Information Weighted Consensus Filters and their Application in Distributed Camera Networks

A. T. Kamal, J. A. Farrell and A. K. Roy-Chowdhury
University of California, Riverside, CA-92521

***Abstract*—Due to their high fault-tolerance and scalability to large networks, consensus-based distributed algorithms have recently gained immense popularity in the sensor networks community. Large scale camera networks are a special case. In a consensus-based state estimation framework, multiple neighboring nodes iteratively communicate with each other, exchanging their own local information about each target's state with the goal of converging to a single state estimate over the entire network. However, the state estimation problem becomes challenging when some nodes have limited observability of the state. In addition, the consensus estimate is sub-optimal when the cross-covariances between the individual state estimates across different nodes are not incorporated in the distributed estimation framework. The cross-covariance is usually neglected because the computational and bandwidth requirements for its computation become unscalable for a large network. These limitations can be overcome by noting that, as the state estimates at different nodes converge, the information at each node becomes correlated. This fact can be utilized to compute the optimal estimate by proper weighting of the prior state and measurement information. Motivated by this idea, we propose information-weighted consensus algorithms for distributed maximum a posteriori parameter estimation, and their extension to the information-weighted consensus filter (ICF) for state estimation. We compare the performance of the ICF with existing consensus algorithms analytically, as well as experimentally by considering the scenario of a distributed camera network under various operating conditions.**

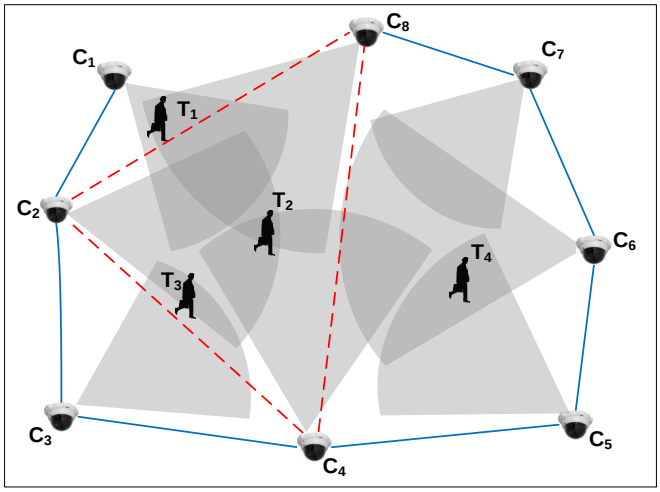

Fig. 1: The figure shows eight sensing nodes, $C_1, C_2, \ldots, C_8$ and four targets T_1, T_2, T_3 and T_4. The solid blue lines show the communication channels between different nodes. This figure also depicts the presence of "naive" nodes. For example, C_1, C_8 and their immediate network neighbors, C_2, C_7 get direct measurements about T_1. However, the rest of the cameras, i.e., C_3, C_4, C_5 and C_6 do not have *direct* access to measurements of T_1 and thus are naive w.r.t. T_1's state. The vision graph for a specific target is a graph where there is an edge between each pair of nodes that are observing that target. The vision graph for T_2 is shown in red dotted lines.

I. Introduction

Distributed estimation schemes are becoming increasingly popular in the sensor networks community due to their scalability for large networks and high fault tolerance. Unlike centralized schemes, distributed schemes usually rely on peer-to-peer communication between sensor nodes and the task of information fusion is distributed across multiple nodes. In a sensor network, each sensor may get multiple or no measurements related to the state of a target. The objective of a distributed estimation framework is to maintain an accurate estimate of the state of all targets using all the measurements in the network without requiring a centralized node for information fusion. In many application domains, e.g., a distributed camera network, it is typical for some nodes to not have observations of some targets whose state is being estimated. In this paper, we study this issue for consensus-based distributed estimation schemes.

Let us consider a sensor network such as that illustrated in Fig. 1, where a collection of N communication nodes $\mathcal{C} = \{C_1, C_2, \ldots, C_N\}$, are connected with each other using a network topology $\mathcal{G}$. Each node consists of one or more heterogeneous sensors. The sensor network is monitoring an area containing N_T moving targets. The vector containing the concatenated state of all targets at time instant t, is represented by $\mathbf{x}(t) \in \mathcal{R}^p$. Each node C_i, through its sensors, gets a measurement $\mathbf{z}_i(t) \in \mathcal{R}^{m_i}$, linearly related to the state $\mathbf{x}(t)$. The specific targets detected by each C_i determines whether $\mathbf{z}_i(t)$ consists of multiple or no measurements and in either case determines the (local) observability of portions of $\mathbf{x}(t)$ from $\mathbf{z}_i(t)$. In this paper, we are interested in estimating the state $\mathbf{x}(t)$ in a distributed framework at each time instant t, using all the measurements available in the network up to time t. We have particular interest in situations where, even though the state is observable from all measurement throughout the system, the full state is not observable to all C_i from the local

Ahmed T. Kamal, Email: akamal@ee.ucr.edu
Jay A. Farrell, Tel: +1-951-827-2159, Email: farrell@ee.ucr.edu, Fax: +1-951-827-2425
Amit K. Roy-Chowdhury, Tel: +1-951-827-7886, Email: amitrc@ee.ucr.edu, Fax: +1-951-827-2425

Mailing Address: Electrical Engineering Suite 343 Winston Chung Hall University of California, Riverside Riverside, CA 92521-0429

This work was partially supported by ONR award N000140910666 titled Distributed Dynamic Scene Analysis in a Self-Configuring Multimodal Sensor Network.

measurements available to each C_i.

A centralized [1] or a spanning-tree based approach [2] could be used to perform the estimation task where all the measurements are brought to a centralized processing node before the data fusion is performed. However, these approaches become unscalable for large networks due to communication bandwidth and processing power limitations and are also less tolerant to node failure. These are some of the many reasons why distributed signal and information processing over networks are becoming increasingly popular and are preferred over centralized and spanning tree based approaches. An example of a distributed estimation approach is the *average consensus* algorithm [3], which will be of special interest for us throughout this paper. In the average consensus algorithm, each node in a network shares its information with its immediate neighbors and corrects its own state using the information sent by its neighbors. By doing so iteratively, assuming a connected undirected graph, each node can asymptotically compute the arithmetic mean of all the states in the network without requiring a centralized server.

For a dynamic state estimation problem, a predictor-corrector solution approach is often used. The Kalman Consensus Filter (KCF) [4] is a popular and efficient algorithm to solve this task in a distributed framework. The KCF algorithm assumes that all targets are observable by each node. However, there are situations (e.g. in Fig. 1), where such a condition cannot be met. The performance of the KCF approach may deteriorate when applied in such a situation. This topic is discussed in greater depth in Sec. VI.

Consensus based approaches asymptotically converge to their estimates through multiple iterations of communication between neighboring nodes. In practice, due to communication bandwidth limitation and target dynamics, only a limited number of consensus iterations K, can be performed at each time step t. Moreover, a node will have limited observability when it does not have any measurement of a target available in its *local neighborhood* (consisting of the node and its immediate network neighbors). Due to limited observability and limited number of iterations, the node becomes *naive* about the target's state. A naive node contains less information about the state. The convergence rate of naive nodes can be significantly slower than that of informed nodes for the KCF algorithm. This issue is exacerbated at subsequent measurement times, as the KCF assumes equal information weighting for the state estimate available from each node. The effect of naivety is larger in a sparse network where the number of communication links between the nodes is small. The concept of naivety is depicted in Fig 1. In the presence of naive nodes, the performance of the KCF algorithm deteriorates.

A. Contributions

The KCF algorithm weights all its neighbors' prior states $\mathbf{x}_j^-$'s equally which causes high estimation error when naive nodes are present. An initial approach to resolve this issue was proposed in [5]. There the Generalized Kalman Consensus Filter (GKCF) algorithm was proposed where the neighbors' prior $\mathbf{x}_j^-$'s were weighted by their covariance matrices $\mathbf{P}_j$'s. The GKCF algorithm outperforms the KCF in the presence of naive nodes. However, the effect of correlation between errors of the nodes' prior estimates was not brought into account in any of the prior methods because it is usually extremely hard to estimate the cross-covariance across all the nodes in a distributed framework. Mainly due to this reason, these distributed estimation schemes are suboptimal.

Naivety relative to the n-th target is likely to be present in the network when nodes on the vision graph for target n are sparsely connected in the communication graph. The vision graph (see [6]) for target T_n is a graph where there is an edge between each pair of nodes that are observing T_n (see Fig. 1). The vision graph is usually time varying, different from the communication graph, and different for different targets. Distributed algorithms that are derived under the assumption that each sensor has full state observability at each time (e.g. KCF), may be adapted to be used in the presence of naive nodes when communication protocols (e.g. [7]) for translating a known vision graph into a set of routing paths that connect nodes in the communication graph are available. However, scalable distributed vision graph discovery algorithms do not exist. Moreover, distributed algorithms that require the knowledge of the vision graphs usually require special target handoff protocols.

Additionally, computation and communication resource constraints are important issues in a distributed estimation framework because in many application scenarios, the nodes are low powered wireless devices. Therefore, a distributed state estimation framework that can guarantee convergence to the optimal centralized estimate while maintaining low computation and communication resource requirements in the presence of naive nodes is desirable. In this paper, we propose such a distributed state estimation framework called the Information-weighted Consensus Filter (ICF). The ICF is guaranteed to converge to the optimal centralized performance under certain reasonable conditions. We also show experimentally that in other conditions it achieves near-optimal performance.

The issue of naivety is handled in the ICF algorithm by proper information weighting in the estimation framework. Optimality is achieved by proper relative weighting between the prior and the measurement information. ICF also supports multiple consensus iterations at each time step t to improve performance. In addition, the experimental results show that ICF outperforms other distributed estimation algorithms at any given computation and communication resource limit. The ICF algorithm does not require handoff protocols or the knowledge of the vision graph.

B. Related work

Recently, distributed decision and control frameworks have gained immense popularity. *Consensus algorithms* [3] are one of the many types of distributed schemes used for collective decision making. Consensus algorithms are protocols that are run individually by each agent where each agent communicates with just its network neighbors and corrects its own information iteratively using the information sent by its neighbors. The protocol, over multiple iterations, ensures

the convergence of all the agents in the network to a single consensus. The consensus they reach is a predefined function of all the information available in the network. It is important to note that this consensus is reached just by peer-to-peer communication without requiring a central fusion node. For example, the item being estimated may be the arithmetic mean (average consensus) [3] or the geometric mean [8] of the initial values. The simplicity and scalability of consensus algorithms makes them extremely useful in distributed estimation tasks in sensor networks.

Consensus algorithms have been extended to perform various linear algebraic operations such as SVD, least squares, PCA, GPCA in a network of agents [6]. It also has been utilized in distributed state estimation framework such as the Kalman Consensus Filter (KCF) [4] and the Generalized Kalman Consensus Filter (GKCF) [5]. The KCF algorithm is a popular distributed estimation framework and has reasonable performance in networks where the entire state is individually observable by each node. However, its performance deteriorates in the presence of naivety [5]. The idea of information-weighted consensus was introduced in [9]; however, that article did not provide the detailed theoretical analysis of its properties, comparisons with other approaches, or the implications for its application in wide-area camera networks. These items are each provided herein. A survey on distributed estimation and control applications using linear consensus algorithms can be found in [10]. Here the authors show how a weighted average can be computed in a distributed framework using two parallel average consensus schemes. Distributed state and parameter estimation frameworks have been applied in various fields including camera networks for distributed implementations of 3-D point triangulation, pose estimation [6], tracking [11], action recognition [11], [12], collaborative tracking and camera control [13], [14], camera calibration [15], [16] etc.

C. Average consensus - Review

Average consensus [3] is a popular distributed algorithm to compute the arithmetic mean of some values $\{a_i\}_{i=1}^N$. Suppose, in a network of N nodes, each node i has a state a_i. We are interested in computing the average value of these states i.e. $\frac{1}{N}\sum_{i=1}^N a_i$, in a distributed manner.

In the average consensus algorithm, each node initializes its consensus state as $a_i(0) = a_i$ and runs the following protocol iteratively

$$a_i(k) = a_i(k-1) + \epsilon \sum_{j \in \mathcal{N}_i} \left(a_j(k-1) - a_i(k-1)\right). \quad (1)$$

At the beginning of iteration k, a node C_i sends its previous state $a_i(k-1)$ to its immediate network neighbors $C_j \in \mathcal{N}_i$ and also receives the neighbors' previous states $a_j(k-1)$. Then it updates its state using (1). By iteratively doing so, the values of the states at all the nodes converge to the average of the initial values. The average consensus algorithm can be used to compute the average of vectors and matrices by applying it to their individual elements separately. Note that average consensus treats all nodes as having equal and uncorrelated information about the quantity a. It is straightforward to show that average consensus preserves the symmetric positive semi-definite property when used on a matrix. This property is especially important when average consensus is applied to covariance matrices.

The rate parameter ϵ should be chosen between 0 and $\frac{1}{\Delta_{max}}$, where Δ_{max} is the maximum degree of the network graph $\mathcal{G}$. Choosing larger values of ϵ will result in faster convergence, but choosing values equal or more than Δ_{max} will render the algorithm unstable. More information about average consensus and about the rate parameter ϵ can be found in [3].

II. Problem Formulation

The communication network is represented by the undirected connected graph $\mathcal{G} = (\mathcal{C}, \mathcal{E})$. The set $\mathcal{C} = \{C_1, ..., C_N\}$ contains the vertices of the graph and represents the communication nodes. The set $\mathcal{E}$ contains the edges of the graph, which represent the available communication channels between different nodes. Also, let $\mathcal{N}_i$ be the set of nodes that have a direct communication channel with node C_i (i.e. shares an edge with C_i). We call the nodes in $\mathcal{N}_i$, the neighbors of C_i. The number of neighbors (degree) of C_i is represented by Δ_i. For simplicity, we will drop the time step index t in places where the time step is not important to explain the issue under consideration. There are N_T targets in the area and the length of their individual state vectors is q. Thus, for $\mathbf{x} \in \mathcal{R}^p$, p would be equal to qN_T. The number of targets can be time-varying, but that will not be a focus of the paper, as it would distract from the main topic.

A data association protocol is required for a multi-target estimation framework. However, in this paper, to analyze the performance of the state estimation approaches independent of a data association method, we assume that the data association is given and without errors.

At each time step t, a node C_i may get some measurements from its sensors. For example, for a target state estimation task in a camera network, a measurement might be the projection of the position of some target onto a camera's pixel coordinate system. We denote the measurement at C_i as $\mathbf{z}_i \in \mathcal{R}^{m_i}$. As, the sensors can be heterogenous and the number of sensors can be different at each communication node, the length of the measurement vector m_i can vary across different nodes. The measurement $\mathbf{z}_i$ is modeled as

$$\mathbf{z}_i = \mathbf{H}_i\mathbf{x} + \boldsymbol{\nu}_i. \quad (2)$$

Here $\mathbf{H}_i \in \mathcal{R}^{m_i \times p}$ is the observation matrix for node C_i. The matrix $\mathbf{H}_i$ can be time-varying.

The noise $\boldsymbol{\nu}_i \in \mathcal{R}^{m_i}$ in the measurement of C_i is modeled as a zero mean Gaussian random variable $\mathcal{N}(\mathbf{0}, \mathbf{R}_i)$ where $\mathbf{R}_i \in \mathcal{R}^{m_i \times m_i}$. As we will derive the estimator in the information form, we will generally refer to the information matrix, $\mathbf{B}_i = \mathbf{R}_i^{-1} \in \mathcal{R}^{m_i \times m_i}$ which can also be time varying. If a node C_i does not have a measurement, we will use $\mathbf{B}_i = \mathbf{0} \in \mathcal{R}^{m_i \times m_i}$ and $\mathbf{z}_i = 0 \in \mathcal{R}^{m_i}$. Note that $\mathbf{H}_i$ is typically not full column rank ($m_i < p$). In fact, we are not assuming that the state $\mathbf{x}$ is observable from $\mathbf{z}_i$. We do assume that $\mathbf{x}$ is observable from $\{\mathbf{z}_i\}_{i=1}^N$. For example, in Fig. 1, $\mathbf{x}$

represents the concatenation of the states of targets $\{T_n\}_{n=1}^{N_T}$ and is observable from $\{\mathbf{z}_i\}_{i=1}^{N}$; however given only $\mathbf{z}_7$, the vector $\mathbf{x}$ is completely unobservable.

The collection of all measurements from all sensors can be expressed as,

$$\boldsymbol{\mathcal{Z}} = \boldsymbol{\mathcal{H}}\mathbf{x} + \boldsymbol{\nu}. \tag{3}$$

Here, $\boldsymbol{\mathcal{Z}} = [\mathbf{z}_1^T, \mathbf{z}_2^T, \dots \mathbf{z}_N^T]^T \in \mathcal{R}^m$ is the concatenation of all measurements in the network where $m = \sum_{i=1}^{N} m_i$ and $\boldsymbol{\mathcal{H}} = [\mathbf{H}_1^T, \mathbf{H}_2^T, \dots \mathbf{H}_N^T]^T \in \mathcal{R}^{m \times p}$ is the stack of all the observation matrices. For the measurement noise vector, $\boldsymbol{\nu} = [\boldsymbol{\nu}_1^T, \boldsymbol{\nu}_2^T, \dots \boldsymbol{\nu}_N^T]^T \in \mathcal{R}^m$, we denote its covariance as $\boldsymbol{\mathcal{R}} \in \mathcal{R}^{m \times m}$ and information matrix as $\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{R}}^{-1} \in \mathcal{R}^{m \times m}$. We assume the measurement noise to be uncorrelated across nodes. Thus, the measurement covariance matrix is $\boldsymbol{\mathcal{R}} = diag(\mathbf{R}_1, \mathbf{R}_2, \dots \mathbf{R}_N)$ and the measurement information matrix is $\boldsymbol{\mathcal{B}} = diag(\mathbf{B}_1, \mathbf{B}_2, \dots \mathbf{B}_N)$.

III. Centralized MAP Estimation

Before considering the decentralized solution for the estimation of $\mathbf{x}$ it is informative to review the centralized solution, i.e., the centralized maximum a posteriori (MAP) estimator. The centralized prior state estimate of $\mathbf{x}$ is denoted as $\mathbf{x}_c^- \in \mathcal{R}^p$ where the error in the prior estimate is $\boldsymbol{\eta}_c = \mathbf{x}_c^- - \mathbf{x}$ with $cov(\boldsymbol{\eta}_c) = \mathbf{P}_c \in \mathcal{R}^{p \times p}$, which is assumed to be nonsingular. The observation model (3) and prior state $\mathbf{x}_c^-$ can be combined into one equation as

$$\begin{bmatrix} \mathbf{x}_c^- \\ \boldsymbol{\mathcal{Z}} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_p \\ \boldsymbol{\mathcal{H}} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \boldsymbol{\eta}_c \\ \boldsymbol{\nu} \end{bmatrix}. \tag{4}$$

Here $\mathbf{I}_p$ is the $p \times p$ identity matrix.

Letting, $\boldsymbol{\mathcal{Y}} = \begin{bmatrix} \mathbf{x}_c^- \\ \boldsymbol{\mathcal{Z}} \end{bmatrix}$, $\boldsymbol{\mathcal{H}}_c = \begin{bmatrix} \mathbf{I}_p \\ \boldsymbol{\mathcal{H}} \end{bmatrix}$ and $\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\eta}_c \\ \boldsymbol{\nu} \end{bmatrix}$, we have $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{H}}_c\mathbf{x} + \boldsymbol{\beta}$, where $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\mathcal{C}})$. We assume that the error in the prior state estimate is uncorrelated to the measurement noise. Thus, we have the block diagonal covariance matrix $\boldsymbol{\mathcal{C}} = diag(\mathbf{P}_c, \boldsymbol{\mathcal{R}})$. Let us define the prior information matrix to be $\mathbf{J}_c^- = (\mathbf{P}_c)^{-1} \in \mathcal{R}^{p \times p}$. Thus, defining $\boldsymbol{\mathcal{J}} = \boldsymbol{\mathcal{C}}^{-1}$ we have $\boldsymbol{\mathcal{J}} = diag(\mathbf{J}_c^-, \boldsymbol{\mathcal{B}})$. The centralized maximum a posteriori (MAP) estimate [17] of the state $\mathbf{x}$ is

$$\begin{aligned} \mathbf{x}_c^+ &= (\boldsymbol{\mathcal{H}}_c^T \boldsymbol{\mathcal{J}} \boldsymbol{\mathcal{H}}_c)^{-1} (\boldsymbol{\mathcal{H}}_c^T \boldsymbol{\mathcal{J}} \boldsymbol{\mathcal{Y}}) \\ &= \left(\mathbf{J}_c^- + \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{H}}\right)^{-1} (\mathbf{J}_c^- \mathbf{x}_c^- + \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{Z}}), \quad (5) \\ \mathbf{J}_c^+ &= \left(\mathbf{J}_c^- + \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{H}}\right). \quad (6) \end{aligned}$$

where $\mathbf{J}_c^+ = (cov(\mathbf{x}_c^+))^{-1}$ quantifies the information about $\mathbf{x}$ in $\mathbf{x}_c^+$. Eqns. (5-6) are useful in the discussion of the physical interpretations of the alternative decentralized algorithms.

IV. Information Consensus based Distributed MAP estimation (IC-MAP)

In a distributed estimation framework, each node C_i possesses a prior estimate of the state vector that we denote as $\mathbf{x}_i^-(t) \in \mathcal{R}^p$ (unlike a single prior state estimate in a centralized solution). The objective of the network is to use distributed computations across the network such that the posterior state estimate $\mathbf{x}_i^+(t) \in \mathcal{R}^p$ at each node converges to the centralized estimate $\mathbf{x}_c^+(t)$. However, due to resource constraints, this convergence may not be fully achieved at any given time. Therefore, if consensus were performed directly on the priors, then at the k-th consensus iteration, the estimate of the i-th node can be modeled as having three components

$$\mathbf{x}_i^{k-} = \mathbf{x} + \boldsymbol{\eta}_c + \boldsymbol{\delta}_i^k,$$

where $\mathbf{x}_c^- = \mathbf{x} + \boldsymbol{\eta}_c$ with the quantities $\mathbf{x}$ and $\boldsymbol{\eta}_c$ having been previously defined, and the consensus algorithm ensures that $\|\boldsymbol{\delta}_i^k\|$ approaches zero as k approaches infinity for all i.

The error covariance in the prior estimate at C_i is $\mathbf{P}_{ii}^k = E[\boldsymbol{\eta}_i^k \left(\boldsymbol{\eta}_i^k\right)^T] \in \mathcal{R}^{p \times p}$, where $\boldsymbol{\eta}_i^k = \boldsymbol{\eta}_c + \boldsymbol{\delta}_i^k$. Similarly, the error cross-covariance between C_i and C_j's prior estimates is $\mathbf{P}_{ij}^k = E[\boldsymbol{\eta}_i^k \left(\boldsymbol{\eta}_j^k\right)^T] \in \mathcal{R}^{p \times p}$. As $k \to \infty$, at each node, consensus forces $\mathbf{x}_i^{k-} \to \mathbf{x}_c^-$. Therefore, for any $\{i, j\}$, $\mathbf{x}_i^{k-}$ and $\mathbf{x}_j^{k-}$ becomes correlated as $\mathbf{x}_i^{k-} \to \mathbf{x}_j^{k-}$ resulting in $\mathbf{P}_{ij}^k \neq 0$. In fact, it is straightforward to show that as the number of consensus iterations k approaches infinity, $\mathbf{P}_{ij}^k$ converges to $\mathbf{P}_c$ for all $\{i, j\}$.

We drop the k and denote the collection of all the state priors from all the nodes as $\boldsymbol{\mathcal{X}}^- = [(\mathbf{x}_1^-)^T, (\mathbf{x}_2^-)^T, \dots, (\mathbf{x}_N^-)^T]^T \in \mathcal{R}^{Np}$. The relationship between the state, the priors and the prior errors can be summarized as,

$$\boldsymbol{\mathcal{X}}^- = \boldsymbol{\mathcal{H}}_I \mathbf{x} + \boldsymbol{\eta}. \tag{7}$$

Here, $\mathbf{x}$ is the true state of the targets, $\boldsymbol{\eta} = [\boldsymbol{\eta}_1^T, \boldsymbol{\eta}_2^T, \dots \boldsymbol{\eta}_N^T]^T \in \mathcal{R}^{Np}$ is the error vector, and $\boldsymbol{\mathcal{H}}_I = [\mathbf{I}_p, \mathbf{I}_p, \dots, \mathbf{I}_p]^T \in \mathcal{R}^{Np \times p}$.

Combining the measurements with the result of a *finite number* k of steps of consensus on the priors yields

$$\begin{bmatrix} \boldsymbol{\mathcal{X}}^- \\ \boldsymbol{\mathcal{Z}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mathcal{H}}_I \\ \boldsymbol{\mathcal{H}} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\nu} \end{bmatrix}. \tag{8}$$

Letting, $\boldsymbol{\mathcal{Y}}' = \begin{bmatrix} \boldsymbol{\mathcal{X}}^- \\ \boldsymbol{\mathcal{Z}} \end{bmatrix}$, $\boldsymbol{\mathcal{H}'}_c = \begin{bmatrix} \boldsymbol{\mathcal{H}}_I \\ \boldsymbol{\mathcal{H}} \end{bmatrix}$ and $\boldsymbol{\beta}' = \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\nu} \end{bmatrix}$, we have $\boldsymbol{\mathcal{Y}}' = \boldsymbol{\mathcal{H}'}_c\mathbf{x} + \boldsymbol{\beta}'$, where $\boldsymbol{\beta}' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\mathcal{C}}')$. The noise term $\boldsymbol{\beta}'$ is Gaussian because it is accumulated through one or more consensus iterations (which are linear operations) performed on Gaussian random variables.

Let us denote the prior information matrix $\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{P}}^{-1}$ where these two matrices can be expressed as $(p \times p)$ blocks,

$$\boldsymbol{\mathcal{P}} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} & \dots & \mathbf{P}_{1N} \\ \mathbf{P}_{21} & \mathbf{P}_{22} & & \vdots \\ \vdots & & \ddots & \\ \mathbf{P}_{N1} & \dots & & \mathbf{P}_{NN} \end{bmatrix}, \; \boldsymbol{\mathcal{F}} = \begin{bmatrix} \mathbf{F}_{11} & \mathbf{F}_{12} & \dots & \mathbf{F}_{1N} \\ \mathbf{F}_{21} & \mathbf{F}_{22} & & \vdots \\ \vdots & & \ddots & \\ \mathbf{F}_{N1} & \dots & & \mathbf{F}_{NN} \end{bmatrix}. \tag{9}$$

Let us define the information matrix of the prior of node i as

$$\mathbf{J}_i^- = (\mathbf{P}_{ii})^{-1} \tag{10}$$

Here, $\mathbf{J}_i^- \in \mathcal{R}^{p \times p}$ and in general, $\mathbf{J}_i^- \neq \mathbf{F}_{ii}$. Assuming that the error in the prior state estimates are uncorrelated to the noise in the new measurements, we have the block diagonal covariance matrix $\boldsymbol{\mathcal{C}}' = diag(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{R}})$. Thus, we get its inverse as $\boldsymbol{\mathcal{J}}' = diag(\boldsymbol{\mathcal{F}}, \boldsymbol{\mathcal{B}})$.

The centralized maximum a posteriori (MAP) estimate of the state $\mathbf{x}$ is

$$\begin{aligned}\mathbf{x}_c^+ &= ({\boldsymbol{\mathcal{H}}'}_c^T \boldsymbol{\mathcal{J}}' \boldsymbol{\mathcal{H}}'_c)^{-1}({\boldsymbol{\mathcal{H}}'}_c^T \boldsymbol{\mathcal{J}}' \boldsymbol{\mathcal{Y}}') \\ &= \left(\boldsymbol{\mathcal{H}}_I^T \boldsymbol{\mathcal{F}} \boldsymbol{\mathcal{H}}_I + \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{H}}\right)^{-1} (\boldsymbol{\mathcal{H}}_I^T \boldsymbol{\mathcal{F}} \boldsymbol{\mathcal{X}}^- + \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{Z}}), \quad (11) \\ \mathbf{J}_c^+ &= \boldsymbol{\mathcal{H}}_I^T \boldsymbol{\mathcal{F}} \boldsymbol{\mathcal{H}}_I + \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{H}}. \quad (12)\end{aligned}$$

Defining

$$\mathbf{F}_i^- = \sum_{j=1}^{N} \mathbf{F}_{ji}, \quad (13)$$

we have

$$\boldsymbol{\mathcal{H}}_I^T \boldsymbol{\mathcal{F}} \boldsymbol{\mathcal{H}}_I = \sum_{i=1}^{N} \mathbf{F}_i^- \quad \text{and} \quad \boldsymbol{\mathcal{H}}_I^T \boldsymbol{\mathcal{F}} \boldsymbol{\mathcal{X}}^- = \sum_{i=1}^{N} \mathbf{F}_i^- \mathbf{x}_i^-. \quad (14)$$

Let us define $\mathbf{U}_i = \mathbf{H}_i^T \mathbf{B}_i \mathbf{H}_i$ and $\mathbf{u}_i = \mathbf{H}_i^T \mathbf{B}_i \mathbf{z}_i$ (with $\mathbf{B}_i$ and $\mathbf{H}_i$ defined right after (2)). Due to the block diagonal structure of $\boldsymbol{\mathcal{B}}$, we get

$$\begin{aligned}\boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{H}} &= \sum_{i=1}^{N} \mathbf{H}_i^T \mathbf{B}_i \mathbf{H}_i = \sum_{i=1}^{N} \mathbf{U}_i, \quad (15) \\ \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{Z}} &= \sum_{i=1}^{N} \mathbf{H}_i^T \mathbf{B}_i \mathbf{z}_i = \sum_{i=1}^{N} \mathbf{u}_i. \quad (16)\end{aligned}$$

Thus, from (11) and (12), we get

$$\begin{aligned}\mathbf{x}_c^+ &= \left(\sum_{i=1}^{N} (\mathbf{F}_i^- + \mathbf{U}_i)\right)^{-1} \sum_{i=1}^{N} (\mathbf{F}_i^- \mathbf{x}_i^- + \mathbf{u}_i), \quad (17) \\ \mathbf{J}_c^+ &= \sum_{i=1}^{N} (\mathbf{F}_i^- + \mathbf{U}_i). \quad (18)\end{aligned}$$

This is the centralized solution that fully accounts for the presence of different priors at each agent, and the cross-correlated errors in the priors between agents, which develop naturally due to consensus, but which are not known in a decentralized implementation. In the following, we show how (17) and (18) can be computed in a distributed manner.

A. Distributed Implementation

To implement eqns. (17-18) in a distributed fashion, define, $\mathbf{V}_i^0 = \mathbf{F}_i^- + \mathbf{U}_i$ and $\mathbf{v}_i^0 = \mathbf{F}_i^- \mathbf{x}_i^- + \mathbf{u}_i$, so that

$$\begin{aligned}\mathbf{x}_c^+ &= \left(\sum_{i=1}^{N} \mathbf{V}_i^0\right)^{-1} \sum_{i=1}^{N} \mathbf{v}_i^0, \quad (19) \\ \mathbf{J}_c^+ &= \sum_{i=1}^{N} \mathbf{V}_i^0. \quad (20)\end{aligned}$$

Under the assumption that a node C_i has information about $\mathbf{F}_i^-$ (methods for computing $\mathbf{F}_i^-$ will be discussed later), C_i could compute $\mathbf{V}_i^0 = \mathbf{F}_i^- + \mathbf{U}_i$ and $\mathbf{v}_i^0 = \mathbf{F}_i^- \mathbf{x}_i^- + \mathbf{u}_i$ from $\mathbf{F}_i^-$, its own state prior $\mathbf{x}_i^-$, measurement $\mathbf{z}_i$, measurement information matrix $\mathbf{B}_i$ and measurement model parameter $\mathbf{H}_i$. Then, each node transmits to its neighbors its own information matrix $\mathbf{V}_i^k \in \mathcal{R}^{p \times p}$ and information vector $\mathbf{v}_i^k \in \mathcal{R}^p$, receives it neighbors information, and uses the average consensus algorithm as described in Sec. I-C to converge toward the global averages of these two quantities. Therefore, from (19) and (20) we get

$$\begin{aligned}\mathbf{x}_c^+ &= \lim_{k \to \infty} \left(N \mathbf{V}_i^k\right)^{-1} \left(N \mathbf{v}_i^k\right) = \lim_{k \to \infty} \left(\mathbf{V}_i^k\right)^{-1} \mathbf{v}_i^k \quad (21) \\ \mathbf{J}_c^+ &= \lim_{k \to \infty} N \mathbf{V}_i^k \quad (22)\end{aligned}$$

From the discussion above, given $\mathbf{F}_i^-$, it is clear that the centralized MAP estimate in (19-20) is achieved using a distributed scheme as $k \to \infty$. We call this distributed approach of computing the MAP estimate, the Information Consensus based MAP (IC-MAP) estimation framework.

B. Computation of $\mathbf{F}_i^-$

To compute the distributed MAP estimate, node C_i needs to have knowledge of $\mathbf{F}_i^-$. In general, computation of $\mathbf{F}_i^-$ requires the knowledge of the entire covariance matrix $\boldsymbol{\mathcal{P}}$ defined in eqn. (9) (i.e. the prior covariances ($\mathbf{P}_{ii}$'s) of each node and the prior cross-covariances ($\mathbf{P}_{ij}$'s) between each pair of nodes). However, computing $\boldsymbol{\mathcal{P}}$ at every time step at each node in a distributed framework is unrealistic as it would require too much information exchange among the nodes. However, in the following, we show that for two special cases (which are of great practical importance), $\mathbf{F}_i^-$ can be computed at each node using only a node's own prior covariance matrix $\mathbf{P}_{ii}$ (or $\mathbf{J}_i^-$ in the information form). The first case is for converged priors which is an important scenario because in a consensus-based framework, with high-enough number of consensus iterations, the prior state information at all the nodes ultimately converge to the same value. The second case is when the prior state estimates across the nodes are uncorrelated to each other. This is generally true at the early steps of consensus when the nodes had no prior information about the target and initialized their prior information with random quantities.

The proposed distributed state estimation framework in Sec. V is based on these two special cases. The practical significance of these two cases can be seen from the experimental results in Sec. VI which implies that the proposed algorithm (derived from these two special cases) is robust even when the assumptions of neither of these two cases are met.

1) Case 1: Converged Priors:

Here we will discuss the case where the estimate of the state vector at each node has converged to the centralized estimate at the previous time step $t-1$. Thus at time t, the prior information at each node is the same and equal to the prior information of a centralized framework (i.e., k sufficiently large such that $\|\boldsymbol{\delta}_i^k(t)\| = 0$ for all i). This case will be of great significance when we will incorporate target dynamics and additional measurement steps to our framework. From (6) and (18) we have

$$\mathbf{J}_c^+ = \mathbf{J}_c^- + \boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{H}} = \sum_{i=1}^{N} (\mathbf{F}_i^- + \mathbf{U}_i). \quad (23)$$

From (15), $\boldsymbol{\mathcal{H}}^T \boldsymbol{\mathcal{B}} \boldsymbol{\mathcal{H}} = \sum_{i=1}^{N} \mathbf{U}_i$. Thus, from (23),

$$\sum_{i=1}^{N} \mathbf{F}_i^- = \mathbf{J}_c^- = \sum_{i=1}^{N} \frac{\mathbf{J}_c^-}{N}. \quad (24)$$

Now, for converged priors, for all i we have $\mathbf{J}_c^- = \mathbf{J}_i^-$. Using this in (24), we have

$$\sum_{i=1}^{N} \mathbf{F}_i^- = \sum_{i=1}^{N} \frac{\mathbf{J}_i^-}{N}. \quad (25)$$

Using this in (17) and (18) and using the fact that $\mathbf{x}_i^- = \mathbf{x}_c^-$ for converged priors, we have

$$\mathbf{x}_c^+ = \left(\sum_{i=1}^{N}\left(\frac{\mathbf{J}_i^-}{N} + \mathbf{U}_i\right)\right)^{-1} \sum_{i=1}^{N}\left(\frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^- + \mathbf{u}_i\right), \quad (26)$$

$$\mathbf{J}_c^+ = \sum_{i=1}^{N}\left(\frac{\mathbf{J}_i^-}{N} + \mathbf{U}_i\right). \quad (27)$$

This can be computed in a distributed manner by initializing $\mathbf{V}_i^0 = \frac{1}{N}\mathbf{J}_i^- + \mathbf{U}_i$ and $\mathbf{v}_i^0 = \frac{1}{N}\mathbf{J}_i^-\mathbf{x}_i^- + \mathbf{u}_i$. This is equivalent to using $\frac{1}{N}\mathbf{J}_i^-$ instead of $\mathbf{F}_i^-$ in (17-18).

The intuition behind this is as follows. After convergence, given the prior at one node, the other priors do not contain any new information. Upon convergence in the previous time step, the prior information matrix at each node $\mathbf{J}_i^-$ is equal to $\mathbf{J}_c^-$: Each agent has an identical copy ($\mathbf{x}_i^- = \mathbf{x}_c^-$) and amount of information ($\mathbf{J}_i^- = \mathbf{J}_c^-$). Thus at the current time step, the prior information matrix $\mathbf{J}_i^-$ should be divided by N as shown in the formulation of eqns. (26-27), so that the effective total weight of all the priors in the estimation scheme remains as $\mathbf{J}_c^-$.

2) Case 2: Uncorrelated Prior Errors:

Now, we consider the case where the errors in the priors are uncorrelated with each other across different nodes. This case can be used at the initial time step if it is known that the prior errors are uncorrelated across different nodes.

When the prior state errors are uncorrelated, the covariance matrix $\boldsymbol{\mathcal{P}}$ will be block diagonal. So, its inverse $\boldsymbol{\mathcal{F}}$ will also be block diagonal as, $\boldsymbol{\mathcal{F}} = diag(\mathbf{F}_{11}, \mathbf{F}_{22}, \dots \mathbf{F}_{NN})$ $= diag(\mathbf{P}_{11}^{-1}, \mathbf{P}_{22}^{-1}, \dots \mathbf{P}_{NN}^{-1})$. In this special case, with $\mathbf{J}_i^-$ defined in (10), we have

$$\mathbf{F}_{ii} = \mathbf{P}_{ii}^{-1} = \mathbf{J}_i^-. \quad (28)$$

As the off-diagonal block elements of $\boldsymbol{\mathcal{F}}$ are zero, from the definition of $\mathbf{F}_i^-$ in (13), we have

$$\mathbf{F}_i^- = \sum_{j=1}^{N} \mathbf{F}_{ji} = \mathbf{F}_{ii} = \mathbf{J}_i^-. \quad (29)$$

Thus, when the prior errors are uncorrelated across the different nodes, using $\mathbf{F}_i^- = \mathbf{J}_i^-$ (in (17-18)) and average consensus, we can compute the centralized MAP estimate in a distributed framework.

V. Information-Weighted Consensus Filter

In the previous section, we derived the Information Consensus based MAP (IC-MAP) estimation framework and proved its optimality for two important scenarios. In this section, we will consider a dynamic model in state space form and extend the IC-MAP framework in Sec. IV for distributed state estimation. We call this distributed state estimation framework, the *Information-Weighted Consensus Filter (ICF)*. We prove theoretically that as the number of consensus iteration $k \to \infty$, the ICF estimates converge to the estimates of the centralized Kalman filter.

Let us consider the following linear dynamical model

$$\mathbf{x}(t+1) = \boldsymbol{\Phi}\mathbf{x}(t) + \boldsymbol{\gamma}(t). \quad (30)$$

Here $\boldsymbol{\Phi} \in \mathcal{R}^{p\times p}$ is the state transition matrix and the process noise is $\boldsymbol{\gamma}(t) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$.

For this model, for the centralized case, we have the following state prediction step [17],

$$\mathbf{J}_c^-(t+1) = \left(\boldsymbol{\Phi}\mathbf{J}_c^+(t)^{-1}\boldsymbol{\Phi}^T + \mathbf{Q}\right)^{-1}, \quad (31)$$

$$\mathbf{x}_c^-(t+1) = \boldsymbol{\Phi}\mathbf{x}_c^+(t). \quad (32)$$

Combining this with the IC-MAP estimation framework proposed in Sec. IV, we get the Information-Weighted Consensus Filter (ICF) in Algorithm 1.

Algorithm 1 ICF at node C_i at time step t

Input: prior state estimate $\mathbf{x}_i^-(t)$, prior information matrix $\mathbf{J}_i^-(t)$, observation matrix $\mathbf{H}_i$, consensus speed factor ϵ and total number of consensus iterations K.

1) Get measurement $\mathbf{z}_i$ and measurement information matrix $\mathbf{B}_i$

2) Compute consensus proposals,

$$\mathbf{V}_i^0 = \frac{1}{N}\mathbf{J}_i^-(t) + \mathbf{H}_i^T\mathbf{B}_i\mathbf{H}_i \quad (33)$$

$$\mathbf{v}_i^0 = \frac{1}{N}\mathbf{J}_i^-(t)\mathbf{x}_i^-(t) + \mathbf{H}_i^T\mathbf{B}_i\mathbf{z}_i \quad (34)$$

3) Perform average consensus on $\mathbf{V}_i^0$ and $\mathbf{v}_i^0$ independently

for k = 1 to K **do**

a) Send $\mathbf{V}_i^{k-1}$ and $\mathbf{v}_i^{k-1}$ to all neighbors $j \in \mathcal{N}_i$

b) Receive $\mathbf{V}_j^{k-1}$ and $\mathbf{v}_j^{k-1}$ from all neighbors $j \in \mathcal{N}_i$

c) Update:

$$\mathbf{V}_i^k = \mathbf{V}_i^{k-1} + \epsilon \sum_{j\in\mathcal{N}_i}\left(\mathbf{V}_j^{k-1} - \mathbf{V}_i^{k-1}\right) \quad (35)$$

$$\mathbf{v}_i^k = \mathbf{v}_i^{k-1} + \epsilon \sum_{j\in\mathcal{N}_i}\left(\mathbf{v}_j^{k-1} - \mathbf{v}_i^{k-1}\right) \quad (36)$$

end for

4) Compute a posteriori state estimate and information matrix for time t

$$\mathbf{x}_i^+(t) = (\mathbf{V}_i^K)^{-1}\mathbf{v}_i^K \quad (37)$$

$$\mathbf{J}_i^+(t) = N\mathbf{V}_i^K \quad (38)$$

5) Predict for next time step $(t+1)$

$$\mathbf{J}_i^-(t+1) = \left(\boldsymbol{\Phi}(\mathbf{J}_i^+(t))^{-1}\boldsymbol{\Phi}^T + \mathbf{Q}\right)^{-1} \quad (39)$$

$$\mathbf{x}_i^-(t+1) = \boldsymbol{\Phi}\mathbf{x}_i^+(t) \quad (40)$$

Output: KF estimate $\mathbf{x}_i^+(t)$ and information matrix $\mathbf{J}_i^+(t)$.

Note that in (33) and (34), we have used the results for the converged prior case (Sec. IV-B1), i.e., using $\frac{1}{N}\mathbf{J}_i^-$ instead of $\mathbf{F}_i^-$. Theoretically, at each time step, if $k \to \infty$, the IC-MAP estimator guarantees that the priors for the next time step at each node will be equal to the optimal centralized one. This convergence further guarantees that the optimal centralized estimate will be reached at the next time steps if $\mathbf{F}_i^- = \frac{1}{N}\mathbf{J}_i^-$ is used. This guarantees the optimality of Algorithm 1 with $k \to \infty$ at each time step.

In practice, due to the fact that the total number of iterations, k, is finite, convergence will not be achieved fully. Therefore,

the distributed implementation will not perform quite as well as the centralized solution. The simulation results in Sec. VI will demonstrate that the ICF has near-optimal performance as either k increases or t increases for a fixed k.

While (21) shows that inclusion of N is inconsequential in the computation of $\mathbf{x}$ in the CKF, the role of N is critical in the distributed implementation. At (33) and (34), due to prior consensus steps, all nodes have the same estimate with the same information $\mathbf{J}_i^-$. If the $\frac{1}{N}$ is neglected, then the measurement information receives too little relative weighting by a factor of N.

A. Initialization and Special Situations

In a practical implementation scenario, at system startup or for the first few iterations in a naive node, $\mathbf{V}_i^K$ in (37) can be $\mathbf{0}$, if there is no prior or measurement information available in the local neighborhood. In this situation, a node will not perform step 4 and 5 in Algorithm 1 until it receives non-zero information from its neighbors (through step 3) or gets a measurement itself (through step 1) yielding $\mathbf{V}_i^K$ to be non-zero. In the target tracking application, this situation occurs when a new target is detected by one or more, but not all of the sensors.

In some situations prior information may be present at system startup. If the priors are known to be equal, then using the standard ICF algorithm as in Algorithm 1, should give optimal results. However, if the priors across the nodes are known to be uncorrelated at the initial time step ($t = 1$), the results for the uncorrelated case (Sec.IV-B2) should be used instead in the first time step. To do this, only at $t = 1$, instead of (33) and (34) in Algorithm 1, the following initializations should be used,

$$\mathbf{V}_i^0 = \mathbf{J}_i^- + \mathbf{H}_i^T \mathbf{B}_i \mathbf{H}_i \quad (41)$$

$$\mathbf{v}_i^0 = \mathbf{J}_i^- \mathbf{x}_i^- + \mathbf{H}_i^T \mathbf{B}_i \mathbf{z}_i. \quad (42)$$

Thus, at $t = 1$, with $k \to \infty$, the states would converge to the optimal centralized estimate. Then for $t > 1$, using Algorithm 1 would guarantee convergence to the optimal centralized estimate for the following time steps (i.e. for $t > 1$).

An example of the estimation results of different methods is shown in Fig. 2. The example includes $N = 15$ nodes and a single target. The target state contains the two dimensional position and velocity. Each line represents the state estimate at one node after each iteration. For clarity, in each experiment only the first component of the state vector for the first 3 node's is shown. State estimation is performed for 3 time steps. At each time step, 30 consensus iterations were performed. The priors were initially uncorrelated and different at each node. For ICF, at $t = 1$, the uncorrelated initializations (41-42) were used. This example shows that the ICF converges to the centralized estimate at each time step after several iterations. The reason that ICF performs better than KCF is because KCF's performance deteriorates in the presence of naivety and the cross-covariances between the priors are implicitly considered in ICF.

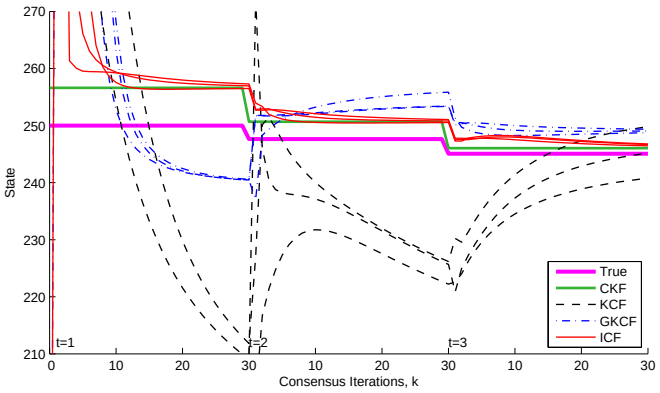

Fig. 2: An example showing the convergence of different algorithms with multiple consensus iterations at different time steps.

B. ICF, GKCF and KCF Comparison

The KCF algorithm in [4] was originally presented using a single consensus step. This section presents the state estimation step of the ICF, GKCF and KCF algorithm in an equivalent form for a single consensus step (i.e., $k = 1$) and compares the differences between the algorithms theoretically. The derivation of the following results are presented in the supplementary materials. Let us define the single step average consensus operation $\mathcal{A}$ on a variable a_i at node C_i as,

$$\mathcal{A}(a_i) = a_i + \epsilon \sum_{j \in \mathcal{N}_i} (a_j - a_i) \quad (43)$$

Also let,

$$\mathbf{S}_i = \sum_{j \in \mathcal{N}_i \cup \{i\}} \mathbf{U}_i \quad (44)$$

$$\mathbf{y}_i = \sum_{j \in \mathcal{N}_i \cup \{i\}} \mathbf{u}_i \quad (45)$$

To facilitate comparison, the estimation steps of the ICF, GKCF and KCF algorithms are presented as follows:

ICF (Asymptotically optimal with equal initial priors) (see Proposition A.1 in Appendix)

$$\mathbf{x}_i^+ = \mathbf{x}_i^- + \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right)^{-1} \left(\mathcal{A}(\mathbf{u}_i) - \mathcal{A}(\mathbf{U}_i)\mathbf{x}_i^- + \epsilon \sum_{j \in \mathcal{N}_i} \frac{\mathbf{J}_j^-}{N} \left(\mathbf{x}_j^- - \mathbf{x}_i^-\right)\right) \quad (46)$$

$$\mathbf{J}_i^+ = N\left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right) \quad (47)$$

GKCF (Suboptimal) (see Proposition A.2 in Appendix)

$$\mathbf{x}_i^+ = \mathbf{x}_i^- + \left(\mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i\right)^{-1} \left(\mathbf{y}_i - \mathbf{S}_i \mathbf{x}_i^- + \epsilon \sum_{j \in \mathcal{N}_i} \mathbf{J}_j^- \left(\mathbf{x}_j^- - \mathbf{x}_i^-\right)\right) \quad (48)$$

$$\mathbf{J}_i^+ = \mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i \quad (49)$$

KCF (Suboptimal)

$$\mathbf{x}_i^+ = \mathbf{x}_i^- + \left(\mathbf{J}_i^- + \mathbf{S}_i\right)^{-1}\left(\mathbf{y}_i - \mathbf{S}_i\mathbf{x}_i^-\right) + \frac{\epsilon}{1+||(\mathbf{J}_i^-)^{-1}||}(\mathbf{J}_i^-)^{-1}\sum_{j\in\mathcal{N}_i}\left(\mathbf{x}_j^- - \mathbf{x}_i^-\right) \quad (50)$$

$$\mathbf{J}_i^+ = \mathbf{J}_i^- + \mathbf{S}_i \quad (51)$$

The following points of comparison are important:
1) In GKCF and KCF, the terms $\mathbf{S}_i$ and $\mathbf{y}_i$ are the summation of the measurement information and the weighted measurements in the local neighborhood. This fusion mechanism, unlike average consensus, does not guarantee a proper global convergence. However, in ICF, $\mathcal{A}(\mathbf{U}_i)$ and $\mathcal{A}(\mathbf{u}_i)$ are used which guarantee convergence to the global average values.
2) In GKCF and KCF, $\mathbf{J}_i^-$ is used instead of $\frac{\mathbf{J}_i^-}{N}$ as in ICF. If the priors are uncorrelated, using $\mathbf{J}_i^-$ is appropriate for a single time step. But, as the nodes converge, which is the goal of consensus, the information becomes redundant at each node and thus dividing the prior information matrices by N is required to match the centralized solution.
3) In the information matrix update equation of the ICF, there is a multiplication factor of N. The total information in an estimate is the sum of the information matrices of the priors and the measurements. However, as the average consensus scheme gives us the *average* information in the priors and measurements, this should be multiplied by N to get the exact measure of the *total* information.
4) In the third term of (50), KCF gives equal weight to all the neighbors' priors. In the presence naivety, this has detrimental effect as the information at different nodes are different and need to be weighted by their information matrices.
5) In (46), ICF uses $\left(\frac{1}{N}\mathcal{A}(\mathbf{J}_i^-) + \mathcal{A}(\mathbf{U}_i)\right)^{-1}$ to normalize both the innovation from the measurements and the innovation from the state priors. Whereas, in (50), the normalizing terms are not balanced because KCF uses $\left(\mathbf{J}_i^- + \mathbf{S}_i\right)^{-1}$ to normalize the measurement innovation and $(\mathbf{J}_i^-)^{-1}$ to normalize the innovation from the priors.
6) In (50), the normalizing term $1 + ||(\mathbf{J}_i^-)^{-1}||$ is a design choice [4] to maintain stability of the algorithm. However, it does not guarantee optimality of KCF.

VI. Experimental Evaluation

In this section, we evaluate the performance of the proposed ICF algorithm in a simulated environment and compare it with other methods: the Centralized Kalman Filter (CKF), the Kalman Consensus Filter (KCF) [4] and the Generalized Kalman Consensus Filter (GKCF) [5]. Comparison in a simulated environment allows an in-depth analysis of the proposed algorithms as parameters are varied to measure performance under different conditions.

We simulate a camera network containing N_T targets randomly moving (with a fixed model) within a 500×500 space. Each target's initial state vector is random. A set of N camera sensors monitor the space (we consider that each communication node consists of one camera). In each experiment, the cameras are randomly distributed in the space with random orientations resulting in overlapping field-of-views (FOVs).

Target State Parameters

Each target was initialized at the center of the simulation grid. The target's state vector was a $4D$ vector, with the $2D$ position and $2D$ velocity components. The initial speed was set to 2 units per time step and with a random direction uniformly chosen from 0 to 2π. The targets evolved for 40 time steps using the target dynamical model of (30). Only the targets which remained in the simulation grid for the 40 time steps were considered. The process covariance $\mathbf{Q}$ is set to $diag(10, 10, 1, 1)$.

For the target state-estimation model, the dynamical model of (30) was also used with the same $\mathbf{\Phi}$ and $\mathbf{Q}$ defined above. The initial prior state $\mathbf{x}_i^-(1)$ and prior covariance $\mathbf{P}_i^-(1)$ is set equal at each node. A diagonal matrix is used for $\mathbf{P}_i^-(1)$ with the main diagonal elements as $\{100, 100, 10, 10\}$. The initial prior state $\mathbf{x}_i^-(1)$ is generated by adding zero-mean Gaussian noise of covariance $\mathbf{P}_i^-(1)$ to the ground truth state.

Sensor Parameters

A total of $N = 15$ nodes were generated at uniformly chosen random locations on the simulation area. The measurement vector length for each sensor is $m_i = 2$. The FOV was chosen to be equilateral triangles. We define the sensing range, SR, for each sensor to be the height of this equilateral triangle. SR was chosen to be 300 units for all the sensors. A sensor can have an observation of a target only if the ground truth position of the target is within the sensor's FOV and in that case, a measurement $\mathbf{z}_i$ was generated using the linear observation model (2) with noise covariance $\mathbf{R}_i = 100\mathbf{I}_2$. The observation matrix $\mathbf{H}_i$ and state transition matrix $\mathbf{\Phi}$ is given below.

$$\mathbf{H}_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \qquad \mathbf{\Phi} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Network Topology Parameters

A circulant graph was chosen as the communication topology for the sensor network to highlight the different issues associated with the sparsity of the communication network. The circulant graph is Δ-regular where the degree of each node is the same, say Δ. The adjacency matrix of a circulant graph is a circulant matrix. So, we denote Δ to be the degree of the communication graph $\mathcal{G}$. In the experiments, $\Delta = 2$ was used, unless stated otherwise. Note that, in this scenario, the maximum degree $\Delta_{max} = \Delta$ because the degree of all the nodes are same.

Consensus Parameters

At step 3 of Algorithm 1, each sensor communicates its measurement information to its neighbors iteratively. The maximum number of consensus iterations K was set to 5 unless stated otherwise. The consensus speed parameter was chosen to be $\epsilon = 0.65/\Delta_{max} = 0.65/\Delta$.

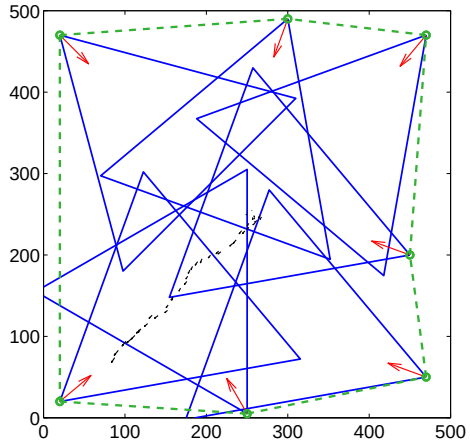

Fig. 3: In this image of an example simulation environment, the red arrows indicate the locations and orientations of the cameras. The camera FOVs are shown in blue triangles. There are 7 cameras in this example. The green dotted lines represent the network connectivity. Each black arrows depict the actual trajectory of a target moving on the grid.

Experimental Description

The parameters that were varied in the experiments are, the maximum number of consensus iterations K, communication bandwidth μ, computation unit τ, degree of the communication network Δ, sensing range SR and the number of cameras N. For each experiment, only one parameter was varied while the others were kept constant. As a measure of performance, we computed the estimation error, e, defined as the Euclidean distance between the ground truth position and the estimated posterior position. An example of the simulation framework is shown in Fig. 3.

For each experiment, 20 different simulation environments differing in camera poses were randomly generated using the method discussed above. For each environment, 20 different randomly generated target tracks were used. Thus, for each experiment, the estimation errors e, were averaged over $20 \times 20 = 400$ random scenarios, 40 time steps and over all sensors N. The mean and standard deviation of the errors for different methods are shown in the following graphs as the results of different experiments. In the graphs, each line (of a unique color) corresponds to the mean error $\bar{e}$ for one estimation method. The variation of the estimation errors over multiple simulation runs are represented by the $\bar{e} \pm 0.2\sigma$ lines drawn in dotted lines of the same color for each method.

The KCF algorithm as originally proposed in [4] uses a single consensus step per measurement update. To compare it with ICF, which supports multiple iterations, we extend KCF for multiple iterations. For this, at each time step, the measurement innovation component is set to zero for $k > 1$ and we consider only the new information provided by the neighboring nodes' estimates.

A. Experiment 1: Varying K

The objective of this experiment is to compare the performance of different estimation algorithms for different K. Here, K was varied from 1 to 20 at increments of 1. The other parameters were kept constant at their default values. The priors were chosen to be equal.

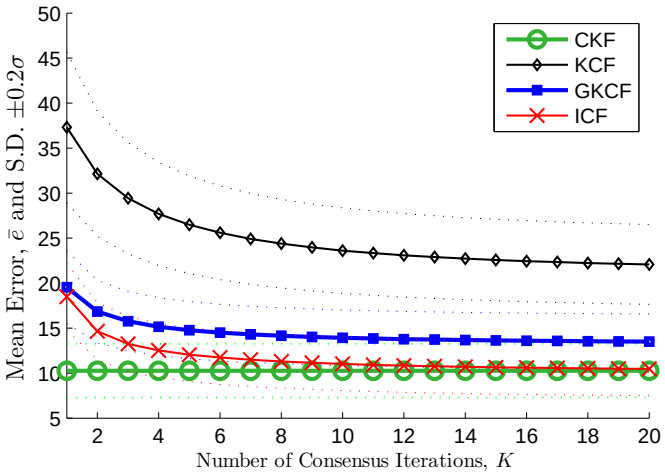

Fig. 4: Equal Priors

The results of this experiment are shown in Fig. 4. The graph shows that for $K = 1$, ICF performs much better than KCF and GKCF performs close to ICF. As, the number of iterations K is increased, the mean error for ICF decreases and approaches the performance of the centralized method at about $K = 10$. The main reason for this difference in performance is that KCF and GKCF do not account for the redundancy in the prior information across different nodes, but ICF does. In this simulation all the initial priors were equal, thus the information in the priors were completely redundant. KCF and GKCF by not taking into account redundancy in priors, gives more weight to prior information and less weight to the measurement information than the optimal weights.

ICF is guaranteed to converge to the optimal centralized estimated if the initial priors are equal. However, to show the robustness of the ICF approach, we conducted the same experiment with unequal and uncorrelated priors (Fig. 5) and with unequal and correlated (with $\rho = 0.5$ correlation-coefficient between the priors across nodes) (Fig. 6) using Algorithm 1. The results show that ICF achieves near-optimal performance even when the optimality constraints are not met. This is because ICF is a consensus based approach and irrespective of the initial condition, after several time steps or consensus iterations, the priors converge. ICF was proved to be optimal with converged priors. Thus, after a few time steps it achieves near-optimal performance as the system approaches the optimality conditions.

B. Experiment 2: Varying μ and τ

The objective of this experiment is to compare the performance of different algorithms at different amounts of communication bandwidth and computational resources. We define the bandwidth, μ, of each method to be the total number of scalars sent at each consensus iteration from a node to each neighbor. As, covariance/information matrices are symmetric, only sending the upper/lower triangular portion of the matrix suffices. Using standard convention, the approximate

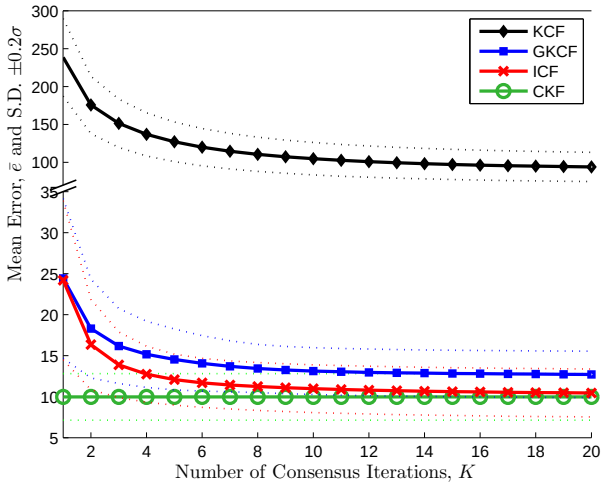

Fig. 5: Uncorrelated Priors

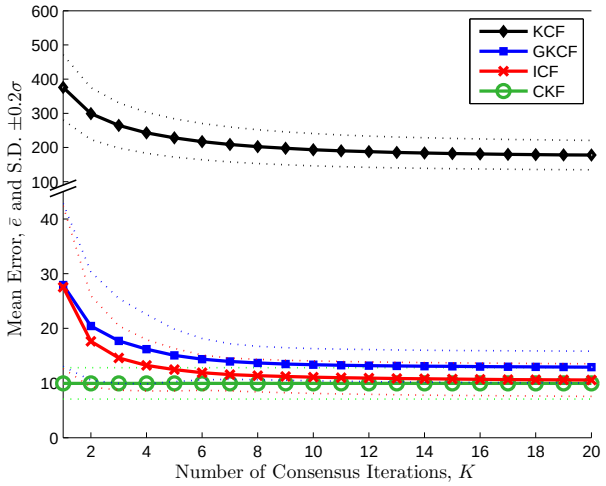

Fig. 6: Correlated Priors ($\rho = .5$)

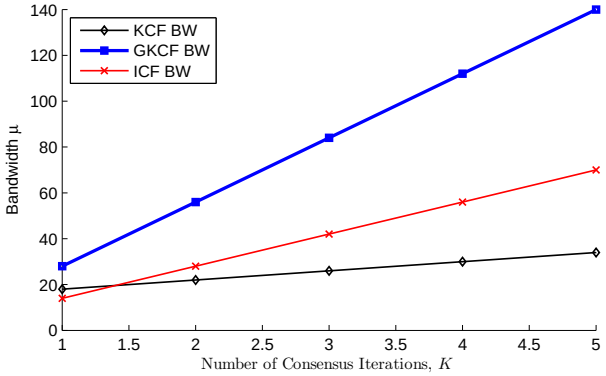

Fig. 7: Bandwidth μ vs. number of consensus iterations K for different approaches.

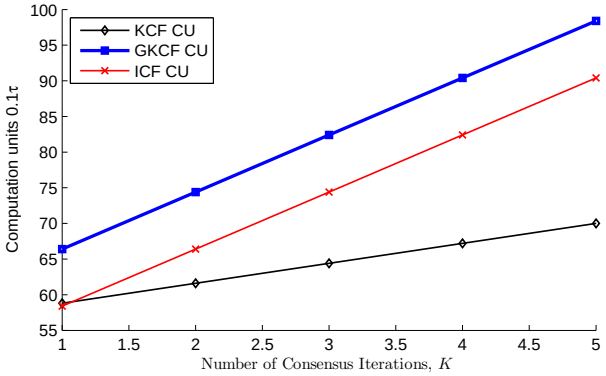

Fig. 8: Computation unit τ vs. number of consensus iterations K for different approaches.

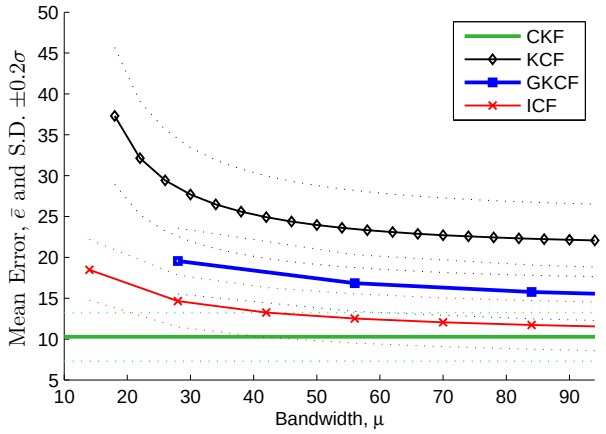

Fig. 9: Performance comparison between different approaches as a function of the bandwidth μ, showing that the ICF performs better than other distributed approaches for each μ.

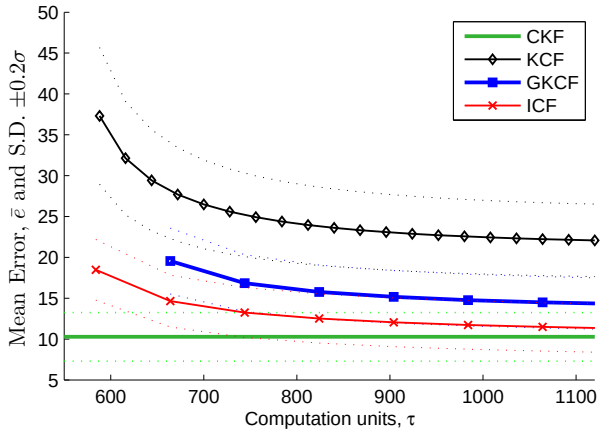

Fig. 10: Performance comparison between different approaches as a function of the computational unit τ, showing that the ICF performs better than other distributed approaches at each τ.

communication data requirement μ and computation cycle requirement τ was computed for $p = 4$ and $m = 2$.

Figs. 7 and 8 show the bandwidth and computational requirements for different algorithms for different numbers of consensus iterations k. For any given k, ICF always requires half the bandwidth of GKCF. The KCF has lower communication and computational requirements.

Figs. 9 and 10 show the performance achieved by each method for different amounts of bandwidth and computational requirements. At any given bandwidth μ or computation unit τ, the ICF performs better than the other distributed methods and the performance approaches that of the CKF with increased resources.

C. Experiment 3: Varying Δ

The objective of this experiment is to compare the performance of different approaches for different values of the degree Δ (i.e., vary the sparsity of the network from sparse connectivity to dense connectivity). For $N = 15$, the maximum possible degree can be $\Delta = 14$ at full mesh connectivity. In this experiment, Δ was varied from 2 to 14 at increments of 2.

The results are shown in Figs 11, where the total number of consensus iterations K was set to 5. It can be seen that the ICF performs better than the other distributed algorithms at all Δ and almost approaches the centralized performance at $\Delta = 4$. For full connectivity, i.e. $\Delta = 14$, where $\mathcal{G}$ is a complete graph, all the distributed methods achieve centralized performance.

D. Experiment 4: Varying SR

The objective of this experiment is to compare the performance of different approaches as a function of the sensor range SR (i.e., varying the area of coverage of each sensor.)

Fig. 12 shows the performance of each approach as SR is varied and the total number of iterations K was set to 5. It can be seen that ICF performed better than the other distributed algorithms at each sensor range.

E. Experiment 5: Varying N

The objective of this experiment is to compare the performance of different approaches as a function of the total number of sensors, N, as it was varied from 5 to 31 at increments of 2. Values less than 5 were not used because at such low number of sensors, even the centralized algorithm cannot perform well due to the high number of time instants at which the number of measurements is insufficient for the state vector to be observable, due to the low percentage coverage of the environment.

Fig. 13 shows results for variable N and fixed $K = 20$. As the number of sensors is increased, the observability of the targets by the network increases, but the number of naive nodes also increases. Due to the amount of information about the targets increasing, the performance of all the approaches (including centralized) improves. For a small N, all the distributed methods performed close to the centralized method, because the number of naive nodes is small and the network distance from a naive node to an informed node is small. As the number of nodes increases, the number of consensus iterations required to reach consensus also increases. Thus we can see that the performance of ICF deviates from the centralized performance for high number of sensors. For all N, ICF outperformed the alternative distributed methods.

F. Experiment 6: Arbitrary Communication Graph

In the previous experiments, for the ease of varying different parameters, we assumed the communication graph to be balanced. To show that ICF is applicable for any connected graph, we conduct experiments on random connected graphs for $N = 15$. Edges were added between five random pairs of nodes on the balanced graph which was used on the previous examples. The result of this experiment is shown in Fig. 14. It depicts the fact that as $K \to \infty$, for any random connected graph, ICF converges to the centralized solution.

G. Experiment 7: Full Observability

Limited local observability (i.e., naive nodes) was one of the motivations for the derivation of the ICF algorithm. To show that ICF is generally applicable even in scenarios with without locally unobservable states, in this experiment, we assume that each node can observe all the targets at all times. From the results in Fig. 15 it can be seen that still ICF outperforms the other methods for most cases and achieves the optimal performance as $K \to \infty$.

H. Experiment 8: Stability Analysis

In this experiment, we experimentally verify the stability of the ICF approach. In Sec. I-C, we have mentioned that the average consensus algorithm is stable if the consensus rate parameter ϵ is chosen between 0 and $\frac{1}{\Delta_{max}}$. The stability of the ICF algorithm is directly related to the stability of the average consensus algorithm. We set $\epsilon = \frac{\alpha}{\Delta_{max}}$ and vary α to perform the stability analysis. From the results in Fig. 16, it can be seen that ICF is stable for $0 < \alpha < 1$ (i.e., $0 < \epsilon < \frac{1}{\Delta_{max}}$), which is the same stability conditions for average consensus algorithm.

I. Experiment 9: Robustness to inaccurate knowledge of N

Unlike CKF, KCF and GKCF; ICF requires the knowledge of the total number of nodes N. Total number of nodes N, can be computed in a distributed framework [10]. In case of node failure or inclusion of new nodes to the system, each agent might have a wrong estimate of N, say $N + \Delta N$. In this experiment, to test the sensitivity of ICF to the error in the value of N, ΔN is varied from $-N+1$ to N. The actual value of N for this experiment is 31. Total number of iterations, K was set to 100. The results are shown in Fig. 17. It shows that ICF is highly tolerant to discrepancies between the actual N and the estimated N and performs better than other methods for most cases.

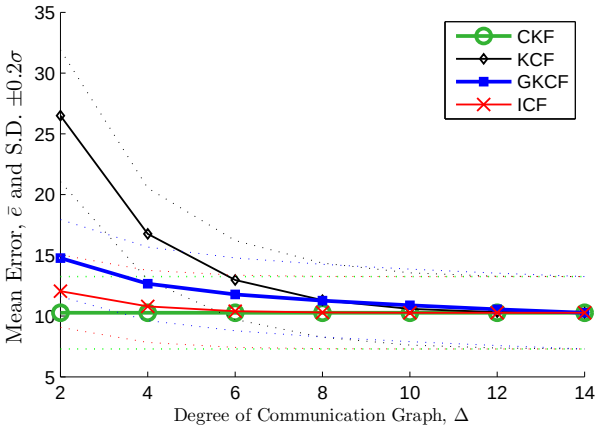

Fig. 11: Performance comparison of different approaches as a function of the degree of communication graph, Δ. Increase in network connectivity increases the performance of all methods.

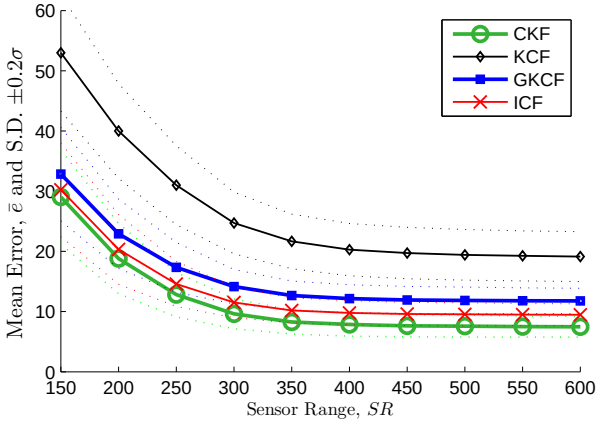

Fig. 12: Performance comparison of different approaches by varying sensor range, SR. As the sensor range increases, the network gets more measurement information and has fewer naive nodes increasing overall performance of all methods (including centralized).

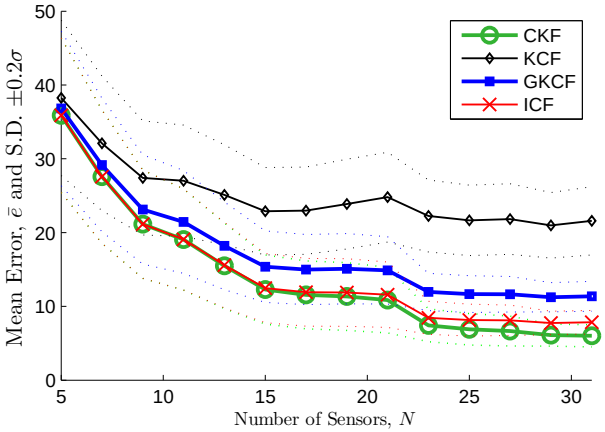

Fig. 13: Performance comparison of different approaches by varying total number of sensors, N.

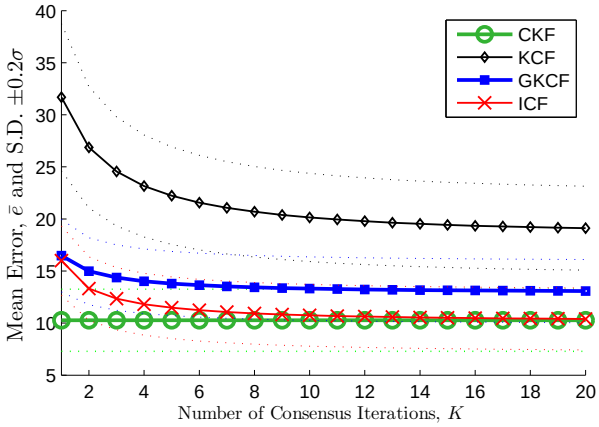

Fig. 14: Performance comparison for imbalanced communication graph.

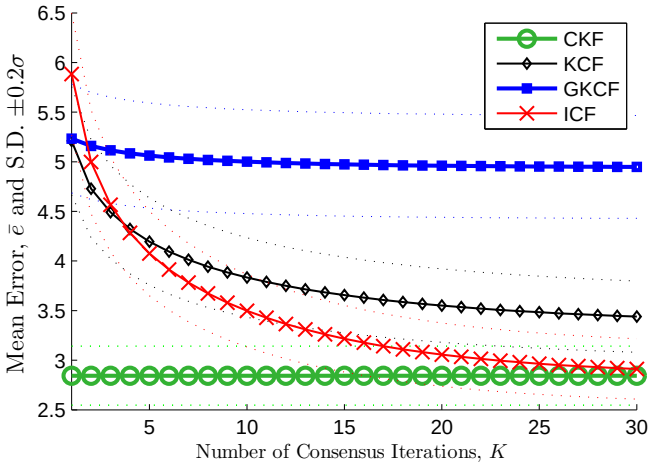

Fig. 15: Performance comparison under unlimited observability.

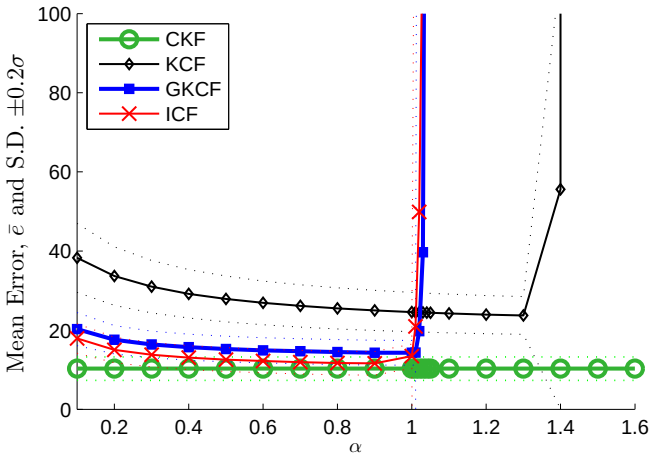

Fig. 16: Stability analysis.

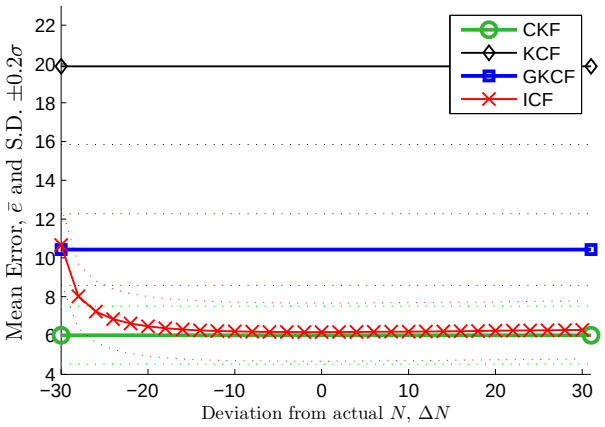

Fig. 17: Robustness to inaccurate knowledge of N.

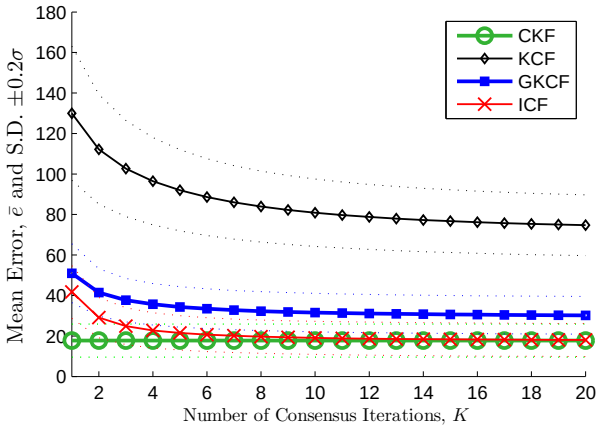

Fig. 18: Robustness to model assumption error.

J. Experiment 10: Robustness to non-linear state propagation

Finally, it is natural to ask whether the performance demonstrated in the previous experiments was highly dependent on the fact that the estimator incorporated the correct model.

In this final experiment, the ground truth tracks were generated for $t = 1$ to 40 using the following non-linear model of (52).

$$\mathbf{x}(t) = 200e^{-\lambda(t-1)} \begin{bmatrix} \cos(\omega t) \\ \sin(\omega t) \\ -\lambda\cos(\omega t) - \omega\cos(\omega t) \\ -\lambda\cos(\omega t) + \omega\cos(\omega t) \end{bmatrix} + \begin{bmatrix} 250 \\ 250 \\ 0 \\ 0 \end{bmatrix} \quad (52)$$

A total of 20 tracks were generated by varying $\omega = \frac{\pi}{80}$ to $\frac{4\pi}{80}$ with increments of $\frac{\pi}{80}$ and $\lambda = 0.02$ to 0.1 with increments of 0.02. The rest of the simulation settings were kept unchanged (similar to that of Experiment 1). Thus, in the estimation step, the linear dynamical model of (30) was used. Fig. 18 shows the performance of different algorithms for this simulation. Comparing with Fig. 4 it can be seen that the performance of all the algorithms deteriorated (including CKF) as one would expect. However, the relative performance of the different algorithms are similar. This shows that the ICF and the performance comparison results are quite robust, even when the assumed dynamical model does not match the true state propagation.

VII. Conclusion

In this paper we have presented a novel distributed state estimation framework called the Information-weighted Consensus Filter (ICF) which is generally applicable to almost any connected sensor network, converges to the optimal centralized estimate under reasonable conditions, does not require target hand-off protocols, and requires computation and communication resource similar to or less than alternative algorithms. The development of this algorithm was motivated by applications in camera networks wherein the observability properties of the state by the nodes is distinct across the nodes. We compared ICF with the state-of-the-art distributed estimation methods such as KCF and GKCF, both theoretically and through simulations. Our simulation results show that ICF outperforms these methods. The results also indicate that ICF is robust to situations where optimality conditions are not met. Future directions of this work might involve the incorporation of data association strategies to ICF to handle multiple targets.

References

[1] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[2] Juan A. Garay, Shay Kutten, and David Peleg. A sub-linear time distributed algorithm for minimum-weight spanning trees. In *SIAM Journal on Computing*, pages 659–668, 1998.

[3] Reza Olfati-Saber, J. Alex Fax, and Richard M. Murray. Consensus and cooperation in networked multi-agent systems. In *Proceedings of the IEEE*, 2007.

[4] R. Olfati-Saber. Kalman-consensus filter: Optimality, stability, and performance. In *IEEE Conference on Decision and Control*, pages 7036 –7042, Dec. 2009.

[5] A. T. Kamal, C. Ding, B. Song, J. A. Farrell, and A. K. Roy-Chowdhury. A generalized Kalman consensus filter for wide area video networks. In *IEEE Conference on Decision and Control*, pages 7863 –7869, Dec. 2011.

[6] R. Tron and R. Vidal. Distributed computer vision algorithms. *Signal Processing Magazine, IEEE*, 28(3):32 –45, May 2011.

[7] Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. Collection tree protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys, pages 1–14, 2009.

[8] Reza Olfati-Saber, Elisa Franco, Emilio Frazzoli, and Jeff S. Shamma. Belief consensus and distributed hypothesis testing in sensor networks. In *Network Embedded Sensing and Control*, pages 169–182. Springer Verlag, 2006.

[9] A. T. Kamal, J. A Farrell, and A. K. Roy-Chowdhury. Information weighted consensus. In *IEEE Conference on Decision and Control*, Dec. 2012.

[10] Federica Garin and Luca Schenato. A Survey on Distributed Estimation and Control Applications Using Linear Consensus Algorithms. In *Networked Control Systems*, volume 406 of *Lecture Notes in Control and Information Sciences*, pages 75–107. Springer, 2011.

[11] Bi Song, A. T. Kamal, C. Soto, Chong Ding, J. A. Farrell, and A. K. Roy-Chowdhury. Tracking and activity recognition through consensus in distributed camera networks. *IEEE Trans. on Image Processing*, 19(10):2564 –2579, Oct. 2010.

[12] A. T. Kamal, Bi Song, and A. K. Roy-Chowdhury. Belief consensus for distributed action recognition. In *Intl. Conf. on Image Processing*, pages 141 –144, Sept. 2011.

[13] Bi Song, Chong Ding, A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury. Distributed camera networks: Integrated sensing and analysis for wide area scene understanding. *IEEE Signal Processing Magazine*, 28(3):20 –31, May 2011.

[14] Chong Ding, Bi Song, Akshay Morye, Jay A. Farrell, and Amit K. Roy-Chowdhury. Collaborative sensing in a distributed PTZ camera network. *IEEE Trans. on Image Processing*, 21(7):3282–3295, 2012.

[15] R. Tron and R. Vidal. Distributed image-based 3-D localization of camera sensor networks. In *IEEE Conference on Decision and Control*, pages 901 –908, Dec. 2009.
[16] D. Borra, E. Lovisari, R. Carli, F. Fagnani, and S. Zampieri. Autonomous calibration algorithms for networks of cameras. In *American Control Conference*, pages 5126 –5131, June 2012.
[17] Steven M. Kay. *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

APPENDIX

Proposition A.1. *The state estimate of ICF using a single step of consensus iteration can be expressed as*

$$\begin{aligned}\mathbf{x}_i^+ &= \mathbf{x}_i^- + \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right)^{-1} \\ &\quad \left(\mathcal{A}(\mathbf{u}_i) - \mathcal{A}(\mathbf{U}_i)\mathbf{x}_i^- + \epsilon \sum_{j\in\mathcal{N}_i} \frac{\mathbf{J}_j^-}{N}\left(\mathbf{x}_j^- - \mathbf{x}_i^-\right)\right) \quad (53)\\ \mathbf{J}_i^+ &= N\left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right) \quad (54)\end{aligned}$$

Proof: Using the shorthand notation $\mathcal{A}()$ for a single step of consensus in (26), for a single step we can write,

$$\begin{aligned}\mathbf{x}_i^+ &= \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right)^{-1} \\ &\quad \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^-\right) + \mathcal{A}(\mathbf{u}_i)\right) \quad (55)\end{aligned}$$

By adding and subtracting $\mathbf{x}_i^-$ in RHS of (55), we get,

$$\begin{aligned}\mathbf{x}_i^+ &= \mathbf{x}_i^- + \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}\left(\mathbf{U}_i\right)\right)^{-1} \\ &\quad \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^-\right) + \mathcal{A}(\mathbf{u}_i) - \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right)\mathbf{x}_i^-\right) \\ &= \mathbf{x}_i^- + \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right)^{-1} \\ &\quad \left(\mathcal{A}(\mathbf{u}_i) - \mathcal{A}(\mathbf{U}_i)\mathbf{x}_i^- + \mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^-\right) - \mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right)\mathbf{x}_i^-\right) \quad (56)\end{aligned}$$

$$\begin{aligned}\text{Now, } &\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^-\right) - \mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right)\mathbf{x}_i^- \\ &= \frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^- + \epsilon\sum_{j\in\mathcal{N}_i}\left(\frac{\mathbf{J}_j^-}{N}\mathbf{x}_j^- - \frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^-\right) - \frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^- \\ &\quad -\epsilon\sum_{j\in\mathcal{N}_i}\left(\frac{\mathbf{J}_j^-}{N}\mathbf{x}_i^- - \frac{\mathbf{J}_i^-}{N}\mathbf{x}_i^-\right) \quad (57)\\ &= \epsilon\sum_{j\in\mathcal{N}_i}\frac{\mathbf{J}_j^-}{N}\left(\mathbf{x}_j^- - \mathbf{x}_i^-\right) \quad (58)\end{aligned}$$

Using this result in (56), we get,

$$\begin{aligned}\mathbf{x}_i^+ &= \mathbf{x}_i^- + \left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right)^{-1} \\ &\quad \left(\mathcal{A}(\mathbf{u}_i) - \mathcal{A}(\mathbf{U}_i)\mathbf{x}_i^- + \epsilon\sum_{j\in\mathcal{N}_i}\frac{\mathbf{J}_j^-}{N}\left(\mathbf{x}_j^- - \mathbf{x}_i^-\right)\right) \quad (59)\end{aligned}$$

Similarly, $\mathbf{J}_i^+$ can be written as

$$\mathbf{J}_i^+ = N\left(\mathcal{A}\left(\frac{\mathbf{J}_i^-}{N}\right) + \mathcal{A}(\mathbf{U}_i)\right) \quad (60)$$

■

Proposition A.2. *The state estimate of GKCF using a single step of consensus iteration can be expressed as*

$$\begin{aligned}\mathbf{x}_i^+ &= \mathbf{x}_i^- + \left(\mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i\right)^{-1} \\ &\quad \left(\mathbf{y}_i - \mathbf{S}_i\mathbf{x}_i^- + \epsilon\sum_{j\in\mathcal{N}_i}\mathbf{J}_j^-\left(\mathbf{x}_j^- - \mathbf{x}_i^-\right)\right) \quad (61)\\ \mathbf{J}_i^+ &= \mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i \quad (62)\end{aligned}$$

Proof: In the GKCF algorithm, after incorporating neighbors priors with a single step consensus we get,

$$\mathbf{x}_i(k+1) = \left(\mathcal{A}(\mathbf{J}_i^-)\right)^{-1}\mathcal{A}(\mathbf{J}_i^-\mathbf{x}_i^-) \quad (63)$$

After incorporating measurement information and adding and subtracting $\mathbf{x}_i^-$ ($\mathbf{S}_i$ and $\mathbf{y}_i$ are defined in (44-45)):

$$\begin{aligned}\mathbf{x}_i^+ &= \mathbf{x}_i^- - \mathbf{x}_i^- + \mathbf{x}_i(k+1) + \left(\mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i\right)^{-1} \\ &\quad \left(\mathbf{y}_i - \mathbf{S}_i\mathbf{x}_i(k+1)\right) \quad (64)\\ &= \mathbf{x}_i^- + \left(\mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i\right)^{-1} \\ &\quad \left(\mathbf{y}_i + \mathcal{A}(\mathbf{J}_i^-)\mathbf{x}_i(k+1) - \mathcal{A}(\mathbf{J}_i^-)\mathbf{x}_i^- - \mathbf{S}_i\mathbf{x}_i^-\right) \\ &= \mathbf{x}_i^- + \left(\mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i\right)^{-1} \\ &\quad \left(\mathbf{y}_i - \mathbf{S}_i\mathbf{x}_i^- + \mathcal{A}(\mathbf{J}_i^-\mathbf{x}_i^-) - \mathcal{A}(\mathbf{J}_i^-)\mathbf{x}_i^-\right) \quad (65)\\ &= \mathbf{x}_i^- + \left(\mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i\right)^{-1} \\ &\quad \left(\mathbf{y}_i - \mathbf{S}_i\mathbf{x}_i^- + \epsilon\sum_{j\in\mathcal{N}_i}\mathbf{J}_j^-\left(\mathbf{x}_j^- - \mathbf{x}_i^-\right)\right) \quad (66)\end{aligned}$$

To get (65), the relation in (63) was used. To get (66), the relation

$$\mathcal{A}(\mathbf{J}_i^-\mathbf{x}_i^-) - \mathcal{A}(\mathbf{J}_i^-)\mathbf{x}_i^- = \epsilon\sum_{j\in\mathcal{N}_i}\mathbf{J}_j^-\left(\mathbf{x}_j^- - \mathbf{x}_i^-\right). \quad (67)$$

was used which can be derived in a similar way (58) was derived. Similarly, $\mathbf{J}_i^+$ can be written as

$$\mathbf{J}_i^+ = \mathcal{A}(\mathbf{J}_i^-) + \mathbf{S}_i. \quad (68)$$

■

Ahmed T. Kamal received the B.S. degree in Electrical and Electronic Engineering from the Bangladesh University of Engineering and Technology, Dhaka in 2008. He received the M.S. degree in Electrical Engineering from the University of California, Riverside in 2010. He is currently a Ph.D. candidate in the Department of Electrical Engineering in the same University. His main research interests include intelligent camera networks, widearea scene analysis, activity recognition and search and distributed information fusion.

Jay A. Farrell earned B.S. degrees in physics and electrical engineering from Iowa State University, and M.S. and Ph.D. degrees in electrical engineering from the University of Notre Dame. At Charles Stark Draper Lab (1989-1994), he received the Engineering Vice President's Best Technical Publication Award in 1990, and Recognition Awards for Outstanding Performance and Achievement in 1991 and 1993. He is a Professor and two time Chair of the Department of Electrical Engineering at the University of California, Riverside. He has served the IEEE Control Systems Society (CSS) as Finance Chair for three IEEE CDC's ('95, '01, and '03), on the Board of Governors for two terms ('03-'06, '12-'14), as Vice President Finance and Vice President of Technical Activities, as CSS General Vice Chair of IEEE CDC-ECC 2011, as General Chair of IEEE CDC 2012, and will serve as President in 2014. He was named a GNSS Leader to Watch for 2009-2010 by GPS World Magazine in May 2009 and a winner of the Connected Vehicle Technology Challenge by the U.S. Department of Transportation's (DOT's) Research and Innovative Technology Administration in July 2011. He is a Fellow of the IEEE, a Fellow of AAAS, a Distinguished Member of IEEE CSS, and author of over 200 technical publications. He is author of the book "Aided Navigation: GPS with High Rate Sensors" (McGraw-Hill 2008). He is also co-author of the books "The Global Positioning System and Inertial Navigation" (McGraw-Hill, 1998) and "Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches" (John Wiley 2006).

Amit K. Roy-Chowdhury received the Bachelors degree in electrical engineering from Jadavpur University, Calcutta, India, the Masters degree in systems science and automation from the Indian Institute of Science, Bangalore, India, and the Ph.D. degree in electrical engineering from the University of Maryland, College Park.

He is a Professor of electrical engineering and a Cooperating Faculty in the Department of Computer Science, University of California, Riverside. His broad research interests include the areas of image processing and analysis, computer vision, and video communications and statistical methods for signal analysis. His current research projects include intelligent camera networks, wide-area scene analysis, motion analysis in video, activity recognition and search, video-based biometrics (face and gait), biological video analysis, and distributed video compression. He is a coauthor of two books Camera Networks: The Acquisition and Analysis of Videos over Wide Areas and Recognition of Humans and Their Activities Using Video. He is the editor of the book Distributed Video Sensor Networks. He has been on the organizing and program committees of multiple computer vision and image processing conferences and is serving on the editorial boards of multiple journals.