Consensus-Based Distributed Mixture Kalman Filter for Maneuvering Target Tracking in Wireless Sensor Networks

Yihua Yu, *Member, IEEE*

***Abstract*—We consider the distributed state estimation for conditional dynamic linear systems (CDLSs) in a wireless sensor network without a data fusion center. Each sensor only exchanges its local information with its neighbors. The mixture Kalman filter (MKF) is an effective technique for the state estimation in the CDLSs. We extend the MKF and develop a distributed MKF for the CDLSs. Each sensor in the network runs a local MKF and interacts with other sensors to generate a mixture Gaussian representation of the global posterior distribution. Since the global likelihood in the CDLSs is not suitable for the distributed computation, we apply the cubature rule for the calculation of likelihood, to enable the distributed implementation of the likelihood computation. Consensus algorithms are executed to fuse the observations from all sensors. For the distributed MKF, the number of particles can be significantly reduced, compared with the distributed particle filtering. Finally, we apply the distributed MKF to the maneuvering target tracking. We consider that the maneuvering variable is discrete and continuous. We also consider that the observation equation is linear and nonlinear. Simulation results illustrate that the performance of the distributed MKF is close to that of the centralized MKF in various scenarios of maneuvering target tracking.**

***Index Terms*—Consensus algorithm, distributed state estimation, maneuvering target tracking, mixture Kalman filter (MKF), wireless sensor network (WSN).**

I. Introduction

WIRELESS sensor networks (WSNs) [1], [2] have attracted much attention in various areas, including target localization and tracking, habitat monitoring, and environment surveillance (e.g., [3]–[6]). A typical WSN consists of a number of battery-powered sensors that use wireless transmission to communicate with the rest of the network. They cooperatively estimate the states of targets based on their local observations. The sensors pose limited sensing and communication range. They also have limited energy and computational capabilities.

In a centralized solution of the WSNs [1], all sensors send their raw observations either directly or via a multihop relay to a data fusion center (FC). The FC is responsible for the state estimation. This strategy requires routing protocols about the network, which is extremely challenging in many applications, particularly when the network topology changes over time, and it is less robust to the unreliable wireless network conditions, such as sensor node failure and communication bottleneck near the FC.

An alternative solution is distributed estimation (e.g., [7]–[10]), where there is no central FC. Each sensor only exchanges its local information with its neighbors and carries out local computation. It is more robust to the unreliable wireless network. The final estimation result can be available at each sensor. The most fundamental distributed technique is the consensus-based estimation regarding the fusion of local observations. Some distributed algorithms, such as consensus algorithms [11], [12], are performed to reach an agreement regarding quantities of interest that depend on the local observations of all sensors. It has been shown that average consensus can be reached by performing a linear iteration in a distributed fashion [11], [12].

We focus on the network of distributed sensors, where no FC is available. The sensors communicate only with their neighbors. The consensus algorithms are utilized for the distributed implementation of global state estimation. Among the consensus-based distributed approaches, distributed Kalman filters (KFs) have been widely explored for the linear and Gaussian systems [13]–[15]. However, many practical systems, such as target localization and tracking, include nonlinear and non-Gaussian elements. The consensus-based distributed particle filters (PFs) [16] are a natural choice for the distributed state estimation of the nonlinear/non-Gaussian systems. The PFs [17], [18] have been proven to be an effective solution to nonlinear/non-Gaussian problems. They form the basis for many Bayesian tracking algorithms.

The distributed PFs are executed by a set of sensors. Each sensor performs a local PF and interacts with other sensors to generate a particle representation of the global posterior distribution. Different consensus-based distributed PFs have been proposed in [19]–[25]. The method in [21] presupposes that the random-number generators are synchronized in the network. The identical sets of particles are generated at each sensor. For each particle, one consensus algorithm is executed to compute the average of local weights, which fuses the observations from all sensors. The methods in [22] and [24] utilize the consensus algorithm to compute an approximation of the global likelihood function at each sensor. The likelihood approximation involves the observations from all sensors. Each sensor independently updates the weights of particles from the

Manuscript received February 11, 2015; revised June 17, 2015, September 20, 2015, and November 29, 2015; accepted December 1, 2015. Date of publication December 17, 2015; date of current version October 13, 2016. The review of this paper was coordinated by Prof. L. Guo.

The author is with the School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: yuyihua2004@163.com).

Digital Object Identifier 10.1109/TVT.2015.2508456

likelihood approximation. A detailed overview of distributed PFs can be found in [16]. Consensus-based distributed PFs are robust to the unreliable wireless network and sensor node failure. They do not require routing protocols about the network and, thus, are suitable for networks with mobile agents. They only require local communications between neighbors. Each sensor has the global estimate. A disadvantage is the (generally) higher communication requirements [16]. The number of consensus iterations increases with the size of the WSNs.

In this paper, we consider conditional dynamic linear systems (CDLSs) [26], which are special types of state-space models. The CDLSs frequently appear in many applications, such as wireless communication and maneuvering target tracking (e.g., [27] and [28]). In the CDLSs, the state variable consists of a component that is conditionally linear and a component that is nonlinear. When the value of the nonlinear component is given, the system is linear and Gaussian about the linear component and the observations. The nonlinear component can be numerically approximated using a PF, and the linear component is analytically handled through KF. This technique is called the mixture Kalman filter (MKF) [26], which is also known as the Rao–Blackwellized PF or the marginalized PF [18]. The MKF reduces the size of state space of the state variable by analytically integrating out the linear component and then increases the efficiency of sampling.

The main contribution of this paper is that we extend the MKF and develop a distributed MKF for the CDLSs in WSNs. In the distributed MKF, each sensor runs a local MKF and interacts with other sensors to generate a mixture Gaussian representation of the global posterior distribution. Since the global likelihood function in the CDLSs is not suitable for the distributed computation, we first apply the cubature rule [29], [30] for the calculation of likelihood, to enable the distributed implementation of the likelihood computation. The global likelihood in the CDLSs can be expressed as an integral, where the integrand is in the form of *nonlinear function* $\times$ *Gaussian density*. The cubature rule can provide an accurate computational result for this type of integral with limited computational complexity. Consensus algorithms are executed to fuse the observations from all sensors when calculating the likelihood at each sensor. Since the distributed MKF reduces the size of state space of the state variable by analytically integrating out the linear component, it can significantly reduce the number of particles compared with the distributed PF and decreases the computational complexity. The posterior distribution of the linear component of the state variable can also be calculated by a distributed scheme at each sensor. Finally, we apply the distributed MKF to the maneuvering target tracking [28]. We consider that the observation equation is linear and nonlinear. We consider that the maneuvering model is a constant-turn model with a known turn rate [28], where the turn rate is a discrete variable. We also consider that the maneuvering model is a constant-turn model with an unknown turn rate [28], where the turn rate is a continuous variable. Simulation results illustrate that the performance of the distributed MKF is close to that of the centralized MKF in various scenarios of maneuvering target tracking. A few works on distributed MKF have been reported in [31], where a local MKF is executed at each sensor, and their estimates are fused with the consensus algorithm. The distributed MKF in [31] is suboptimal, and it cannot approach the performance of the centralized MKF.

The rest of this paper is organized as follows. Section II describes the CDLSs under consideration. Section III briefly reviews the centralized MKF. In Section IV, we develop the distributed MKF for the CDLSs. Applications of the distributed MKF to maneuvering target tracking are provided in Section V, and conclusions are drawn in Section VI.

II. Problem Statement

Consider a set of N sensors spatially distributed over some region. The interaction topology between sensors is represented using an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the set of nodes $\mathcal{V} = \{1, \ldots, N\}$ and the set of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. Each edge $(i, j) \in \mathcal{E}$ is an unordered pair of distinct nodes. The set of neighbors of node i is denoted by $\mathcal{N}_i = \{j | (i, j) \in \mathcal{E}\}$. A graph is connected if, for any two nodes i and j, there exists a sequence of edges $(i, k_1), (k_1, k_2), \ldots, (k_{s-1}, k_s), (k_s, j)$ in $\mathcal{E}$.

The discrete-time CDLS in the WSN is described as

$$\boldsymbol{r}_t \sim p(\boldsymbol{r}_t | \boldsymbol{r}_{t-1}) \tag{1}$$

$$\boldsymbol{x}_t = \boldsymbol{A}_t(\boldsymbol{r}_t)\boldsymbol{x}_{t-1} + \boldsymbol{u}_t(\boldsymbol{r}_t) \tag{2}$$

$$\boldsymbol{y}_{n,t} = \boldsymbol{B}_{n,t}(\boldsymbol{r}_t)\boldsymbol{x}_t + \boldsymbol{v}_{n,t}(\boldsymbol{r}_t), \quad n = 1, \ldots, N \tag{3}$$

where $\boldsymbol{x}_t \in \mathbf{R}^{d_x}$ and $\boldsymbol{r}_t \in \mathbf{R}^{d_r}$ are the state variables, and $\boldsymbol{y}_{n,t} \in \mathbf{R}^{d_{n,y}}$ is the observation of the nth sensor at time t. It is assumed that the state noise $\boldsymbol{u}_t(\boldsymbol{r}_t)$ and the observation noise $\boldsymbol{v}_{n,t}(\boldsymbol{r}_t)$ are mutually uncorrelated when given $\boldsymbol{r}_t$ and are zero-mean Gaussian processes with covariance matrices $\boldsymbol{Q}_t(\boldsymbol{r}_t)$ and $\boldsymbol{R}_{n,t}(\boldsymbol{r}_t)$, respectively. When $\boldsymbol{r}_t$ is given, the coefficient matrices $\boldsymbol{A}_t(\boldsymbol{r}_t)$ and $\boldsymbol{B}_{n,t}(\boldsymbol{r}_t)$ are known, and systems (2) and (3) are linear and Gaussian for $\boldsymbol{x}_t$ and $\boldsymbol{y}_{n,t}$.

At any discrete time t, the sensors cooperatively estimate states $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$ based on their local observations. The aggregate observation of all sensors is described as

$$\boldsymbol{y}_t = \boldsymbol{B}_t(\boldsymbol{r}_t)\boldsymbol{x}_t + \boldsymbol{v}_t(\boldsymbol{r}_t) \tag{4}$$

where $\boldsymbol{y}_t \triangleq (\boldsymbol{y}_{1,t}^T, \ldots, \boldsymbol{y}_{N,t}^T)^T$, $\boldsymbol{B}_t(\boldsymbol{r}_t) \triangleq (\boldsymbol{B}_{1,t}^T(\boldsymbol{r}_t), \ldots, \boldsymbol{B}_{N,t}^T(\boldsymbol{r}_t))^T$, $\boldsymbol{v}_t(\boldsymbol{r}_t) \triangleq (\boldsymbol{v}_{1,t}^T(\boldsymbol{r}_t), \ldots, \boldsymbol{v}_{N,t}^T(\boldsymbol{r}_t))^T$, and the covariance matrix of $\boldsymbol{v}_t(\boldsymbol{r}_t)$ is $\boldsymbol{R}_t(\boldsymbol{r}_t) \triangleq \mathrm{diag}(\boldsymbol{R}_{1,t}(\boldsymbol{r}_t), \ldots, \boldsymbol{R}_{N,t}(\boldsymbol{r}_t))$. In the Bayesian filtering paradigm, the posterior distribution $p(\boldsymbol{x}_t, \boldsymbol{r}_t | \boldsymbol{y}_{1:t})$ provides a complete statistical description of states $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$, where $\boldsymbol{y}_{1:t} \triangleq \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t\}$. From the posterior distribution, one can calculate various estimates of $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$.

The objective in the distributed filter is to estimate the unknown states $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$ at each sensor in the network, where each sensor only exchanges information with its neighbors. The main challenge is to obtain a state estimate for each sensor that is as accurate as that of the centralized filter.

III. Centralized Mixture Kalman Filter

In the centralized filter, each sensor sends its observation to a data FC, which is responsible for the calculation of posterior

$p(\boldsymbol{x}_t, \boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t})$, where $\boldsymbol{r}_{1:t} \triangleq \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_t\}$, which can be written as

$$p(\boldsymbol{x}_t, \boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t}) = p(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t})p(\boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t}). \quad (5)$$

Systems (2) and (3) are linear and Gaussian for $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$ when given the value of $\boldsymbol{r}_{1:t}$. The MKF [26] applies the PF to estimate $p(\boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t})$ and then use the KF to analytically compute the posterior $p(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t})$. By exploiting the principle of importance sampling and resampling, the PF recursively generates in time a set of randomly weighted particles to represent $p(\boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t})$. Specifically, if we denote the particle presentation of $p(\boldsymbol{r}_{1:t-1}|\boldsymbol{y}_{1:t-1})$ as $\{(\boldsymbol{r}_{1:t-1}^{(j)}, w_{t-1}^{(j)})\}_{j=1}^{J}$, where $\boldsymbol{r}_{1:t-1}^{(j)}$ is the particle and $w_{t-1}^{(j)}$ is the associated weight, the PF generates the particle presentation $\{(\boldsymbol{r}_{1:t}^{(j)}, w_t^{(j)})\}_{j=1}^{J}$ of $p(\boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t})$ from $\{(\boldsymbol{r}_{1:t-1}^{(j)}, w_{t-1}^{(j)})\}_{j=1}^{J}$ with the arrival of a new observation $\boldsymbol{y}_t$. The algorithm proceeds in a sequential manner as follows [18], [26].

For $j = 1, \ldots, J$, do the following.

- Draw a particle $\boldsymbol{r}_t^{(j)}$ from a proposal distribution (PD) $q(\boldsymbol{r}_t|\boldsymbol{r}_{1:t-1}^{(j)}, \boldsymbol{y}_{1:t})$, and generate $\boldsymbol{r}_{1:t}^{(j)} = \{\boldsymbol{r}_{1:t-1}^{(j)}, \boldsymbol{r}_t^{(j)}\}$.
- Calculate the importance weight, i.e.,

$$w_t^{(j)} \propto w_{t-1}^{(j)} \frac{p\left(\boldsymbol{r}_{1:t}^{(j)}|\boldsymbol{y}_{1:t}\right)}{q\left(\boldsymbol{r}_t|\boldsymbol{r}_{1:t-1}^{(j)}, \boldsymbol{y}_{1:t}\right) p\left(\boldsymbol{r}_{1:t-1}^{(j)}|\boldsymbol{y}_{1:t-1}\right)}. \quad (6)$$

One popular choice of PD is the prior distribution, i.e.,

$$q(\boldsymbol{r}_t|\boldsymbol{r}_{1:t-1}, \boldsymbol{y}_{1:t}) = p(\boldsymbol{r}_t|\boldsymbol{r}_{t-1}). \quad (7)$$

From the Bayes' rule, we have

$$p(\boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t}) = \frac{p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1})p(\boldsymbol{r}_t|\boldsymbol{r}_{t-1})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})} p(\boldsymbol{r}_{1:t-1}|\boldsymbol{y}_{1:t-1}). \quad (8)$$

By substituting (7) and (8) into (6), the corresponding weight is

$$w_t^{(j)} \propto w_{t-1}^{(j)} p\left(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right). \quad (9)$$

For the computation of likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ in (9), we first write it as

$$p\left(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \int p\left(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{r}_t^{(j)}\right) p\left(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{x}_t. \quad (10)$$

From the observation equation (4), we have

$$p\left(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{r}_t^{(j)}\right) = \mathcal{N}\left(\boldsymbol{y}_t|\boldsymbol{B}_t\left(\boldsymbol{r}_t^{(j)}\right)\boldsymbol{x}_t, \boldsymbol{R}_t\left(\boldsymbol{r}_t^{(j)}\right)\right). \quad (11)$$

The second term $p(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ in the integral of (10) can be expressed as

$$p\left(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \int p\left(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{r}_t^{(j)}\right) p\left(\boldsymbol{x}_{t-1}|\boldsymbol{r}_{1:t-1}^{(j)}, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{x}_{t-1} \quad (12)$$

where from the state equation (2), we have

$$p\left(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{r}_t^{(j)}\right) = \mathcal{N}\left(\boldsymbol{x}_t|\boldsymbol{A}_t\left(\boldsymbol{r}_t^{(j)}\right)\boldsymbol{x}_{t-1}, \boldsymbol{Q}_t\left(\boldsymbol{r}_t^{(j)}\right)\right) \quad (13)$$

and the second term $p(\boldsymbol{x}_{t-1}|\boldsymbol{r}_{1:t-1}^{(j)}, \boldsymbol{y}_{1:t-1})$ is the objective distribution at the previous time $t-1$, which has been computed by KF in the iterative process, and we denote it as

$$p\left(\boldsymbol{x}_{t-1}|\boldsymbol{r}_{1:t-1}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \mathcal{N}\left(\boldsymbol{x}_{t-1}|\boldsymbol{x}_{t-1|t-1}^{(j)}, \boldsymbol{P}_{t-1|t-1}^{(j)}\right). \quad (14)$$

Substituting (13) and (14) into (12), we have

$$p\left(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \mathcal{N}\left(\boldsymbol{x}_t|\boldsymbol{x}_{t|t-1}^{(j)}, \boldsymbol{P}_{t|t-1}^{(j)}\right) \quad (15)$$

where $\boldsymbol{x}_{t|t-1}^{(j)} = \boldsymbol{A}_t(\boldsymbol{r}_t^{(j)})\boldsymbol{x}_{t-1|t-1}^{(j)}$, and $\boldsymbol{P}_{t|t-1}^{(j)} = \boldsymbol{Q}_t(\boldsymbol{r}_t^{(j)}) + \boldsymbol{A}_t(\boldsymbol{r}_t^{(j)})\boldsymbol{P}_{t-1|t-1}^{(j)}\boldsymbol{A}_t(\boldsymbol{r}_t^{(j)})^T$. Moreover, substituting (11) and (15) into (10), the likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ can be obtained as

$$p\left(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \mathcal{N}\left(\boldsymbol{y}_t|\boldsymbol{\mu}_t^{(j)}, \boldsymbol{\Sigma}_t^{(j)}\right) \quad (16)$$

where $\boldsymbol{\mu}_t^{(j)} = \boldsymbol{B}_t(\boldsymbol{r}_t^{(j)})\boldsymbol{x}_{t|t-1}^{(j)}$, and $\boldsymbol{\Sigma}_t^{(j)} = \boldsymbol{R}_t(\boldsymbol{r}_t^{(j)}) + \boldsymbol{B}_t(\boldsymbol{r}_t^{(j)})\boldsymbol{P}_{t|t-1}^{(j)}\boldsymbol{B}_t(\boldsymbol{r}_t^{(j)})^T$.

Finally, the posterior $p(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t})$ in (5) can be calculated by KF as

$$p\left(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t}\right) = \mathcal{N}\left(\boldsymbol{x}_t|\boldsymbol{x}_{t|t}^{(j)}, \boldsymbol{P}_{t|t}^{(j)}\right) \quad (17)$$

where $\boldsymbol{x}_{t|t}^{(j)} = \boldsymbol{x}_{t|t-1}^{(j)} + \boldsymbol{K}_t^{(j)}(\boldsymbol{y}_t - \boldsymbol{B}_t(\boldsymbol{r}_t^{(j)})\boldsymbol{x}_{t|t-1}^{(j)})$, $\boldsymbol{P}_{t|t}^{(j)} = \boldsymbol{P}_{t|t-1}^{(j)} - \boldsymbol{K}_t^{(j)}\boldsymbol{B}_t(\boldsymbol{r}_t^{(j)})\boldsymbol{P}_{t|t-1}^{(j)}$, and $\boldsymbol{K}_t^{(j)} = \boldsymbol{P}_{t|t-1}^{(j)}\boldsymbol{B}_t(\boldsymbol{r}_t^{(j)})^T(\boldsymbol{\Sigma}_t^{(j)})^{-1}$.

IV. Distributed Mixture Kalman Filter

A. Basic Structure

Here, we extend the centralized MKF and develop a consensus-based distributed MKF for the CDLSs. In the distributed MKF, the sensors establish an agreement on some global quantities among all sensors by consensus algorithms, which fuse the observations from all sensors. Each sensor runs local MKF simultaneously and uses these quantities to generate a mixture Gaussian representation of the global posterior $p(\boldsymbol{x}_t, \boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t})$.

Suppose that at time $t-1$, the nth sensor has the weighted particles $\{(\boldsymbol{r}_{n,1:t-1}^{(j)}, w_{n,t-1}^{(j)})\}_{j=1}^{J}$ to represent $p(\boldsymbol{r}_{1:t-1}|\boldsymbol{y}_{1:t-1})$. Moreover, since systems (2) and (3) are linear and Gaussian for $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$ when given $\boldsymbol{r}_{1:t-1} = \boldsymbol{r}_{n,1:t-1}^{(j)}$, it has an analytical expression about $p(\boldsymbol{x}_{t-1}|\boldsymbol{r}_{n,1:t-1}^{(j)}, \boldsymbol{y}_{1:t-1})$ from the KF in the iterative process, and we denote it as

$$p\left(\boldsymbol{x}_{t-1}|\boldsymbol{r}_{n,1:t-1}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \mathcal{N}\left(\boldsymbol{x}_{t-1}|\boldsymbol{x}_{n,t-1|t-1}^{(j)}, \boldsymbol{P}_{n,t-1|t-1}^{(j)}\right). \quad (18)$$

At time t, the nth sensor will carry out two tasks. The first task is that the nth sensor applies the PF to generate the particle representation $\{(\boldsymbol{r}_{n,1:t}^{(j)}, w_{n,t}^{(j)})\}_{j=1}^{J}$ of $p(\boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t})$ from

$\{(\boldsymbol{r}_{n,1:t-1}^{(j)}, w_{n,t-1}^{(j)})\}_{j=1}^{J}$ after each sensor obtains its observation of time t. Specifically, the nth sensor can use the prior PD, i.e.,

$$q\left(\boldsymbol{r}_t|\boldsymbol{r}_{n,1:t-1}^{(j)}, \boldsymbol{y}_{1:t}\right) = p\left(\boldsymbol{r}_t|\boldsymbol{r}_{n,t-1}^{(j)}\right) \tag{19}$$

to generate one particle $\boldsymbol{r}_{n,t}^{(j)}$. Then, it constructs the particle $\boldsymbol{r}_{n,1:t}^{(j)}$ as $\boldsymbol{r}_{n,1:t}^{(j)} \triangleq \{\boldsymbol{r}_{n,1:t-1}^{(j)}, \boldsymbol{r}_{n,t}^{(j)}\}$. This step can be realized at each sensor locally. From (9), the particle's weight is calculated as

$$w_{n,t}^{(j)} \propto w_{n,t-1}^{(j)} p\left(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right). \tag{20}$$

The second task is that the nth sensor computes the analytical expression of the posterior $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t})$ by KF, which is denoted as

$$p\left(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t}\right) = \mathcal{N}\left(\boldsymbol{x}_t|\boldsymbol{x}_{n,t|t}^{(j)}, \boldsymbol{P}_{n,t|t}^{(j)}\right). \tag{21}$$

In the distributed filter, the computation of the likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ and the posterior $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t})$ requires the observation $\boldsymbol{y}_t$ from all sensors. Distributed schemes will be executed to fuse the observations from all sensors when calculating $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ and $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t})$ at each sensor.

B. Distributed Computation of the Likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$

From the system models (1)–(3), the observation noise levels $\boldsymbol{v}_{n,t}(\boldsymbol{r}_t)$ are assumed to be mutually uncorrelated zero-mean Gaussian processes when given the value of $\boldsymbol{r}_t$. Hence, if the state variables $\boldsymbol{x}_t$ and $\boldsymbol{r}_{1:t}$ are known, the observations $\boldsymbol{y}_{1,t}$, …, $\boldsymbol{y}_{N,t}$ from different sensors are mutually uncorrelated, and we have

$$p\left(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1}\right) = \prod_{m=1}^{N} p\left(\boldsymbol{y}_{m,t}|\boldsymbol{x}_t, \boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1}\right). \tag{22}$$

This factorization is indispensable for the development of the distributed PF such as that in [19]–[25]. However, if only $\boldsymbol{r}_{1:t}$ is known, the observations $\boldsymbol{y}_{1,t}$, …, $\boldsymbol{y}_{N,t}$ from different sensors are all related to the same variable $\boldsymbol{x}_t$; then, they are not mutually uncorrelated, and

$$p\left(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1}\right) \neq \prod_{m=1}^{N} p\left(\boldsymbol{y}_{m,t}|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1}\right). \tag{23}$$

This relationship can also come from the analytical expression of $p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1})$ in (16), whose covariance $\boldsymbol{\Sigma}_t$ is not a block diagonal matrix.

To obtain a distributed implementation, some approximations are typically required as that in [16], [22], and [24]. We write the likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1})$ as

$$p\left(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \int p\left(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{r}_{n,t}^{(j)}\right) p\left(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{x}_t. \tag{24}$$

From the state equation (2), the second term $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ of the integrand is

$$p\left(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) = \mathcal{N}\left(\boldsymbol{x}_t|\boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)}, \boldsymbol{P}_{n,t|t-1}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right) \tag{25}$$

where $\boldsymbol{P}_{n,t|t-1}(\boldsymbol{r}_{n,t}^{(j)}) = \boldsymbol{Q}_t(\boldsymbol{r}_{n,t}^{(j)}) + \boldsymbol{A}_t(\boldsymbol{r}_{n,t}^{(j)})\boldsymbol{P}_{n,t-1|t-1}^{(j)}\boldsymbol{A}_t^T(\boldsymbol{r}_{n,t}^{(j)})$.

The integrand in (24) is in the form of *nonlinear function* × *Gaussian density*, where $p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{r}_{n,t}^{(j)})$ is the nonlinear function of $\boldsymbol{x}_t$, and $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ is the Gaussian density. This type of integral can be accurately computed by the cubature rule [29], [30] with limited computational complexity. In the cubature rule, a set of points ξ_c and weights ω_c, which are called the cubature points, is used for the numerical integration. The cubature points are defined as

$$\xi_c = \sqrt{\frac{C}{2}}[1]_c \tag{26}$$

$$\omega_c = \frac{1}{C}, \quad c = 1, 2, \ldots, C = 2d_x \tag{27}$$

where the notation $[1]$ denotes the set of points that are located at the intersection of the unit sphere and the axes, and $[1]_c$ denotes the cth element of set $[1]$. For example, $[1] \in \mathbf{R}^2$ represents the following set of points:

$$\left\{\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}0\\-1\end{pmatrix}\right\} \tag{28}$$

which are the points at the intersection of the unit sphere and the x- and y-axes in $\mathbf{R}^2$.

It factorizes the covariance $\boldsymbol{P}_{n,t|t-1}(\boldsymbol{r}_{n,t}^{(j)})$ of the Gaussian density $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ as

$$\boldsymbol{P}_{n,t|t-1}\left(\boldsymbol{r}_{n,t}^{(j)}\right) = \Psi_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\Psi_{n,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right) \tag{29}$$

and evaluates

$$\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) = \Psi_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\xi_c + \boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)} \quad c = 1, \ldots, C. \tag{30}$$

The integral in (24) is then computed with a weighted sum as

$$p\left(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) \approx \frac{1}{C}\sum_{c=1}^{C} p\left(\boldsymbol{y}_t|\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{r}_{n,t}^{(j)}\right). \tag{31}$$

The computation of $p(\boldsymbol{y}_t|\Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{r}_{n,t}^{(j)})$ can be realized in a distributed scheme. Since the observation noise levels $\boldsymbol{v}_{n,t}(\boldsymbol{r}_t)$ are mutually uncorrelated when given $\boldsymbol{r}_t$, we have

$$p\left(\boldsymbol{y}_t|\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{r}_{n,t}^{(j)}\right) = \prod_{m=1}^{N} p\left(\boldsymbol{y}_{m,t}|\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{r}_{n,t}^{(j)}\right) \tag{32}$$

where $p(\boldsymbol{y}_{m,t}|\Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{r}_{n,t}^{(j)}) = \mathcal{N}(\boldsymbol{y}_{m,t}|\boldsymbol{B}_{m,t}(\boldsymbol{r}_{n,t}^{(j)}) \Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{R}_{m,t}(\boldsymbol{r}_{n,t}^{(j)}))$. Hence, we have

$$
\begin{aligned}
&p\left(\boldsymbol{y}_t|\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{r}_{n,t}^{(j)}\right) \\
&= \prod_{m=1}^{N} \mathcal{N}\left(\boldsymbol{y}_{m,t}|\boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) \Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right) \\
&= \prod_{m=1}^{N} \frac{1}{2\pi \left|\boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right|^{\frac{1}{2}}} \\
&\quad \times \exp\left\{-\frac{1}{2}\left(\boldsymbol{y}_{m,t} - \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) \Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^T \right. \\
&\qquad \left. \times \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)^{-1} \left(\boldsymbol{y}_{m,t} - \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) \Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)\right\} \\
&= \frac{1}{(2\pi)^N \left(\prod_{m=1}^{N} \left|\boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right|\right)^{\frac{1}{2}}} \\
&\quad \times \exp\left\{-\frac{1}{2}\left(\sum_{m=1}^{N} \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)^{-1} \boldsymbol{y}_{m,t}\right)\right. \\
&\qquad + \left(\sum_{m=1}^{N} \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)^{-1} \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right) \Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) \\
&\qquad - \frac{1}{2} \Theta_{c,n,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right) \\
&\qquad \times \left(\sum_{m=1}^{N} \boldsymbol{B}_{m,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right) \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)^{-1} \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right) \\
&\qquad \left. \times \Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right\}. \qquad (33)
\end{aligned}
$$

If $\boldsymbol{r}_t$ is discrete and takes value in a finite set $\mathcal{M}$, we define

$$\phi_t(\boldsymbol{r}) = \frac{1}{N} \sum_{m=1}^{N} \left(-\log|\boldsymbol{R}_{m,t}(\boldsymbol{r})| - \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}(\boldsymbol{r})^{-1} \boldsymbol{y}_{m,t}\right) \qquad (34)$$

$$\varsigma_t(\boldsymbol{r}) = \frac{1}{N} \sum_{m=1}^{N} \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}(\boldsymbol{r})^{-1} \boldsymbol{B}_{m,t}(\boldsymbol{r}) \qquad (35)$$

$$\psi_t(\boldsymbol{r}) = \frac{1}{N} \sum_{m=1}^{N} \boldsymbol{B}_{m,t}^T(\boldsymbol{r}) \boldsymbol{R}_{m,t}(\boldsymbol{r})^{-1} \boldsymbol{B}_{m,t}(\boldsymbol{r}) \qquad (36)$$

for each $\boldsymbol{r} \in \mathcal{M}$. These averages can be calculated in a distributed fashion by consensus algorithms among all sensors, and the estimation results are available at each sensor.

If the state variable $\boldsymbol{r}_t$ is continuous, (34)–(36) are not available for each $\boldsymbol{r} \in \mathcal{M}$. In this case, we take advantage of the techniques in [21] and presuppose that the local random-number generators in the entire network are synchronized. Since the same pseudorandom numbers are obtained at all sensors, we have $\boldsymbol{r}_{1,t}^{(j)} = \cdots = \boldsymbol{r}_{N,t}^{(j)}$ for $j = 1, \ldots, J$. Then, the likelihood in (33) can be written as

$$
\begin{aligned}
&p\left(\boldsymbol{y}_t|\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{r}_{n,t}^{(j)}\right) \\
&= \frac{1}{(2\pi)^N \left(\prod_{m=1}^{N} \left|\boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)\right|\right)^{\frac{1}{2}}} \\
&\quad \times \exp\left\{-\frac{1}{2}\left(\sum_{m=1}^{N} \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)^{-1} \boldsymbol{y}_{m,t}\right)\right. \\
&\qquad + \left(\sum_{m=1}^{N} \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)^{-1} \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)\right) \Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) \\
&\qquad - \frac{1}{2} \Theta_{c,n,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right) \\
&\qquad \times \left(\sum_{m=1}^{N} \boldsymbol{B}_{m,t}^T\left(\boldsymbol{r}_{m,t}^{(j)}\right) \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)^{-1} \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)\right) \\
&\qquad \left. \times \Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right\}. \qquad (37)
\end{aligned}
$$

For $j = 1, \ldots, J$, we define

$$
\begin{aligned}
\phi_t(j) = \frac{1}{N} \sum_{m=1}^{N} \Bigg(&-\log\left|\boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)\right| \\
&- \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)^{-1} \boldsymbol{y}_{m,t}\Bigg) \qquad (38)
\end{aligned}
$$

$$\varsigma_t(j) = \frac{1}{N} \sum_{m=1}^{N} \boldsymbol{y}_{m,t}^T \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)^{-1} \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right) \qquad (39)$$

$$\psi_t(j) = \frac{1}{N} \sum_{m=1}^{N} \boldsymbol{B}_{m,t}^T\left(\boldsymbol{r}_{m,t}^{(j)}\right) \boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right)^{-1} \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{m,t}^{(j)}\right). \qquad (40)$$

These averages can be calculated in a distributed scheme by consensus algorithms among all sensors, and the estimation results are available at each sensor.

For the cubature rule in (31), there are $C = 2d_x$ elements in the summation, and the likelihood $p(\boldsymbol{y}_t|\Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{r}_{n,t}^{(j)})$ will be computed C times. The likelihood in (24) can also be simply approximated from (25) as

$$p\left(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1}\right) \approx p\left(\boldsymbol{y}_t|\boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right) \boldsymbol{x}_{n,t-1|t-1}^{(j)}, \boldsymbol{r}_{n,t}^{(j)}\right) \qquad (41)$$

where we only use one point, i.e., the mean of $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ in (25), when calculating the integral in (24). The likelihood $p(\boldsymbol{y}_t|\boldsymbol{A}_t(\boldsymbol{r}_{n,t}^{(j)})\boldsymbol{x}_{n,t-1|t-1}^{(j)}, \boldsymbol{r}_{n,t}^{(j)})$ in (41) is computed with the same distributed method as $p(\boldsymbol{y}_t|\Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{r}_{n,t}^{(j)})$ in (32). The method has lower computational complexity than the cubature rule at the cost of some performance loss.

C. Computation of Average Consensus

The objective of average consensus is that the sensors are to converge to the average of their initial values. The average consensus problems have been addressed in the literature [11], [12],

where Olfati-Saber *et al.* showed that average consensus can be reached by performing a linear combination of its own value and the values of its neighbors in a distributed fashion. Average consensus problems in networks have also been developed to operate under disturbances due to delays [32], packet drops [33], or dynamical interaction [34]. In our paper, we consider the average consensus under a fixed interconnection topology and without disturbances due to delays and packet drops, which is the same as that in [19]–[24], where the distributed PFs are addressed.

Consensus algorithms [11], [12] are a linear iterative method, which only require local communications between neighboring sensors. At the initial step $h = 0$, the nth sensor initializes its state as $\vartheta_n(0) = D_n$, where D_n is the element of the averages in (34)–(36) or (38)–(40). In the following step, each sensor updates its state with a linear combination of its own state and the states at its neighbors as

$$\vartheta_n(h+1) = W_{nn}\vartheta_n(h) + \sum_{m\in\mathcal{N}_n} W_{nm}\vartheta_m(h) \tag{42}$$

where W_{nm} is the linear weight on $\vartheta_m(h)$ at the nth sensor. We use the Metropolis weights [35] as

$$W_{nm} = \begin{cases} \frac{1}{1+\max\{|\mathcal{N}_n|,|\mathcal{N}_m|\}}, & \text{if } (n,m)\in\mathcal{E} \\ 1 - \sum\limits_{\{n,l\}\in\mathcal{E}} W_{nl}, & \text{if } n = m \\ 0, & \text{otherwise} \end{cases}$$

where $|\mathcal{N}_n|$ denotes the cardinality of set $\mathcal{N}_n$.

With the consensus algorithms, $\phi_t(\cdot)$, $\varsigma_t(\cdot)$, and $\psi_t(\cdot)$ can be computed at each sensor, which fuse the observations from all sensors. We denote the results of these quantities computed at the nth sensor as $\phi_{n,t}(\cdot)$, $\varsigma_{n,t}(\cdot)$, and $\psi_{n,t}(\cdot)$, respectively. These quantities will slightly differ at different sensors because only a finite number of consensus iterations have been performed. With these quantities, at the nth sensor, the likelihood $p(\boldsymbol{y}_t|\Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{r}_{n,t}^{(j)})$ in (33) when $\boldsymbol{r}_t$ is discrete will be computed as

$$\begin{aligned} p\left(\boldsymbol{y}_t|\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{r}_{n,t}^{(j)}\right) &= \frac{1}{(2\pi)^N} \\ &\times \exp\left\{-\frac{N}{2}\phi_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) + N\varsigma_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right. \\ &\left. - \frac{N}{2}\Theta_{c,n,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right)\psi_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right\} \end{aligned} \tag{43}$$

and the likelihood $p(\boldsymbol{y}_t|\Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{r}_{n,t}^{(j)})$ in (37) when $\boldsymbol{r}_t$ is continuous will be computed as

$$\begin{aligned} p\left(\boldsymbol{y}_t|\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right), \boldsymbol{r}_{n,t}^{(j)}\right) &= \frac{1}{(2\pi)^N} \\ &\times \exp\left\{-\frac{N}{2}\phi_{n,t}(j) + N\varsigma_{n,t}(j)\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right. \\ &\left. - \frac{N}{2}\Theta_{c,n,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right)\psi_{n,t}(j)\Theta_{c,n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right\} \end{aligned} \tag{44}$$

where the number of sensors N may be provided beforehand, or a distributed algorithm for counting N may be used [36].

D. Distributed Computation of the Posterior $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t})$

After the weighted particles $\{(\boldsymbol{r}_{n,1:t}^{(j)}, w_{n,t}^{(j)})\}_{j=1}^J$ have been generated to represent the posterior $p(\boldsymbol{r}_{1:t}|\boldsymbol{y}_{1:t})$ at the nth sensor, the mean and covariance of the posterior $p(\boldsymbol{x}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t})$ in (21) can be calculated from the KF at the nth sensor as

$$\begin{aligned} \boldsymbol{x}_{n,t|t}^{(j)} &= \boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)} + \boldsymbol{P}_{n,t|t}^{(j)}\boldsymbol{B}_t^T\left(\boldsymbol{r}_{n,t}^{(j)}\right) \\ &\times \left(\boldsymbol{R}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^{-1}\left(\boldsymbol{y}_t - \boldsymbol{B}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)}\right) \\ &= \boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)} + \boldsymbol{P}_{n,t|t}^{(j)} \\ &\times \left[\left(\sum_{m=1}^N \boldsymbol{y}_{m,t}^T\boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)^{-1}\boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^T\right. \\ &\quad - \left(\sum_{m=1}^N \boldsymbol{B}_{m,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)^{-1}\right. \\ &\quad \left.\left.\times \boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)\boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)}\right] \\ &= \boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)} + \boldsymbol{P}_{n,t|t}^{(j)} \\ &\times \left[N\left(\varsigma_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^T\right. \\ &\quad \left. - N\psi_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{A}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_{n,t-1|t-1}^{(j)}\right] \end{aligned} \tag{45}$$

$$\begin{aligned} \boldsymbol{P}_{n,t|t}^{(j)} &= \left[\left(\boldsymbol{P}_{n,t|t-1}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^{-1}\right. \\ &\quad \left. -\boldsymbol{B}_t^T\left(\boldsymbol{r}_{n,t}^{(j)}\right)\left(\boldsymbol{R}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^{-1}\boldsymbol{B}_t\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right]^{-1} \\ &= \left[\left(\boldsymbol{P}_{n,t|t-1}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^{-1}\right. \\ &\quad \left. -\left(\sum_{m=1}^N \boldsymbol{B}_{m,t}^T\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{R}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)^{-1}\boldsymbol{B}_{m,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)\right]^{-1} \\ &= \left[\left(\boldsymbol{P}_{n,t|t-1}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right)^{-1} - N\psi_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\right]^{-1} \end{aligned} \tag{46}$$

where $\varsigma_{n,t}(\cdot)$ and $\psi_{n,t}(\cdot)$ have been calculated in (35) and (36) when $\boldsymbol{r}_t$ is discrete or (39) and (40) when $\boldsymbol{r}_t$ is continuous with consensus algorithms.

E. Communication Requirements

When computing $\phi_t(\cdot)$, $\varsigma_t(\cdot)$, and $\psi_t(\cdot)$ in (34)–(36) or in (38)–(40) with the average consensus algorithm, it requires the

communication of the quantities $-\log|\boldsymbol{R}_{m,t}(\cdot)| - \boldsymbol{y}_{m,t}^T\boldsymbol{R}_{m,t}(\cdot)^{-1}\boldsymbol{y}_{m,t}$, $\boldsymbol{y}_{m,t}^T\boldsymbol{R}_{m,t}(\cdot)^{-1}\boldsymbol{B}_{m,t}(\cdot)$, and $\boldsymbol{B}_{m,t}(\cdot)^T\boldsymbol{R}_{m,t}(\cdot)^{-1}\boldsymbol{B}_{m,t}(\cdot)$ between sensors, respectively. The quantity $-\log|\boldsymbol{R}_{m,t}(\cdot)| - \boldsymbol{y}_{m,t}^T\boldsymbol{R}_{m,t}(\cdot)^{-1}\boldsymbol{y}_{m,t}$ is a scalar, $\boldsymbol{y}_{m,t}^T\boldsymbol{R}_{m,t}(\cdot)^{-1}\boldsymbol{B}_{m,t}(\cdot)$ is a vector with dimension d_x, and $\boldsymbol{B}_{m,t}(\cdot)^T\boldsymbol{R}_{m,t}(\cdot)^{-1}\boldsymbol{B}_{m,t}(\cdot)$ is a symmetrical matrix with dimension $d_x \times d_x$. In the applications of Section V, the dimension d_x of state $\boldsymbol{x}_t$ is $d_x = 4$. These quantities are not related to the dimension $d_{m,y}$ of the observation variable $\boldsymbol{y}_{m,t}$ at each sensor.

If variable $\boldsymbol{r}_t$ is discrete and matrices $\boldsymbol{B}_{m,t}(\boldsymbol{r}_t)$ and $\boldsymbol{R}_{m,t}(\boldsymbol{r}_t)$ in the observation equation (3) are independent of $\boldsymbol{r}_t$ and time invariant (e.g., the first application in Section V-A), then $\phi_t(\cdot)$, $\varsigma_t(\cdot)$, and $\psi_t(\cdot)$ in (34)–(36) are not related to $\boldsymbol{r}_t$, and $\psi_t(\cdot)$ is time invariant. Hence, $\psi_t(\cdot)$ can be computed before the whole estimation process, and at each time step t and for all particles, one average consensus algorithm is required for computing $\phi_t(\cdot)$ and $\varsigma_t(\cdot)$. Then, at each time step, each sensor will transmit $1 + d_x$ real numbers to its neighbors at each iteration of the average consensus algorithm.

If $\boldsymbol{r}_t$ is discrete and $\boldsymbol{B}_{m,t}(\boldsymbol{r}_t)$ and $\boldsymbol{R}_{m,t}(\boldsymbol{r}_t)$ in (3) are dependent on $\boldsymbol{r}_t$ (e.g., the second application in Section V-B), then at each time step t and for each $\boldsymbol{r} \in \mathcal{M}$, one average consensus algorithm is required for computing $\phi_t(\boldsymbol{r})$, $\varsigma_t(\boldsymbol{r})$, and $\psi_t(\boldsymbol{r})$. Then, at each time step, each sensor will transmit $(1 + d_x + (d_x + d_x^2)/2) \cdot |\mathcal{M}|$ real numbers to its neighbors at each iteration of the average consensus algorithm. It is noted that the cardinality of set $\mathcal{M}$ is usually much smaller than the number of particles, where $|\mathcal{M}| = 3$ in the application of Section V-B.

If $\boldsymbol{r}_t$ is continuous and $\boldsymbol{B}_{m,t}(\boldsymbol{r}_t)$ and $\boldsymbol{R}_{m,t}(\boldsymbol{r}_t)$ in (3) are independent of $\boldsymbol{r}_t$ and time invariant (e.g., the third application in Section V-C), $\phi_t(\cdot)$, $\varsigma_t(\cdot)$, and $\psi_t(\cdot)$ in (38)–(40) are not related to $\boldsymbol{r}_t$, and $\psi_t(\cdot)$ is time invariant. Hence, $\psi_t(\cdot)$ can be computed before the whole estimation process, and at each time step t and for all particles, one average consensus algorithm is required for computing $\phi_t(\cdot)$ and $\varsigma_t(\cdot)$. Then, at each time step, each sensor will transmit $1 + d_x$ real numbers to its neighbors at each iteration of the average consensus algorithm.

If $\boldsymbol{r}_t$ is continuous and $\boldsymbol{B}_{m,t}(\boldsymbol{r}_t)$ and $\boldsymbol{R}_{m,t}(\boldsymbol{r}_t)$ in (3) are dependent on $\boldsymbol{r}_t$ (e.g., the fourth application in Section V-D), then at each time step t and for each particle $\boldsymbol{r}_{m,t}^{(j)}$, one average consensus algorithm is required for computing $\phi_t(j)$, $\varsigma_t(j)$, and $\psi_t(j)$ in (38)–(40). Then, at each time step, each sensor will transmit $(1 + d_x + (d_x + d_x^2)/2) \cdot J$ real numbers to its neighbors at each iteration of the average consensus algorithm. Since the MKF reduces the size of state space of the state variable by analytically integrating out the state variable $\boldsymbol{x}_t$, it can significantly decrease the required number J of particles compared with the standard PF. The distributed MKF can save much computational cost compared with the distributed PF.

F. Computational Complexity

From (9), the likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ in the centralized MKF at the data FC will be computed one time for each weighted particle. From (20), the likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ in the distributed MKF at each sensor will be computed one time for each weighted particle, which is the same as that in the centralized MKF. Hence, the difference in the computational complexity between the distributed MKF and the centralized MKF is in the different methods for computing the likelihood.

For the centralized MKF, the likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ is analytically computed in (16). For the distributed MKF, to obtain a distributed implementation, some approximations are executed, and we consider two methods for computing $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$. One method is based on the cubature rule in (31), where the likelihood $p(\boldsymbol{y}_t|\Theta_{c,n,t}(\boldsymbol{r}_{n,t}^{(j)}), \boldsymbol{r}_{n,t}^{(j)})$ will be computed $C = 2d_x$ times when computing $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$. Hence, the computational complexity of the distributed MKF with (31) is C times that of the centralized MKF. Another method is based on (41), which has some performance loss. In (41), the likelihood $p(\boldsymbol{y}_t|\boldsymbol{A}_t(\boldsymbol{r}_{n,t}^{(j)})\boldsymbol{x}_{n,t-1|t-1}^{(j)}, \boldsymbol{r}_{n,t}^{(j)})$ will be computed one time when computing $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$. Hence, the computational complexity of the distributed MKF with (41) is the same as that of the centralized MKF.

V. Applications to Maneuvering Target Tracking

Here, we apply the proposed distributed MKF to the maneuvering target tracking problems [28]. Four examples of maneuvering target tracking are considered. In the first two examples, the maneuvering model is a constant-turn model with a known turn rate [28], and the turn rate is a discrete variable. The observation equation in the first example is linear, and the observation equation in the second example is nonlinear. In the last two examples, the maneuvering model is a constant-turn model with an unknown turn rate [28], and the turn rate is a continuous variable. The observation equation in the third example is linear, and the observation equation in the fourth example is nonlinear.

A. Constant-Turn Model With Known Turn Rate and Linear Observation

The target state is denoted by $\boldsymbol{x}_t \triangleq (p_{x,t}, v_{x,t}, p_{y,t}, v_{y,t})^T$, where $\boldsymbol{p}_t \triangleq (p_{x,t}, p_{y,t})^T$ represents the target position, and $\boldsymbol{v}_t \triangleq (v_{x,t}, v_{y,t})^T$ represents the target velocity in the (x, y) coordinate system. The target dynamics are described by the constant-turn model with a known turn rate [28] as

$$\boldsymbol{x}_t = \boldsymbol{A}_t(\boldsymbol{r}_t)\boldsymbol{x}_{t-1} + \boldsymbol{u}_t(\boldsymbol{r}_t) \tag{47}$$

where the matrix $\boldsymbol{A}_t(\boldsymbol{r}_t)$ is

$$\boldsymbol{A}_t(\boldsymbol{r}_t) = \begin{bmatrix} 1 & \frac{\sin(\boldsymbol{r}_t T)}{\boldsymbol{r}_t} & 0 & -\frac{1-\cos(\boldsymbol{r}_t T)}{\boldsymbol{r}_t} \\ 0 & \cos(\boldsymbol{r}_t T) & 0 & -\sin(\boldsymbol{r}_t T) \\ 0 & \frac{1-\cos(\boldsymbol{r}_t T)}{\boldsymbol{r}_t} & 1 & \frac{\sin(\boldsymbol{r}_t T)}{\boldsymbol{r}_t} \\ 0 & \sin(\boldsymbol{r}_t T) & 0 & \cos(\boldsymbol{r}_t T) \end{bmatrix} \tag{48}$$

and $\boldsymbol{r}_t$ denotes the turn rate, and $T = 1$ (s) is the sampling time period. The state noise $\boldsymbol{u}_t(\boldsymbol{r}_t)$ is zero-mean white Gaussian noise with covariance matrix, i.e.,

$$\boldsymbol{Q}_t(\boldsymbol{r}_t) = \sigma_{\boldsymbol{r}_t}^2 \boldsymbol{I}_2 \otimes \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix} \tag{49}$$

where $\otimes$ is the Kronecker product, and $\boldsymbol{I}_2$ denotes the 2×2 identity matrix. In the simulations, three coordinated turn models are considered, i.e., $\boldsymbol{r}_t \in \mathcal{M} = \{-10^\circ/\text{s}, 0^\circ/\text{s}, 10^\circ/\text{s}\}$, which correspond to counterclockwise turn rate $10^\circ/\text{s}$, turn rate $0^\circ/\text{s}$, and clockwise turn rate $10^\circ/\text{s}$, respectively. The deviations in the different coordinated turn models are considered to be $\sigma_{-10} = 3^\circ$, $\sigma_0 = 2^\circ$, and $\sigma_{10} = 3^\circ$. The switching between the three models is governed by a first-order homogeneous Markov chain with known transition probabilities $\pi_{ii} = 0.8$ $(i = -10, 0, 10)$ and $\pi_{ij} = 0.1$ $(i \neq j)$. We choose the prior distribution of the initial target state as $\boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ with $\boldsymbol{\mu}_0 = (4, 0.05, 5, 0.05)^T$ and $\boldsymbol{\Sigma}_0 = \text{diag}\{1, 0.001, 1, 0.001\}$, and $\boldsymbol{r}_0$ takes values in $\mathcal{M}$ with discrete uniform distribution.

In the simulations, there are $N = 25$ sensors in the network that are uniformly deployed within a rectangular region of size 40 m $\times$ 40 m. Each sensor communicates with other sensors within a range of 18 m. The observation of the nth sensor in the network is

$$\boldsymbol{y}_{n,t} = \boldsymbol{B}_{n,t}\boldsymbol{x}_t + \boldsymbol{v}_{n,t} \tag{50}$$

where $\boldsymbol{B}_{n,t} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$, and $\boldsymbol{v}_{n,t}$ is zero-mean white Gaussian noise with covariance matrix $\boldsymbol{R}_{n,t} = \sigma_n^2 \boldsymbol{I}_2$. Equations (47) and (50), as well as the switching probabilities π_{ij}, conform to a discrete-time CDLS, where the state variable includes $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$, and $\boldsymbol{r}_t$ is discrete and takes values in the finite set $\mathcal{M}$.

We evaluate the performance of the distributed MKF for the maneuvering target tracking, where two methods in (31) and (41) are considered for the calculation of the likelihood function $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$. If the likelihood is calculated by the cubature rule in (31), it is denoted as DMKF-CR. If the likelihood is calculated by the simple approximation in (41), it is denoted as DMKF-SA. We consider the prior PD in (19) to generate particle $\boldsymbol{r}_{n,t}^{(j)}$. It is noted that the matrices $\boldsymbol{B}_{n,t}$ and $\boldsymbol{R}_{n,t}$ in the observation equation (50) are independent of $\boldsymbol{r}_t$ and time invariant, then $\phi_t(\cdot)$, $\varsigma_t(\cdot)$, and $\psi_t(\cdot)$ in (34)–(36) are also independent of $\boldsymbol{r}_t$, and $\psi_t(\cdot)$ is time invariant. We denote them as ϕ_t, ς_t, and ψ, where ψ can be computed with the average consensus algorithm before the whole estimation process. Hence, at each time t and for all particles, the consensus algorithm is only required to be executed one time to compute φ_t and ς_t. The resulting algorithm is summarized in Algorithm I.

Algorithm I Distributed MKF for Constant-Turn Models With Known Turn Rate and Linear Observation

Initialization: For $n = 1, \ldots, N$, the nth sensor generates particles $\{(\boldsymbol{r}_{n,0}^{(j)}, \boldsymbol{x}_{n,0|0}^{(j)}, \boldsymbol{P}_{n,0|0}^{(j)}, w_{n,0}^{(j)})\}_{j=1}^{J}$ from the prior distribution of $\boldsymbol{x}_0$ and $\boldsymbol{r}_0$.

For $t = 1, 2, \ldots$,

1) For $n = 1, \ldots, N$, the nth sensor uses the prior PD in (19) to generate $\boldsymbol{r}_{n,t}^{(j)}$, and constructs the particle $\boldsymbol{r}_{n,1:t}^{(j)} \triangleq \{\boldsymbol{r}_{n,1:t-1}^{(j)}, \boldsymbol{r}_{n,t}^{(j)}\}$ for $j = 1, \ldots, J$.
2) Consensus algorithm is executed to compute φ_t and ς_t among all sensors.
3) For $n = 1, \ldots, N$, the nth sensor computes the weight $w_{n,t}^{(j)}$ with (20), where the likelihood is computed with (31) or (41).
4) For $n = 1, \ldots, N$, the nth sensor computes $\boldsymbol{x}_{n,t|t}^{(j)}$ and $\boldsymbol{P}_{n,t|t}^{(j)}$ with (45) and (46).

End for

Independent Monte Carlo runs are repeated for the performance evaluation of the methods, where the total number of Monte Carlo runs is $L = 1200$. The total length of the time interval in each Monte Carlo run is $A = 100$. The time-averaged root mean square error (RMSE) of the target position is used to assess the performance, where the RMSE_n of the nth sensor is defined as

$$\text{RMSE}_n = \left(\frac{1}{LA} \sum_{l=1}^{L} \sum_{t=1}^{A} \left\| \boldsymbol{p}_t^{(l)} - \widetilde{\boldsymbol{p}}_{n,t}^{(l)} \right\|_2^2 \right)^{\frac{1}{2}} \tag{51}$$

where l denotes the lth Monte Carlo run, $\boldsymbol{p}_t^{(l)} \triangleq (p_{x,t}^{(l)}, p_{y,t}^{(l)})^T$ is the simulated value, and $\widetilde{\boldsymbol{p}}_{n,t}^{(l)} \triangleq (\widetilde{p}_{n,x,t}^{(l)}, \widetilde{p}_{n,y,t}^{(l)})^T$ is the estimated value of the nth sensor in the lth Monte Carlo run. The average RMSE of all sensors is defined as $\text{RMSE} = (1/N)\sum_{n=1}^{N} \text{RMSE}_n$.

For the performance comparison, we present the estimation result with the centralized MKF. We denote the centralized MKF as CMKF. In the CMKF, each sensor sends its data to a data FC, and the FC is responsible for the execution of the CMKF. The RMSE of the CMKF is defined as

$$\text{RMSE} = \left(\frac{1}{LA} \sum_{l=1}^{L} \sum_{t=1}^{A} \left\| \boldsymbol{p}_t^{(l)} - \widehat{\boldsymbol{p}}_t^{(l)} \right\|_2^2 \right)^{\frac{1}{2}} \tag{52}$$

where $\widehat{\boldsymbol{p}}_t^{(l)} \triangleq (\widehat{p}_{x,t}^{(l)}, \widehat{p}_{y,t}^{(l)})^T$ is the estimated value of the FC in the lth Monte Carlo run. We also evaluate the effect when the likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{n,1:t}^{(j)}, \boldsymbol{y}_{1:t-1})$ is calculated with (31) or (41). If the CMKF is applied but the likelihood is calculated with (31), it is denoted as CMKF-CR. If the CMKF is applied but the likelihood is calculated with (41), it is denoted as CMKF-SA.

In the first simulation, we investigate the effect of the number of consensus iterations on the performance of the distributed MKF. Fig. 1 presents the RMSEs of DMKF-CR and DMKF-SA for different numbers of consensus iterations. The RMSEs of CMKF, CMKF-CR, and CMKF-SA are also described in Fig. 1. It is noted that the RMSEs of these three methods are not related to the number of consensus iterations. The number of particles is set to $J = 40$. The variance of observation noise is set to $\sigma_n^2 = 1.0$. In Fig. 1, the RMSE of CMKF-CR is nearly identical to that of CMKF. It shows that the calculation of likelihood with (31) nearly induces no error. Compared with CMKF-CR, CMKF-SA has little performance loss, but it has lower computational complexity. As the number of consensus iterations increases, the RMSEs of DMKF-CR and DMKF-SA can be arbitrarily close to that of CMKF.

In the second simulation, we investigate the effect of the number of particles for all methods. Fig. 2 presents the RMSEs

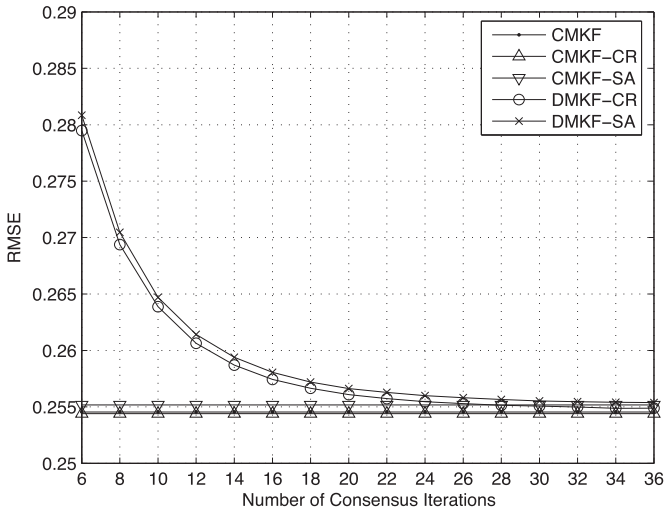

Fig. 1. RMSEs versus number of consensus iterations. The variance of observation noise is set to $\sigma_n^2 = 1.0$. The number of particles is $J = 40$.

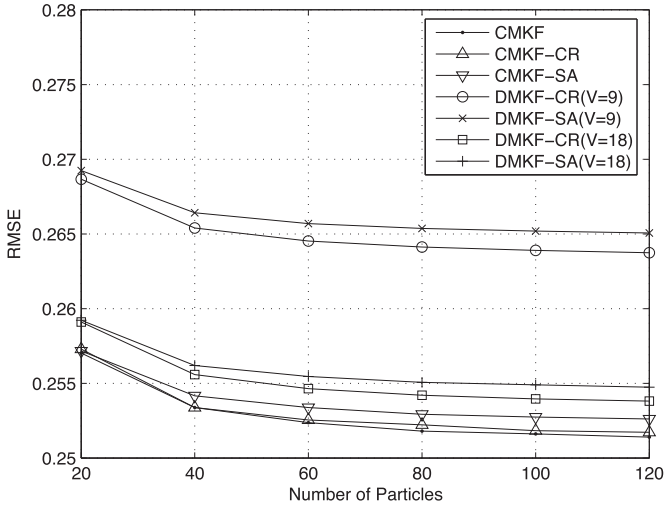

Fig. 2. RMSEs versus number of particles J. The variance of observation noise is set to $\sigma_n^2 = 1.0$. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA.

of all methods, where the variance of observation noise is $\sigma_n^2 = 1.0$, and the number of consensus iterations is set to $V = 9$ and $V = 18$ for the distributed MKF. In Fig. 2, the RMSE performances of all methods are improved as the number of particles increases. Compared with DMKF-CR, DMKF-SA has little performance loss, and both are close to the CMKF as the number of consensus iterations increases.

In the third simulation, we investigate the performance of different methods for different values of observation noise variance σ_n^2. Fig. 3 presents the RMSEs of CMKF, DMKF-CR, and DMKF-SA. The number of particles is set to $J = 40$, and the number of consensus iterations is set to $V = 9$ and $V = 18$. In Fig. 3, the RMSEs of DMKF-CR and DMKF-SA with $V = 18$ are arbitrarily close to that of CMKF.

B. Constant-Turn Model With Known Turn Rate and Nonlinear Observation

The dynamics of the target state are similarly described as in (47). The statistical characteristics of the state noise $\boldsymbol{u}_t(\boldsymbol{r}_t)$

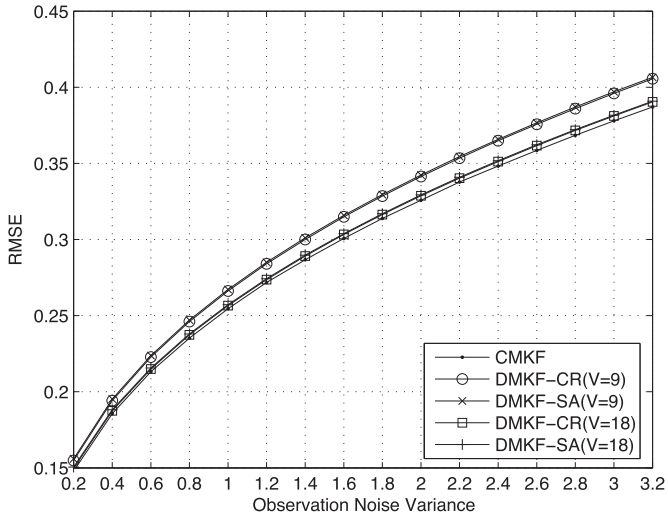

Fig. 3. RMSEs versus observation noise variance σ_n^2. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA. The number of particles is $J = 40$.

and the state variable $\boldsymbol{r}_t$ are the same as that in Section V-A. Different from the linear observation (50) in Section V-A, we consider the nonlinear observation in this section, where the observation of the nth sensor is described as

$$y_{n,t} = \|\boldsymbol{p}_t - \boldsymbol{s}_n\|_2 + v_{n,t} \tag{53}$$

which is the distance of the nth sensor and the target, where $\boldsymbol{s}_n = (s_{n,x}, s_{n,y})^T$ denotes the position of the nth sensor, and $v_{n,t}$ is zero-mean Gaussian noise with variance σ_n^2.

After the generation of particle $\boldsymbol{r}_{n,t}^{(j)}$ at the nth sensor, it linearizes the observation (53) on the point $\widetilde{\boldsymbol{x}}_{n,t|t-1}(\boldsymbol{r}_{n,t}^{(j)}) \triangleq \boldsymbol{A}_t(\boldsymbol{r}_{n,t}^{(j)})\widetilde{\boldsymbol{x}}_{n,t-1|t-1}$, where $\widetilde{\boldsymbol{x}}_{n,t-1|t-1} = \sum_{j=1}^{J} w_{n,t-1}^{(j)} \boldsymbol{x}_{n,t-1|t-1}^{(j)}$ is the estimate of the state $\boldsymbol{x}_{t-1}$ at the nth sensor. The linearized observation is described as

$$\widetilde{y}_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) \approx \boldsymbol{B}_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right)\boldsymbol{x}_t + v_{n,t} \tag{54}$$

where

$$\widetilde{y}_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) = y_{n,t} + \left(\frac{p_{x,t} - s_{n,x}}{\|\boldsymbol{p}_t - \boldsymbol{s}_n\|_2}p_{x,t} + \frac{p_{y,t} - s_{n,y}}{\|\boldsymbol{p}_t - \boldsymbol{s}_n\|_2}p_{y,t} - \|\boldsymbol{p}_t - \boldsymbol{s}_n\|_2\right)\Big|_{\boldsymbol{x}_t = \widetilde{\boldsymbol{x}}_{n,t|t-1}\left(\boldsymbol{r}_{n,t}^{(j)}\right)}$$

$$\boldsymbol{B}_{n,t}\left(\boldsymbol{r}_{n,t}^{(j)}\right) = \left(\frac{p_{x,t} - s_{n,x}}{\|\boldsymbol{p}_t - \boldsymbol{s}_n\|_2}, 0, \frac{p_{y,t} - s_{n,y}}{\|\boldsymbol{p}_t - \boldsymbol{s}_n\|_2}, 0\right)\Big|_{\boldsymbol{x}_t = \widetilde{\boldsymbol{x}}_{n,t|t-1}\left(\boldsymbol{r}_{n,t}^{(j)}\right)}.$$

Equations (47) and (54) as well as the switching probabilities π_{ij} conform to a discrete-time CDLS, where the state variable includes $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$, and $\boldsymbol{r}_t$ is discrete and takes values in the finite set $\mathcal{M}$.

We apply the distributed MKF in Section IV to the CDLS. It is noted that $\widetilde{y}_{n,t}(\cdot)$ and $\boldsymbol{B}_{n,t}(\cdot)$ in the observation (54) are related to the state variable $\boldsymbol{r}_t$. At each time step t and for each $\boldsymbol{r} \in \mathcal{M}$, one average consensus algorithm is required for computing $\phi_t(\boldsymbol{r})$, $\varsigma_t(\boldsymbol{r})$, and $\psi_t(\boldsymbol{r})$, where $|\mathcal{M}| = 3$. In the

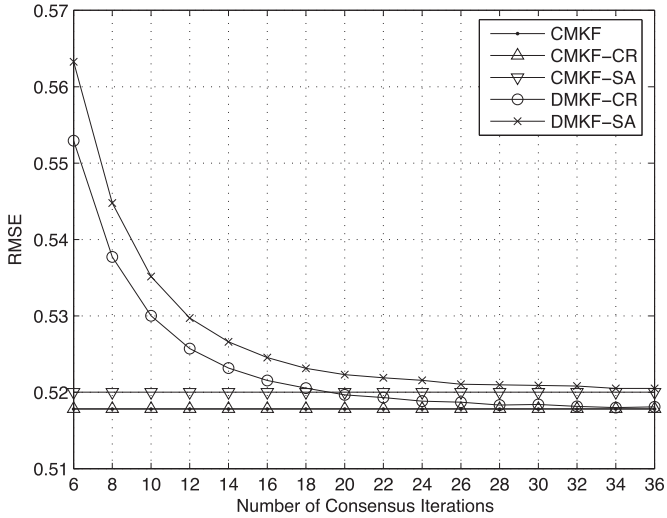

Fig. 4. RMSEs versus number of consensus iterations. The variance of observation noise is set to $\sigma_n^2 = 2.0$. The number of particles is $J = 40$.

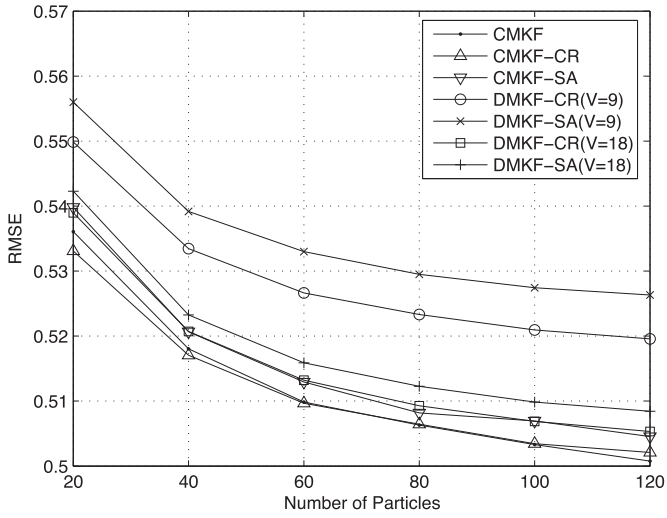

Fig. 5. RMSEs versus number of particles J. The variance of observation noise is set to $\sigma_n^2 = 2.0$. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA.

simulations, the prior distribution of the initial target states $\boldsymbol{x}_0$ and $\boldsymbol{r}_0$ and the statistical characteristics of sensors are all the same as that in Section V-A. The resulting algorithm is summarized in Algorithm II.

Algorithm II Distributed MKF for Constant-Turn Models With Known Turn Rate and Nonlinear Observation

Initialization: For $n = 1, \ldots, N$, the nth sensor generates particles $\{(\boldsymbol{r}_{n,0}^{(j)}, \boldsymbol{x}_{n,0|0}^{(j)}, \boldsymbol{P}_{n,0|0}^{(j)}, w_{n,0}^{(j)})\}_{j=1}^{J}$ from the prior distribution of $\boldsymbol{x}_0$ and $\boldsymbol{r}_0$.

For $t = 1, 2, \ldots$,

1) For $n = 1, \ldots, N$, the nth sensor uses the prior PD in (19) to generate $\boldsymbol{r}_{n,t}^{(j)}$, and constructs the particle $\boldsymbol{r}_{n,1:t}^{(j)} \triangleq \{\boldsymbol{r}_{n,1:t-1}^{(j)}, \boldsymbol{r}_{n,t}^{(j)}\}$ for $j = 1, \ldots, J$.
2) For $n = 1, \ldots, N$, the nth sensor linearizes the observation (53) on $\widetilde{\boldsymbol{x}}_{n,t|t-1}(\boldsymbol{r})$ for each $\boldsymbol{r} \in \mathcal{M}$ and obtains (54).
3) Consensus algorithm is applied to compute $\phi_t(\boldsymbol{r})$, $\varsigma_t(\boldsymbol{r})$, and $\psi_t(\boldsymbol{r})$ among all sensors for each $\boldsymbol{r} \in \mathcal{M}$.
4) For $n = 1, \ldots, N$, the nth sensor computes the weight $w_{n,t}^{(j)}$ with (20), where the likelihood is computed with (31) or (41).
5) For $n = 1, \ldots, N$, the nth sensor computes $\boldsymbol{x}_{n,t|t}^{(j)}$ and $\boldsymbol{P}_{n,t|t}^{(j)}$ with (45) and (46).

End for

Fig. 4 presents the RMSEs of DMKF-CR and DMKF-SA for different numbers of consensus iterations. For comparison, the RMSEs of CMKF, CMKF-CR, and CMKF-SA are also described in Fig. 4. The number of particles is set to $J = 40$. The variance of observation noise is set to $\sigma_n^2 = 2.0$. In Fig. 4, the RMSE of CMKF-CR is nearly identical to that of CMKF, and CMKF-SA has little performance loss. As the number of consensus iterations increases, the RMSEs of DMKF-CR and DMKF-SA can be arbitrarily close to that of CMKF.

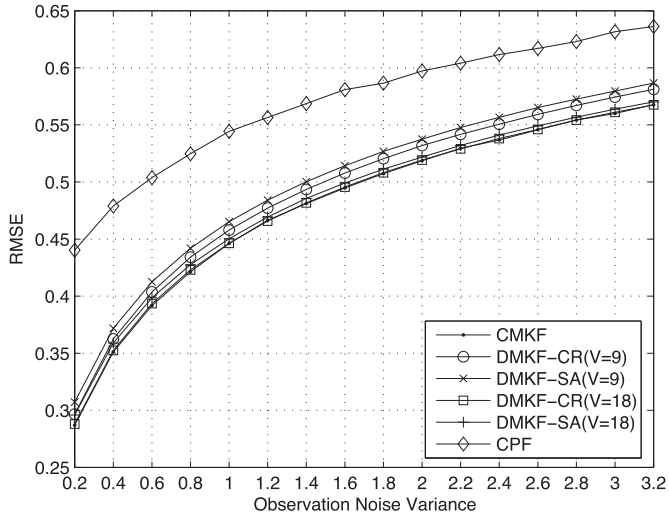

Fig. 6. RMSEs versus observation noise variance σ_n^2. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA. The number of particles is $J = 40$.

Fig. 5 presents the RMSEs of all methods for different numbers of particles, where the variance of observation noise is set to $\sigma_n^2 = 2.0$, and the number of consensus iterations is set to $V = 9$ and $V = 18$. In Fig. 5, the RMSE performances of all methods are improved as the number of particles increases. Compared with DMKF-CR, DMKF-SA has little performance loss, and both are close to the CMKF as the number of consensus iterations increases.

Fig. 6 presents the RMSEs of CMKF, DMKF-CR, and DMKF-SA for different values of observation noise variance. Since the observation equation (53) is nonlinear, we also apply the centralized PF to systems (47) and (53) for the state estimation, and the result is also shown in Fig. 6. We denote the centralized PF as CPF. The number of particles is set to $J = 40$, and the number of consensus iterations is set to $V = 9$ and

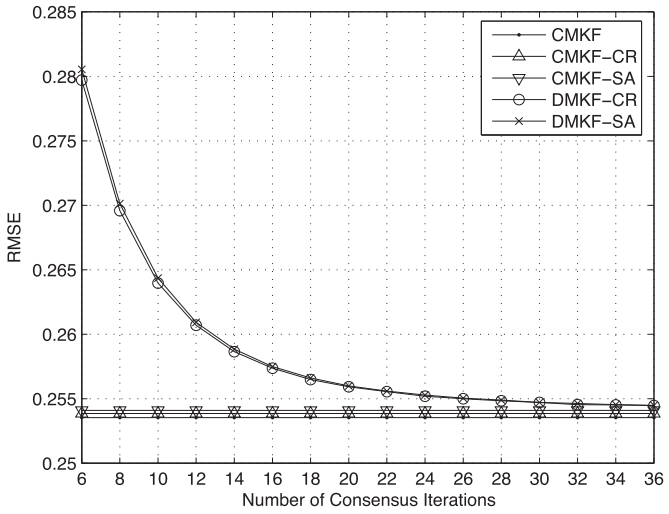

Fig. 7. RMSEs versus number of consensus iterations. The variance of observation noise is set to $\sigma_n^2 = 1.0$. The number of particles is $J = 40$.

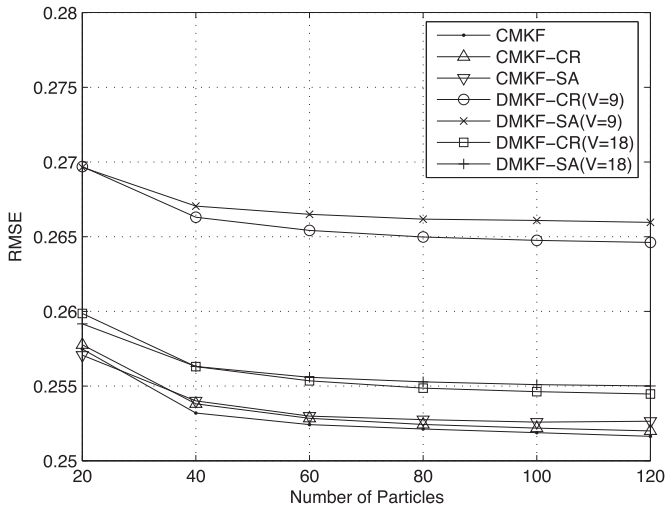

Fig. 8. RMSEs versus number of particles J. The variance of observation noise is set to $\sigma_n^2 = 1.0$. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA.

$V = 18$. In Fig. 6, the RMSEs of DMKF-CR and DMKF-SA with $V = 18$ are arbitrarily close to that of CMKF. The RMSEs of DMKF-CR, DMKF-SA, and CMK are significantly better than that of the CPF.

C. Constant-Turn Model With Unknown Turn Rate and Linear Observation

Different from the constant-turn model with a known turn rate in Section V-A, we consider that the turn rate in this section is described by the Wiener process [28], i.e.,

$$\boldsymbol{r}_t = \boldsymbol{r}_{t-1} + \omega_t \tag{55}$$

where ω_t is zero-mean Gaussian noise with variance $\sigma_{\omega,t}^2$. The dynamic of the target state $\boldsymbol{x}_t$ is described by (47). The observation of the nth sensor is described by (50). Equations (47), (50), and (55) conform to a discrete-time CDLS, whereas the state variable includes $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$, and $\boldsymbol{r}_t$ is a continuous variable.

In the simulations, we choose the prior distribution of $\boldsymbol{r}_0$ as $\boldsymbol{r}_0 \sim \mathcal{N}(0, \sigma_{\boldsymbol{r}_0}^2)$ with $\sigma_{\boldsymbol{r}_0} = 3^\circ$. The noise variance of ω_t is set to $\sigma_{\omega,t} = 3^\circ$. The state noise variance of $\boldsymbol{u}_t(\boldsymbol{r}_t)$ in (47) is set to $\sigma_{\boldsymbol{r}_t} = 3^\circ$. The prior distribution of $\boldsymbol{x}_0$ and the statistical characteristics of sensors are all the same as that in Section V-A. Since the matrices $\boldsymbol{B}_{n,t}$ and $\boldsymbol{R}_{n,t}$ in the observation equation (50) are independent of $\boldsymbol{r}_t$ and time invariant, then $\phi_t(\cdot)$, $\varsigma_t(\cdot)$, and $\psi_t(\cdot)$ in (38)–(40) are also independent of $\boldsymbol{r}_t$, and $\psi_t(\cdot)$ is time invariant. We denote them as ϕ_t, ς_t, and ψ, where ψ can be computed with the average consensus algorithm before the whole estimation process. Hence, at each time t and for all particles, one average consensus algorithm is required for computing φ_t and ς_t. In this case, the synchronization of random-number generators in the network is not required. Each sensor independently generates the particles and updates their weights from the global likelihood approximation. The resulting algorithm has the same structure as Algorithm I.

Fig. 7 presents the RMSEs of all methods for different numbers of consensus iterations. The number of particles is set to $J = 40$. The variance of observation noise is set to $\sigma_n^2 = 1.0$. In Fig. 7, the RMSE of CMKF-CR is nearly identical to that of CMKF, and CMKF-SA only has little performance loss. As the number of consensus iterations increases, the RMSEs of DMKF-CR and DMKF-SA can be arbitrarily close to that of CMKF.

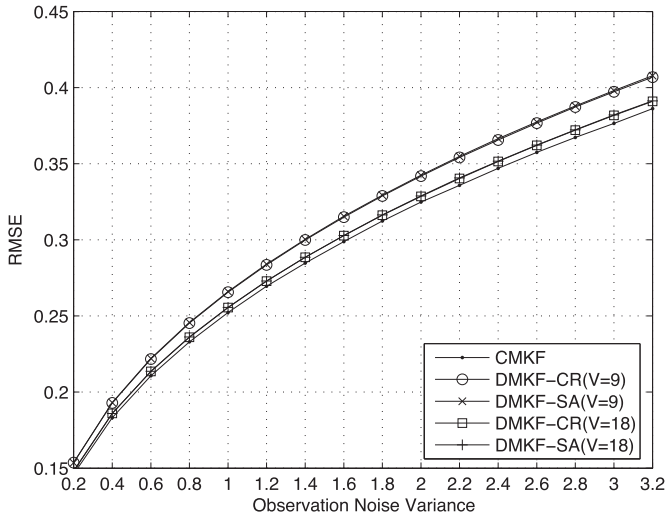

Fig. 9. RMSEs versus observation noise variance σ_n^2. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA. The number of particles is $J = 40$.

Fig. 8 presents the RMSEs of all methods for different numbers of particles. The variance of observation noise is set to $\sigma_n^2 = 1.0$. The number of consensus iterations is set to $V = 9$ and $V = 18$. In Fig. 8, the RMSE performances of all methods are improved as the number of particles increases. The RMSEs of DMKF-CR and DMKF-SA are close to that of the CMKF as the number of consensus iterations increases.

Fig. 9 presents the RMSEs of CMKF, DMKF-CR, and DMKF-SA for different values of observation noise variance. The number of particles is set to $J = 40$, and the number of consensus iterations is set to $V = 9$ and $V = 18$. The RMSEs of DMKF-CR and DMKF-SA with $V = 18$ are arbitrarily close to that of CMKF.

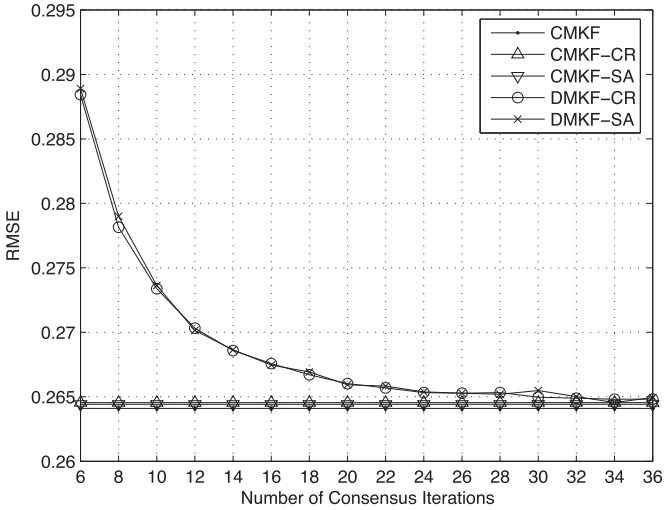

Fig. 10. RMSEs versus number of consensus iterations. The variance of observation noise is set to $\sigma_n^2 = 0.5$. The number of particles is $J = 40$.

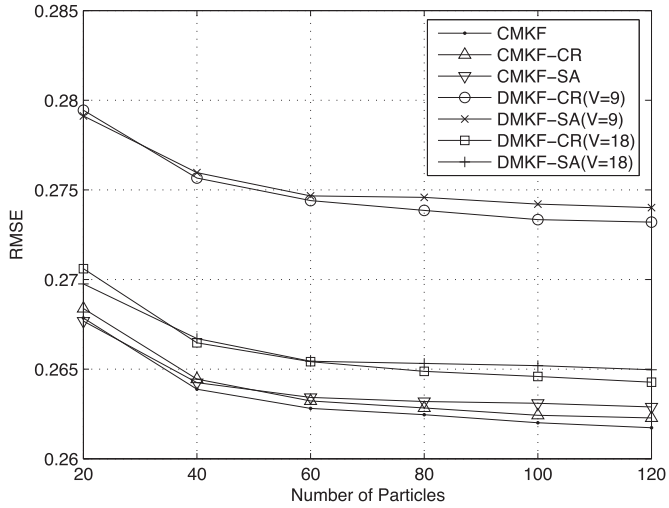

Fig. 11. RMSEs versus number of particles J. The variance of observation noise is set to $\sigma_n^2 = 0.5$. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA.

D. Constant-Turn Models With Unknown Turn Rate and Nonlinear Observation

The dynamics of the target states $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$ are described as that in (47) and (55), respectively, which are the same as the example in Section V-C. Different from the linear observation in Section V-C, we consider the nonlinear observation in this section, and the observation of the nth sensor is described as that in (53).

After the generation of particle $\boldsymbol{r}_{n,t}^{(j)}$ at the nth sensor, it linearizes the observation (53) on the point $\widetilde{\boldsymbol{x}}_{n,t|t-1}(\boldsymbol{r}_{n,t}^{(j)}) \triangleq \boldsymbol{A}_t(\boldsymbol{r}_{n,t}^{(j)})\widetilde{\boldsymbol{x}}_{n,t-1|t-1}$, where $\widetilde{\boldsymbol{x}}_{n,t-1|t-1} = \sum_{j=1}^{J} w_{n,t-1}^{(j)} \boldsymbol{x}_{n,t-1|t-1}^{(j)}$. The linearized observation is described as that in (54). Equations (47), (54), and (55) conform to a discrete-time CDLS, where the state variable includes $\boldsymbol{x}_t$ and $\boldsymbol{r}_t$, and $\boldsymbol{r}_t$ is a continuous variable. We apply the distributed MKF in Section IV to the CDLS. It is noted that $\widetilde{y}_{n,t}(\cdot)$ and $\boldsymbol{B}_{n,t}(\cdot)$ in the observation equation (54) are related to the state variable $\boldsymbol{r}_t$. When DMKF-CR and DMKF-SA are applied, the local random-number generators in the entire network are synchronized. The average consensus algorithm is executed for each particle to compute $\phi_t(j)$, $\varsigma_t(j)$, and $\psi_t(j)$ in (38)–(40). In the simulations, the prior distribution of the initial target states $\boldsymbol{x}_0$ and $\boldsymbol{r}_0$ and the statistical characteristics of sensors are all the same as that in Section V-C. The resulting algorithm has the same structure as Algorithm II.

Fig. 10 presents the RMSEs of all methods for different numbers of consensus iterations. The number of particles is set to $J = 40$. The variance of observation noise is set to $\sigma_n^2 = 0.5$. In Fig. 10, the RMSEs of CMKF-CR and CMKF-SA are nearly identical to that of CMKF. As the number of consensus iterations increases, the RMSEs of DMKF-CR and DMKF-SA can be arbitrarily close to that of CMKF.

Fig. 11 presents the RMSEs of all methods for different numbers of particles. The variance of observation noise is set to $\sigma_n^2 = 0.5$. The number of consensus iterations is set to $V = 9$ and $V = 18$. The RMSEs of DMKF-CR and DMKF-SA with $V = 18$ are arbitrarily close to that of the CMKF.

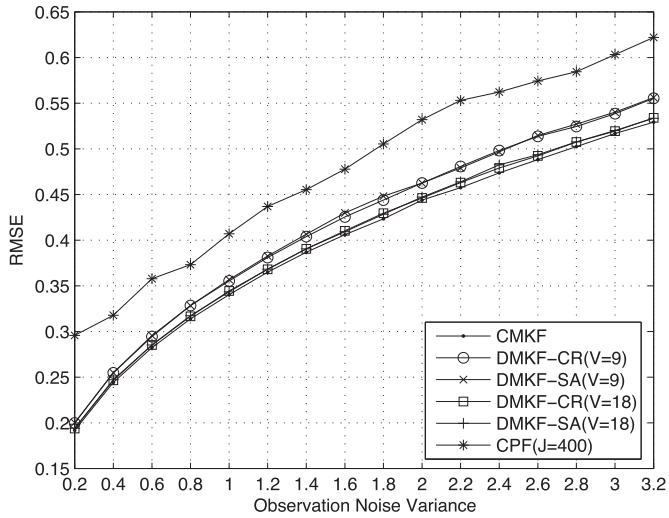

Fig. 12. RMSEs versus observation noise variance σ_n^2. The number of consensus iterations is set to $V = 9$ and $V = 18$ for DMKF-CR and DMKF-SA. The number of particles is $J = 40$. The number of particles for CPF is $J = 400$.

Fig. 12 presents the RMSEs of CMKF, DMKF-CR, and DMKF-SA for different values of observation noise variance. The number of particles is set to $J = 40$, and the number of consensus iterations is set to $V = 9$ and $V = 18$. Since the observation equation (53) is nonlinear, we also apply the CPF to systems (47), (53), and (55) for the state estimation. Since the RMSE of the CPF with the number of particles $J = 40$ is too big in the simulations, we choose the number of particles $J = 400$ for the CPF. The estimation result is also shown in Fig. 12. In Fig. 12, the RMSEs of DMKF-CR and DMKF-SA with $V = 18$ are arbitrarily close to that of CMKF. The RMSEs of DMKF-CR, DMKF-SA, and CMK are significantly better than that of the CPF.

VI. Conclusion

We have considered the distributed estimation for the CDLSs in a WSN without a data FC. Each sensor only exchanges

its local information with its neighbors and carries out local computation. We developed a distributed MKF for the CDLSs. Each sensor in the network runs a local MKF and interacts with other sensors to generate a mixture Gaussian representation of the global posterior distribution. We applied the cubature rule for the calculation of the global likelihood $p(\boldsymbol{y}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t-1})$, to enable the distributed implementation of the likelihood computation. Consensus algorithms are executed to fuse the observations from all sensors when calculating the likelihood at each sensor. Distributed schemes are also executed to calculate the posterior $p(\boldsymbol{x}_t|\boldsymbol{r}_{1:t}, \boldsymbol{y}_{1:t})$ at each sensor. Finally, we applied the distributed MKF to the maneuvering target tracking. We considered the constant-turn models with a known turn rate where the maneuvering variable is discrete and the constant-turn models with an unknown turn rate where the maneuvering variable is continuous. We also considered that the observation equation is linear and nonlinear. Simulation results illustrate that the performance of the distributed MKF is close to that of the centralized MKF in various scenarios of maneuvering target tracking.

REFERENCES

[1] F. Zhao and L. J. Guibas, *Wireless Sensor Networks: An Information Processing Approach.* Amsterdam, The Netherlands: Morgan Kaufmann, 2004.

[2] T. He *et al.*, "Achieving real-time target tracking using wireless sensor networks," in *Proc. IEEE Real-Time Embed. Technol. Appl. Symp.*, Apr. 2006, pp. 37–48.

[3] W. Li, Z. Yang, and H. Hu, "Sequential particle-based sum–product algorithm for distributed inference in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 62, no. 1, pp. 341–348, Jan. 2013.

[4] Z. Shen, H. Jiang, and Z. Yan, "Fast data collection in linear duty-cycled wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 4, pp. 1951–1957, May 2014.

[5] S. Guo, C. Wang, and Y. Yang, "Joint mobile data gathering and energy provisioning in wireless rechargeable sensor networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 12, pp. 2836–2852, Dec. 2014.

[6] T. Wang, R. C. de Lamare, and A. Schmeink, "Alternating optimization algorithms for power adjustment and receive filter design in multihop wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 1, pp. 173–184, Jan. 2015.

[7] W. Li and Y. Jia, "Consensus-based distributed multiple model UKF for jump Markov nonlinear systems," *IEEE Trans. Autom. Control.*, vol. 57, no. 1, pp. 230–236, Jan. 2012.

[8] W. Li, Y. Jia, J. Du, and J. Zhang, "Distributed multiple-model estimation for simultaneous localization and tracking with NLOS mitigation," *IEEE Trans. Veh. Technol.*, vol. 62, no. 6, pp. 2824–2830, Jul. 2013.

[9] R. Carli and S. Zampieri, "Network clock synchronization based on the second-order linear consensus algorithm," *IEEE Trans. Autom. Control.*, vol. 59, no. 2, pp. 409–422, Feb. 2014.

[10] F. Penna and S. Stańczak, "Decentralized eigenvalue algorithms for distributed signal detection in wireless networks," *IEEE Trans. Signal Process.*, vol. 63, no. 2, pp. 427–440, Jan. 2015.

[11] R. Olfati-Saber and R. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Autom. Control.*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

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

[13] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *Proc. 44th IEEE Conf. Decision Control/Eur. Control Conf.*, Dec. 2005, pp. 8179–8184.

[14] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Distributed Kalman filtering based on consensus strategies," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 4, pp. 622–633, May 2008.

[15] F. S. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Trans. Autom. Control*, vol. 55, pp. 2069–2084, Sep. 2010.

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

[17] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[18] O. Cappé, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proc. IEEE*, vol. 95, no. 5, pp. 899–924, May 2007.

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

[20] O. Ozdemir, R. Niu, and P. K. Varshney, "Tracking in wireless sensor networks using particle filtering: Physical layer considerations," *IEEE Trans. Signal Process.*, vol. 57, no. 5, pp. 1987–1999, May 2009.

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

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

[23] J. Read, K. Achutegui, and J. Míguez, "A distributed particle filter for nonlinear tracking in wireless sensor networks," *Signal Process.*, vol. 98, pp. 121–134, May 2014.

[24] O. Hlinka, F. Hlawatsch, and P. M. Djurić, "Consensus-based distributed particle filtering with distributed proposal adaptation," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3029–3041, Jun. 2014.

[25] A. Mohammadi and A. Asif, "Distributed consensus + innovation particle filtering for bearing range tracking with communication constraints," *IEEE Trans. Signal Process.*, vol. 63, no. 3, pp. 620–635, Feb. 2015.

[26] R. Chen and J. S. Liu, "Mixture Kalman filters," *J. R. Stat. Soc. B, Stat. Methodol.*, vol. 62, no. 3, pp. 493–508, 2000.

[27] K. J. Kim, M. Pun, and R. A. Iltis, "Joint carrier frequency offset and channel estimation for uplink MIMO-OFDMA systems using parallel Schmidt Rao-Blackwellized particle filters," *IEEE Trans. Commun.*, vol. 58, no. 9, pp. 2697–2708, Sep. 2010.

[28] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I: Dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, Oct. 2003.

[29] I. Arasaratnam, S. Haykin, and R. J. Elliott, "Discrete-time nonlinear filtering algorithms using Gauss–Hermite quadrature," *Proc. IEEE*, vol. 95, no. 5, pp. 953–976, May 2007.

[30] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Autom. Control*, vol. 54, no. 6, pp. 1254–1269, Jun. 2009.

[31] C. Xu and K. Lum, "Jump-Markov process estimation based on distributed Rao-Blackwellized particle filter," in *Proc. 11th IEEE ICCA*, Jun. 2014, pp. 953–958.

[32] C. N. Hadjicostis and T. Charalambous, "Average consensus in the presence of delays in directed graph topologies," *IEEE Trans. Autom. Control*, vol. 59, no. 3, pp. 763–768, Mar. 2014.

[33] F. Fagnani and S. Zampieri, "Average consensus with packet drop communication," *SIAM J. Control Optim.*, vol. 48, no. 1, pp. 102–133, Feb. 2009.

[34] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Trans. Autom. Control*, vol. 50, no. 2, pp. 169–182, Feb. 2005.

[35] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, Los Angeles, CA, USA, Apr. 2005, pp. 63–70.

[36] D. Mosk-Aoyama and D. Shah, "Fast distributed algorithms for computing separable functions," *IEEE Trans. Inf. Theory*, vol. 54, no. 7, pp. 2997–3007, Jul. 2008.

Yihua Yu (M'13) received the B.S. and Ph.D. degrees from the School of Mathematical Sciences, Peking University, Beijing, China, in 1997 and 2003, respectively.

He is currently an Associate Professor with the School of Science, Beijing University of Posts and Telecommunications. His research interests include statistical signal processing for digital communication, linear and nonlinear Bayesian estimation, and tracking in wireless networks.