4$, but varying total number of nodes, N. The results in this case are similar to those of Figure 3. Specifically, the KCIF and ICF have approximately 20% and 30% larger mean RMSE, respectively, for a large range of in-degree values. Moreover, we can see that the performance advantage of the proposed method is most pronounced in sparser graphs.

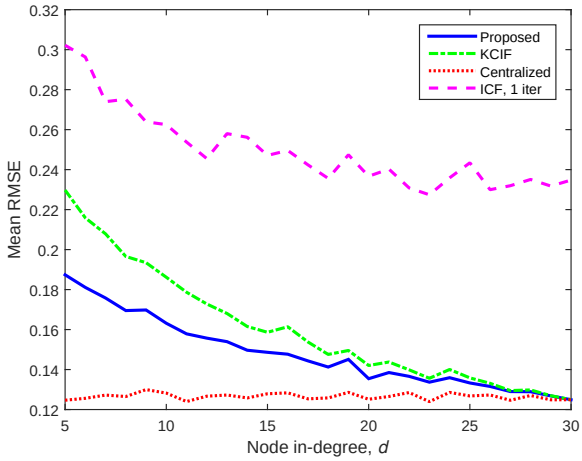

Fig. 3. Comparison of the accuracy (mean RMSE) in graphs of a fixed size, $N = 30$, but with varying node in-degree.

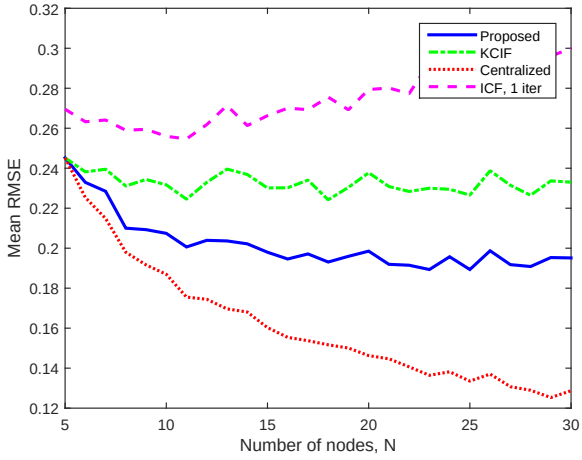

Fig. 4. Comparison of the accuracy (mean RMSE) in graphs of a varying size, but with fixed node in-degree, $d = 4$.

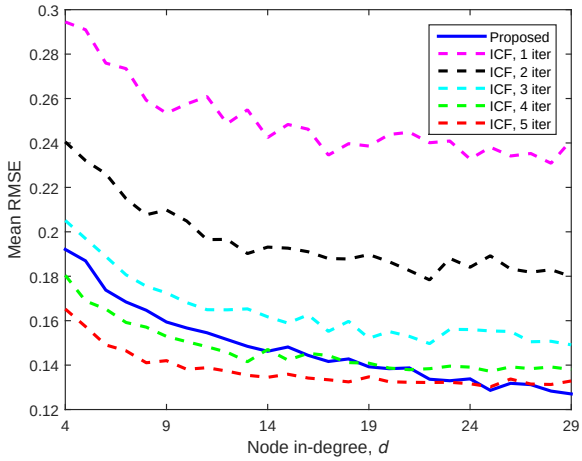

Fig. 5. Comparison of the accuracy (mean RMSE) of the proposed method vs. the ICF in a graph of fixed size, $N = 30$, with varying node in-degree. The number of ICF iterations ranges between one and five.

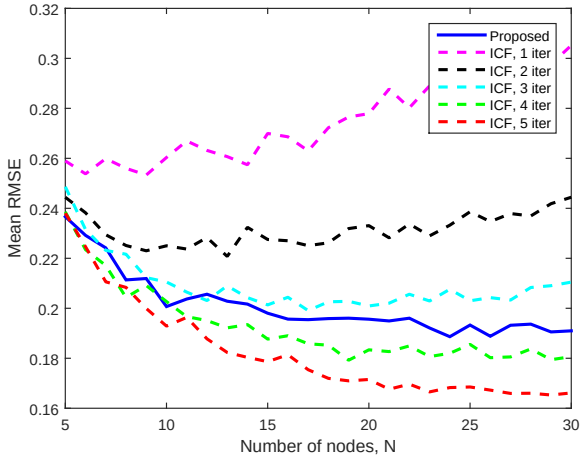

Fig. 6. Comparison of the accuracy (mean RMSE) of the proposed method vs. the ICF in a graph of fixed node in-degree, $d = 4$, with varying size. The number of ICF iterations ranges between one and five.

When the ICF algorithm only performs one iteration per update step, it has the same communication requirements as the method proposed here. However, it is also interesting to examine how our approach compares against the ICF when the latter is allowed to perform multiple iterations per update step. These results are shown in Figures 5 and 6, which show the mean RMSE of the proposed method against that of the ICF with the iteration number varying from one to five (the simulation setup is identical to those in Figures 3 and 4, respectively). These results show that, in most cases in this test, the ICF would require between three and five iterations (i.e., between three and five times more communication) to yield estimation accuracy similar to the proposed method.

C. Nonlinear example

As discussed in Section III-A, the proposed estimation method can be employed for estimating the state of nonlinear dynamical models, via linearization. In this case, each node linearizes the process and/or measurement models by evaluating Jacobians using its own state estimates. As a result different nodes will, in general, employ different linearization points. Despite this, in our tests we have verified that the proposed approach can produce consistent state estimates (in the sense defined in [25, Section 5.4]).

We here present a representative simulation result, in which the proposed estimator is used to track the position of target moving in 2D. The target is tracked using relative-distance measurements from eight sensor nodes. The sensors are placed at known, constant locations in the 2D plane. The position of the target is modelled by a constant-velocity model (with the addition of Gaussian process noise). Figure 7 shows the time-evolution of the estimation errors, as well as the reported $\pm 3\sigma$ envelope for the position of the target. We can clearly observe that the estimation errors are commensurate with the reported estimation covariance, which was one of the key motivations for the development of the proposed

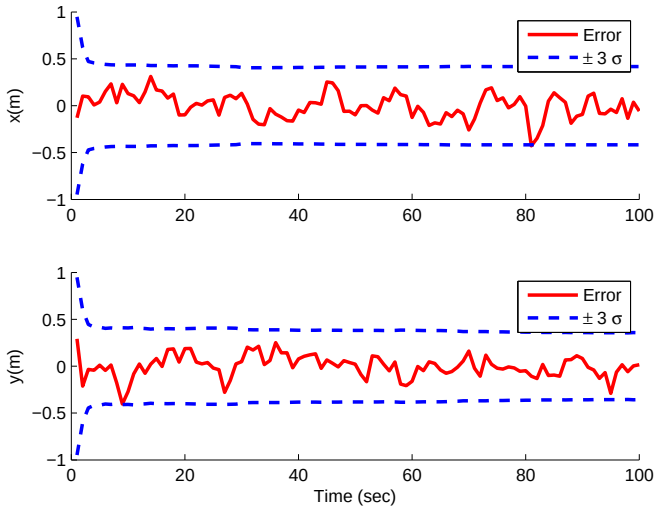

Fig. 7. Error and $\pm 3\sigma$ envelope reported by the proposed approach used to track the position of a moving target using relative-distance measurements.

method.

V. Conclusion

In this paper, we presented a method for distributed estimation of the state of a dynamical system. The key characteristic of the proposed approach is that it utilizes knowledge of the communication topology to explicitly model the accuracy of individual nodes' estimates, as well as their correlations. As a result, it is able to optimally fuse the measurements and state estimates of a node's neighbors, and to produce accurate state estimates with a single round of communication. Our results demonstrate that when compared against popular consensus-based algorithms, such as the KCIF and the ICF, the proposed approach attains higher estimation precision for the same amount of inter-node communication. Maintaining an accurate description of the correlations between nodes' estimates incurs a per-node computational cost that scales quadratically with the size of the network. Therefore, this method would be ideally suited for medium-sized networks (e.g., consisting of up to a few tens of nodes).

Acknowledgement

This work was supported by the National Science Foundation (grant no. IIS-1253314 and IIS-1316934).

References

[1] M. Saqib and C. Lee, "Traffic control system using wireless sensor network," in *Proceedings of the 12th International Conference on Advanced Communication Technology (ICACT)*, vol. 1, Phoenix park, Korea, Feb. 2010, pp. 352–357.

[2] A. Pascale, M. Nicoli, F. Deflorio, B. Dalla Chiara, and U. Spagnolini, "Wireless sensor networks for traffic management and road safety," *IET Intelligent Transport Systems*, vol. 6, no. 1, pp. 67–77, Mar. 2012.

[3] K. K. Khedo, R. Perseedoss, and A. Mungur, "A wireless sensor network air pollution monitoring system," *International Journal of Wireless and Mobile Networks*, vol. 2, no. 2, pp. 31–45, 2010.

[4] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, GA, USA, Sept. 2002, pp. 88–97.

[5] T. Gao, D. Greenspan, M. Welsh, R. Juang, and A. Alm, "Vital signs monitoring and patient tracking over a wireless network," in *Proceedings of the 27th Annual International Conference of the Engineering in Medicine and Biology Society*, Shanghai, China, Jan. 2006, pp. 102–105.

[6] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: a survey," *Computer Networks*, vol. 54, pp. 2688–2710, 2010.

[7] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J. A. Stankovic, and T. Abdelzaher, "Achieving real-time target tracking using wireless sensor networks," in *Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium*, San Jose, CA, USA, Apr. 2006.

[8] H.-W. Tsai, C.-P. Chu, and T.-S. Chen, "Mobile object tracking in wireless sensor networks," *Computer Communications*, vol. 30, no. 8, pp. 1811–1825, 2007.

[9] H. Durrant-Whyte, B. Rao, and H. Hu, "Toward a fully decentralized architecture for multi-sensor data fusion," in *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, vol. 2, Cincinnati, OH, USA, May 1990, pp. 1331–1336.

[10] B. Rao and H. Durrant-Whyte, "Fully decentralised algorithm for multisensor Kalman filtering," *Control Theory and Applications, IEE Proceedings D*, vol. 138, no. 5, pp. 413–420, Sept. 1991.

[11] S. Grime and H. Durrant-Whyte, "Data fusion in decentralized sensor networks," *Control Engineering Practice*, vol. 2, no. 5, pp. 849–863, 1994.

[12] W. Ren, R. Beard, and D. Kingston, "Multi-agent Kalman consensus with relative uncertainty," in *Proceedings of the 2005 American Control Conference*, vol. 3, Portland, OR, USA, Jun. 2005, pp. 1865–1870.

[13] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, USA, Dec. 2007, pp. 5492–5498.

[14] A. Kamal, C. Ding, B. Song, J. Farrell, and A. Roy-Chowdhury, "A generalized Kalman consensus filter for wide-area video networks," in *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, Dec. 2011, pp. 7863–7869.

[15] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *Proceedings of the 44th IEEE Conference on Decision and Control and 2005 European Control Conference*, Seville, Spain, Dec. 2005, pp. 8179–8184.

[16] ——, "Kalman-consensus filter: optimality, stability, and performance," in *Proceedings of the 48th IEEE Conference on Decision and Control, held jointly with the 28th Chinese Control Conference*, Shanghai, China, Dec. 2009, pp. 7036–7042.

[17] A. Kamal, J. Farrell, and A. Roy-Chowdhury, "Information weighted consensus filters and their application in distributed camera networks," *IEEE Transactions on Automatic Control*, vol. 58, no. 12, pp. 3112–3125, Dec. 2013.

[18] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, pp. 1520–1533, Sept. 2004.

[19] N. G. Wah and Y. Rong, "Comparison of decentralized tracking algorithms," in *Proceedings of the 6th International Conference on Information Fusion*, vol. 1, Cairns, Queensland, Australia, Jul. 2003, pp. 107–113.

[20] Q. Gan and C. Harris, "Comparison of two measurement fusion methods for Kalman-filter-based multisensor data fusion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 1, pp. 273–279, Jan. 2001.

[21] K. Kim, "Development of track to track fusion algorithms," in *Proceedings of the 1994 American Control Conference*, vol. 1, Baltimore, MD, USA, Jun. 1994, pp. 1037–1041.

[22] H. Chen, T. Kirubarajan, and Y. Bar-Shalom, "Performance limits of track-to-track fusion vs. centralized estimation: theory and application," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, pp. 386–400, 2003.

[23] S. li Sun, "Multi-sensor optimal information fusion Kalman filters with applications," *Aerospace Science and Technology*, vol. 8, no. 1, pp. 57–62, Jan. 2004.

[24] R. Hunger, "Floating point operations in matrix-vector calculus," Munich University of Technology, Tech. Rep., 2007.

[25] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, 2001.