2020 59th IEEE Conference on Decision and Control (CDC)
Jeju Island, Republic of Korea, December 14-18, 2020

Distributed Kalman Filter for 3-D Moving Object Tracking over Sensor Networks

Pengxiang Zhu and Wei Ren

Abstract— **This paper studies the problem of distributed state estimation (DSE) over sensor networks. Unlike the existing filtering algorithms that only consider targets moving in two-dimension (2-D) environments, we address this problem in three-dimension (3-D) scenarios where each agent equipped with the communication and sensing capabilities cooperatively track the state of a 3-D moving object. First, it is shown that the existing distributed Kalman filter (DKF) algorithms cannot solve the quaternion-based six degree-of-freedom (6-DoF) motion tracking. Then, a novel DKF applicable for the 3-D tracking is introduced for a general nonlinear system. The proposed algorithm is fully distributed and robust to time-varying communication topologies and changing blind agents (the agents that lose sight of the whole target object). Finally, we apply the proposed algorithm to a camera network to track the 6-DoF pose (position and orientation) of a moving target object. The effectiveness of our approach is demonstrated through Monte-Carlo simulations.**

I. Introduction

State estimation in sensor networks has a wide range of applications such as target tracking, environmental monitoring and surveillance. It is assumed that a state of interest is evolving according to noisy dynamics, and each agent may or may not get measurements that are related to the state. The objective is to obtain an accurate estimator of this state on every agent. In conventional centralized algorithms, all the agents send their measurements to a fusion center that runs a centralized Kalman filter (CKF) to get an optimal estimator. This estimator is then sent back to every agent. This approach requires expensive communication and computational resources. Moreover, it has the potential for the failure on the fusion center. In contrast, distributed approaches that have the advantages of effectiveness, scalability and robustness have drawn more attentions in the research community. In 3-D environments, quaternions have been introduced to express orientations due to its unambiguity and computational efficiency. Furthermore, compared to the Euler angle expression, quaternions avoid singularity when calculating rotations [1]. However, quaternions are not valid vector quantities, which makes the existing DKF algorithms not suitable for the quaternion-based 3-D motion tracking in sensor networks.

Due to the aptitude for distributed computing, most of the existing DKF algorithms are derived from the *information filter* (IF) (information form of the Kalman filter) which propagates and updates, instead of the state estimate and the covariance, the information pair that contains the information matrix and the information vector [1]. The consensus algorithm as a tool of distributed averaging has been exploited in DSE. Three kind of consensus filters are proposed in [2] where the consensus-on-information algorithm performs the consensus on the prior information pair and the consensus-on-measurements algorithm performs the consensus on the measurements. These two algorithms are then combined to provide a hybrid consensus filter. Ref. [3] develops a Kullback–Leibler average consensus filter where the local measurement is first used to update the local prior information pairs and then the consensus is exploited on the resulting posterior information pairs. Some other consensus filtering algorithms derived from the IF can be found in [4]–[7].

Apart from the consensus-based algorithms that require several communication iterations for each measurement, a more efficient kind of DKF is based on the covariance intersection (CI) algorithm presented in [8], [9]. The CI algorithm is proposed to obtain an improved and consistent estimator from the fusion of multiple estimators with unknown correlations by using a convex combination of the local information pairs. The weights are chosen to minimize the trace or determinant of the fused covariance. Refs. [10] and [11] let each agent compute an estimator by using its own measurements independently and then fuse the resulting posterior information pairs among the neighborhood with CI to obtain an improved estimator. Only the posterior information pairs are transmitted. Another typical CI-based approach is proposed in [12]–[14] where the prior information pairs are first fused with CI and then the resulting prior estimator at every agent is further updated with all the measurements among the neighborhood. Here, the prior information pairs and the local measurements are transmitted. Some other CI-based DKF can be found in [15], [16].

Both the IF-derived consensus filters and the CI-based DKF algorithms need to compute the information vector. However, we cannot calculate the information vector for a quaternion due to the mismatching dimension between a quaternion and the corresponding covariance [17]. We further explore the existing DKF without the need of computing the information vector. The Kalman consensus filter (KCF) in

This work was supported by National Science Foundation under Grant CMMI-2027139.

Pengxiang Zhu and Wei Ren are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521, USA (pzhu008@ucr.edu, ren@ee.ucr.edu).

[1]Given a true value $\mathbf{x}$, $\bar{\mathbf{x}}$ and $\hat{\mathbf{x}}$ denote, respectively, the prior and posterior estimates with the corresponding covariances $\bar{\mathbf{p}}$ and $\mathbf{p}$. The information pair can be divided into the prior and posterior information pair. The prior information pair contains the information matrix $\bar{\mathbf{p}}^{-1}$ and the information vector $\bar{\mathbf{p}}^{-1}\bar{\mathbf{x}}$, while the posterior information pair contains the information matrix $\mathbf{p}^{-1}$ and the information vector $\mathbf{p}^{-1}\hat{\mathbf{x}}$.

978-1-7281-7446-4/20/$31.00 ©2020 IEEE

[18] and the Generalized KCF (GKCF) in [19] perform an average consensus on the prior estimates in the update step. KCF use equal weights, which causes large estimation errors with blind agents. This issue is avoided by using GKCF that weights the prior estimates by their covariances. Ref. [20] presents the diffusion Kalman filter where each agent first updates the local estimates using its own and the neighbors' measurements, and then computes a convex combination of the resulting estimates. Nevertheless, quaternions are not in the vector space. Then, the arithmetic mean (average consensus) or a convex combination computed is no longer a valid quaternion and has no physical meaning, which renders the approaches in [18]–[20] not applicable.

Indirect Kalman filter has been proposed in [17], [21] to address the problem of *single* robot 3-D localization where the quaternion is used to represent the orientation. But how to fuse the information especially the quaternions from other sensors in a sensor network has not been addressed. From the above observations, it is clear that the existing DKF algorithms are not applicable for the quaternion-based 3-D tracking, which limits their applications in many real-world scenarios where the target exhibits 3-D motion. Hence, the main contribution of this paper is that a novel DKF suitable for the 3-D DSE over sensor networks is proposed and further applied to camera networks. The proposed approach is fully distributed and applicable for generic target motion and measurement models. It can also handle time-varying communication topologies and changing blind agents.

II. Preliminaries

A. Quaternion

A quaternion consists of a vector and scalar portion as

$$\bar{q} = q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} + q_4.$$

For notation simplicity, $\bar{q}$ can be further written as a four-dimensional column matrix given by

$$\bar{q} = \begin{bmatrix}\mathbf{q}\\ q_4\end{bmatrix} = \begin{bmatrix}q_1 & q_2 & q_3 & q_4\end{bmatrix}^\mathsf{T}.$$

Orientation is represented as a unit quaternion [22] which satisfies $|\bar{q}| = \sqrt{|\mathbf{q}|^2 + q_4^2} = 1$. A rotation can be represented by a unit quaternion

$$\bar{q} = \begin{bmatrix}\mathbf{q}\\ q_4\end{bmatrix} = \begin{bmatrix}\mathbf{m}\sin(\theta/2)\\ \cos(\theta/2)\end{bmatrix}, \tag{1}$$

where $\mathbf{m}$ is the unit vector defining the rotation axis and θ is the angle of rotation. A rotation can also be described by a rotation matrix $\mathbf{R}$ which is related to the unit quaternion $\bar{q}$ by

$$\mathbf{R} = \left(2q_4^2 - 1\right)\mathbf{I}_3 - 2q_4\lfloor\mathbf{q}\times\rfloor + 2\mathbf{q}\mathbf{q}^\mathsf{T}$$

where $\lfloor\cdot\times\rfloor$ denotes the skew symmetric matrix. Moreover, $\bar{q}$ and $-\bar{q}$ describe the same rotation [23].

The error vector and its covariance are usually expressed in terms of the arithmetic difference between the true and estimated values. However, using this representation for a quaternion would make the corresponding covariance singular [24]. Instead, the quaternion error $\delta\bar{q}$ is represented as a rotation between the estimated and true quaternion as $\bar{q} = \hat{\bar{q}} \otimes \delta\bar{q}$, where $\otimes$ denotes the quaternion multiplication. When representing the uncertainty of a quaternion error, a minimal representation of the 3-dimension vector is required [25]. Since the rotation associated with $\delta\bar{q}$ can be assumed to be very small, the mapping between $\delta\bar{q}$ and the minimal representation, the rotation angle error $\delta\boldsymbol{\theta} \in \mathcal{R}^3$, is obtained from (1) with small angle approximation as

$$\delta\bar{q} = \begin{bmatrix}\delta\mathbf{q}\\ \delta q_4\end{bmatrix} = \begin{bmatrix}\mathbf{m}\sin(\delta\theta/2)\\ \cos(\delta\theta/2)\end{bmatrix} \approx \begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\theta}\\ 1\end{bmatrix},$$

where $\delta\boldsymbol{\theta} = \mathbf{m}\delta\theta$. Then, the uncertainty of $\delta\bar{q}$ is represented as the covariance of $\delta\boldsymbol{\theta}$. It is evident that for a quaternion estimate $\hat{\bar{q}}$, we cannot compute its information vector, as the dimension for the covariance of $\hat{\bar{q}}$ is 3×3 but $\hat{\bar{q}}$ is 4×1.

B. Notation and Definitions

$\mathbf{I}_{m\times n}$ ($\mathbf{0}_{m\times n}$) is the identity (zero) matrix of size $m\times n$. If $m = n$, for simplicity, we use $\mathbf{I}_m$ ($\mathbf{0}_m$) to denote the square identity (zero) matrix. We denote G, T and C_i, respectively, as the global frame, the target's body frame and the ith camera frame. ${}^T_G\bar{q}$, the target's orientation, describes the rotation from G to T. ${}^T_G\mathbf{R}$ is the rotation matrix associated with ${}^T_G\bar{q}$. ${}^G\mathbf{p}_T$, the target's global position, denotes the position of T in G. For vector quantities, the error $\delta\mathbf{x}$ is defined as the standard additive error $\delta\mathbf{x} = \mathbf{x} - \hat{\mathbf{x}}$. For a vector $\mathbf{x} = [x\ y\ z]^\mathsf{T}$, the projection function is defined as $\Pi(\mathbf{x}) = \frac{1}{z}[x\ y]^\mathsf{T}$ whose state Jacobian

$$\mathbf{H}_p(x) = \frac{1}{z}\begin{bmatrix}1 & 0 & -\frac{x}{z}\\ 0 & 1 & -\frac{y}{z}\end{bmatrix}.$$

We define a directed communication graph $\mathcal{G}^k = (\mathcal{V}, \mathcal{E}^k)$, where $\mathcal{V}$ is the agent set and $\mathcal{E}^k$ is the edge set defined as $\mathcal{E}^k \subseteq \mathcal{V}\times\mathcal{V}$. $\mathcal{E}^k$ stands for the communication links between agents at timestep k. We assume that self edge $(i,i) \in \mathcal{E}^k$, $\forall i \in \mathcal{V}$, exists in the communication graph. If there exists an edge $(j,i) \in \mathcal{E}^k$, where $j \neq i$, which means that agent i can receive information from agent j, then agent j is a communicating neighbor of agent i. The communicating neighbor set of agent i at timestep k can be defined as $\mathcal{N}_i^k = \{i | (l,i) \in \mathcal{E}^k, l \in \mathcal{V}\}$. Note that $i \in \mathcal{N}_i^k$.

III. 3-D Distributed State Estimation Algorithm

A. Problem Formulation

Consider a network of agents, where each agent has the ability to communicate with its neighbors and sense the target with limited sensing region. The target is moving in a 3-D environment. Without loss of generality, we represent the 3-D motion of the target with the following state,

$$\mathbf{x} = \begin{bmatrix}{}^T_G\bar{q}\\ \mathbf{x}_v\end{bmatrix} = \begin{bmatrix}{}^T_G\bar{q}^\mathsf{T} & {}^G\mathbf{p}_T{}^\mathsf{T} & {}^G\mathbf{v}_T{}^\mathsf{T}\end{bmatrix}^\mathsf{T}, \tag{2}$$

where $\mathbf{x}$ includes the target's 6-DoF pose, ${}^T_G\bar{q}$ and ${}^G\mathbf{p}_T$ in addition to the target's global linear velocity ${}^G\mathbf{v}_T$; $\mathbf{x}_v$ contains all the vector quantities in $\mathbf{x}$. Consider the following nonlinear motion model as the dynamics of the target object:

$$\mathbf{x}^k = \mathbf{f}(\mathbf{x}^{k-1}, \mathbf{n}^{k-1}), \tag{3}$$

where $\mathbf{x}^k$ is the target's state at timestep k, $\mathbf{n}$ is the zero-mean white Gaussian noise with covariance $\mathbf{O}$. The local measurement $\mathbf{z}_i^k$ obtained by each agent i, $i \in \mathcal{V}$, is given by the following general nonlinear model:

$$\mathbf{z}_i^k = \mathbf{h}_i(\mathbf{x}^k, \mathbf{w}_i^k), \tag{4}$$

where $\mathbf{w}_i$ is the local measurement noise assumed to be zero-mean white Gaussian with covariance $\mathbf{R}_i$. We further suppose that the measurement and target process noises are mutually uncorrelated. The objective is to compute an accurate estimate of the target's state $\mathbf{x}$ on every agent by only using the information from itself and the one-hop communicating neighbors.

B. Proposed Distributed Kalman Filter

Suppose that at timestep k, each agent maintains a prior estimator $(\bar{\mathbf{x}}_i^k, \bar{\mathbf{p}}_i^k)$ after propagation. Now, agent i aims to update its local estimator $(\bar{\mathbf{x}}_i^k, \bar{\mathbf{p}}_i^k)$ by using its local information and the information from its one-hop communicating neighbors. The first step is to fuse all prior estimation pairs among the neighborhood, i.e., $(\bar{\mathbf{x}}_j^k, \bar{\mathbf{p}}_j^k)$, $\forall j \in \mathcal{N}_i^k$. Recall that we cannot directly compute the information vector of a quaternion and then use the consensus or CI algorithms to fuse the prior estimation pairs. Instead, we first *weighted synchronize* the prior estimation pairs to reduce its uncertainty. The weight π_j satisfies $\pi_j \in [0, 1]$ and $\sum_{j\in\mathcal{N}_i} \pi_j = 1$, which makes sure that we do not overuse the information among the neighborhood. For the estimates of the vector quantities $\mathbf{x}_v$ in $\mathbf{x}$, we compute

$$\check{\mathbf{x}}_{v_i}^k = \sum_{j\in\mathcal{N}_i^k} \pi_j^k \bar{\mathbf{x}}_{v_j}^k. \tag{5}$$

Note that for the quaternions, our objective is to average the orientations described by the quaternions, not the average of the quaternion. Simply taking the same form of $\mathbf{x}_{v_i}$ cannot even get a valid quaternion (e.g., change the sign of a quaternion should not change the described orientation). Here, we employ the method in [26] which provides a closed form solution of the averaged quaternion ${}^{T_i}_{G}\check{\bar{q}}$ by the following maximization procedure

$$\begin{aligned} {}^{T_{i,k}}_{G}\check{\bar{q}} &= \arg\max_{\bar{q}\in\mathbb{S}^3} \bar{q}^{\mathsf{T}}\mathbf{M}\bar{q}, \\ \mathbf{M} &= \sum_{j\in\mathcal{N}_i^k} \pi_j^k ({}^{T_{j,k}}_{G}\bar{\bar{q}})^{\mathsf{T}} {}^{T_{j,k}}_{G}\bar{\bar{q}}, \end{aligned} \tag{6}$$

where ${}^{T_{j,k}}_{G}\bar{\bar{q}}$ is agent j's prior estimate of ${}^{T_k}_{G}\bar{q}$; $\mathbb{S}^3$ denotes the unit 3-sphere. Solving (6) in fact gives a quaternion that minimizes the weighted sum of the orientation errors. We define a compatible symbol $\boxtimes$ for computing the weighted average and then we obtain

$$\check{\mathbf{x}}_i^k = \begin{bmatrix} {}^{T_{i,k}}_{G}\check{\bar{q}} \\ \check{\mathbf{x}}_{v_i}^k \end{bmatrix} = \sum_{j\in\mathcal{N}_i^k} \pi_j^k \boxtimes \bar{\mathbf{x}}_j^k.$$

As for the synchronized covariance, we can directly compute $\check{\mathbf{p}}_i^k = \sum_{j\in\mathcal{N}_i^k} \pi_j^k \bar{\mathbf{p}}_j^k$, since the quaternion error is represented by the error of the rotational angle that is a vector quantity. The weight π_j^k is chosen to minimize the determinant or the trace of $\check{\mathbf{p}}_i^k$.

For the sake of computational simplicity, we use the simplified algorithm in [27] to calculate π_j^k as

$$\pi_j^k = \frac{1/\mathrm{Tr}(\bar{\mathbf{p}}_j)}{\sum_{j\in\mathcal{N}_i^k} 1/\mathrm{Tr}(\bar{\mathbf{p}}_j)},$$

where $\mathrm{Tr}(\cdot)$ computes the trace of a matrix. Clearly, more weights will be given to the prior estimation pairs with small covariances.

The second step is to fuse the intermediate estimation pair $(\check{\mathbf{x}}_i^k, \check{\mathbf{p}}_i^k)$ with all the local measurements $\mathbf{z}_j^k$, $\forall j \in \mathcal{N}_i^k$. If agent j cannot sense the target directly, we assume infinite uncertainties in $\mathbf{z}_j^k$, that is, $\mathbf{R}_j^k = \infty$. After linearization of $\mathbf{z}_i^k$ about the current estimated state, we compute

$$\mathbf{s}_i^k = (\mathbf{H}_i^k)^{\mathsf{T}}(\mathbf{R}_i^k)^{-1}\mathbf{H}_i^k, \quad \mathbf{y}_i^k = (\mathbf{H}_i^k)^{\mathsf{T}}(\mathbf{R}_i^k)^{-1}\tilde{\mathbf{z}}_i^k, \tag{7}$$

where $\tilde{\mathbf{z}}_i = \mathbf{z}_i - \mathbf{h}_i(\bar{\mathbf{x}}_i)$ and $\mathbf{H}_i = \frac{\partial \mathbf{h}_i}{\partial \mathbf{x}_i}(\bar{\mathbf{x}}_i)$. Then, we obtain the updated covariance $\mathbf{p}_i^k$ and the state correction $\delta\mathbf{x}_i^k$ according to

$$\begin{aligned} \mathbf{p}_i^k &= \left[\left(\check{\mathbf{p}}_i^k\right)^{-1} + \sum_{j\in\mathcal{N}_i^k} \mathbf{s}_j^k \right]^{-1}, \\ \delta\mathbf{x}_i^k &= \begin{bmatrix} \delta\boldsymbol{\theta}_i^k \\ \delta\mathbf{x}_{v_i}^k \end{bmatrix} = \mathbf{p}_i^k \sum_{j\in\mathcal{N}_i^k} \mathbf{y}_j^k, \end{aligned} \tag{8}$$

where $\delta\boldsymbol{\theta}_i$ is the orientation correction while $\delta\mathbf{x}_{v_i}$ is the corrections of the vector quantities. Next, we update $\check{\mathbf{x}}_i$ by using $\delta\mathbf{x}_i$. For the vector quantities $\check{\mathbf{x}}_{v_i}$ in $\check{\mathbf{x}}_i$, we have $\hat{\mathbf{x}}_{v_i}^k = \check{\mathbf{x}}_{v_i}^k + \delta\mathbf{x}_{v_i}^k$. We update the quaternion ${}^{T_{i,k}}_{G}\check{\bar{q}}$ according to

$${}^{T_{i,k}}_{G}\hat{\bar{q}} = {}^{T_{i,k}}_{G}\check{\bar{q}} \otimes \delta\bar{q}_i \tag{9}$$

where $\delta\bar{q}_i$ represents a rotation that is supposed to be a unit quaternion. Recall that $\delta\bar{q}_i$ is approximately equal to $[\frac{1}{2}\delta\theta_i^{\mathsf{T}}\ 1]^{\mathsf{T}}$, which is however not a unit quaternion. To obtain a unit quaternion, we let $\delta\bar{q}_i = \frac{1}{\sqrt{1+\frac{1}{4}\delta\boldsymbol{\theta}_i^{\mathsf{T}}\delta\boldsymbol{\theta}_i}} \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta}_i \\ 1 \end{bmatrix}$. We define a compatible symbol $\boxplus$ for updating $\check{\mathbf{x}}_i^k$. Then we have

$$\hat{\mathbf{x}}_i^k = \begin{bmatrix} {}^{T_{i,k}}_{G}\hat{\bar{q}} \\ \hat{\mathbf{x}}_{v_i}^k \end{bmatrix} = \check{\mathbf{x}}_i^k \boxplus \delta\mathbf{x}_i^k. \tag{10}$$

By adding the standard propagation step, the proposed 3-D DKF algorithm is summarised in Algorithm I.

IV. Simulations

In this section, we apply the proposed 3-D DKF to address the DSE problem over a camera network where 10 cameras are employed to track a drone executing 3-D motion (see Fig. 1). Each camera has a limited field of view. The status of which cameras are directly sensing the target over the tracking period is shown in Fig. 2. Clearly, all of the cameras could turn into blind cameras for long time periods. Moreover, each camera's intrinsic parameters are known via prior calibration [28]. We perform extensive Monte-Carlo simulations to validate the effectiveness of the proposed algorithm.

Algorithm I: 3-D DKF Algorithm Implemented by Agent i at Timestep k.

Propagation:

$$\boldsymbol{\Phi}_i^{k-1} = \tfrac{\partial \mathbf{f}}{\partial \mathbf{x}_i}(\hat{\mathbf{x}}_i^{k-1}), \quad \mathbf{G}_i^{k-1} = \tfrac{\partial \mathbf{f}}{\partial \mathbf{n}}(\hat{\mathbf{x}}_i^{k-1}),$$
$$\mathbf{Q}_i^{k-1} = \mathbf{G}_i^{k-1}\mathbf{O}^{k-1}(\mathbf{G}_i^{k-1})^\mathsf{T},$$
$$\bar{\mathbf{p}}_i^k = \boldsymbol{\Phi}_i^{k-1}\mathbf{p}_i^{k-1}(\boldsymbol{\Phi}_i^{k-1})^\mathsf{T} + \mathbf{Q}_i^{k-1},$$
$$\bar{\mathbf{x}}_i^k = \mathbf{f}(\hat{\mathbf{x}}_i^{k-1}).$$

Update:

(1) compute the update terms $\mathbf{s}_i^k$, $\mathbf{y}_i^k$;
(2) receive $\mathbf{s}_j^k$, $\mathbf{y}_j^k$, $\bar{\mathbf{x}}_j^k$, $\bar{\mathbf{p}}_j^k$ from agent j, $\forall j \in \mathcal{N}_i^k$;
(3) update $\bar{\mathbf{x}}_i^k$, $\mathbf{p}_i^k$ according to

$$\mathbf{p}_i^k = \left[\left(\sum_{j\in\mathcal{N}_i^k} \pi_j^k \bar{\mathbf{p}}_j^k\right)^{-1} + \sum_{j\in\mathcal{N}_i^k} \mathbf{s}_j^k\right]^{-1}$$
$$\hat{\mathbf{x}}_i^k = \left(\sum_{j\in\mathcal{N}_i^k} \pi_j^k \boxtimes \bar{\mathbf{x}}_j^k\right) \boxplus \left(\mathbf{p}_i^k \sum_{j\in\mathcal{N}_i^k} \mathbf{y}_j^k\right)$$

The time-varying weight π_j^k subject to $\pi_j^k \in [0,1]$ is selected to minimize $\mathrm{Tr}\{\mathbf{p}_i^k\}$.

A. State Vector and Models

As vision algorithms can yield many features on the target, like [29] we represent the 3-D rigid body target as the point cloud constructed by the tracked corner features. One of these features is chosen as the representative feature where the target's state is defined while all the other features are the non-representative features that can provide additional observations. These non-representative features' positions are also unknown. We include the non-representative features' relative position in the target's body frame in our estimation state to provide reobservation constraints. Therefore, the target state (2) is extended to

$$\mathbf{x} = \begin{bmatrix} {}^T_G\bar{q} \\ \mathbf{x}_v \end{bmatrix} = \begin{bmatrix} {}^T_G\bar{q}^\mathsf{T} & {}^G\mathbf{p}_T{}^\mathsf{T} & {}^G\mathbf{v}_T{}^\mathsf{T} & {}^T\mathbf{p}_f{}^\mathsf{T} \end{bmatrix}^\mathsf{T},$$
$$ {}^T\mathbf{p}_f = \begin{bmatrix} {}^T\mathbf{p}_{f_1} & \cdots & {}^T\mathbf{p}_{f_n} \end{bmatrix}^\mathsf{T}, \tag{11}$$

where ${}^T\mathbf{p}_f$ contains n non-representative features' relative positions in T.

At timestep k, suppose that the target moves according to the following dynamics [17]

$$ {}^{T_k}_G\dot{\bar{q}} = \frac{1}{2}\boldsymbol{\omega}^k \otimes {}^{T_k}_G\bar{q}, \quad {}^G\dot{\mathbf{p}}_{T_k} = {}^G\mathbf{v}_{T_k},$$
$$ {}^G\dot{\mathbf{v}}_{T_k} = {}^{T_k}_G\mathbf{R}^\mathsf{T}\mathbf{a}^k, \tag{12}$$

where $\boldsymbol{\omega}$ and $\mathbf{a}$ are the actual local angular velocity and linear acceleration. The corresponding noisy angular velocity and linear acceleration are given as $\boldsymbol{\omega}_m = \boldsymbol{\omega} + \mathbf{n}_\omega$ and $\mathbf{a}_m = \mathbf{a} + \mathbf{n}_a$, where $\mathbf{n}_\omega$ and $\mathbf{n}_a$ are zero-mean white Gaussian noise. $\boldsymbol{\omega}_m$ and $\mathbf{a}_m$ are known to each agent. After linearizing (12), the corresponding error state obtained by camera i evolves according to

$$\begin{bmatrix} {}^G\dot{\delta\boldsymbol{\theta}}_{T_{i,k}} \\ {}^G\dot{\delta\mathbf{p}}_{T_{i,k}} \\ {}^G\dot{\delta\mathbf{v}}_{T_{i,k}} \end{bmatrix} = \mathbf{F}_i^k \begin{bmatrix} {}^G\delta\boldsymbol{\theta}_{T_{i,k}} \\ {}^G\delta\mathbf{p}_{T_{i,k}} \\ {}^G\delta\mathbf{v}_{T_{i,k}} \end{bmatrix} + \mathbf{L}_i^k\mathbf{n}^k \tag{13}$$

where $\mathbf{n} = \begin{bmatrix} \mathbf{n}_\omega^\mathsf{T} & \mathbf{n}_a^\mathsf{T} \end{bmatrix}^\mathsf{T}$ with the covariance $\mathbf{O}$,

$$\mathbf{F}_i = \begin{bmatrix} -\lfloor\boldsymbol{\omega}_{m\times}\rfloor & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ -\lfloor {}^{T_i}_G\bar{\mathbf{R}}^\mathsf{T}\mathbf{a}_{m\times}\rfloor & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix}, \quad \mathbf{L}_i = \begin{bmatrix} -\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & -{}^{T_i}_G\bar{\mathbf{R}}^\mathsf{T} \end{bmatrix}.$$

Then, we discretize (13) and obtain the first-order approximation. By noting that ${}^T\mathbf{p}_f$ does not evolve over time as we assume a rigid-body target, we obtain the discrete-time transition matrix and the noise covariance

$$\boldsymbol{\Phi}_i = \begin{bmatrix} \mathbf{F}_i\delta t + \mathbf{I}_9 & \mathbf{0}_{9\times 3n} \\ \mathbf{0}_{3n\times 9} & \mathbf{I}_{3n} \end{bmatrix}, \quad \mathbf{Q}_i = \begin{bmatrix} \mathbf{L}_i\mathbf{O}\mathbf{L}_i^\mathsf{T}\delta t & \mathbf{0}_{9\times 3n} \\ \mathbf{0}_{3n\times 9} & \mathbf{0}_{3n} \end{bmatrix},$$

where δt is the sampling time. With $\boldsymbol{\Phi}_i$ and $\mathbf{Q}_i$, we can perform the propagation step in Algorithm I.

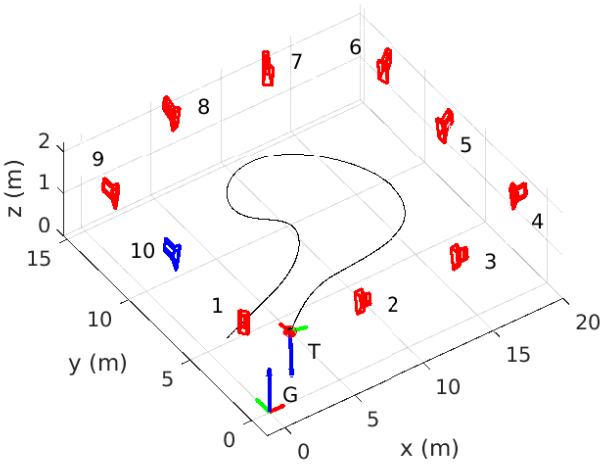

Fig. 1: 3-D moving object tracking over camera networks. G and T are respectively, the global frame and the target's body frame. The Blue camera denotes the camera currently sensing the target directly while the red ones are the blind cameras. The 3-D trajectory followed by the target is the black line.

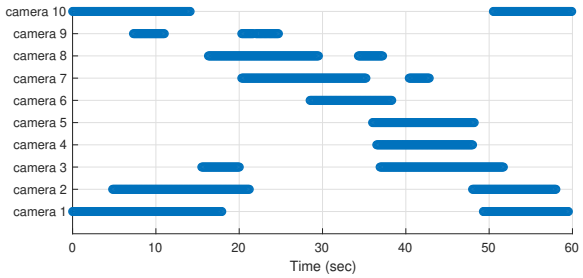

Fig. 2: Status of cameras directly sensing the target. The bold blue lines indicate the time intervals when the cameras can directly sense the target.

As the target explores the environment, the target features are captured by the cameras. Each camera i is assumed to be static with the global pose $({}^{C_i}_{G}\bar{q}, {}^{G}\mathbf{p}_{C_i})$. At timestep k, the measurements of the representative features take the form

$$\mathbf{z}^k_{T_i} = \Pi({}^{C_i}\mathbf{p}_{T_k}) + \mathbf{w}^k_i, \tag{14}$$

$${}^{C_i}\mathbf{p}_{T_k} = {}^{C_i}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T_k} - {}^{G}\mathbf{p}_{C_i}\right), \tag{15}$$

where ${}^{C_i}\mathbf{p}_T$ denotes the target's position in the ith camera frame; $\mathbf{w}_i$ is the zero-mean white Gaussian noise with covariance $\mathbf{R}_i$. By linearization of (14) and (15), we obtain the state Jacobian

$$\mathbf{H}_i = \mathbf{H}_p({}^{C_i}\bar{\mathbf{p}}_T){}^{C_i}_{G}\mathbf{R}\begin{bmatrix}\mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_{3\times 3(n+1)}\end{bmatrix}.$$

For a non-representative feature ${}^{T}\mathbf{p}_{f_1}$ (for notation simplicity, consider the first feature in ${}^{T}\mathbf{p}_f$), then (15) is replaced with

$${}^{C_i}\mathbf{p}_{T_k} = {}^{C_i}_{G}\mathbf{R}\left({}^{T_k}_{G}\mathbf{R}^{\mathsf{T}}{}^{T}\mathbf{p}_{f_1} + {}^{G}\mathbf{p}_{T_k} - {}^{G}\mathbf{p}_{C_i}\right). \tag{16}$$

Note that (16) puts constraints not only on the target's position ${}^{G}\mathbf{p}_{T_k}$ as (15) does, but also on the relative position ${}^{T}\mathbf{p}_{f_1}$ and the rotation matrix ${}^{T_k}_{G}\mathbf{R}$ associated with the target's orientation. By linearization of (14) and (16), we obtain the state Jacobian

$$\mathbf{H}_i = \mathbf{H}_p({}^{C_i}\bar{\mathbf{p}}_T){}^{C_i}_{G}\mathbf{R}[-\lfloor {}^{T_i}_{G}\bar{\mathbf{R}}^{\mathsf{T}}{}^{T}\bar{\mathbf{p}}_{f_1\times}\rfloor \quad \mathbf{I}_3 \quad \mathbf{0}_3 \quad {}^{T_k}_{G}\bar{\mathbf{R}}^{\mathsf{T}} \quad \mathbf{0}_{3\times 3(n-1)}].$$

With $\mathbf{H}_i$, we can perform the update step in Algorithm I.

B. Simulations Results

The target is moving following a pre-designed 3-D trajectory. Further, the non-representative features are generated around the target's body frame. Each camera has the resolution of $[752, 480]$ and its maximum sensing distance is purposely set to 5 m. The linear acceleration and angular velocity noise are 0.4 m/s^2 and 0.03 rad/s, while the camera measurements are corrupted by 1 pixel noise. Then we perform 50 Monte-Carlo simulations and the results are quantified by the root mean squared error (RMSE).

To show the benefits of cooperative tracking, we assume that each camera can communicate with the other cameras with certain percentages. For example, 40% means that each camera can communicate with another camera with the probability of 40%. Hence, each camera's communicating neighbors are randomly chosen at every timestep and the communication graph is time varying. We compare the results of the proposed distributed algorithm (3-D DKF) against the one obtained by the benchmark (CKF) where all the cameras can communicate with the fusion center perfectly. Fig. 3 shows the averaged position RMSE (PRMSE) and the orientation RMSE (ORMSE) results for the CKF and the 3-D DKF over all trials and all cameras. It becomes clear that as the communication percentage increases, the estimation errors of the 3-D DKF reduce in both the positions and orientations. In particular, the performance of the 3-D DKF with 60% communication percentage is comparable to the CKF's performance.

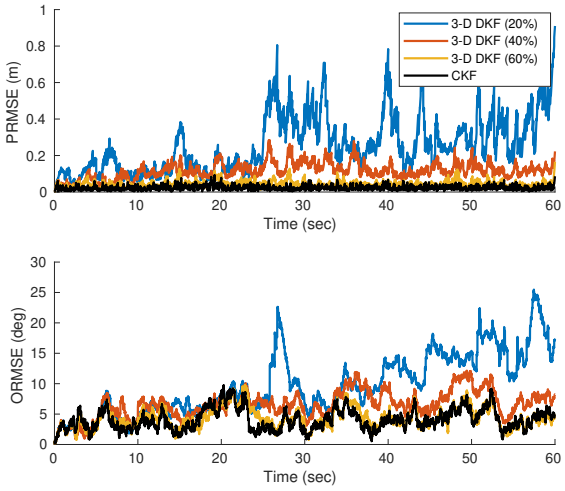

Fig. 3: Averaged RMSE for the estimated target pose over 50 Monte-Carlo runs and ten cameras.

Further, to show the performance of individual cameras, Table I provides the averaged RMSE results with different communication percentages for the first four cameras (cams 1, 2, 3 and 4) over all trials and all timesteps. Obviously, none of the cameras can successfully track the target with 0% communication (no collaboration between cameras). While as the communication percentage increases, all the estimators maintained by each camera become more accurate. When the communication percentage is 40%, the estimated trajectories obtained by the first four cameras are plotted against the groundtruth in Fig. 4, which shows that our approach can well track the 3-D trajectory of the target.

TABLE I: Averaged RMSE for the estimated target pose over 50 Monte-Carlo runs and all timesteps.

communication (3-D DKF)		0 %	20 %	40%	60%
Cam 1	PRMSE (m)	22.654	0.239	0.119	0.041
	ORMSE (deg)	22.246	9.902	6.992	4.320
Cam 2	PRMSE (m)	65.005	0.265	0.123	0.040
	ORMSE (deg)	36.935	10.306	6.917	4.285
Cam 3	PRMSE (m)	72.067	0.217	0.117	0.042
	ORMSE (deg)	26.246	9.970	6.900	4.337
Cam 4	PRMSE (m)	56.871	0.281	0.124	0.043
	ORMSE (deg)	38.271	10.245	6.925	4.314
CKF	PRMSE (m)	0.022			
	ORMSE (deg)	4.128			

V. Conclusion

In this paper, we have introduced a new DKF that is applicable for tacking the 6-DoF motion of a target moving in 3-D environments over sensor networks. The proposed algorithm enjoys the property of being fully distributed as it only uses its own and one-hop neighbors' information. Moreover, it only requires a single communication iteration in the update step and is robust to the time-varying changes in the network such as the communication topology, the blind

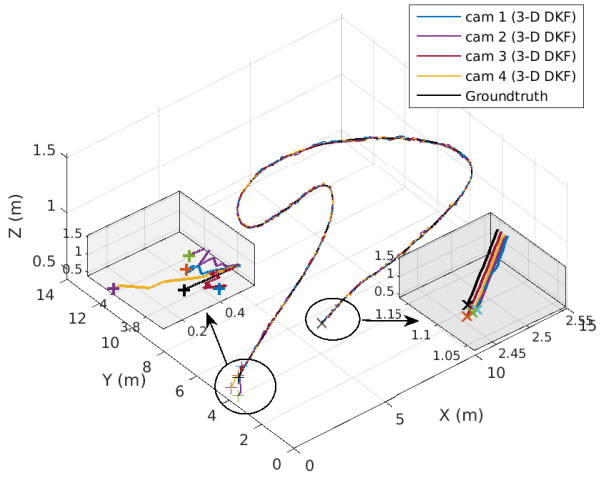

Fig. 4: Estimated 3-D trajectories of the first four cameras. '+' denotes the start position while 'x' denotes the end point. The start and end areas are enlarged in the built-in figures.

agents, and the network size. It also deals with the generic target and measurement models. These properties ensure that our approach is applicable in a wide range of cooperative target tacking scenarios. The performance is tested with the application to camera networks via Monte-Carlo simulations.

REFERENCES

[1] E. Kaplan and C. Hegarty, *Understanding GPS: principles and applications*. Artech House, 2005.

[2] G. Battistelli, L. Chisci, G. Mugnai, A. Farina, and A. Graziano, "Consensus-based linear and nonlinear filtering," *IEEE Transactions on Automatic Control*, vol. 60, no. 5, pp. 1410–1415, 2014.

[3] G. Battistelli and L. Chisci, "Kullback–leibler average, consensus on probability densities, and distributed state estimation with guaranteed stability," *Automatica*, vol. 50, no. 3, pp. 707–718, 2014.

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

[5] G. Wei, W. Li, D. Ding, and Y. Liu, "Stability analysis of covariance intersection-based Kalman consensus filtering for time-varying systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018.

[6] G. Battistelli and L. Chisci, "Stability of consensus extended Kalman filter for distributed state estimation," *Automatica*, vol. 68, pp. 169–178, 2016.

[7] X. He, C. Hu, Y. Hong, L. Shi, and H. Fang, "Distributed Kalman filters with state equality constraints: Time-based and event-triggered communications," *IEEE Transactions on Automatic Control*, 2019.

[8] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the American Control Conference*, 1997, pp. 2369–2373.

[9] S. Julier and J. K. Uhlmann, "General decentralized data fusion with covariance intersection," in *Handbook of multisensor data fusion*. CRC Press, 2017, pp. 339–364.

[10] P. O. Arambel, C. Rago, and R. K. Mehra, "Covariance intersection algorithm for distributed spacecraft state estimation," in *Proceedings of the American Control Conference*, 2001, pp. 4398–4403.

[11] X. He, W. Xue, and H. Fang, "Consistent distributed state estimation with global observability over sensor network," *Automatica*, vol. 92, pp. 162–172, 2018.

[12] J. Hu, L. Xie, and C. Zhang, "Diffusion Kalman filtering based on covariance intersection," *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 891–902, 2011.

[13] S. Wang and W. Ren, "On the convergence conditions of distributed dynamic state estimation using sensor networks: A unified framework," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 4, pp. 1300–1316, 2017.

[14] S. Wang, Y. Lyu, and W. Ren, "Unscented-transformation-based distributed nonlinear state estimation: Algorithm, analysis, and experiments," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 5, pp. 2016–2029, 2018.

[15] G. Battistelli, L. Chisci, and D. Selvi, "A distributed Kalman filter with event-triggered communication and guaranteed stability," *Automatica*, vol. 93, pp. 75–82, 2018.

[16] S. Wang, W. Ren, and J. Chen, "Fully distributed dynamic state estimation with uncertain process models," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 4, pp. 1841–1851, 2017.

[17] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3d attitude estimation," *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep*, vol. 2, p. 2005, 2005.

[18] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proceedings of the IEEE Conference on Decision and Control*, 2007, pp. 5492–5498.

[19] 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 *Proceedings of the IEEE Conference on Decision and Control, and the European Control Conference*. IEEE, 2011, pp. 7863–7869.

[20] F. S. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 2069–2084, 2010.

[21] J. Sola, "Quaternion kinematics for the error-state kf," *Laboratoire dAnalyse et dArchitecture des Systemes-Centre national de la recherche scientifique (LAAS-CNRS), Toulouse, France, Tech. Rep*, 2012.

[22] W. Breckenridge, "Quaternions proposed standard conventions," *Jet Propulsion Laboratory, Pasadena, CA, Interoffice Memorandum IOM*, pp. 343–79, 1999.

[23] M. D. Shuster, "A survey of attitude representations," *Navigation*, vol. 8, no. 9, pp. 439–517, 1993.

[24] E. Leffens, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[25] D. P. Koch, D. O. Wheeler, R. Beard, T. McLain, and K. M. Brink, "Relative multiplicative extended Kalman filter for observable gps-denied navigation," *[Online]. Available: https://scholarsarchive.byu.edu/facpub/1963/*, 2017.

[26] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman, "Averaging quaternions," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 4, pp. 1193–1197, 2007.

[27] W. Niehsen, "Information fusion based on fast covariance intersection filtering," in *Proceedings of the IEEE International Conference on Information Fusion*, vol. 2, 2002, pp. 901–904.

[28] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1280–1286.

[29] K. Eckenhoff, Y. Yang, P. Geneva, and G. Huang, "Tightly-coupled visual-inertial localization and 3-d rigid-body target tracking," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1541–1548, 2019.