June 2019 Volume 14 Number 1 ISSN 1557-6418

JAIF 9

Journal of Advances in Information Fusion

A semi-annual archival publication of the International Society of Information Fusion

Regular Papers Page

ISIF

INTERNATIONAL SOCIETY OF INFORMATION FUSION

The International Society of Information Fusion (ISIF) is the premier professional society and global information resource for multidisciplinary approaches for theoretical and applied INFORMATION FUSION technologies. Technical areas of interest include target tracking, detection theory, applications for information fusion methods, image fusion, fusion systems architectures and management issues, classification, learning, data mining, Bayesian and reasoning methods.

Journal of Advances in Information Fusion (ISSN 1557-6418) is published semi-annually by the International Society of Information Fusion. The responsibility for the contents rests upon the authors and not upon ISIF, the Society, or its members. ISIF is a California Nonprofit Public Benefit Corporation at P.O. Box 4631, Mountain View, California 94040.

Game-Theoretic Defense of Adversarial Distributed Support Vector Machines

RUI ZHANG
QUANYAN ZHU

With a large number of sensors and control units in networked systems, distributed support vector machines (DSVMs) play a fundamental role in scalable and efficient multi-sensor classification and prediction tasks. However, DSVMs are vulnerable to adversaries who can modify and generate data to deceive the system to misclassification and misprediction. This work aims to design defense strategies for DSVM learner against a potential adversary. We establish a game-theoretic framework to capture the conflicting interests between the DSVM learner and the attacker. The Nash equilibrium of the game allows predicting the outcome of learning algorithms in adversarial environments, and enhancing the resilience of the machine learning through dynamic distributed learning algorithms. We show that the DSVM learner is less vulnerable when he uses a balanced network with fewer nodes and higher degree. We also show that adding more training samples is an efficient defense strategy against an attacker. We present secure and resilient DSVM algorithms with verification method and rejection method, and show their resiliency against adversary with numerical experiments.

Manuscript received May 15, 2017; released for publication July 1, 2017.

Refereeing of this contribution was handled by Huimin Chen.

Authors' address: Department of Electrical and Computer Engineering, New York University, Brooklyn, NY, 11201 USA (E-mail: {rz885,qz494}@nyu.edu).

This research is partially supported by a DHS grant through Critical Infrastructure Resilience Institute (CIRI) and grants SES-1541164, and ECCS-1550000 from National Science Foundation (NSF).

A preliminary version of this work has been presented at the 18th International Conference on Information fusion, Washington, D.C., 2015 [34].

1. INTRODUCTION

Support Vector Machines (SVMs) have been widely used in multi-sensor data fusion problems, such as motor fault detection [1], land cover classification [30], and gas prediction [39]. In these applications, a fusion center is required to collect data from each sensor and train the SVM classifier. However, the computations in the fusion center and its communications with sensors become costly when the size of data and network becomes large [11].

To solve the large-scale data fusion problems, several methods have been developed to speed up SVMs. For example, in [28], Tsang et al. have introduced an approximation method to scale up SVMs. In [10], Dong et al. have presented an efficient SVM algorithm using parallel optimization. These methods only speed up the computations in the fusion center, but the data transmissions between fusion center and sensors still require a significant amount of time and channel usages.

Efficiency is not the only drawback of the centralized SVM using fusion center. Sensors that collect sensitive or private information to design the classifier may not be willing to share their training data [13]. Moreover, a compromised fusion center attacked by an adversary may give erroneous information to all the sensors in the network. Furthermore, compromised sensors may also provide misleading information to the fusion center, and consequently affect uncompromised sensors [6].

Distributed support vector machines (DSVMs) draw attentions recently as it does not require a fusion center to process data collections and computations [13, 23, 29]. Each node in the network solves decentralized sub-problems themselves using their own data, and only a small amount of data is transferred between nodes, which makes DSVMs more efficient and private than the centralized counterpart.

However, DSVMs are also vulnerable. For example, misleading information can be spread to the whole network, and the large number of nodes and complex connections in a network makes it harder to detect and track the source of the incorrect information [5]. Moreover, even though we can find the compromised nodes, an adversary can attack other nodes and spread misleading information.

Thus, it is important to design secure and resilient distributed support vector machines algorithms against potential attacks from an adversary. In this paper, we focus on a consensus based DSVM algorithm where SVM problem is captured by a set of decentralized convex optimization sub-problems with consensus constraints on their decision variables [13, 34]. We aim to design defense strategies against potential attacks by analyzing the equilibrium of the game-theoretic model between a DSVM learner and an attacker.

In our previous work [34], we have built a game-theoretic framework to capture the conflict of interests between the DSVM learner and the attacker who can

modify the training data. In the two-person nonzero-sum game, the learner aims to decentralize the computations over a network of nodes and minimize the error with an effort to minimize misclassification, while the attacker seeks to modify strategically the training data and maximize the error constrained by its computational capabilities.

The game formulation of the security problem enables a formal analysis of the impact of the DSVM algorithm in adversarial environments. The Nash equilibrium of the game enables the prediction of the outcome, and yields an optimal response strategy to the adversary behaviors. The game framework also provides a theoretic basis for developing dynamic learning algorithms that will enhance the security and the resilience of DSVMs.

In this paper, we propose several defense strategies for a DSVM learner against a potential attacker, and we show the effectiveness of our defense strategies using numerical experiments. The major contributions of this work are multi-fold.

Firstly, we capture the attacker's objective and constrained capabilities in a game-theoretic framework, and develop a nonzero-sum game to model the strategic interactions between an attacker and a learner with a set of nodes. We then fully characterize the Nash equilibrium by showing the strategic equivalence between the original nonzerosum game and a zero-sum game.

Secondly, we develop secure and resilient distributed algorithms based on alternating direction method of multipliers (ADMoM) [4]. Each node communicates with its neighboring nodes, and updates its decision strategically in response to adversarial environments. We present a summary of numerical results in [34].

Lastly, we present four defense strategies against potential attackers. The first defense strategy is to use balanced networks with fewer nodes and higher degrees. In the second defense strategy, we show that adding training samples to compromised nodes can reduce the vulnerability of the learning system. Adding samples to uncompromised nodes at the beginning of the training process also makes the learner less vulnerable. The third defense strategy is to use verification method where each node verifies its received data, and only accepts reasonable information from neighboring nodes to prevent misleading or illegitimate information sent to uncompromised nodes. The fourth defense strategy is to use rejection method where each node rejects unacceptable updates. Thus, not only misleading information is kept from affecting uncompromised nodes, but also wrong updates could be prevented in compromised nodes.

1.1. Related Works

Our work intersects the research areas on data fusion, machine learning, cyber security and machine learning. Machine learning tools have been used to tackle data fusion problems, e.g., [9, 16, 31]. However, machine learning systems can be insecure [2]. For example, in [17], Huang et al. have shown that Spam-Bayes and PCA-based network anomaly detection are vulnerable to causative attacks. In [3], Biggio et al. have shown that popular classification algorithms can be evaded even if the attacker has limited knowledge of learner's system.

With distributed machine learning tools developed for solving large-scale multi-sensor data fusion problems, each sensor solves sub-problems themselves and transmits information with neighboring sensors [24]. However, cyber security becomes another problem as an attacker may launch malicious cyber attacks to the data fusion networks [7]. Thus, it is important for the machine learning learner to analyze the equilibrium of the game with an adversary and design defense strategies against potential attacks.

Game theory is a natural tool to address this problem. It has been used in the study of the security of machine learning. For example, in [21], Liu et al. have modeled the interaction between a learner and an attacker as a two-person sequential noncooperative Stackelberg game. In [19], Kantarcioglu et al. have used game theory to analyze the equilibrium behavior of adversarial learning.

Game theory has also been used widely in cyber security as it provides mathematical tools for modeling situations of conflicts and predicting the behaviors of the attacker and defender in network security [22, 38, 40–43]. For example, in [26], Shen et al. have built an adaptive Markov game model to infer possible cyber attack patterns. In [18], Jiang et al. have presented an attack prediction and optimal active defense method using a stochastic game.

With game theory, we are able to analyze the game between a distributed machine learning learner with an adversary in a network, and further design defense strategies for the learner against the attacker. In our work, we focus on a class of consensus-based distributed support vector machines algorithms [13]. We assume that the attacker has the ability to modify training data to achieve his objectives.

In our previous works [33–37], we have built a game-theoretic model to capture the conflicts between a DSVM learner and an adversary who can modify training data or labels, and we have solved the game-theoretic problem with ADMoM [4]. In this work, we further analyze the equilibrium behaviors, and design defense strategies for DSVMs against potential attacks. We use numerical experiments to verify the effectiveness of our strategies.

1.2. Organization of the Paper

The rest of this paper is organized as follows. Section 2 outlines the consensus-based distributed support vector machines. In Section 3, we establish game-theoretic models for the learner and the attacker. Section 4 deals with the distributed and dynamic algorithms

for the learner and the attacker. Section 5 summarizes our previous numerical experiments. Section 6 presents four different defense strategies and their corresponding numerical experiments. Section 7 provides conclusion remarks.

1.3. Summary of Notations

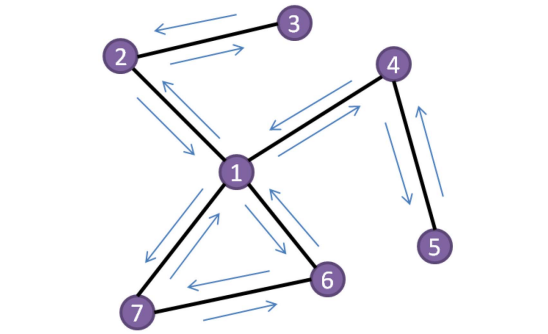
Notations in this paper are summarized as follows. Boldface letters are used for matrices (column vectors); $(\cdot)^T$ denotes matrix and vector transposition; $(\cdot)^{(t)}$ denotes values at step t; $[\cdot]_{vu}$ denotes the vuth entry of a matrix; diag($\mathbf{X}$) is the diagonal matrix with $\mathbf{X}$ on its main diagonal; $\|\cdot\|$ is the norm of the matrix or vector; $\mathcal{V}$ denotes the set of nodes in a network; $\mathcal{B}_v$ denotes the set of neighboring nodes of node v; $\mathcal{U}$ denotes the action set used by the attacker.

2. DISTRIBUTED SUPPORT VECTOR MACHINES

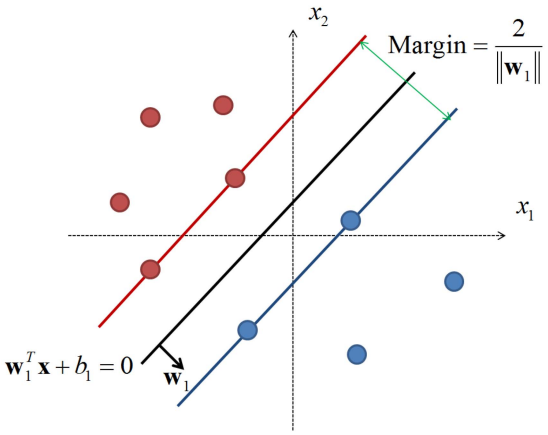
In this section, we present a distributed support vector machines learner in the network modeled by an undirected graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ with $\mathcal{V} := \{1,\dots,V\}$ representing the set of nodes, and $\mathcal{E}$ representing the set of links between nodes. Node $v \in \mathcal{V}$ communicates only with his neighboring nodes $\mathcal{B}_v \subseteq \mathcal{V}$. Note that without loss of generality, graph $\mathcal{G}$ is assumed to be connected; in other words, any two nodes in graph $\mathcal{G}$ are connected by a path. However, nodes in $\mathcal{G}$ do not have to be fully connected, which means that nodes are not required to directly connect to all the other nodes in the network. The network can contain cycles. At every node $v \in \mathcal{V}$, a labelled training set $\mathcal{D}_v := \{(\mathbf{x}_{vn}, y_{vn}) : n = 1,\dots,N_v\}$ of size N_v is available, where $\mathbf{x}_{vn} \in \mathbb{R}^p$ represents a p-dimensional pattern, and they are divided into two groups with labels $y_{vn} \in \{+1,-1\}$. Examples of a network of distributed nodes are illustrated in Fig. 1(a).

The goal of the learner is to design DSVM algorithms for each node in the network based on its local training data $\mathcal{D}_v$, so that each node has the ability to give new input $\mathbf{x}$ a label of $+1$ or -1 without communicating $\mathcal{D}_v$ to other nodes $v' \neq v$. To achieve this, the learner aims to find local maximum-margin linear discriminant functions $g_v(\mathbf{x}) = \mathbf{x}^T\mathbf{w}_v^* + b_v^*$ at every node $v \in \mathcal{V}$ with the consensus constraints $\mathbf{w}_1^* = \mathbf{w}_2^* = \cdots = \mathbf{w}_V^*$, $b_1^* = b_2^* = \cdots = b_V^*$, forcing all the local variables $\{\mathbf{w}_v^*, b_v^*\}$ to agree across neighboring nodes. Variables $\mathbf{w}_v^*$ and b_v^* of the local discriminant functions $g_v(\mathbf{x})$ can be obtained by solving the following convex optimization problem [13]:

$$
\begin{aligned}
\min_{\{\mathbf{w}_v, b_v, \{\xi_{vn}\}\}} \ & \frac{1}{2}\sum_{v\in\mathcal{V}} \|\mathbf{w}_v\|_2^2 + VC_l \sum_{v\in\mathcal{V}}\sum_{n=1}^{N_v} \xi_{vn} \\
& y_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn} + b_v) \geq 1 - \xi_{vn}, \quad \forall v \in \mathcal{V},\ n = 1,\dots,N_v; \\
\text{s.t.} \ & \xi_{vn} \geq 0, \qquad\qquad\qquad\quad \forall v \in \mathcal{V},\ n = 1,\dots,N_v; \\
& \mathbf{w}_v = \mathbf{w}_u,\ b_v = b_u, \qquad\quad \forall v \in \mathcal{V},\ u \in \mathcal{B}_v.
\end{aligned}
\tag{1}
$$



(a) Network example.



(b) SVMs at node 1.

Fig. 1. Network example. (a) There are 7 nodes in this network. (b) Each node contains a labelled training set $\mathcal{D}_v := \{(\mathbf{x}_{vn}, y_{vn}) : n = 1,\dots,N_v\}$. Each node can communicate with its neighbors. In each node, the learner aims to find the best linear discriminant line (Black solid line).

In the above problem, slack variables ξ_{vn} account for non-linearly separable training sets. C_l is a tunable positive scalar for the learner.

To solve Problem (1), we first define $\mathbf{r}_v := [\mathbf{w}_v^T, b_v]^T$, the augmented matrix $\mathbf{X}_v := [(\mathbf{x}_{v1},\dots,\mathbf{x}_{vN_v})^T, \mathbf{1}_v]$, the diagonal label matrix $\mathbf{Y}_v := \text{diag}([y_{v1},\dots,y_{vN_v}])$, and the vector of slack variables $\xi_v := [\xi_{v1},\dots,\xi_{vN_v}]^T$. With these definitions, it follows readily that $\mathbf{w}_v = (\mathbf{I}_{p+1} - \Pi_{p+1})\mathbf{r}_v$, where Π_{p+1} is a $(p+1)\times(p+1)$ matrix with zeros everywhere except for the $(p+1,p+1)$st entry, given by $[\Pi_{p+1}]_{(p+1)(p+1)} = 1$. Thus, Problem (1) can be rewritten as

$$
\begin{aligned}
\min_{\{\mathbf{r}_v, \xi_v, \omega_{vu}\}} \ & \frac{1}{2}\sum_{v\in\mathcal{V}} \mathbf{r}_v^T(\mathbf{I}_{p+1} - \Pi_{p+1})\mathbf{r}_v + VC_l \sum_{v\in\mathcal{V}} \mathbf{1}_v^T\xi_v \\
& \mathbf{Y}_v\mathbf{X}_v\mathbf{r}_v \geq \mathbf{1}_v - \boldsymbol{\xi}_v, \quad \forall v \in \mathcal{V}; && \text{(2a)} \\
\text{s.t.} \ & \boldsymbol{\xi}_v \geq \mathbf{0}_v, \quad \forall v \in \mathcal{V}; && \text{(2b)} \\
& \mathbf{r}_v = \omega_{vu},\ \omega_{vu} = \mathbf{r}_u, \quad \forall v \in \mathcal{V},\ \forall u \in \mathcal{B}_v. && \text{(2c)}
\end{aligned}
\tag{2}
$$

Note that ω_{vu} is used to decompose the decision variable $\mathbf{r}_v$ to its neighbors $\mathbf{r}_u$, where $u \in \mathcal{B}_v$. Problem (2) is a min-problem with matrix form coming from Problem (1).

With alternating direction method of multipliers [4], Problem (2) can be solved distributedly in the following lemma [13],

LEMMA 1 *With arbitrary initialization* $\mathbf{r}_v^{(0)}$, $\lambda_v^{(0)}$, $\omega_{vu}^{(0)}$ *and* $\alpha_v^{(0)} = \mathbf{0}_{(p+1)\times 1}$, *the iterations per node are given by:*

$$\lambda_v^{(t+1)} \in \arg \max_{\mathbf{0}\le\boldsymbol{\lambda}_v\le VC_l\mathbf{1}_v} -\tfrac{1}{2}\lambda_v^T\mathbf{Y}_v\mathbf{X}_v\mathbf{U}_v^{-1}\mathbf{X}_v^T\mathbf{Y}_v\lambda_v + (\mathbf{1}_v + \mathbf{Y}_v\mathbf{X}_v\mathbf{U}_v^{-1}\mathbf{f}_v^{(t)})^T\lambda_v, \quad (3)$$

$$\mathbf{r}_v^{(t+1)} = \mathbf{U}_v^{-1}(\mathbf{X}_v^T\mathbf{Y}_v\lambda_v^{(t+1)} - \mathbf{f}_v^{(t)}), \quad (4)$$

$$\omega_{vu}^{(t+1)} = \tfrac{1}{2}(\mathbf{r}_v^{(t+1)} + \mathbf{r}_u^{(t+1)}), \quad (5)$$

$$\alpha_v^{(t+1)} = \alpha_v^{(t)} + \frac{\eta}{2}\sum_{u\in\mathcal{B}_v}[\mathbf{r}_v^{(t+1)} - \mathbf{r}_u^{(t+1)}], \quad (6)$$

where $\mathbf{U}_v = (\mathbf{I}_{p+1} - \Pi_{p+1}) + 2\eta|\mathcal{B}_v|\mathbf{I}_{p+1}$, $\mathbf{f}_v^{(t)} = 2\alpha_v^{(t)} - 2\eta \times \sum_{u\in\mathcal{B}_v}\omega_{vu}^{(t)}$.

The proof of Lemma 1 can be found in [13]. Iteration (3) is a quadratic programming problem. λ_v are the Lagrange multipliers per node corresponding to constraint (2a). Iteration (4) computes the decision variables $\mathbf{r}_v$, note that the inverse of $\mathbf{U}_v$ always exists and easy to solve. Iteration (5) yields the consensus variables ω_{vu}. Iteration (6) computes α_v, e.g., the Lagrange multipliers corresponding to the consensus constraint (2c). Iterations (3)–(6) are summarized into Algorithm 1. Note that at any given iteration t of the algorithm, each node $v \in \mathcal{V}$ computes its own local discriminant function $g_v^{(t)}(\mathbf{x})$ for any vector $\mathbf{x}$ as

$$g_v^{(t)}(\mathbf{x}) = [\mathbf{x}^T, 1]\mathbf{r}_v^{(t)}. \quad (7)$$

ALGORITHM 1: *ADMoM-DSVM*

Randomly initialize $\mathbf{r}_v^{(0)}$, $\lambda_v^{(0)}$, $\omega_{vu}^{(0)}$ and $\alpha_v^{(0)} = \mathbf{0}_{(p+1)\times 1}$.

for $t = 0,1,2,\ldots$ **do**
  for all $v \in \mathcal{V}$ **do**
    Compute $\lambda_v^{(t+1)}$ via (3).
    Compute $\mathbf{r}_v^{(t+1)}$ via (4).
  end for
  for all $v \in \mathcal{V}$ **do**
    Broadcast $\mathbf{r}_v^{(t+1)}$ to all neighbors $u \in \mathcal{B}_v$.
  end for
  for all $v \in \mathcal{V}$ **do**
    Compute $\omega_{vu}^{(t+1)}$ via (5).
    Compute $\alpha_v^{(t+1)}$ via (6).
  end for
end for

Algorithm 1 solves the DSVM problem using ADMoM technique. It is a fully decentralized network operation, and it does not require exchanging training data or the value of decision functions, which meets the reduced communication overhead and privacy preservation requirements at the same time. However, information transmitted in the network not only helps improve the performance of each node, but also increases the damages from the attacker, as the misleading information can be spread to every node. To design a secure and resilient DSVM algorithm, we first build the attack model to capture the attacker's intentions of breaking the training process of the learner.

3. DISTRIBUTED SUPPORT VECTOR MACHINES WITH ADVERSARY

In this section, we present the game-theoretic framework of a DSVM learner and an attacker who takes over a set of nodes with the aim of breaking the training process of the learner. We assume that the attacker has a complete knowledge of the learner's Problem (1) by Kerckhoffs's principle: the enemy knows the system [25], which enables us to anticipate the interactions of the learner and the attacker in a worst-case scenario. Moreover, the attacker can easily acquire the complete knowledge of the learning systems in reality, for example, by node capture attacks [27] and computer worms [8], an attacker can compromise the whole network through connections between neighboring nodes, and thus obtain the private and sensitive information of the learner.

To achieve the malicious goal, the attacker takes over a set of nodes $\mathcal{V}_a := \{1,\ldots,V_a\}$ and changes $\mathbf{x}_{vn}$ into

$$\hat{\mathbf{x}}_{vn} = \mathbf{x}_{vn} - \delta_{vn},$$

where $\delta_{vn} \in \mathcal{U}_v$, and $\mathcal{U}_v$ is the attacker's action set at node v. Note that we use $\mathcal{V}_l = \{1,\ldots,V_l\}$ to represent nodes without the attacker. $V = V_a + V_l$ and $\mathcal{V} = \mathcal{V}_l \cup \mathcal{V}_a$. A node in the network is either under attack or not under attack. An example of the impact of the attacker on the learner is shown in Fig. 2. This type of attacks represents a challenge for the learner. On the one hand, the learner will find the incorrect classifiers at the compromised nodes, and communications in the network may lead to unanticipated results as misleading information from compromised nodes can be spread to and then used by uncompromised nodes. On the other hand, it is difficult for the learner to identify modified data, and furthermore, in distributed settings, the learner may not even be able to detect which nodes are under attack. Potential real world examples of the attackers are discussed as follows.

EXAMPLE 1 (Air pollution detection) [20].

Consider an air pollution detection system which uses DSVM as the classifiers to determine whether certain areas have air pollution. An attacker can modify the training data of the certain areas to let the system fail to recognize the air pollution. Moreover, the attacker can even modify other areas' training data to achieve his goal, since misleading information can be spread among the whole system by the communications between neighboring nodes. However, with a large amount of training data and areas, the learner will fail to detect

the compromised data and areas, and the results of the air pollution detection system will be untrustworthy.

EXAMPLE 2 (Distributed medical databases) [13].

Suppose several medical centers aim to find classifiers together on a certain disease using DSVM. An attacker can modify the training data of one medical center, which affects not only the compromised medical center, but also the uncompromised medical centers, as the misleading information can be spread among the network. As a result, all the medical centers might give inaccurate diagnosis on the disease. To find out the compromised training data, the learner is required to examine all the training data from all the medical centers, which is costly and sometimes even unrealistic.

Now Problem (1) changes to,

$$\min_{\{\mathbf{w}_v,b_v,\{\xi_{vn}\}\}} \frac{1}{2}\sum_{v\in\mathcal{V}}\|\mathbf{w}_v\|_2^2 + VC_l\sum_{v\in\mathcal{V}}\sum_{n=1}^{N_v}\xi_{vn}$$

s.t.

$$\begin{aligned}
&y_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn}+b_v)\geq 1-\xi_{vn}, && \forall v\in\mathcal{V}_l,\ n=1,\ldots,N_v;\\
&y_{vn}(\mathbf{w}_v^T\hat{\mathbf{x}}_{vn}+b_v)\geq 1-\xi_{vn}, && \forall v\in\mathcal{V}_a,\ n=1,\ldots,N_v;\\
&\xi_{vn}\geq 0, && \forall v\in\mathcal{V},\ n=1,\ldots,N_v;\\
&\mathbf{w}_v=\mathbf{w}_u,\ b_v=b_u, && \forall v\in\mathcal{V},\ u\in\mathcal{B}_v.
\end{aligned} \tag{8}$$

By minimizing the objective function in Problem (8), the learner can obtain the optimal variables $\{\mathbf{w}_v, b_v\}$, which can be used to build up the discriminant function to classify the testing data. The attacker, on the other hand, aims to find an optimal way to modify the data using variables $\{\delta_{vn}\}$ to maximize the same objective function. The behavior of the attacker can thus be captured as follows:

$$\max_{\{\delta_{vn}\}} \frac{1}{2}\sum_{v\in\mathcal{V}}\|\mathbf{w}_v\|_2^2 + VC_l\sum_{v\in\mathcal{V}}\sum_{n=1}^{N_v}\xi_{vn} - C_a\sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v}\|\delta_{vn}\|_0$$

s.t.

$$\begin{aligned}
&y_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn}+b_v)\geq 1-\xi_{vn}, && \forall v\in\mathcal{V}_l,\ n=1,\ldots,N_v;\\
&y_{vn}(\mathbf{w}_v^T(\mathbf{x}_{vn}-\delta_{vn})+b_v)\geq 1-\xi_{vn}, && \forall v\in\mathcal{V}_a,\ n=1,\ldots,N_v;\\
&\xi_{vn}\geq 0, && \forall v\in\mathcal{V},\ n=1,\ldots,N_v;\\
&\mathbf{w}_v=\mathbf{w}_u,\ b_v=b_u, && \forall v\in\mathcal{V},\ u\in\mathcal{B}_v;\\
&\delta_{vn}\in\mathcal{U}_v, && \forall v\in\mathcal{V}_a.
\end{aligned} \tag{9}$$

In above problem, the term $C_a\sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v}\|\delta_{vn}\|_0$ represents the cost function for the attacker. l_0-norm is defined as $\|x\|_0 := |\{i : x_i \neq 0\}|$, i.e., a total number of nonzero elements in a vector. Here, we use the l_0-norm to denote the number of elements which are changed by the attacker. The objective function with l_0-norm captures the fact that the attacker aims to make the largest impact on the learner by changing the least number of elements. Constraint $\delta_{vn}\in\mathcal{U}_v$ indicates the action set of the attacker. In this paper, we use the following form of $\mathcal{U}_v$:

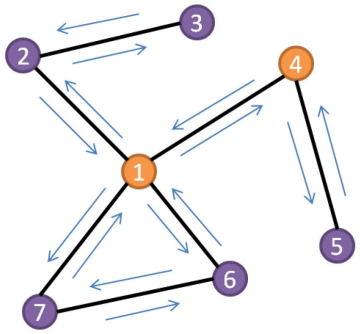
$$\mathcal{U}_v = \left\{(\delta_{v1},\ldots,\delta_{vN_v}) \,\middle|\, \sum_{n=1}^{N_v}\|\delta_{vn}\|_2^2 \leq C_{v,\delta}\right\},$$

which is related to the atomic action set $\mathcal{U}_{v0} = \{\delta_v \mid \|\delta_v\|_2^2 \leq C_{v,\delta}\}$. $C_{v,\delta}$ indicates the bound of the sum of the norm of all the changes at node v. A higher $C_{v,\delta}$ indicates that the attacker has a larger degree of freedom in changing the value $\mathbf{x}_{vn}$. Thus, training these data will lead to a higher risk for the learner. Notice that $C_{v,\delta}$ can vary at different nodes, and we use C_δ to represent the situation when $C_{v,\delta}$ are equal at every node. $\delta_v \in \mathbb{R}^p$ from the atomic action set has the same form with δ_{vn}, but δ_v and $(\delta_{v1},\ldots,\delta_{vN_v})$ are bounded by same $C_{v,\delta}$. Furthermore, the atomic action set $\mathcal{U}_{v0}$ has the following properties [32].

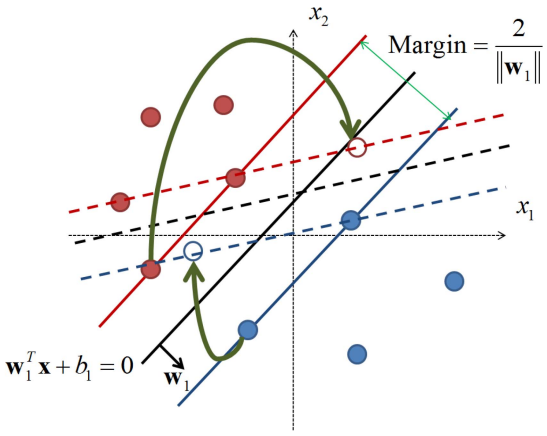
(P1) $\mathbf{0}\in\mathcal{U}_{v0}$;

(P2) For any $\mathbf{w}_0\in\mathbb{R}^p$:

$$\max_{\delta_v\in\mathcal{U}_{v0}}[\mathbf{w}_0^T\delta_v] = \max_{\delta_v'\in\mathcal{U}_{v0}}[-\mathbf{w}_0^T\delta_v'] < +\infty.$$



(a) Network under attack.



(b) SVMs at compromised node 1.

Fig. 2. Network with attacker. (a) Node 1 and 4 are under attack. (b) In compromised node, for example, node 1, an attacker modifies the training data which leads to a wrong linear discriminant line (Black dotted line).

The first property (P1) states that the attacker can choose not to change the value of $\mathbf{x}_{vn}$. Property (P2) states that the atomic action set is bounded and symmetric. Here, "bounded" means that the attacker has the limit on the capability of changing $\mathbf{x}_{vn}$. It is reasonable since changing the value significantly will result in the evident detection of the learner.

For the learner, the learning process is to find the discriminant function which separates the training data into two classes with less error, and then use the discriminant function to classify testing data. Since the attacker has the ability to change the value of original data $\mathbf{x}_{vn} \in \mathcal{X}$ into $\hat{\mathbf{x}}_{vn} \in \hat{\mathcal{X}}$, the learner will find the discriminant function that separates the data in $\hat{\mathcal{X}}$ more accurate, rather than the data in $\mathcal{X}$. As a result, when using the discriminant function to classify the testing data $\mathbf{x} \in \mathcal{X}$, it will be prone to be misclassified.

Since the learner aims at a high classification accuracy, while the attacker seeks to lower the accuracy, we can capture the conflicting goals of the players as a two-person nonzero-sum game by combining Problem (8) and Problem (9) together. The solution to the game problem is described by Nash equilibrium, which yields the equilibrium strategies for both players, and predicts the outcome of machine learning in the adversarial environment. By comparing Problem (8) with Problem (9), we notice that they contain the same terms in their objective functions and the constraints in the two problems are uncoupled. As a result, the nonzero-sum game can be reformulated into a zero-sum game, which takes the minimax or max-min form as follows:

$$\min_{\{\mathbf{w}_v,b_v,\{\xi_{vn}\}\}} \max_{\{\delta_{vn}\}} \frac{1}{2}\sum_{v\in\mathcal{V}} \|\mathbf{w}_v\|_2^2 + VC_l \sum_{v\in\mathcal{V}}\sum_{n=1}^{N_v} \xi_{vn} - C_a \sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v} \|\delta_{vn}\|_0$$

s.t.

$$\begin{aligned}
&\mathrm{y}_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn} + b_v) \geq 1 - \xi_{vn}, && \forall v \in \mathcal{V}_l,\ n = 1,\ldots,N_v;\\
&\mathrm{y}_{vn}(\mathbf{w}_v^T(\mathbf{x}_{vn} - \delta_{vn}) + b_v) \geq 1 - \xi_{vn}, && \forall v \in \mathcal{V}_a,\ n = 1,\ldots,N_v;\\
&\xi_{vn} \geq 0, && \forall v \in \mathcal{V},\ n = 1,\ldots,N_v;\\
&\mathbf{w}_v = \mathbf{w}_u, b_v = b_u, && \forall v \in \mathcal{V},\ u \in \mathcal{B}_v;\\
&\delta_{vn} \in \mathcal{U}_v, && \forall v \in \mathcal{V}_a.
\end{aligned} \tag{10}$$

Note that the first and fourth constraints only contribute to the minimization part of the problem, the fifth constraint only affects the maximization part. The second and third constraints contribute to both the minimization and the maximization part. The first term of the objective function is the inverse of the distance of margin. The second term is the sum of all the slack variables which captures the error penalties. On one hand, minimizing the objective function captures the trade-off between a larger margin and a small error penalty of the learner, while on the other hand, maximizing the objective function captures the trade-off between a large error penalty and a small cost of the attacker. As a result, solving Problem (10) can be understood as finding the saddle-point equilibrium of the zero-sum game between the attacker and the learner.

DEFINITION 1 Let $\mathcal{S}_L$ and $\mathcal{S}_A$ be the action sets for the DSVM learner and the attacker, respectively. Notice that here $\mathcal{S}_A = \{\mathcal{U}_v\}_{v\in\mathcal{V}_a}$. Then, the strategy pair $(\{\mathbf{w}_v^*, b_v^*, \{\xi_{vn}^*\}\}, \{\delta_{vn}^*\})$ is a saddle-point equilibrium solution of the zero-sum game defined by the triple $G_z := \langle\{L,A\},\{\mathcal{S}_L,\mathcal{S}_A\},K\rangle$, if $K(\{\mathbf{w}_v^*, b_v^*, \{\xi_{vn}^*\}\}, \{\delta_{vn}\}) \leq K(\{\mathbf{w}_v^*, b_v^*, \{\xi_{vn}^*\}\}, \{\delta_{vn}^*\}) \leq K(\{\mathbf{w}_v, b_v, \{\xi_{vn}\}\}, \{\delta_{vn}^*\}), \forall v \in \mathcal{V}$, where K is the objective function of Problem (10).

Based on the property of the action set and atomic action set, Problem (10) can be further simplified as stated in the following lemma [34].

LEMMA 2 *Assume that $\mathcal{U}_v$ is an action set with corresponding atomic action set $\mathcal{U}_{v0}$. Then, Problem (10) is equivalent to the following optimization problem:*

$$\min_{\{\mathbf{w}_v,b_v,\{\xi_{vn}\}\}} \max_{\{\delta_v\}} \frac{1}{2}\sum_{v\in\mathcal{V}} \|\mathbf{w}_v\|_2^2 + VC_l \sum_{v\in\mathcal{V}}\sum_{n=1}^{N_v} \xi_{vn} + \sum_{v\in\mathcal{V}_a} (V_a C_l \mathbf{w}_v^T\delta_v - C_a\|\delta_v\|_0)$$

s.t.

$$\begin{aligned}
&\mathrm{y}_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn} + b_v) \geq 1 - \xi_{vn}, && \forall v \in \mathcal{V},\ n = 1,\ldots,N_v;\\
&\xi_{vn} \geq 0, && \forall v \in \mathcal{V},\ n = 1,\ldots,N_v;\\
&\mathbf{w}_v = \mathbf{w}_u,\ b_v = b_u, && \forall v \in \mathcal{V},\ u \in \mathcal{B}_v;\\
&\delta_v \in \mathcal{U}_{v0}, && \forall v \in \mathcal{V}_a.
\end{aligned} \tag{11}$$

PROOF See Appendix A.

In Problem (10), the second and third constraints are the coupled terms with the second term of the objective function. But in Problem (11), the only coupled term is $V_a C_l \mathbf{w}_v^T\delta_v$, which is linear in the decision variables of the attacker and the learner, respectively.

4. ADMOM-DSVM AND DISTRIBUTED ALGORITHM

In the previous section, we have combined Problem (8) for the learner with Problem (9) for the attacker into one minimax Problem (10), and have showed its equivalence to Problem (11). In this section, we develop iterative algorithms to find equilibrium solutions to Problem (11). Using a similar method in Section II, Problem (11) can be rewritten into matrix

form as

$$\min_{\{\mathbf{r}_v,\xi_v,\omega_{vu}\}} \max_{\{\delta_v\}} \frac{1}{2}\sum_{v\in\mathcal{V}} \mathbf{r}_v^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{r}_v + VC_l\sum_{v\in\mathcal{V}}\mathbf{1}_v^T\xi_v + \sum_{v\in\mathcal{V}_a}(V_aC_l\mathbf{r}_v^T(\mathbf{I}_{p+1}-\Pi_{p+1})\delta_v - C_a\|\delta_v\|_0)$$

s.t.

$$\begin{aligned}
&\mathbf{Y}_v\mathbf{X}_v\mathbf{r}_v \geq \mathbf{1}_v - \boldsymbol{\xi}_v, && \forall v\in\mathcal{V}; && \text{(12a)}\\
&\boldsymbol{\xi}_v \geq \mathbf{0}_v, && \forall v\in\mathcal{V}; && \text{(12b)}\\
&\mathbf{r}_v = \omega_{vu}, \omega_{vu} = \mathbf{r}_u, && \forall v\in\mathcal{V},\ \forall u\in\mathcal{B}_v; && \text{(12c)}\\
&\delta_v \in \mathcal{U}_{v0}, && \forall v\in\mathcal{V}_a. && \text{(12d)}
\end{aligned} \tag{12}$$

To solve problem (12), we use best response dynamics to construct the best response for the min-problem and max-problem separately. The min-problem and max-problem are archived by fixing $\{\delta_v\}$ and $\{\mathbf{r}_v\}$, respectively. With ADMoM [12], we can develop a method of solving Problem (12) in a distributed way as follows: The first step is that each node randomly picks an initial $\mathbf{r}_v^{(0)}$, $\delta_v^{(0)}$ and $\alpha_v = \mathbf{0}_{(p+1)\times 1}$, then solve the max-problem with $\{\mathbf{r}_v^{(0)}\}$, and obtain$\{\delta_v^{(1)}\}$. The next step is to solve the min-problem with $\{\delta_v^{(1)}\}$ and obtain $\{\mathbf{r}_v^{(1)}\}$, then we repeat solving the max-problem with $\{\mathbf{r}_v\}$ from the previous step and the min-problem with $\{\delta_v\}$ from the previous step until the pair $\{\mathbf{r}_v,\delta_v\}$ achieves convergence. The iterations of solving Problem (12) can be summarized as follows [34].

LEMMA 3 *With arbitrary initialization $\boldsymbol{\delta}_v^{(0)}$, $\mathbf{r}_v^{(0)}$, $\lambda_v^{(0)}$, $\omega_{vu}^{(0)}$ and $\alpha_v^{(0)} = \mathbf{0}_{(p+1)\times 1}$, the iterations per node are given by:*

$$\begin{aligned}
\delta_v^{(t+1)} \in \arg\max_{\{\delta_v,s_v\}} & V_aC_l\mathbf{r}_v^{(t)T}(\mathbf{I}_{p+1}-\Pi_{p+1})\delta_v - \mathbf{1}^Ts_v\\
\text{s.t.}\quad & C_a\delta_v \leq s_v, \quad \forall v\in\mathcal{V}_a;\\
& C_a\delta_v \geq -s_v, \quad \forall v\in\mathcal{V}_a;\\
& \delta_v \in \mathcal{U}_{v0}, \quad \forall v\in\mathcal{V}_a.
\end{aligned} \tag{13}$$

$$\lambda_v^{(t+1)} \in \arg\max_{\mathbf{0}\leq\boldsymbol{\lambda}_v\leq VC_l\mathbf{1}_v} -\tfrac{1}{2}\lambda_v^T\mathbf{Y}_v\mathbf{X}_v\mathbf{U}_v^{-1}\mathbf{X}_v^T\mathbf{Y}_v\lambda_v + (\mathbf{1}_v + \mathbf{Y}_v\mathbf{X}_v\mathbf{U}_v^{-1}\mathbf{f}_v^{(t)})^T\lambda_v, \tag{14}$$

$$\mathbf{r}_v^{(t+1)} = \mathbf{U}_v^{-1}(\mathbf{X}_v^T\mathbf{Y}_v\lambda_v^{(t+1)} - \mathbf{f}_v^{(t)}), \tag{15}$$

$$\omega_{vu}^{(t+1)} = \tfrac{1}{2}(\mathbf{r}_v^{(t+1)} + \mathbf{r}_u^{(t+1)}), \tag{16}$$

$$\alpha_v^{(t+1)} = \alpha_v^{(t)} + \frac{\eta}{2}\sum_{u\in\mathcal{B}_v}[\mathbf{r}_v^{(t+1)} - \mathbf{r}_u^{(t+1)}], \tag{17}$$

where $\mathbf{U}_v = (\mathbf{I}_{p+1}-\Pi_{p+1}) + 2\eta|\mathcal{B}_v|\mathbf{I}_{p+1}$, $\mathbf{f}_v^{(t)} = V_aC_l\delta_v^{(t)} + 2\alpha_v^{(t)} - 2\eta\sum_{u\in\mathcal{B}_v}\omega_{vu}^{(t)}$.

PROOF See Appendix B.

Iteration (13) corresponds to the attacker's Max-Problem (9), while iterations (14)–(17) correspond to the learner's Min-Problem (8). The Minimax Problem (11) is solved by iterating them together. Note that, iterations (14)–(17) differ from iterations (3)–(6) only in $\mathbf{f}_v$. In (14)–(17), $\mathbf{f}_v$ adds another term $V_aC_l\delta_v$ which captures the attacker's impact on the learner. Iterations (13)–(17) are summarized into Algorithm 2.

ALGORITHM 2: *DSVM under attack*

Randomly initialize $\boldsymbol{\delta}_v^{(0)}, \mathbf{r}_v^{(0)}, \lambda_v^{(0)}, \omega_{vu}^{(0)}$ and $\alpha_v^{(0)} = \mathbf{0}_{(p+1)\times 1}$.
for $t = 0,1,2,\ldots$ **do**
  for all $v\in\mathcal{V}$ **do**
    Compute $\delta_v^{(t+1)}$ via (13).
  end for
  for all $v\in\mathcal{V}$ **do**
    Compute $\lambda_v^{(t+1)}$ via (14).
    Compute $\mathbf{r}_v^{(t+1)}$ via (15).
  end for
  for all $v\in\mathcal{V}$ **do**
    Broadcast $\mathbf{r}_v^{(t+1)}$ to all neighbors $u\in\mathcal{B}_v$.
  end for
  for all $v\in\mathcal{V}$ **do**
    Compute $\omega_{vu}^{(t+1)}$ via (16).
    Compute $\alpha_v^{(t+1)}$ via (17).
  end for
end for

Algorithm 2 solves the Minimax Problem (11) using ADMoM technique. It is a fully distributed algorithm which only requires transmitting $\mathbf{r}_v$ between each nodes. The attacker's behavior is captured as calculating δ_v by solving the linear programming Problem (13) with the learner's decision variable $\mathbf{r}_v$. The learner's behavior is captured as computing (14)–(17) with δ_v from the attacker. Since the learner transmits $\mathbf{r}_v$ to each neighboring nodes, misleading information will eventually spread in the whole network, which leads to misclassifications in all nodes.

5. NUMERICAL RESULTS

In this section, we summarize numerical results of DSVM under adversarial environments. We use empirical risk to measure the performance of DSVM. The empirical risk at node v at step t is defined as follows:

$$\mathbf{R}_v^{(t)} := \frac{1}{\tilde{N}_v}\sum_{n=1}^{\tilde{N}_v}\frac{1}{2}|\tilde{y}_{vn} - \hat{y}_{vn}^{(t)}|, \tag{18}$$

where $\tilde{y}_{vn}$ is the true label; $\hat{y}_{vn}^{(t)}$ is the predicted label; and $\tilde{N}_v$ represents the number of testing samples in node v. The empirical risk (18) sums over the number of misclassified samples in node v, and then divides it by the number of all testing samples in node v. Notice that testing samples can vary for different nodes. In order to measure the global performance, we use the global empirical risk defined as follows:

$$\mathbf{R}_G^{(t)} := \frac{1}{\tilde{N}}\sum_{v\in\mathcal{V}}\sum_{n=1}^{\tilde{N}_v}\frac{1}{2}|\tilde{y}_{vn} - \hat{y}_{vn}^{(t)}|, \tag{19}$$

where $\tilde{N} = \sum_{v\in\mathcal{V}} \tilde{N}_v$, representing the total number of testing samples. Clearly, a higher global empirical risk shows that there are more testing samples being misclassified, i.e., a worse performance of DSVM. We use the first experiment to illustrate the significant impact of the attacker.

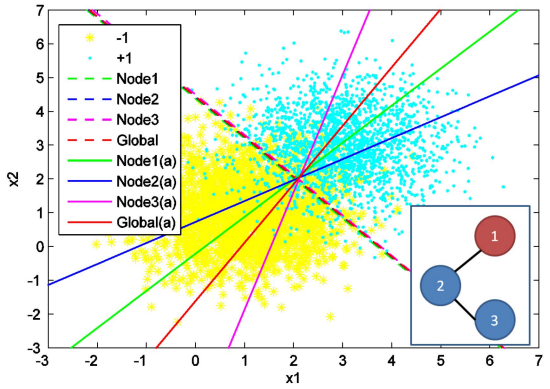
Consider a network with 3 nodes, which can be seen at the bottom right corner of Fig. 3(a). Each node contains 80 training samples and 1000 testing samples from the same global training dataset, which is shown as points and stars in Fig. 3(a). Yellow stars and magenta points are labelled as -1 and $+1$, respectively. They are generated from two-dimensional Gaussian distributions with mean vectors $[1,1]$ and $[3,3]$, with the same covariance matrix $[1,0;0,1]$. The learner has the ability $C_l = 1$ and $\eta = 1$. The attacker has the atomic action set parameter $C_{1,\delta} = 9{,}000{,}000$, and the cost parameter $C_a = 1$. The attacker only attacks Node 1 and the attack starts from the beginning of the training process. Numerical results are shown in Fig. 3(b). Notice that the risks when there is an attacker are much higher than the risks when there is no attacker, which indicates that the attacker has a significant impact on the learner. Also, we can conclude that the risks at the node under attack are much higher than the risks in nodes without attack, but both of them are higher than the risks when there is no attacker in the network. This shows that the attacker has the ability to affect uncompromised nodes through network connections. We can also observe from Fig. 3(a) that the solid lines, which represent the situation when there is an attacker, cannot separate yellow stars and magenta points.

It is clear that the attacker can cause disastrous results for the learner. In our previous work [34], we have shown that results of the game between the DSVM learner and the attacker are affected by both the attacker's ability and the network topologies. We summarize our previous numerical results from [34] in the following observations.

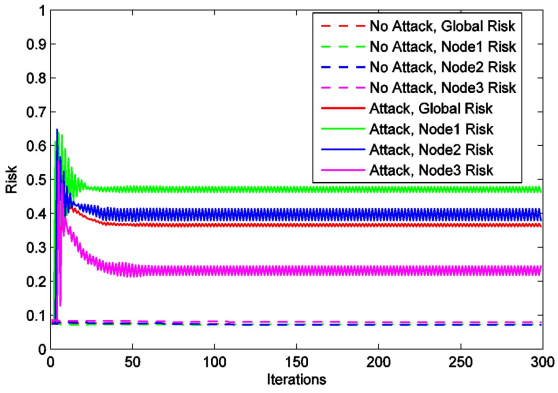
OBSERVATION 1

The attacker's ability is captured by four measures, i.e., (i) the time t for the attacker to take an action, (ii) the atomic action set parameter $C_{v,\delta}$, (iii) the cost parameter C_a, and (iv) the number of compromised nodes $|\mathcal{V}_a|$. The impact of them is summarized as follows.

- *The time t for an attacker to take an action does not affect the equilibrium risks.*
- *A larger $C_{v,\delta}$ increases the equilibrium risk, as a larger $C_{v,\delta}$ indicates that the attacker can make a larger modification on training data.*
- *A larger C_a decreases the equilibrium risk, as a larger C_a restricts the attacker's actions to make changes.*



(a)



(b)

Fig. 3. Evolution of the empirical risks of ADMoM-DSVM with an attacker at a network with 3 nodes shown at the bottom right corner of figure (a). The attacker only attacks red node 1 from the beginning of the training process. Training data and testing data are generated from two Gaussian classes. Dotted lines and solid lines show the results when there is no attacker and there is an attacker, respectively. Different colors represent risks or discriminant lines of different nodes.

- *A larger number of compromised nodes $|\mathcal{V}_a|$ increases the equilibrium risk as attacking more nodes gives the attacker access to modify more training samples.*

OBSERVATION 2 *Denote the degree of node v as $|\mathcal{B}_v|/(|\mathcal{V}|-1)$ and the degree of a network as the average degree of all the nodes. The impact of network topologies are summarized as follows.*

- *Networks with higher degrees and fewer nodes are less vulnerable to attackers.*
- *Balanced networks, i.e., nodes in these networks have the same number of neighboring nodes, are more secure than unbalanced networks.*

Notice that here we assume that each node in the network contains the same number of training samples.

OBSERVATION 3 *For a specified network, assuming that all the nodes contain the same number of training samples, the impact of a node is summarized as follows.*

- *Nodes with higher degrees are more vulnerable, i.e., attacking nodes with higher degrees leads to a higher global equilibrium risk.*

- *Attacking nodes with lower degrees can lead to a higher global equilibrium risk if the network contains nodes with higher degrees, comparing to networks without high degree nodes but has the same average degree.*

Observations 1, 2 and 3 summarize our previous numerical experiments in [34]. From Observation 1, the attacker makes a larger impact when he has a higher capability, such as, he has a larger $C_{v,\delta}$ and a smaller C_a, or he can attack more nodes. From Observation 2, on the one hand, the attacker can choose to attack unbalanced networks with lower degrees and more nodes to make a more significant impact on the learner, on the other hand, the learner should select balanced networks with higher degrees and fewer nodes to reduce potential damages from attacker. From Observation 3, the attacker benefits more from attacking nodes with higher degrees, while the learner should avoid using high degree nodes. These observations provide both players the strategies to make a larger impact on the other ones. In the following subsections, we present in detail how the attacker and the learner can find better strategies against each other.

5.1. Attacker's Strategies

Consider that a DSVM learner operates training data on an unbalanced network. We assume that the attacker knows the learner's algorithm and the network topology. We also assume that the attacker has the ability to attack any nodes in this network with $\sum_{v=1}^{V_a} C_{v,\delta} \leq C_{V_a,\delta}$, i.e., a total sum of all changed values in the network should be bounded by $C_{V_a,\delta}$. Notice that bounded $C_{V_a,\delta}$ represents a trade-off between attacking more nodes V_a and attacking each nodes with larger $C_{v,\delta}$. Since attacking different nodes leads to different global equilibrium risks, and the attacker prefers higher risks, there exists an optimal strategy of selecting $\mathcal{V}_a$ and $\{C_{v,\delta}\}_{v\in\mathcal{V}_a}$ for the attacker which has the highest equilibrium global risk with a bounded $C_{V_a,\delta}$. The optimal strategy can be found by solving the following problem:

$$\max_{\{\mathcal{V}_a, C_{v,\delta}\}} \min_{\{\mathbf{w}_v, b_v, \{\xi_{vn}\}\}} \max_{\{\delta_{vn}\}} \frac{1}{2}\sum_{v\in\mathcal{V}} \|\mathbf{w}_v\|_2^2 + VC_l \sum_{v\in\mathcal{V}}\sum_{n=1}^{N_v} \xi_{vn} - C_a \sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v} \|\delta_{vn}\|_0 - \sum_{v\in\mathcal{V}_a} h_v$$

s.t.

$$\begin{aligned}
&\mathrm{y}_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn} + b_v) \geq 1 - \xi_{vn}, && \forall v \in \mathcal{V}_l,\ n = 1,\ldots,N_v;\\
&\mathrm{y}_{vn}(\mathbf{w}_v^T(\mathbf{x}_{vn} - \delta_{vn}) + b_v) \geq 1 - \xi_{vn}, && \forall v \in \mathcal{V}_a,\ n = 1,\ldots,N_v;\\
&\xi_{vn} \geq 0, && \forall v \in \mathcal{V},\ n = 1,\ldots,N_v;\\
&\mathbf{w}_v = \mathbf{w}_u,\ b_v = b_u, && \forall v \in \mathcal{V},\ u \in \mathcal{B}_v;\\
&\delta_{vn} \in \mathcal{U}_v,\ \text{i.e.,}\ \textstyle\sum_{n=1}^{N_v} \|\delta_{vn}\|_2^2 \leq C_{v,\delta}, && \forall v \in \mathcal{V}_a;\\
&\textstyle\sum_{v=1}^{V_a} C_{v,\delta} \leq C_{V_a,\delta}.
\end{aligned} \tag{20}$$

Note that Problem (20) extends Problem (10) by maximizing over variables $\mathcal{V}_a$ and $\{C_{v,\delta}\}$ with a new constraint $\sum_v^{V_a} C_{v,\delta} \leq C_{V_a,\delta}$ that captures a bound on the attacker's ability. The last term h_v in the objective function represents the cost of attacking node v.

Problem (20) is based on the assumption that the attacker has the knowledge of the learner's algorithm and the network topology. The learner aims to minimize the classification errors in Problem (10), while the attacker aims to maximize those errors. In Problem (20), the attacker has two components to maximize. Maximizing over $\{\delta_{vn}\}$ is the same as in Problem (10). Maximizing over $\mathcal{V}_a$ and $\{C_{v,\delta}\}$ indicates the objective of the attacker to maximize the equilibrium risk of the original game with a bounded $C_{V_a,\delta}$ and a cost h_v. By solving Problem (20), the attacker can find the optimal strategy of $\mathcal{V}_a$ and $\{C_{v,\delta}\}_{v\in\mathcal{V}_a}$, which has the maximized equilibrium risk.

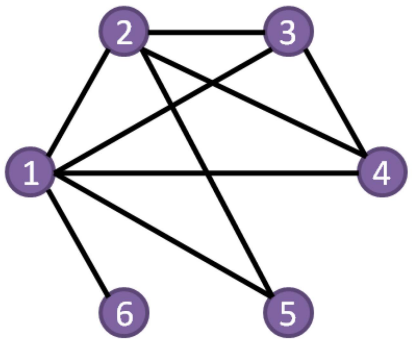
However, solving Problem (20) can be a challenge as the decision variables $\mathcal{V}_a$ and $C_{v,\delta}$ are coupled with the decisions of the learner and the attacker. The attacker is still able to make a larger impact on the learner by Observation 1, 2 and 3. For example, instead of randomly picking nodes to attack and assigning $C_{v,\delta}$, the attacker can strategically attack high degree nodes, which leads to a higher risk from our observations. One numerical example is shown in Fig. 4.

Consider the learner operates on a network shown in Fig. 4(a). We assume that the attacker can only attack 2 nodes with the bound $C_{V_a,\delta} = 2\times10^8$, and the cost of attacking node v, i.e., h_v are the same for every node. A naive attacker may randomly attack 1 node with $C_{v,\delta} = 2\times10^8$. However, a smart attacker will choose 2 nodes with higher degrees, and by modifying the value of $C_{v,\delta}$ in both nodes, he can make a larger impact on the learner. Numerical results are shown in Fig. 4(b).

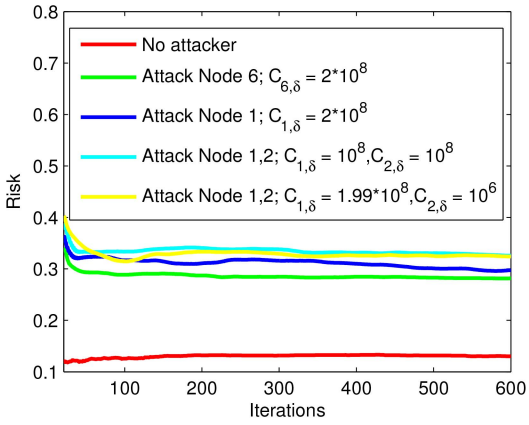
From Fig. 4, the attacker has four different strategies, (i) the attacker only attacks Node 6, (ii) the attacker only attacks Node 1, (iii) the attacker attacks Node 1,2 with balanced ability, and (iv) the attacker attacks Node 1,2 with unbalanced ability. We can see that when the attacker choose Strategy (iii), the risk is the highest. However, if we take the cost of attacking different nodes into consideration, this strategy may not be the best as attacking 2 nodes may cost too much. But from the example, we can see that Observations 1, 2 and 3 provide us a way to find a better strategy for the attacker. They also provide us tools of finding better strategies for the learner.

5.2. Learner's Strategies

A DSVM learner aims to find the best discriminant functions with the least classification errors. Since an attacker will increase the classification errors, a better strategy of the learner is to reduce the attacker's impact as much as possible. In this section, we assume that the learner is trying to find the strategy of network topology that has a smallest risk with potential attacks.



(a)



(b)

Fig. 4. Evolution of moving average of global empirical risks of ADMoM-DSVM with the attacker on Spam dataset [14]. Each node contains 40 training samples. Attacker has four strategies with same $C_{V_a,\delta} = 2 \times 10^8$ and $C_a = 0.01$.

We assume that the learner has the ability to select any kinds of network topologies and assign any number of training samples in each node. The learner's strategy can be found by solving the following problem,

$$\min_{\{\mathcal{V},\mathcal{B}_v,N_v\}} \min_{\{\mathbf{w}_v,b_v,\{\xi_{vn}\}\}} \max_{\{\delta_{vn}\}} \frac{1}{2}\sum_{v\in\mathcal{V}} \|\mathbf{w}_v\|_2^2 + VC_l \sum_{v\in\mathcal{V}}\sum_{n=1}^{N_v} \xi_{vn}$$
$$- C_a \sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v} \|\delta_{vn}\|_0 - \sum_{v\in\mathcal{V}} T_v(N_v) - \sum_{v\in\mathcal{V}} B_v(\mathcal{B}_v)$$

s.t.

$$\begin{aligned}
&y_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn} + b_v) \geq 1 - \xi_{vn}, && \forall v \in \mathcal{V}_l,\ n = 1,\ldots,N_v;\\
&y_{vn}(\mathbf{w}_v^T(\mathbf{x}_{vn} - \delta_{vn}) + b_v) \geq 1 - \xi_{vn}, && \forall v \in \mathcal{V}_a,\ n = 1,\ldots,N_v;\\
&\xi_{vn} \geq 0, && \forall v \in \mathcal{V},\ n = 1,\ldots,N_v;\\
&\mathbf{w}_v = \mathbf{w}_u,\ b_v = b_u, && \forall v \in \mathcal{V},\ u \in \mathcal{B}_v;\\
&\delta_{vn} \in \mathcal{U}_v,\ \text{i.e.,}\ \textstyle\sum_{n=1}^{N_v} \|\delta_{vn}\|_2^2 \leq C_{v,\delta}, && \forall v \in \mathcal{V}_a.
\end{aligned} \tag{21}$$

Note that Problem (21) extends Problem (10) by minimizing over variables $\mathcal{V}$, $\mathcal{B}_v$ and N_v with new costs $T_v(N_v)$ and $B_v(\mathcal{B}_v)$. $T_v(N_v)$ represents the cost of training N_v samples in node v, $B_v(\mathcal{B}_v)$ represents the cost of sending information from node v to his neighboring nodes $u \in \mathcal{B}_v$. Problem (21) can be understood as the learner's objective of minimizing equilibrium risk of the game with potential attacks by finding the best network topology $\mathcal{V}$, $\mathcal{B}_v$ and training samples' assignments N_v.

Solving Problem (21) can be a challenge as $\mathcal{V}$, $\mathcal{B}_v$ and N_v are coupled with the decisions of the learner and the attacker. But the learner can benefit from Observation 1, 2 and 3. For example, the learner should select a balanced network with fewer nodes and higher degree, which has a smaller equilibrium risk. However, in reality, the learner may not be able to modify network topologies as the connections between nodes can be fixed, or it may not be possible to add connections between nodes. Thus, to reduce the impact of the attacker, the learner requires actionable defense strategies.

In the following sections, we present four different defense strategies, and we verify their effectiveness with numerical experiments.

6. DSVM DEFENSE STRATEGIES

In this section, we present four defense strategies (DSs) for the DSVM learner. We show their effectiveness with numerical experiments.

6.1. DSVM Defense Strategy 1: Selecting Network Topology

DS 1 for the learner is to find a network topology that has a smaller risk when there is an attacker. From the last section, the learner can find the network topology by solving Problem (21). However, Problem (21) is difficult to solve. But we are still able to find a secure network topology using Observation 2 and 3. The network topology should be close to a balanced network with fewer nodes and a higher degree. A numerical experiment is shown in Fig. 5.

Consider that a DSVM learner trains 300 samples, and he aims to select a secure network topology from four topologies shown in Fig. 5(a). DS 1 indicates that we should select network A or B as network A has the smallest number of nodes among all the networks, and network B has the highest degree among networks B, C, D. Numerical results in Fig. 5(b) show that DS 1 has smaller risks.

Though selecting a network with fewer nodes reduces the vulnerability of the learner, but each node is required to train more training samples, which takes more time and memory usages. In addition, the learner may not have the ability to select a proper network topology as most networks are fixed. Moreover, improving the degree of the network may not be always applicable as adding connections between nodes is costly. Thus DS 1 is suitable for cases when the network connections are convenient to modify.

Consider the application in which several wireless temperature sensors in the building aim to decide whether to open their air conditioners or not. Since a large building may have hundreds of sensors and the

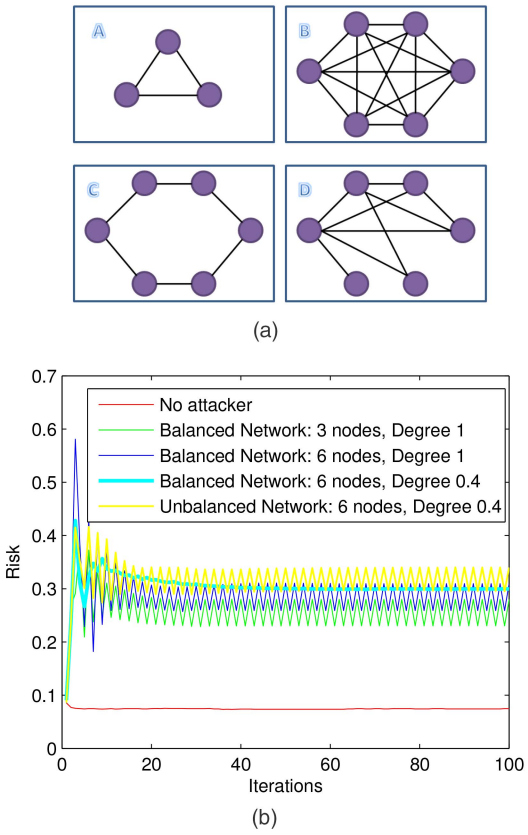


Fig. 5. Evolution of the global empirical risks of ADMoM-DSVM with an attacker on a random dataset. The learner has four options of network topologies which are shown in figure (a). Topology A is a balanced network with 3 nodes and degree 1, each node in this network contains 80 training samples. Network B is a balanced network with 6 nodes and degree 1. Network C is a balanced network with 6 nodes and degree 0.4. Network D is an unbalanced network with 6 nodes and degree 0.4. Each node in network B,C and D contains 40 training samples. Attacker attacks 1 node in network A, but he attacks 2 nodes in network B,C,D, so the attacker can modify the same number of training samples in different network topologies. The attacker has $C_{v,\delta} = 5 \times 10^5$ and $C_a = 0.01$.

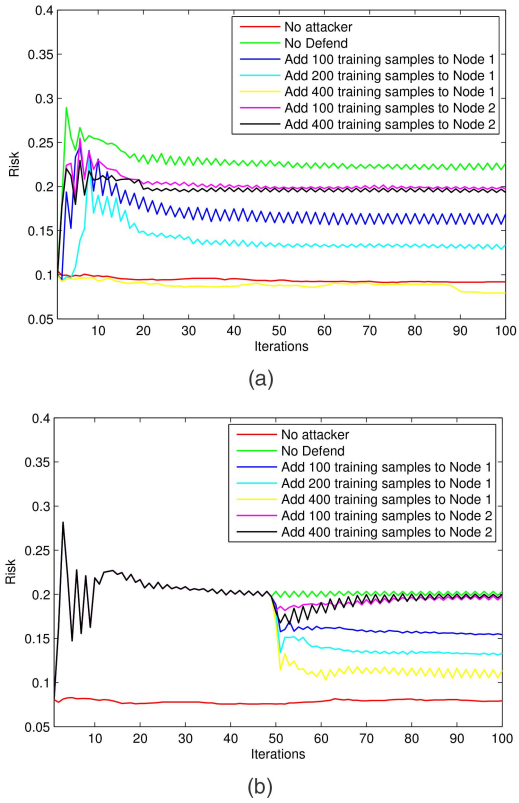


Fig. 6. Evolution of global empirical risks of ADMoM-DSVM with an attacker at a balanced network with 6 nodes and degree 0.4 on random dataset, which is shown in Fig. 5(a) as Network C. Each node contains 40 training samples. Attacker only attacks node 1 with $C_{1,\delta} = 10^6$ and $C_a = 0.01$. (a) Defense starts from step 0. (b) Defense starts from step 50.

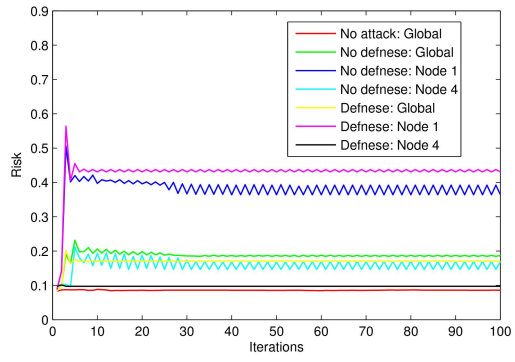
temperatures are always changing with time, centralized classifications may take a significant amount of time to collect, transmit, and process the data. DSVMs can be used here as each sensor operates on its own data, and only a small amount of information is transmitted between sensors. But if there is an attacker who has the ability to modify the training data in several sensors, then the sensors in the building will lead to wrong decisions. In this case, wireless temperature sensors can adapt and modify their network topology. Thus, a secure strategy here is to use DS 1 to create a balanced network with fewer sensors and a higher average degree.

6.2. DSVM Defense Strategy 2: Adding Training Samples

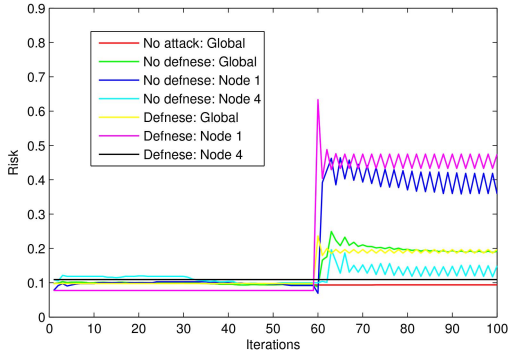
Since the attacker is limited to making modifications on the training data, a higher volume of training data will decrease the ratio of incorrect data at a node. As long as most of the data are correct, the learner can find the discriminant function with small classification errors. Thus adding more training samples becomes a reasonable defense strategy. Numerical experiments are shown in Fig. 6.

From Fig. 6, when we add training samples to network, the risk is lower. Thus adding training samples is a proper defense strategy. Note that more samples we add, the lower the risk will be. Adding training samples to compromised nodes turns out to be more efficient than adding to uncompromised nodes. However, training more samples requires more time and memory usages, which sacrifices efficiency. Thus, DS 2 is a trade-off between efficiency and security.

DS 2 is suitable for the case when the learner cannot change the network topology, but the size of training data is sufficiently large and each node has a strong computing capability. For example, consider an application where several environmental stations plan to detect whether some areas are under pollution with a wired communication network. DSVMs are suitable to process a large amount of data computations and transmissions. However, if an attacker modifies the training data, environmental stations may lead to misdetection. In this case, DS 1 may not be applicable as the wired connections between each station are fixed. However, since each station can collect enough training data and



(a)



(b)

Fig. 7. Evolution of the empirical risks of ADMoM-DSVM with the attacker at a balanced network with 4 nodes degree 0.4 on random dataset. Each node contains 60 training samples. Attacker only attacks Node 1 with $C_{1,\delta} = 10^5$ and $C_a = 0.01$. $\tau = 0.1$. (a) Attack starts from step 0. (b) Attack starts from step 60.

has a higher computation capability, DS 2 is more appropriate and the learner can add more training samples to each node to make the training process more secure. Note that using more samples requires additional time to train and more spaces to store data.

6.3. DSVM Defense Strategy 3: Verification Method

DS 1 suggests that the learner uses a balanced network with fewer nodes and a higher degree. However, using fewer nodes requires that each node trains more training samples, which sacrifices the efficiency. Increasing the degree of the network requires creating more connections between nodes, which are usually not applicable as building new lines may incur a high cost. DS 2 indicates that adding more training samples can reduce the vulnerability of the network, which also sacrifices the efficiency. Thus, both DS 1 and DS 2 have their limitations on securing a training process. In this section, we present a verification method that reduces the vulnerability without modifying the network topology or adding training samples.

In ADMoM-DSVM Algorithm 1, each node in the network receives $\mathbf{r}_u$ from his neighboring nodes and it also sends his $\mathbf{r}_v$ to his neighboring nodes at each step. Since $\mathbf{r}_u$ from neighboring nodes of node v contributes to the updates of $\mathbf{r}_v$, a wrong $\mathbf{r}_u$ can lead to an incorrect update of $\mathbf{r}_v$. As a result, if node v is protected from receiving wrong $\mathbf{r}_u$ from compromised nodes, it can have a correct discriminant function.

Recall DSVM Problem (2), note that consensus constraints $\mathbf{r}_v = \omega_{vu}, \omega_{vu} = \mathbf{r}_u$ force all the local decision variables $\mathbf{r}_v$ to agree with each other. Thus, $\mathbf{r}_1^{(t)} \approx \cdots \approx \mathbf{r}_V^{(t)}$ should hold for every step t during the training process. Thus, if $\mathbf{r}_v$ violates this, then the learner can tell that node v is under attack. With Algorithm 1, if node v finds $\mathbf{r}_u$ is significantly different from $\mathbf{r}_v$, then he will reject using $\mathbf{r}_u$ to update himself. We call this method as the verification method. The ADMoM-DSVM algorithm with verification method can be summarized as Algorithm 3.

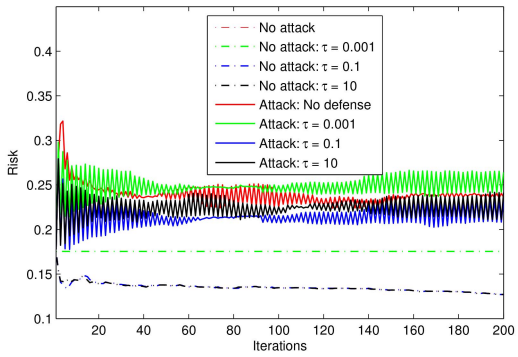
ALGORITHM 3: *DSVM with Verification*

Randomly initialize $\mathbf{r}_v^{(0)}$, $\lambda_v^{(0)}$, $\omega_{vu}^{(0)}$, set $\alpha_v^{(0)} = \mathbf{0}_{(p+1)\times 1}$, set $\widehat{\mathcal{B}_v} = \mathcal{B}_v$.

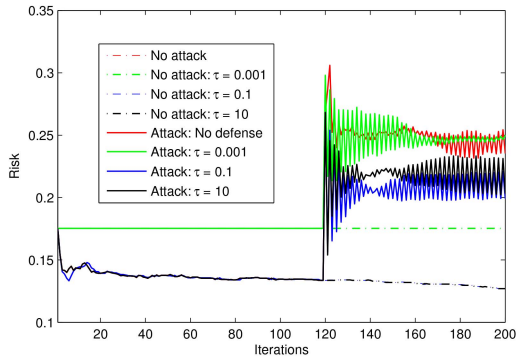
1: **for** $t = 0, 1, 2, \ldots$ **do**
2: **for all** $v \in \mathcal{V}$ **do**
3: Compute $\lambda_v^{(t+1)}$ via (3) with $\widehat{\mathcal{B}_v}$.
4: Compute $\mathbf{r}_v^{(t+1)}$ via (4) with $\widehat{\mathcal{B}_v}$.
5: **end for**
6: **for all** $v \in \mathcal{V}$ **do**
7: Broadcast $\mathbf{r}_v^{(t+1)}$ to all neighbors $u \in \mathcal{B}_v$.
8: **end for**
9: **for all** $v \in \mathcal{V}$ **do**
10: Set $\hat{\mathcal{B}}_v = \emptyset$.
11: **for all** $u \in \mathcal{B}_v$ **do**
12: **if** $\left| 1 - \frac{\|\mathbf{r}_u^{(t+1)}\|_2}{\|\mathbf{r}_v^{(t+1)}\|_2} \right| < \tau$
13: Set $u \in \widehat{\mathcal{B}_v}$.
14: **end if**
15: **end for**
16: **end for**
17: **for all** $v \in \mathcal{V}$ **do**
18: Compute $\omega_{vu}^{(t+1)}$ via (5) with $\widehat{\mathcal{B}_v}$.
19: Compute $\alpha_v^{(t+1)}$ via (6) with $\widehat{\mathcal{B}_v}$.
20: **end for**
21: **end for**

Algorithm 3 differs from Algorithm 1 in the verification method. Each node computes with information only from trusted neighboring nodes $u \in \hat{\mathcal{B}}_v$. The verification method is based on the inequality in step 12 of Algorithm 3. τ indicates the tolerance of indifference from $\mathbf{r}_u$ to $\mathbf{r}_v$, and $\tau \geq 0$. When τ is close to 0, node v is very sensitive to the information from other nodes, and it only uses $\mathbf{r}_u$ that is very close to $\mathbf{r}_v$. Numerical experiments are shown in Fig. 7 and Fig. 8.

We can see from Fig. 7 that the global risk has decreased when there is a verification method. Note that in uncompromised node 4, the risk is close to the risk when there is no attacker, while in compromised node 1, the risk is higher than the risk when there is



(a)



(b)

Fig. 8. Evolution of the global empirical risks of ADMoM-DSVM with the attacker at a balanced network with 4 nodes degree 0.4 on Spambase dataset [14]. Each node contains 60 training samples. Attacker only attacks node 1 with $C_{1,\delta} = 10^6$ and $C_a = 0.01$. (a) Attack starts from step 0. (b) Attack starts from step 120.

no defense. This indicates that, though the verification method protects uncompromised nodes from receiving misleading information, it also prevents compromised nodes from receiving correct information.

Fig. 8 compares the global risks when the learner uses different τ. We can see that when $\tau = 10$, the risk is higher than the risk when $\tau = 0.1$, thus some of the misleading information is still able to be spread in the network. When $\tau = 0.001$, we can see that the risk is even higher than the risk when there is no defense. Also note that when there is no attacker, the risk of DSVM with $\tau = 0.001$ does not converge to the risk of normal DSVM. This indicates that when τ is close to 0, the misleading information cannot be spread to other nodes, but the useful information is also forbidden to transmit. Thus DS 3 requires a proper selection of τ.

DS 3 is suitable for the case when training data are used in a large network. Since it is difficult for the attacker to attack many nodes at the same time, for a network with a large number of nodes, all the uncompromised nodes can be kept from being affected by the compromised nodes. Moreover, the learner can distinguish compromised nodes by their high local classification risks, and thus, without revoking the training process and retraining all the data in every node, the learner is able to maintain the resilience of the training process by deleting or correcting the compromised nodes. Comparing to DS 1 and 2, DS 3 does not sacrifice efficiency to maintain security, but the compromised nodes may result in worse performances.

6.4. DSVM Defense Strategy 4: Rejection Method

DSs 1, 2 and 3 have shown that with selecting proper network topologies, adding training samples and verification method, DSVM learner can be less vulnerable to attacks. However, DSs 1 and 2 will sacrifice efficiency. In DS 3, compromised nodes may result in worse performances. In this section, we present the rejection method where each node rejects unreasonable updates. With the rejection method, once there is an attacker, the iteration will terminate to prevent further damages caused by the attacker.

The rejection method relies on a combined residual, which measures both the primal and dual error simultaneously:

$$J^{(t+1)} = \eta \sum_{v\in\mathcal{V}} \sum_{u\in B_v} \|\omega_{vu}^{(t+1)} - \omega_{vu}^{(t)}\|_2^2 + \frac{2}{\eta} \sum_{v\in\mathcal{V}} \|\alpha_v^{(t+1)} - \alpha_v^{(t)}\|_2^2. \tag{22}$$

Note that the combined residual contains two terms. The first term measures the dual residual. The second term measures the primal residual. The combined residual has the following lemma [15].

LEMMA 4 *Iterations (3)–(6) satisfy that* $J^{(t+1)} \leq J^{(t)}$, *which can also be rewritten as:*

$$\begin{aligned} &\eta \sum_{v\in\mathcal{V}} \sum_{u\in B_v} \|\omega_{vu}^{(t+1)} - \omega_{vu}^{(t)}\|_2^2 + \frac{2}{\eta} \sum_{v\in\mathcal{V}} \|\alpha_v^{(t+1)} - \alpha_v^{(t)}\|_2^2 \\ &\quad \leq \eta \sum_{v\in\mathcal{V}} \sum_{u\in B_v} \|\omega_{vu}^{(t)} - \omega_{vu}^{(t-1)}\|_2^2 + \frac{2}{\eta} \sum_{v\in\mathcal{V}} \|\alpha_v^{(t)} - \alpha_v^{(t-1)}\|_2^2. \end{aligned} \tag{23}$$

A proof of Lemma 4 can be found in [15]. Lemma 4 indicates that the combined residual always decreases over time. Since the attacker aims to break the training process, this inequality will not be satisfied when there is an attacker. Note that computing Inequality (23) requires ω_{vu} and α_v from every node, which can be achieved by a fusion center in centralized machine learning problems. However, since the learner uses a fully distributed network without a fusion center, we decentralize Inequality (23) into $|\mathcal{V}|$ distributed inequalities, for $v \in \mathcal{V}$:

$$\begin{aligned} &\eta \sum_{u\in B_v} \|\omega_{vu}^{(t+1)} - \omega_{vu}^{(t)}\|_2^2 + \frac{2}{\eta} \|\alpha_v^{(t+1)} - \alpha_v^{(t)}\|_2^2 \\ &\quad \leq \eta \sum_{u\in B_v} \|\omega_{vu}^{(t)} - \omega_{vu}^{(t-1)}\|_2^2 + \frac{2}{\eta} \|\alpha_v^{(t)} - \alpha_v^{(t-1)}\|_2^2. \end{aligned} \tag{24}$$

Note that there is no guarantee that Inequality (24) holds based on Inequality (23). As a result, we relax the

distributed inequality with a parameter $\rho > 1$, which is summarized in the following proposition.

PROPOSITION 1 *Iterations (3)–(6) satisfy that* $J_v^{(t+1)} \leq \rho J_v^{(t)}$, *where*

$$J_v^{(t)} = \eta \sum_{u \in B_v} \|\omega_{vu}^{(t)} - \omega_{vu}^{(t-1)}\|_2^2 + \frac{2}{\eta} \|\alpha_v^{(t)} - \alpha_v^{(t-1)}\|_2^2. \tag{25}$$

PROOF

Let us assume that $J_v^{(t+1)} \leq \rho J_v^{(t)}$ does not hold for $v = v_0$, we have $J_{v_0}^{(t+1)} > \rho J_{v_0}^{(t)}$ and $J_{v \neq v_0}^{(t+1)} \leq \rho J_{v \neq v_0}^{(t)}$. As a result, $J_{v_0}^{(t+1)} > \rho^{(t+1)} J_{v_0}^{(0)}$ which increases exponentially with $\rho > 1$. Since $J_v^{(t)}$ is always larger than 0, Inequality (23) will be violated eventually. Proposition 1 holds.

With the inequality in Proposition 1, the new DSVM algorithm with rejection method can be summarized into Algorithm 4. In Algorithm 4, if the inequality at Step 15 is satisfied, the current update will be rejected. $J_v^{(0)}$ should be set to be sufficiently large to pass the first rejection test. Numerical experiments are shown in Fig. 9, Fig. 10, and Fig. 11.

ALGORITHM 4: *DSVM with Rejection*

Randomly initialize $\mathbf{r}_v^{(0)}$, $\lambda_v^{(0)}$, $\omega_{vu}^{(0)}$ and $\alpha_v^{(0)} = \mathbf{0}_{(p+1)\times 1}$, set $J_v^{(0)}$ very large.

1: **for** $t = 0, 1, 2, \ldots$ **do**
2: **for all** $v \in \mathcal{V}$ **do**
3: Compute $\lambda_v^{(t+1)}$ via (3).
4: Compute $\mathbf{r}_v^{(t+1)}$ via (4).
5: **end for**
6: **for all** $v \in \mathcal{V}$ **do**
7: Broadcast $\mathbf{r}_v^{(t+1)}$ to all neighbors $u \in \mathcal{B}_v$.
8: **end for**
9: **for all** $v \in \mathcal{V}$ **do**
10: Compute $\omega_{vu}^{(t+1)}$ via (5).
11: Compute $\alpha_v^{(t+1)}$ via (6).
12: Compute $J_v^{(t+1)}$ via (25).
13: **end for**
14: **for all** $v \in \mathcal{V}$ **do**
15: **if** $J_v^{(t+1)} > \rho J_v^{(t)}$
16: $\lambda_v^{(t+1)} = \lambda_v^{(t)}$, $\mathbf{r}_v^{(t+1)} = \mathbf{r}_v^{(t)}$,
17: $\alpha_v^{(t+1)} = \alpha_v^{(t)}$, $\omega_{vu}^{(t+1)} = \omega_v^{(t)}$,
18: $J_v^{(t+1)} = J_v^{(t)}$.
19: **end if**
20: **end for**
21: **end for**

From Fig. 9, we can see that the DSVM algorithm with rejection method has a lower risk than the normal algorithm when there is an attacker. And it has the same performance when there is no attacker, which indicates that when $\rho = 1.5$, rejection method does not affect the training process. Fig. 10 and Fig. 11 show the results when $\rho = 1$ and $\rho = 100$, respectively. We can see from

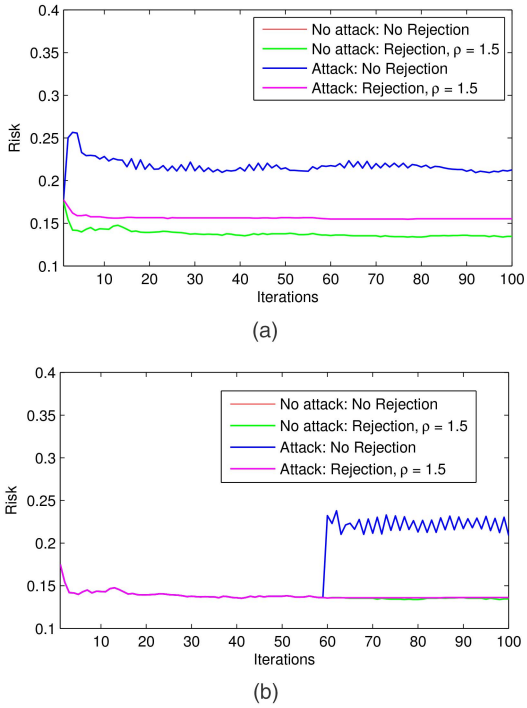


Fig. 9. Evolution of the empirical risks of ADMoM-DSVM Rejection with the attacker at a balanced network with 4 nodes of degree 0.4 on Spambase dataset [14]. Each node has 60 training samples. The attacker only attacks 1 node with $C_{1,\delta} = 10^5$ and $C_a = 0.01$. The rejection method has $\rho = 1.5$. (a) Attack starts from step 0. (b) Attack starts from step 60.

Fig. 10 that when $\rho = 1$, the risk is lower when there is an attacker, but convergence slows down when there is no attacker. We can see from Fig. 11 that when $\rho = 100$, the risk with rejection method is even higher than the risk of the standard algorithm, because wrong updates can still be treated as a correct update and accumulates as iteration goes.

From the numerical experiments, the value of ρ is important to the rejection method. A smaller ρ may slow down the convergence of the DSVM algorithm without attacker, a larger ρ does not prevent attacks. With a properly selected ρ, the training process becomes less vulnerable to attackers.

DS 4 is suitable for a wide range of applications as wrong updates will be rejected. Comparing to DSs 1 and 2, DS 4 does not sacrifice efficiency. Comparing to DS 3, compromised nodes in DS 4 has been kept from being further damaged by the attacker. One possible drawback of DS 4 is that it may require insights of the problem to find a proper ρ.

Each defense strategy is suitable for a different scenario and applications. The choice of defense strategies will depend on the applications and the constraints on the defender's actions. Though four defense strategies have their own advantages and disadvantages, a combination of all the defense strategies can be used to secure the training process of the learner.

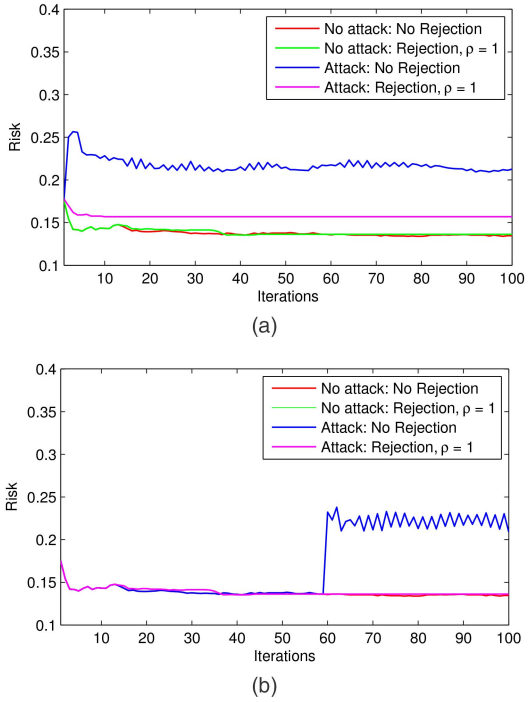


Fig. 10. Evolution of the empirical risks of ADMoM-DSVM Rejection with the attacker at a balanced network with 4 nodes of degree 0.4 on Spambase dataset [14]. Each node has 60 training samples. The attacker only attacks 1 node with $C_{1,\delta} = 10^5$ and $C_a = 0.01$. The rejection method has $\rho = 1$. (a) Attack starts from step 0. (b) Attack starts from step 60.

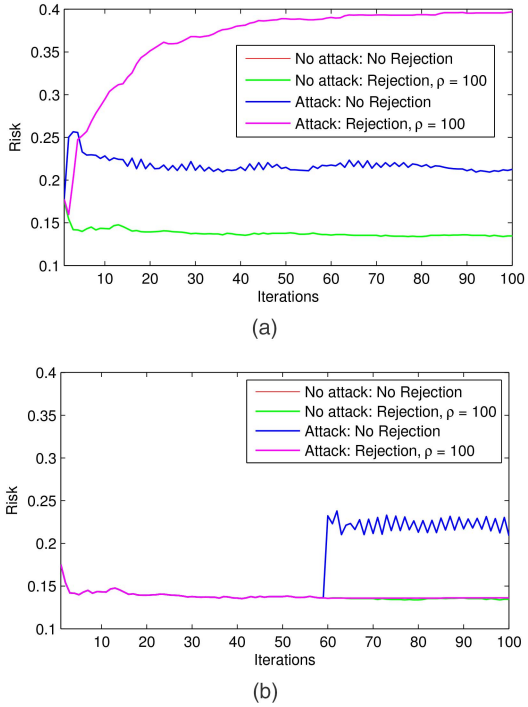


Fig. 11. Evolution of the empirical risks of ADMoM-DSVM Rejection with the attacker at a balanced network with 4 nodes of degree 0.4 on Spambase dataset [14]. Each node has 60 training samples. The attacker only attacks 1 node with $C_{1,\delta} = 10^5$ and $C_a = 0.01$. The rejection method has $\rho = 100$. (a) Attack starts from step 0. (b) Attack starts from step 60.

7. CONCLUSION

Distributed support vector machines are ubiquitous but inherently vulnerable to adversaries. This paper has investigated defense strategies of DSVM against potential attackers. We have established a game-theoretic framework to capture the strategic interactions between an attacker and a learner with a network of distributed nodes. We have shown that the nonzero-sum game is strategically equivalent to a zero-sum game, and hence, its equilibrium can be characterized by a saddle-point equilibrium solution to a minimax problem. By using the technique of ADMoM, we have developed secure and resilient algorithms that can respond to the adversarial environment. We have shown that a balanced network with fewer nodes and a higher degree is less vulnerable to the attacker. Moreover, adding more training samples has been proved to reduce the vulnerability of the system. We have shown that verification method where each node verifies information from neighboring nodes can protect uncompromised nodes from receiving misleading information, but compromised nodes are also prevented from receiving correct information. We have shown that rejection method where each node rejects unreasonable updates can stop global training process from deterioration, thus wrong information is thwarted from affecting the system. One direction of future works is to extend the current framework to investigate nonlinear DSVM and other machine learning algorithms.

APPENDIX A: PROOF OF LEMMA 2

A detailed proof of Lemma 2 can be found in our previous work [34]. By using hinge loss function, we reformulate Problem (10) into the following problem:

$$\begin{aligned}\min_{\{\mathbf{w}_v,b_v\}}\max_{\{\delta_{vn}\}}\ & \frac{1}{2}\sum_{v\in\mathcal{V}}\|\mathbf{w}_v\|_2^2 \\ & + V_l C_l \sum_{v\in\mathcal{V}_l}\sum_{n=1}^{N_v}[1-\mathrm{y}_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn}+b_v)]_+ \\ & + V_a C_l \sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v}[1-\mathrm{y}_{vn}(\mathbf{w}_v^T(\mathbf{x}_{vn}-\delta_{vn})+b_v)]_+ \\ & - C_a \sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v}\|\delta_{vn}\|_0\end{aligned}$$

s.t.

$$\begin{aligned}&\mathbf{w}_v = \mathbf{w}_u,\ b_v = b_u, \quad \forall v\in\mathcal{V},\ u\in\mathcal{B}_v; \\ &(\delta_{v1},\ldots,\delta_{vN_v})\in\mathcal{U}_v, \quad \forall v\in\mathcal{V}_a.\end{aligned} \tag{26}$$

Similarly, Problem (11) can be reformulated into the following problem:

$$\begin{aligned}\min_{\{\mathbf{w}_v,b_v\}}\max_{\{\delta_v\}}\ &\frac{1}{2}\sum_{v\in\mathcal{V}}\|\mathbf{w}_v\|^2\\&+V_lC_l\sum_{v\in\mathcal{V}_l}\sum_{n=1}^{N_v}[1-\mathrm{y}_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn}+b_v)]_+\\&+V_aC_l\sum_{v\in\mathcal{V}_a}\sum_{n=1}^{N_v}[1-\mathrm{y}_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn}+b_v)]_+\\&+\sum_{v\in\mathcal{V}_a}(V_aC_l\mathbf{w}_v^T\delta_v-C_a\|\delta_v\|_0)\\\text{s.t.}\ &\mathbf{w}_v=\mathbf{w}_u,\ b_v=b_u,\quad\forall v\in\mathcal{V},\ u\in\mathcal{B}_v;\\&\delta_v\in\mathcal{U}_{v0},\qquad\qquad\forall v\in\mathcal{V}_a.\end{aligned}\tag{27}$$

As a result, we only need to prove that problem (26) is equivalent to problem (27). Since both of problems are min-max problems with the same variables, we only need to prove that we minimize the same maximization problem. Moreover, since $\{\delta_{vn}\}$ is independent in the maximization part of (26), and δ_v is independent in the maximization part of (27), we can separate maximization problem into V_a sub-maximization problems, and solving the sub-problems is equivalent to solving the global maximization problem. As a result, we only need to show that the following sub-problem

$$\begin{aligned}\max_{\{\delta_{vn}\}\in\mathcal{U}_v}S(\{\delta_{vn}\})\triangleq&V_aC_l\sum_{n=1}^{N_v}[1-\mathrm{y}_{vn}(\mathbf{w}_v^T(\mathbf{x}_{vn}-\delta_{vn})+b_v)]_+\\&-C_a\sum_{n=1}^{N_v}\|\delta_{vn}\|_0\end{aligned}\tag{28}$$

is equivalent to the following sub-problem

$$\begin{aligned}\max_{\delta_v\in\mathcal{U}_{v0}}\ &V_aC_l\sum_{n=1}^{N_v}[1-\mathrm{y}_{vn}(\mathbf{w}_v^T\mathbf{x}_{vn}+b_v)]_+\\&+V_aC_l\mathbf{w}_v^T\delta_v-C_a\|\delta_v\|_0.\end{aligned}\tag{29}$$

We adopt the similar proof in [32], recall the properties of sublinear aggregated action set, $\mathcal{U}_v^-\subseteq\mathcal{U}_v\subseteq\mathcal{U}_v^+$, where

$$\mathcal{U}^-\triangleq\bigcup_{t=1}^{n}\mathcal{U}_t^-,\ \mathcal{U}_t^-\triangleq\left\{(\delta_1,\ldots,\delta_n)\left|\begin{array}{l}\delta_t\in\mathcal{U}_0;\\\delta_i=\mathbf{0},\ i\neq t.\end{array}\right.\right\};$$

$$\mathcal{U}^+\triangleq\left\{(\alpha_1\delta_1,\ldots,\alpha_n\delta_n)\left|\begin{array}{l}\sum_{i=1}^n\alpha_i=1;\ \alpha_i\geq0,\\\delta_i\in\mathcal{U}_0,\ i=1,\ldots,n\end{array}\right.\right\}.$$

Hence, fixing any $(\mathbf{w}_v,b_v)\in\mathbb{R}^{p+1}$, we have the following inequalities:

$$\max_{\{\delta_{vn}\}\in\mathcal{U}_v^-}S(\{\delta_{vn}\})\leq\max_{\{\delta_{vn}\}\in\mathcal{U}_v}S(\{\delta_{vn}\})\leq\max_{\{\delta_{vn}\}\in\mathcal{U}_v^+}S(\{\delta_{vn}\})\tag{30}$$

We can show that (29) is no larger than the leftmost term and no smaller than the rightmost term [34]. Thus, the equivalence between (28) and (29) holds. Hence, Lemma 2 holds.

APPENDIX B: PROOF OF LEMMA 3

We use best response dynamics to construct the best response for the min-problem and max-problem separately. The min-problem and max-problem are achieved by fixing $\{\mathbf{r}_v,\xi_v\}$ and $\{\delta_v\}$, respectively. For fixed $\{\mathbf{r}_v^*,\xi_v^*\}$,

$$\begin{aligned}\delta_v^*\in\arg\max_{\{\delta_v\}}&\sum_{v\in\mathcal{V}_a}(V_aC_l\mathbf{r}_v^{*T}(\mathbf{I}_{p+1}-\Pi_{p+1})\delta_v-C_a\|\delta_v\|_0)\\\text{s.t.}\quad&\delta_v\in\mathcal{U}_{v0},\ \forall v\in\mathcal{V}_a.\end{aligned}\tag{31}$$

We relax l_0 norm to l_1 norm to represent the cost function of the attacker. By writing the dual form of the l_1 norm, we arrive at

$$\begin{aligned}\delta_v^*\in\arg\max_{\{\delta_v,s_v\}}\ &V_aC_l\mathbf{r}_v^{*T}(\mathbf{I}_{p+1}-\Pi_{p+1})\delta_v-\mathbf{1}^Ts_v\\&C_a\delta_v\leq s_v,\\\text{s.t.}\quad&C_a\delta_v\geq-s_v,\\&\delta_v\in\mathcal{U}_{v0}.\end{aligned}\tag{32}$$

For fixed $\{\delta_v^*\}$, we have

$$\begin{aligned}\min_{\{\mathbf{r}_v,\omega_{vu},\boldsymbol{\xi}_v\}}\ &\frac{1}{2}\sum_{v\in\mathcal{V}}\mathbf{r}_v^T(\mathbf{I}_{p+1}-\Pi_{p+1})\mathbf{r}_v\\&+V_aC_l\sum_{v\in\mathcal{V}_a}\mathbf{r}_v^T(\mathbf{I}_{p+1}-\Pi_{p+1})\delta_v^*+VC_l\sum_{v\in\mathcal{V}}\mathbf{1}_v^T\boldsymbol{\xi}_v\\&\mathbf{Y}_v\mathbf{X}_v\mathbf{r}_v\geq\mathbf{1}_v-\boldsymbol{\xi}_v,\quad\forall v\in\mathcal{V};\\\text{s.t.}\ &\boldsymbol{\xi}_v\geq\mathbf{0}_v,\qquad\forall v\in\mathcal{V};\\&\mathbf{r}_v=\omega_{vu},\ \omega_{vu}=\mathbf{r}_u,\quad\forall v\in\mathcal{V},\ \forall u\in\mathcal{B}_u.\end{aligned}\tag{33}$$

Note that term $-C_a\|\delta_v^*\|_0$ is removed since it does not play a role in the minimization problem. Based on (32) and (33), we have the method of solving Problem (12) as follows, first step is that we randomly pick initial $\{\mathbf{r}_v^{(0)},\delta_v^{(0)}\}$, and then we solve Max-problem (32) with $\{\mathbf{r}_v^{(0)}\}$ to obtain $\{\delta_v^{(1)}\}$. In next step, we solve Min-problem (33) to obtain $\{\mathbf{r}_v^{(1)}\}$ with $\{\delta_v^{(1)}\}$ from the previous step. We repeat solving the max-problem with $\{\mathbf{r}_v^{(t-1)}\}$ and solving the min-problem with $\{\delta_v^{(t)}\}$ until convergence. Furthermore, we use the alternating direction method of multipliers (ADMoM) to solve Problem (33).

The ADMoM is a distributed optimization algorithm solving the following problem:

$$\begin{aligned}\min_{\mathbf{r},\omega}\ &f(\mathbf{r})+g(\omega)\\\text{s.t.}\quad&\mathbf{M}\mathbf{r}=\omega,\end{aligned}\tag{34}$$

where f and g are convex functions [12].

The augmented Lagrangian corresponding to (34) is

$$L(\mathbf{r},\omega,\alpha) = f(\mathbf{r}) + g(\omega) + \alpha^T(\mathbf{M}\mathbf{r} - \omega) + \frac{\eta}{2}\|\mathbf{M}\mathbf{r} - \omega\|^2, \tag{35}$$

where α denotes the Lagrange multiplier.

Then, the ADMoM solves problem (34) by the update rules below:

$$\mathbf{r}^{(t+1)} \in \arg\min_{\mathbf{r}} L(\mathbf{r},\omega^{(t)},\alpha^{(t)}); \tag{36}$$

$$\omega^{(t+1)} \in \arg\min_{\omega} L(\mathbf{r}^{(t+1)},\omega,\alpha^{(t)}); \tag{37}$$

$$\alpha^{(t+1)} = \alpha^{(t)} + \eta(\mathbf{M}\mathbf{r}^{(t+1)} - \omega^{(t+1)}). \tag{38}$$

The objective here is to transform Problem (33) into the form of (34), and then we can solve Problem (33) by iterations (36), (37), and (38). We adopt a similar method in [13], which leads to the following result.

REMARK 1 Each node iterates $\lambda_v^{(t)}, \mathbf{r}_v^{(t)}$ and $\alpha_v^{(t)}$, given by

$$\lambda_v^{(t+1)} \in \arg\max_{\mathbf{0}\le\boldsymbol{\lambda}_v\le VC_l\mathbf{1}_v} -\tfrac{1}{2}\lambda_v^T\mathbf{Y}_v\mathbf{X}_v\mathbf{U}_v^{-1}\mathbf{X}_v^T\mathbf{Y}_v\lambda_v + (\mathbf{1}_v + \mathbf{Y}_v\mathbf{X}_v\mathbf{U}_v^{-1}\mathbf{f}_v^{(t)})^T\lambda_v, \tag{39}$$

$$\mathbf{r}_v^{(t+1)} = \mathbf{U}_v^{-1}(\mathbf{X}_v^T\mathbf{Y}_v\lambda_v^{(t+1)} - \mathbf{f}_v^{(t)}), \tag{40}$$

$$\omega_{vu}^{(t+1)} = \tfrac{1}{2}(\mathbf{r}_v^{(t+1)} + \mathbf{r}_u^{(t+1)}), \tag{41}$$

$$\alpha_v^{(t+1)} = \alpha_v^{(t)} + \frac{\eta}{2}\sum_{u\in\mathcal{B}_v}[\mathbf{r}_v^{(t+1)} - \mathbf{r}_u^{(t+1)}], \tag{42}$$

where $\mathbf{U}_v = (\mathbf{I}_{p+1} - \Pi_{p+1}) + 2\eta|\mathcal{B}_v|\mathbf{I}_{p+1}$, $\mathbf{f}_v^{(t)} = V_aC_l\delta_v^* + 2\alpha_v^{(t)} - 2\eta\sum_{u\in\mathcal{B}_v}\omega_{vu}^{(t)}$.

By combining the above remark with Problem (32), we can obtain Lemma 3.

REFERENCES

[1] T. P. Banerjee and S. Das
Multi-sensor data fusion using support vector machine for motor fault detection
Information Sciences, 217 (Dec. 2012), pp. 96–107.

[2] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar
Can machine learning be secure?
in Proceedings of the 2006 ACM Symposium on Information, computer and communications security, ACM, Mar. 2006, pp. 16–25.

[3] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli
Evasion attacks against machine learning at test time
in Machine Learning and Knowledge Discovery in Databases, Springer, Sep. 2013, pp. 387–402.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein
Distributed optimization and statistical learning via the alternating direction method of multipliers
Foundations and Trends® in Machine Learning, 3 (July 2011), pp. 1–122.

[5] H. Chan and A. Perrig
Security and privacy in sensor networks
Computer, 36 (Oct. 2003), pp. 103–105.

[6] R. Chen, J.-M. Park, and K. Bian
Robust distributed spectrum sensing in cognitive radio networks
in INFOCOM 2008. The 27th Conference on Computer Communications. IEEE, IEEE, Apr. 2008.

[7] X. Chen, K. Makki, K. Yen, and N. Pissinou
Sensor network security: a survey
IEEE Communications Surveys & Tutorials, 11 (Apr. 2009).

[8] Z. Chen, L. Gao, and K. Kwiat
Modeling the spread of active worms
in INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, vol. 3, IEEE, Mar. 2003, pp. 1890–1900.

[9] M. Dalponte, L. Bruzzone, and D. Gianelle
Fusion of hyperspectral and lidar remote sensing data for classification of complex forest areas
IEEE Transactions on Geoscience and Remote Sensing, 46 (May 2008), pp. 1416–1427.

[10] J.-x. Dong, A. Krzyzak, and C. Y. Suen
Fast svm training algorithm with decomposition on very large data sets
IEEE transactions on pattern analysis and machine intelligence, 27 (Apr. 2005), pp. 603–618.

[11] H. F. Durrant-Whyte, B. Rao, and H. Hu
Toward a fully decentralized architecture for multi-sensor data fusion
in Robotics and Automation, 1990. Proceedings, 1990 IEEE International Conference on, IEEE, May 1990, pp. 1331–1336.

[12] J. Eckstein and W. Yao
Augmented lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results
RUTCOR Research Reports, 32 (Dec. 2012), p. 3.

[13] P. A. Forero, A. Cano, and G. B. Giannakis
Consensus-based distributed support vector machines
Journal of Machine Learning Research, 11 (2010), pp. 1663–1707.

[14] A. Frank and A. Asuncion
UCI machine learning repository [http://archive. ics. uci. edu/ml]. irvine, ca: University of california
School of Information and Computer Science, 213 (2010).

[15] B. He and X. Yuan
On non-ergodic convergence rate of douglas-rachford alternating direction method of multipliers
Numerische Mathematik, 130 (July 2012), pp. 567–577.

[16] J. Hert, P. Willett, D. J. Wilton, P. Acklin, K. Azzaoui, E. Jacoby, and A. Schuffenhauer
New methods for ligand-based virtual screening: use of data fusion and machine learning to enhance the effectiveness of similarity searching
Journal of chemical information and modeling, 46 (Mar. 2006), pp. 462–470.

[17] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. Tygar
Adversarial machine learning
in Proceedings of the 4th ACM workshop on Security and artificial intelligence, ACM, Oct. 2011, pp. 43–58.

[18] W. Jiang, Z.-h. Tian, H.-l. Zhang, and X.-f. Song
A stochastic game theoretic approach to attack prediction and optimal active defense strategy decision
in Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on, IEEE, Apr. 2008, pp. 648–653.

[19] M. Kantarcioglu, B. Xi, and C. Clifton
A game theoretical framework for adversarial learning
in CERIAS 9th Annual Information Security Symposium, Citeseer, 2008.

[20] K. K. Khedo, R. Perseedoss, A. Mungur, et al.
A wireless sensor network air pollution monitoring system
arXiv preprint arXiv:1005.1737, (May 2010).

[21] W. Liu and S. Chawla
A game theoretical model for adversarial learning
in Data Mining Workshops, 2009. ICDMW'09. IEEE International Conference on, IEEE, Dec. 2009, pp. 25–30.

[22] M. H. Manshaei, Q. Zhu, T. Alpcan, T. Bacşar, and J.-P. Hubaux
Game theory meets network security and privacy
ACM Computing Surveys (CSUR), 45 (June 2013), p. 25.

[23] A. Navia-Vázquez and E. Parrado-Hernandez
Distributed support vector machines
IEEE Transactions on Neural Networks, 17 (July 2006), pp. 1091–1097.

[24] J. B. Predd, S. R. Kulkarni, and H. V. Poor
Distributed learning in wireless sensor networks
John Wiley & Sons: Chichester, UK, Oct. 2007.

[25] C. E. Shannon
Communication theory of secrecy systems
Bell Labs Technical Journal, 28 (Oct. 1949), pp. 656–715.

[26] D. Shen, G. Chen, E. Blasch, and G. Tadda
Adaptive markov game theoretic data fusion approach for cyber network defense
in Military Communications Conference, 2007. MILCOM 2007. IEEE, IEEE, Oct. 2007, pp. 1–7.

[27] P. Tague and R. Poovendran
Modeling node capture attacks in wireless sensor networks
in Communication, Control, and Computing, 2008 46th Annual Allerton Conference on, IEEE, Sep. 2008, pp. 1221–1224.

[28] I. W. Tsang, J. T. Kwok, and P.-M. Cheung
Core vector machines: Fast svm training on very large data sets
Journal of Machine Learning Research, 6 (2005), pp. 363–392.

[29] D. Wang and Y. Zhou
Distributed support vector machines: An overview
in Control and Decision Conference (CCDC), 2012 24th Chinese, IEEE, May 2012, pp. 3897–3901.

[30] B. Waske and J. A. Benediktsson
Fusion of support vector machines for classification of multisensor data
IEEE Transactions on Geoscience and Remote Sensing, 45 (Dec. 2007), pp. 3858–3866.

[31] G. Wu, Y. Wu, L. Jiao, Y.-F. Wang, and E. Y. Chang
Multi-camera spatio-temporal fusion and biased sequence-data learning for security surveillance
in Proceedings of the eleventh ACM international conference on Multimedia, ACM, Nov. 2003, pp. 528–538.

[32] H. Xu, C. Caramanis, and S. Mannor
Robustness and regularization of support vector machines
Journal of Machine Learning Research, 10 (2009), pp. 1485–1510.

[33] R. Zhang and Q. Zhu
A game-theoretic defense against data poisoning attacks in distributed support vector machines
in Decision and Control (CDC), 2017 IEEE 56th Annual Conference on, IEEE, Dec. 2017, pp. 4582–4587.

[34] ———
Secure and resilient distributed machine learning under adversarial environments
in Information Fusion (Fusion), 2015 18th International Conference on, IEEE, July 2015, pp. 644–651.

[35] ———
Student research highlight: Secure and resilient distributed machine learning under adversarial environments
IEEE Aerospace and Electronic Systems Magazine, 31 (Mar. 2016), pp. 34–36.

[36] ———
A game-theoretic analysis of label flipping attacks on distributed support vector machines
in Information Sciences and Systems (CISS), 2017 51st Annual Conference on, IEEE, Mar. 2017, pp. 1–6.

[37] ———
A game-theoretic approach to design secure and resilient distributed support vector machines
IEEE Transactions on Neural Networks and Learning Systems, (Mar. 2018).

[38] R. Zhang, Q. Zhu, and Y. Hayel
A bi-level game approach to attack-aware cyber insurance of computer networks
IEEE Journal on Selected Areas in Communications, 35 (Mar. 2017), pp. 779–794.

[39] X.-H. Zhao, W. Gang, K.-K. Zhao, and D.-J. Tan
On-line least squares support vector machine algorithm in gas prediction
Mining Science and Technology (China), 19 (Mar. 2009), pp. 194–198.

[40] Q. Zhu and T. Basar
Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: games-in-games principle for optimal cross-layer resilient control systems
IEEE control systems, 35 (Feb. 2015), pp. 46–65.

[41] Q. Zhu and T. Başar
Game-theoretic approach to feedback-driven multi-stage moving target defense
in International Conference on Decision and Game Theory for Security, Springer, Nov. 2013, pp. 246–263.

[42] Q. Zhu, H. Tembine, and T. Başar
Heterogeneous learning in zero-sum stochastic games with incomplete information
in Decision and Control (CDC), 2010 49th IEEE Conference on, IEEE, Dec. 2010, pp. 219–224.

[43] ———
Distributed strategic learning with application to network security
in American Control Conference (ACC), 2011, IEEE, June 2011, pp. 4057–4062.

Rui Zhang received the B.S. degree in optical information science and technology from Wuhan University in 2014, and the M.S. degree in electrical engineering from New York University in 2016, where he is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering. He won the Runner Up Best Student Award at the International Conference on Information Fusion 2015. His research interests include cyber-insurance, network security, machine learning, optimal transport, and cyber-physical systems.

Quanyan Zhu received B. Eng. in Honors Electrical Engineering from McGill University in 2006, M.A.Sc. from University of Toronto in 2008, and Ph.D. from the University of Illinois at Urbana-Champaign (UIUC) in 2013. After stints at Princeton University, he is currently an assistant professor at the Department of Electrical and Computer Engineering, New York University. He is a recipient of many awards including NSERC Canada Graduate Scholarship (CGS), Mavis Future Faculty Fellowships, and NSERC Postdoctoral Fellowship (PDF). He spearheaded and chaired INFOCOM Workshop on Communications and Control on Smart Energy Systems (CCSES), and Midwest Workshop on Control and Game Theory (WCGT). His current research interests include Internet of things, cyber-physical systems, security and privacy, game theory, and system and control.

A belief combination rule for a large number of sources

KUANG ZHOU
ARNAUD MARTIN
QUAN PAN

The theory of belief functions is widely used for data from multiple sources. Different evidence combination rules have been proposed in this framework according to the properties of the sources to combine. However, most of these combination rules are not efficient when there are a large number of sources. This is due to either the complexity or the existence of an absorbing element such as the total conflict mass function for the conjunctive based rules when applied on unreliable evidence. In this paper, based on the assumption that the majority of sources are reliable, a combination rule for a large number of sources is proposed using a simple idea: the more common ideas the sources share, the more reliable these sources are supposed to be. This rule is adaptable for aggregating a large number of sources which may not all be reliable. It will keep the spirit of the conjunctive rule to reinforce the belief on the focal elements with which the sources are in agreement. The mass on the empty set will be kept as an indicator of the conflict.

The proposed rule, called LNS-CR **(Conjunctive combination Rule for a Large Number of Sources), is evaluated on synthetic mass functions. The experimental results verify that the rule can be effectively used to combine a large number of mass functions and to elicit the major opinion.**

Manuscript received September 17, 2016; revised March 2, 2017 and October 24, 2017; released for publication July 11, 2018.

Refereeing of this contribution was handled by Ramona Georgescu.

Authors' addresses: K. Zhou and Q. Pan, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, PR China. A. Martin, DRUID, IRISA, University of Rennes 1, Rue E. Branly, 22300 Lannion, France.

This paper is an extension and revision of [1].

I. INTRODUCTION

In recent years, Dempster-Shafer Theory (DST), also called the theory of belief functions, has gained increasing attention in the scientific community as it allows to the deal with the imprecise and uncertain information. It has been applied in various domains, such as data classification [2, 3], data clustering [4, 5], social network analysis [6], etc. In complex environment, multiple stake-holders attempt to reach a decision by combining several sources of information and aggregating their points of view by stressing common agreement. The theory of belief functions, which has provided many rules to combine information represented by mass functions [7], are widely used for decision making. In real applications, there are usually a large number of sources. Most of the existing combination rules are not applicable in this case, and cannot be used to find the major opinion from many participants.

One of the most famous combination rule in belief function framework is the Dempster's rule [7]. Smets [8] proposed a modification of Dempster's rule, often called "conjunctive rule," where the empty set can be assigned with a non-null mass under the Transferable Belief Model (TBM) [9]. In fact, the conjunctive rule is equivalent to the Dempster rule without the normalization process. It has a fast and clear convergence towards a solution. But this rule has a strong assumption that all the sources are reliable. In real applications, it is difficult to be either satisfied or verified. Moreover, the more sources there are, the more chance that there is some unreliable evidence.

Smets [8] reasoned that the mass on the empty set can play the role of alarm. When the global conflict (the mass assigned to the empty set) is high, it indicates that there is strong disagreement among the sources of mass functions to combine. However, as observed in [10, 11, 12], the mass on the empty set is not sufficient to exactly describe the conflict since it includes an amount of auto-conflict [13]. Sometimes when there is only a small amount of concordant evidence, the total conflict mass function, i.e. $m(\emptyset) = 1$ will be an absorbing element. Consequently, when combining a large number of (incompatible) mass functions using the conjunctive rule, the global conflict may tend to 1. This makes it impossible to reveal the cause of high global conflict. We do not know whether it is due to the sources to fuse or caused by the absorption power of the empty set [10, 14]. In other words, even the combined mass function by the conjunctive rule is $m(\emptyset) \approx 1$, the proposition that the sources are highly conflicting may be incorrect.

In order to rectify the drawbacks of the classical Dempster's rule and Smets' conjunctive rule, many approaches have been made through the modification of the combination rule. Some authors tried to find alternative repartitions of the conflict. A plethora of combination rules have been brought forward in this way.

For example, Yager [15] and Dubois and Prade [16] suggested assigning the highly conflicting mass to the whole set or a particular set. The Proportional Conflict Redistribution (PCR) rule, which can distribute the partial conflicts among the involved focal elements rather than to their union, is developed in [13, 17]. Apart from these approaches working directly on the combination rule, some studies manage the conflict through evidence discounting, where the reliability of sources is automatically and adaptively taken into account [10, 16, 18, 19].

Most of the existing combination rules are not efficient when applied on a large number of sources due to the ineffective way to handle conflict or the high complexity of the computation. Orponen [20] proved that the complexity of the conjunctive rule is NP-hard, but the complexity depends on the way to program the belief functions [21]. Some rules can manage efficiently the conflict but have large complexity [13, 16, 22, 23], making them infeasible when applied to combine a large number of mass functions.

In this paper, a conjunctive-based combination rule, named LNS-CR (Large Number of Sources), is proposed to aggregate a large number of mass functions. Our perspective on belief function combination is that combining mass functions from different sources is similar to combining opinions from multiple stake-holders in group decision-making [24], i.e. the more one's opinion is consistent with the other experts, the more reliable the source is. We assume that all the mass functions available are separable mass functions, which means they can be expressed by a group of simple support mass functions. In many applications, the mass assignments are directly in the form of Simple Support Functions (SSF) [25]. The advantage of SSFs is that we can group the mass functions in such a way that sources in the same group share the same viewpoint. Mass functions in each small group are first fused and then discounted according to the proportions. After that the number of mass functions participating the next global combination process is independent of the number of sources, but only depends on the number of classes. As a result, the problem brought by the absorbing element (the empty set) using the conjunctive rule can be avoided. Moreover, an approximation method when the number of mass functions is large enough is presented. The main contributions of this paper are as follows:

- A new conjunctive-based combination rule, named LNS-CR rule, is brought froward. The property to reinforce the belief on the focal elements with which most of the sources agree is preserved in the proposed rule;
- The assumption of the LNS-CR rule on the reliability of the sources is more relaxed, as it does not require all the sources are reliable, but only at least half of them are reliable.
- LNS-CR can be used to combine mass functions from a large number of sources, especially can be used to elicit the major opinion;
- Derivation that the LNS-CR rule is within acceptable complexity.

The rest of this paper is organized as follows. In Section 2, some basic knowledge of belief function theory is briefly introduced. The proposed evidence combination approach is presented in detail in Section 3. Numerical examples are employed to compare different combination rules and show the effectiveness of LNS-CR rule in Section 4. Finally, Section 5 concludes the paper.

II. BACKGROUND

A. Basic knowledge of belief function theory

Let $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$ be the discernment frame. A mass function is defined on the power set $2^\Theta = \{A : A \subseteq \Theta\}$. The mass function $m : 2^\Theta \to [0,1]$ is said to be a Basic Belief Assignment (bba) on 2^Θ, if it satisfies:

$$\sum_{A \subseteq \Theta} m(A) = 1. \tag{1}$$

Every $A \in 2^\Theta$ such that $m(A) > 0$ is called a focal element, and the set of focal elements is denoted by $\mathcal{F}$. In a practical way of programming, the element of 2^Θ can be arranged by natural order [26]: $\theta_1, \theta_2, \{\theta_1, \theta_2\}, \theta_3, \ldots, \{\theta_1, \theta_2, \theta_3\}, \theta_4, \ldots, \Theta$.

The frame of discernment can also be a focal element. If Θ is a focal element, the mass function is called non-dogmatic. The mass assigned to the frame of discernment, $m(\Theta)$, is interpreted as a degree of ignorance. In the case of total ignorance, $m(\Theta) = 1$. This type of mass assignment is vacuous. If there is only one focal element, i.e. $m(A) = 1$, $A \subset \Theta$, the mass function is categorical. Another special case of assignment is named consonant mass functions, where the focal elements include each other as a subset, i.e. if $A, B \in \mathcal{F}$, $A \subset B$ or $B \subset A$.

The credibility and plausibility functions are derived from a bba m as in Eqs. (2) and (3):

$$Bel(A) = \sum_{B \subseteq A, B \neq \emptyset} m(B), \quad \forall A \subseteq \Theta, \tag{2}$$

$$Pl(A) = \sum_{B \cap A \neq \emptyset} m(B), \quad \forall A \subseteq \Theta. \tag{3}$$

Each quantity $Bel(A)$ measures the minimal belief on A justified by available information on $B (B \subseteq A)$, while $Pl(A)$ is the maximal belief on A justified by information on B which are not contradictory with A ($A \cap B \neq \emptyset$). The commonality function q and the implicability function b are defined respectively as

$$q(A) = \sum_{A \subseteq B} m(B), \quad \forall A \subseteq \Theta \tag{4}$$

and

$$b(A) = Bel(A) + m(\emptyset), \quad \forall A \subseteq \Theta. \tag{5}$$

A bba m can be recovered from any of these functions. For instance,

$$m(A) = \sum_{B \supseteq A} (-1)^{|B|-|A|} q(B), \quad \forall A \subseteq \Theta \tag{6}$$

and

$$m(A) = \sum_{B \subseteq A} (-1)^{|A|-|B|} b(B), \quad \forall A \subseteq \Theta. \tag{7}$$

Belief functions can be transformed into a probability function by Smets' method [27], where each mass of belief $m(A)$ is equally distributed among the elements of A. This leads to the concept of pignistic probability, BetP. For all $\theta_i \in \Theta$, we have

$$\text{BetP}(\theta_i) = \sum_{A \subseteq \Theta | \theta_i \in A} \frac{m(A)}{|A|(1 - m(\emptyset))}, \tag{8}$$

where $|A|$ is the cardinality of set A (number of elements of Θ in A). Pignistic probabilities can help make a decision.

B. Consistency of mass assignments

The consistency between two bbas can be defined in two different ways. Suppose the sets of focal elements for m_1 and m_2 are $\mathcal{F}_1$ and $\mathcal{F}_2$ respectively. Mass functions m_1 and m_2 are called strong consistent if and only if

$$\cap_{E \in \{\mathcal{F}_1 \cup \mathcal{F}_2\}} \neq \emptyset. \tag{9}$$

Meanwhile, bbas m_1 and m_2 are called weak consistent if and only if

$$\forall A \in \mathcal{F}_1,\ B \in \mathcal{F}_2,\ A \cap B \neq \emptyset. \tag{10}$$

Strong consistent evidence means that there is at least one element that is common to all subsets [28]. It is easy to see that, when m_1 and m_2 are strong consistent, they are sure to be weak consistent. This is the definition of consistency between belief functions. The inconsistency within an individual mass assignment can be defined similarly [12].

C. Reliability-based discounting

When the sources of evidence are not completely reliable, the discounting operation proposed by Shafer [25] and justified by Smets [29] could be applied. Denote the reliability degree of mass function m by $\alpha \in [0,1]$, then the discounting operation can be defined as:

$$m'(A) = \begin{cases} \alpha \times m(A) & \forall A \subset \Theta, \\ 1 - \alpha + \alpha \times m(\Theta) & \text{if } A = \Theta. \end{cases} \tag{11}$$

If $\alpha = 1$, the evidence is completely reliable and the bba will remain unchanged. On the contrary, if $\alpha = 0$, the evidence is completely unreliable. In this case the so-called vacuous belief function, $m(\Theta) = 1$, could be got. It describes the total ignorance.

Before evoking the discounting process, the reliability of each sources should be known. One possible way to estimate the reliability is to use confusion matrices [30]. Generally, the goal of discounting is to reduce global conflict before combination. One can assume that the conflict comes from the unreliability of the sources. Therefore, the source reliability estimation is to some extent linked to the estimation of conflict between sources.

Hence, Martin et al. [10] proposed to use a conflict measure to evaluate the relative reliability of experts. Once the degree of conflict is computed, the relative reliability of the source can be computed accordingly. Suppose there are S sources, $\mathcal{S} = \{s_1, s_2, \ldots, s_S\}$, the reliability discounting factor α_j of source s_j can be defined as follows:

$$\alpha_j = f(\text{Conf}(s_j, \mathcal{S})), \tag{12}$$

where $\text{Conf}(s_j, \mathcal{S})$ quantifies the degree that source s_j conflicts with the other sources in $\mathcal{S}$, and f is a decreasing function. The following function is suggested by the authors:

$$\alpha_j = (1 - \text{Conf}(s_j, \mathcal{S})^{\lambda})^{1/\lambda}, \tag{13}$$

where $\lambda > 0$.

In [31], the authors considered to use those two possible conflict origins, extrinsic measure and intrinsic measure, to estimate reliability. In their opinion, conflict may not only come from the source's contradiction (extrinsic measure), but also from the confusion rate of a source (intrinsic measure). The reliability discounting factor, called Generic Discounting Factor (GDF), is then suggested to be a weighted sum of the two items:

$$\alpha = \frac{k\delta + l\beta}{k + l}, \tag{14}$$

where $k > 0$, $l > 0$ are the weight factors. In the above equation, δ denotes the internal conflict measure of the treated source indicating its confusion rate while β is the average distance between the treated sources s_i and s_j where $j \in \mathcal{S},\ j \neq i$. Different intrinsic and extrinsic conflict measures can be adopted here.

There are some other methods to estimate the reliability. In [32], the authors proposed to estimate the reliability of sources based on a degree of falsity. The bbas are sequentially and incrementally discounted until the mass assigned to the empty set is smaller than a given threshold k. After that the discounted mass functions can be combined using the conjunctive rule since there is little global conflict at this time. In [33], the source reliability is obtained by minimizing the distance between the pignistic probabilities computed from the discounted beliefs and the actual value of the data. In Samet et al. [34], the authors proposed two different versions of generic discounting approaches: weighted GDA and exponent GDA. A new degree of disagreement is proposed by Yang et al. [35], where the reliability discounting factor can be generated. Klein and Colot [36] viewed the

degree of conflict as a function of discounting rates and introduced a new criterion assessing bbas' reliability. These reliability estimation methods either consider the distance (or dissimilarity) between each pair of bbas, or the mass assigned to the empty set after the conjunctive combination. However, these methods are of high complexity and not suitable for large data applications.

D. Simple support function

Suppose m is a bba defined on the frame of discernment Θ. If there exists a subset $A \subseteq \Theta$ such that m could be expressed in the following form:

$$m(X) = \begin{cases} w & X = \Theta, \\ 1-w & X = A, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

where $w \in [0,1]$, then the belief function related to bba m is called a Simple Support Function (SSF) (also called simple mass function) [25] focused on A. Such a SSF can be denoted by $A^w(\cdot)$ where the exponent w of the focal element A is the basic belief mass (bbm) given to the frame of discernment Θ, $m(\Theta)$. The complement of w to 1, i.e. $1-w$, is the bbm allocated to A [37]. If $w = 1$ the mass function represents the total ignorance, if $w = 0$ the mass function is a categorical bba on A.

A belief function is separable if it is a SSF or if it is the conjunctive combination of some SSFs [38]. In the work of [38], this kind of separable masses is called u-separable where "u" stands for "unnormalized," indicating the conjunctive rule is the unnormalized version of Dempster-Shafer rule. The set of separable mass functions is not obvious to obtain. It is easy to see consonant mass functions (the focal element are nested) are separable [39]. Smets [37] defined the Generalized Simple Support Function (GSSF) by relaxing the weight w to $[0,\infty)$. Those GSSFs with $w \in (1,\infty)$ are called Inverse Simple Support Functions (ISSF). Smets proved all non-dogmatic mass functions are separable if one uses GSSFs. For any non-dogmatic belief function m_0, the canonical decomposition method proposed by Smets is as follows. First, calculate the commonality number for all focal elements, which is given by

$$Q_0(X) = \sum_{B \supseteq X} m_0(B). \tag{16}$$

Secondly for any $A \subseteq \Theta$, calculate w_A value as follows:

$$w_A = \prod_{X \supseteq A} Q_0(X)^{(-1)^{|X|-|A|+1}}. \tag{17}$$

Then the belief function m_0 can be represented by the conjunctive combination of all the functions A_{w_A}, i.e.

$$m_0 = \bigcirc\!\!\!\!\!\!\cap_{A \subseteq \Theta} A^{w_A}, \tag{18}$$

where $\cap$ denotes the conjunctive combination rule. For fast computation, the Fast Möbius Transform (FMT) method [40] can be evoked.

E. Some combination rules

How to combine efficiently several bbas coming from distinct sources is a major information fusion problem in the belief function framework. Many rules have been proposed for such a task. Here we just briefly recall how some most popular rules are mathematically defined.

When information sources are reliable, the used fusion operators can be based on the conjunctive combination. If bbas m_j, $j = 1,2,\ldots,S$ describing S distinct items of evidence on Θ, the included result of the **conjunctive rule** [9] is defined as

$$m_{\text{conj}}(X) = \left(\bigcirc\!\!\!\!\!\!\cap_{j=1,\ldots,S} m_j \right)(X) = \sum_{Y_1 \cap \cdots \cap Y_S = X} \prod_{j=1}^{S} m_j(Y_j), \tag{19}$$

where $m_j(Y_j)$ is the mass allocated to Y_j by expert j. To apply this rule, the sources are assumed reliable and cognitively independent.

Another kind of conjunctive combination is **Dempster's rule** [41]. Assuming that $m_{\text{conj}}(\emptyset) \neq 1$, the result of the combination by Dempster's rule is

$$m_{\text{Dempster}}(X) = \begin{cases} 0 & \text{if } X = \emptyset, \\ \dfrac{m_{\text{conj}}(X)}{1 - m_{\text{conj}}(\emptyset)} & \text{otherwise.} \end{cases} \tag{20}$$

The item

$$\kappa \triangleq m_{\text{conj}}(\emptyset) = \sum_{Y_1 \cap \cdots \cap Y_S = \emptyset} \prod_{j=1}^{S} m_j(Y_j)$$

is generally called Dempster's degree of conflict of the combination or the inconsistency of the combination. As the conjunctive rule is not idempotent, $m_{\text{conj}}(\emptyset)$ includes an amount of auto-conflict [42], and it is called global conflict to make the difference.

The conjunctive rule can be applied only if all the experts are reliable. In the other case, the **disjunctive rule** [43], which only assumes that at least one of the sources is reliable, can be used. The disjunctive combination of S sources can be defined as

$$m_{\text{disj}}(X) = \left(\bigcirc\!\!\!\!\!\!\cup_{j=1,\ldots,S} m_j \right)(X) = \sum_{Y_1 \cup \cdots \cup Y_S = X} \prod_{j=1}^{S} m_j(Y_j). \tag{21}$$

The conjunctive and disjunctive rules can be conveniently expressed by means of the commonality function q (Eq. (4)) and the implacability function b (Eq. (5)) [43]. Let q_i and b_i be the commonality function and implacability function respectively (associated with m_i), then the commonality function of the conjunctive combination of S bbas is

$$q_{\text{conj}}(A) = \prod_{i=1}^{S} q_i(A), \quad \forall A \subseteq \Theta \tag{22}$$

while the implacability function of the disjunctive combination of S bbas is

$$b_{\text{disj}}(A) = \prod_{i=1}^{S} b_i(A), \quad \forall A \subseteq \Theta. \tag{23}$$

Since functions m, q and b (as well as bel and pl) are equivalent representations, the mass function m can be recovered using the Fast Möbius Transform (FMT) method given the functions q and b. The conversion can be done in time proportional to $n2^n$ [44].[1] For the conjunctive combination of S sources, the S bbas should be converted into commonality functions first. After calculating the product of S commonality functions, another transformation from m to q should be evoked. Overall the total complexity is $O(Sn2^n + S2^n + n2^n)$, and the time needed is proportional to $Sn2^n$ [44, 45].

The conflict could be redistributed on partial ignorance like in the Dubois and Prade rule (**DP rule**) [16], which can be seen as a mixed conjunctive and disjunctive rule. For all $X \subseteq \Theta$, $X \neq \emptyset$:

$$m_{\text{DP}}(X) = \sum_{Y_1 \cap \cdots \cap Y_S = X} \prod_{j=1}^{S} m_j(Y_j) + \sum_{\substack{Y_1 \cup \cdots \cup Y_S = X \\ Y_1 \cap \cdots \cap Y_S = \emptyset}} \prod_{j=1}^{S} m_j(Y_j), \tag{24}$$

where m_j is the mass function delivered by expert j. In a general case, this rule cannot be programmed with the Fast Möbius Transform method because all the partial conflict must be considered. If the implementation is made like that in Ref. [46], it takes much more time than the conjunctive rule.

Denœux [38] proposed a family of conjunctive and disjunctive rules using triangular norms. The **cautious rule** [47, 48] belongs to that family and could be used to combine mass functions for which independence assumption is not verified. Cautious combination of S non-dogmatic mass functions m_j, $j = 1, 2, \ldots, S$ is defined by the bba with the following weight function:

$$w(A) = \bigwedge_{j=1}^{S} w_j(A), \quad A \in 2^\Theta \setminus \Theta. \tag{25}$$

We thus have

$$m_{\text{Cautious}}(X) = \bigcirc_{A \not\subseteq \Theta} A^{\bigwedge_{j=1}^{S} w_j(A)}, \tag{26}$$

where $A^{w_j(A)}$ is the simple support function focused on A with weight function $w_j(A)$ issued from the canonical decomposition of m_j. Note also that $\bigwedge$ is the min operator. The time consumption of the cautious rule includes the canonical decomposition of non-dogmatic mass functions and is therefore bigger than the conjunctive rule. If this rule is implemented in Fast Möbius Transform method, the complexity is proportional to $Sn2^n$.

Murphy [49] presented the **average combination rule** and proposed to utilize the mean of the basic belief assignments as the fusion of evidence. Therefore, for each focal element $X \in 2^\Theta$ of S mass functions, the combined one is defined as follows:

$$m_{\text{Ave}}(X) = \frac{1}{S} \sum_{j=1}^{S} m_j(X), \quad \forall X \subseteq \Theta. \tag{27}$$

The complexity of the average is proportional to $S2^n$.

A family of fusion rules based on new Proportional Conflict Redistributions (PCR) for the combination of uncertainty and conflicting information have been developed in Dezert-Smarandache Theory (DSmT) framework [50]. Among them, the fusion rule called PCR6 proposed by Martin and Osswald [13] is one of the most popular one among the PCR rules. For the combination of $S > 2$ sources, the fused mass is given by $m_{\text{PCR6}}(\emptyset) = 0$, and for $X \neq \emptyset$ in 2^Θ

$$m_{\text{PCR6}}(X) = m_{\text{conj}}(X) + \sum_{i=1}^{S} \left\{ (m_i(X))^2 \sum_{\substack{\bigcap_{k=1}^{S-1} Y_{\sigma_i(k)} \cap X \equiv \emptyset \\ (Y_{\sigma_i(1)}, \ldots, Y_{\sigma_i(S-1)}) \in (2^\Theta)^{S-1}}} \times \left(\frac{\prod_{j=1}^{S-1} m_{\sigma_i(j)}(Y_{\sigma_i(j)})}{m_i(X) + \sum_{j=1}^{S-1} m_{\sigma_i(j)}(Y_{\sigma_i(j)})} \right) \right\}, \tag{28}$$

where σ_i counts from 1 to S avoiding i:

$$\begin{cases} \sigma_i(j) = j & \text{if } j < i, \\ \sigma_i(j) = j + 1 & \text{if } j \geq i. \end{cases} \tag{29}$$

As Y_i is a focal element of expert/source i, we have $m(Y_i) > 0$. Then

$$m_i(X) + \sum_{j=1}^{S-1} m_{\sigma_i(j)}(Y_{\sigma_i(j)}) \neq 0.$$

In Eq. (28), m_{conj} is the conjunctive rule given by Eq. (19). Here again, the Fast Möbius Transform method to program the belief functions is not generally the best way. If the implementation is made like that in Ref. [46], the time consumption is very high.

III. A COMBINATION RULE FOR A LARGE NUMBER OF MASS FUNCTIONS

The main idea of the conjunctive combination rule is to reinforce the belief on the focal elements with which

[1] This is based on the assumption that the mass functions are arranged in natural order. If not, the complexity is proportional to $n^2 2^n$. The complexity analysis in this work all assumes that the bbas to be combined are encoded using the natural order.

most of the sources agree. Martin et al. [10] showed that the mass on the empty set, which is an absorbing element, tends quickly to 1 with the number of sources when combining inconsistent bbas. Consequently, when using Dempster rule (Eq. (20)), the gap between κ and 1 may rapidly exceed machine precision, even if the combination is valid theoretically. In that case the fused bba by the conjunctive rules (normalized or not) and the pignistic probability are inefficient. Moreover, the assumption that all the sources are reliable for the conjunctive combination rule is difficult to reach in real applications. The more sources there are, the less chance that this assumption is valid.

The principle of the conjunctive rule with the reinforcement of belief and the role of the empty set as an alarm are essential in the theory of belief functions. In order to propose a rule which can be adapted to the combination of a large number of mass functions and keep the previous behavior, the following assumptions are made:

- The majority of sources are reliable;
- The larger extent one source is consistent with others, the more reliable the source is;
- The sources are cognitively independent [43].

These assumptions seem reasonable if we consider combing mass functions as some kind of group decision making problems. As a result, the proposed rule will give more importance to the groups of mass functions that are in a domain, and it is without auto-conflict [13, 14]. In order to take into account this effect, this rule will discount the mass functions according to the number of sources giving bbas with the same focal elements. The discounting factor is directly given by the proportion of mass functions with the same focal elements. This procedure is for the elicitation of the majority opinion.

The simple support mass functions are considered here. In this case, the mass functions can be grouped in the light of their focal elements (except the frame Θ). To make the rule applicable on separable mass functions, the decomposition process should be performed to decompose each bba into simple support mass functions. In most of applications, the basic belief can be defined using separable mass functions, such as simple support functions [2] and consonant mass functions [51, 52].

Hereafter we describe the proposed LNS-CR rule for simple support functions, and then an approximation calculation method of LNS-CR rule is suggested.

A. LNS-CR rule for simple support functions

Suppose that each evidence is represented by a SSF. Then all the bbas can be divided into at most 2^n groups (where $n = |\Theta|$). It is easy to see that there is no conflict at all in each group because of consistency. The focal elements of the SSF are singletons and Θ itself. For the combination of bbas inside each group, the conjunctive rule can be employed directly. Then the fused bbas are discounted according to the number of mass functions in each group. Finally, the global combination of the bbas of different groups is preformed also using the conjunctive rule. Suppose that all bbas are defined on the frame of discernment $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$, and denoted by $m_j = (A_i)^{w_j}$, $j = 1, \ldots, S$ and $i = 1, 2, \ldots, c$, where $c \leq 2^n$. The detailed process of the combination is listed as follows. Our proposed rule called LNS-CR for Large Number of Sources rule is composed of the four following steps:

1) Cluster the simple bbas into c groups based on their focal element A_i. For the convenience, each class is labeled by its corresponding focal element.
2) Combine the bbas in the same group. Denote the combined bba in group A_k by SSF

$$\hat{m}_k = (A_k)^{\hat{w}_k}, \quad k = 1, 2, \ldots, c.$$

Let the number of bbas in group A_k is s_k. If the conjunctive rule is adopted, we have

$$\hat{m}_k = \bigcirc\!\!\!\!\!\!\cap_{j=1,\ldots,s_k} m_j = (A_k)^{\prod_{j=1}^{s_k} w_j}. \tag{30}$$

3) Reliability-based discounting. Suppose the fused bba of all the mass functions in A_k is $\hat{m}_k$. At this time, each group can be regarded as a source, and there are c sources in total. The reliability of one source can be estimated as compared to a group of sources. In our opinion, the reliability of source A_k is related to the proportion of bbas in this group. The larger the number of bbas in group A_k is, the more reliable A_k is. Then the reliability discounting factor of $\hat{m}_k$ can be defined as:

$$\alpha_k = \frac{s_k}{\sum_{i=1}^{c} s_i}. \tag{31}$$

In order to keep the mass function representing total ignorance as a neutral element of the rule, in Eq. (31) we let $a_k = 0$ for the group with $A_k = \Theta$. Another version of the discounting can be given by a factor taking into account the precision of the group by:

$$\alpha_k = \frac{\beta_k^{\eta} s_k}{\sum_{i=1}^{c} \beta_i^{\eta} s_i}, \tag{32}$$

where

$$\beta_k = \frac{|\Theta|}{|A_k|}. \tag{33}$$

Parameter η can be used to adjust the precision of the combination results. The larger the value of η is, the less imprecise the resulting bba is. The discounted bba of $\hat{m}_k$ can be denoted by SSF $\hat{m}'_k = (A_k)^{\hat{w}'_k}$ with $\hat{w}'_k = 1 - \alpha_k + \alpha_k \hat{w}_k$. As we can see, when the number of bbas in one group is larger, α is closer to 1. That is to say, the fused mass in this group is more reliable.

4) Global combine the fused bbas in different groups using the conjunctive rule:

$$m_{\text{LNS-CR}} = \bigcirc\!\!\!\!\!\cap_{k=1,\ldots,c} \hat{m}'_k = \bigcirc\!\!\!\!\!\cap_{k=1,\ldots,c} (A_k)^{\hat{w}'_k}. \quad (34)$$

REMARKS

- The reliability estimation method proposed here is very simple compared with the previous mentioned methods in Section II-C, where usually the distance between bbas should be calculated or a special learning process is required. In the LNS-CR rule, to evaluate the reliability discounting factor, we only need to count the number of SSFs in each group. Note that other reliability estimation methods can also be used here.
- In the last step of combination, as the number of mass functions that take part in the global combination is small (at most 2^n), other combination rules such as DP rule and PCR rules are also possible in practice instead of Eq. (34).

B. LNSa-CR rule for the approximated combination

If there is a large number of mass functions in each group, an approximation method is suggested here to calculate the combined mass in the given group. Suppose the mass functions in group with focal element A_k $(k = 1,2,\ldots,c)$ are:

$$m_j(A) = \begin{cases} 1 - w_j & A = A_k, \\ w_j & A = \Theta, \\ 0 & \text{otherwise}, \end{cases}$$
$$0 \le w_j < 1,\ j = 1,2,\ldots,s_k. \quad (35)$$

The combination of the masses in this group using the conjunctive rule is

$$\hat{m}_k(A) = \begin{cases} 1 - \prod_{j=1}^{s_k} w_j & A = A_k, \\ \prod_{j=1}^{s_k} w_j & A = \Theta, \\ 0 & \text{otherwise}. \end{cases} \quad (36)$$

It is easy to get

$$\lim_{s_k \to \infty} \hat{m}_k(A) = \begin{cases} 1 & A = A_k, \\ 0 & A = \Theta, \\ 0 & \text{otherwise}. \end{cases} \quad (37)$$

This is an illustration of the conjunctive property. After the discounting with factor α_k, the fused bba using for the global combination is

$$\lim_{n_k \to \infty} \hat{m}'_k(A) = \begin{cases} \alpha_k & A = A_k, \\ 1 - \alpha_k & A = \Theta, \\ 0 & \text{otherwise}. \end{cases} \quad (38)$$

It can be represented by SSF

$$\hat{m}'_k = (A_k)^{1-\alpha_k}, \quad (39)$$

where α_k is shown in Eq. (31) or (32). If the conjunctive rule is adopted for the global combination at step 4, the final bba we get is

$$m_{\text{LNSa-CR}} = \bigcirc\!\!\!\!\!\cap\ (A_k)^{1-\alpha_k}. \quad (40)$$

In this approximate rule for the large number of sources, the initial mass functions is no longer considered, and the combination process of the bbas inside each group is not required any more. This can accelerate the algorithm to a large extent. The LNS-CR and LNSa-CR rule provide different results when the number of sources is small. However, when the number of sources is large enough, they can be regarded as equivalent.

C. Properties

The proposed rule is commutative, but not associative. The rule is not idempotent, but there is no absorbing element. The vacuous mass function is a neutral element of the LNS-CR rule.

There are four steps when applying LNS-CR rule[2]: decomposition (not necessary for simple support mass functions), inner-group combination, discounting and global combination. The LNS-CR rule has the same memory complexity as some other rules such as conjunctive, Dempster and cautious rules if all the rules are combined globally using FMT method. Only DP and PCR6 rules have higher memory complexity because of the partial conflict to manage. Suppose the number of mass functions to combine is S, and the number of elements in the frame of discernment is n. The complexity for decomposing[3] mass functions to SSFs is $O(Sn2^n)$. For combining the mass functions in each group, due to the structure of the simple support mass functions, we only need to calculate the product of the masses on only one focal element Θ. Thus the complexity is $O(S)$. The complexity of the discounting is $O(2^n)$. In the process of global combination, the bbas are all SSFs. If we use the Fast Möbius Transform method, the complexity is $O(n2^n)$. And there are at most 2^n mass functions participating the following discounting and global conjunctive combination processes. Since in most application cases with a large number of mass functions, we have $2^n \ll S$, the last two steps are not very time-consuming. The total complexity of LNS-CR is $O(Sn2^n + S + 2^n + n2^n)$ and so is approximately equivalent to $O(Sn2^n)$.

For the approximate method, we can also save the time for inner combination and the discounting. The fused mass in each group is calculated by the proportions, and the complexity is also $O(S)$. Although the approximate method does not reduce the complexity,

[2]The source code for LNS-CR rule can be found in R package *ibelief* [53].

[3]In the decomposing process, the Fast Möbius Transform method is used.

TABLE I
The combination of six masses. For the names of columns, θ_{ij} is used to denote $\{\theta_i,\theta_j\}$.

	Conjunctive	Dempster	Disjunctive	DP	PCR6	Cautious	Average	LNS-CR
Ø	0.49313	0.00000	0.00000	0.00000	0.00000	0.15200	0.00000	0.06849
$\{\theta_1\}$	0.02595	0.05120	0.00000	0.02595	0.04783	0.00800	0.11333	0.36408
$\{\theta_2\}$	0.45687	0.90136	0.00000	0.45687	0.56639	0.79800	0.15833	0.08984
$\{\theta_1,\theta_2\}$	0.00000	0.00000	0.00004	0.49313	0.00000	0.00000	0.00000	0.00000
$\{\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_1,\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_2,\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
Θ	0.02405	0.04744	0.99996	0.02405	0.38578	0.04200	0.72833	0.47759

in the experimental part, we will show that it will save some running time in applications when S is quite large.

We remark here that one of the assumptions of LNS-CR rule is that the majority of sources are reliable. However, this condition is not always satisfied in every applicative context. Consider here an example with two sensor technologies: TA and TB. The system has two TA-sensors (S_1 and S_2), and one TB-sensor S_3. Suppose also a parasite signal causes TA sensors to malfunction. In this situation, the majority of sensors are unreliable. And we could not get a good result if the LNS-CR rule is used directly as LNS-CR(S_1,S_2,S_3) at this time. Actually there is an underlying hierarchy in the sources of information, LNS-CR rule could be evoked according to the hierarchy, such as LNS-CR(LNS-CR(S_1,S_2),S_3). We will study that more in the future work.

IV. EXPERIMENTS

In this section, several experiments will be conducted to illustrate the behavior of the proposed combination rule LNS-CR and to compare with other classical rules. Some different types of randomly generated mass functions will be used. The function *RandomMass* in R package *ibelief* [53] is adopted to generate random mass functions [54].

EXPERIMENT 1 (Elicitation of the majority opinion). In some applications, the elicitation of the majority opinion is very important. In this experiment, it is assumed that reliable sources can provide some imprecise and uncertain information, which is assumed to be in the form of the mass functions m_j ($j = 1,2,\ldots,6$) over the same discernment frame $\Theta = \{\theta_1,\theta_2,\theta_3\}$:

$$
\begin{aligned}
&m_1 : m_1(\{\theta_1\}) = 0.12, \quad m_1(\Theta) = 0.88,\\
&m_2 : m_2(\{\theta_1\}) = 0.16, \quad m_2(\Theta) = 0.84,\\
&m_3 : m_3(\{\theta_1\}) = 0.15, \quad m_3(\Theta) = 0.85,\\
&m_4 : m_4(\{\theta_1\}) = 0.11, \quad m_4(\Theta) = 0.89,\\
&m_5 : m_5(\{\theta_1\}) = 0.14, \quad m_5(\Theta) = 0.86,\\
&m_6 : m_6(\{\theta_2\}) = 0.95, \quad m_6(\Theta) = 0.05.
\end{aligned}
$$

As can be seen, the first five sources share similar belief (supporting $\{\theta_1\}$) whereas the sixth one delivers a mass function strongly committed to another solution (supporting $\{\theta_2\}$). These six mass functions cannot be regarded as conflicting, because the majority of evidence shows the preference of $\{\theta_1\}$. Here, source 6, is assumed not reliable since it contradicts with all the other sources.

The combination results by conjunctive rule, Dempster rule, disjunctive rule, DP rule, PCR6 rule, cautious rule, average rule and the proposed LNS-CR rule[4] are depicted in Table I. As can be observed, the conjunctive rule assigns most of the belief to the empty set, regarding the sources as highly conflictual. Dempster rule, DP rule, PCR6 rule and average rule redistribute all the global conflict to other focal elements. The disjunctive rule gives the total ignorance mass functions. The cautious rule and the proposed LNS-CR rule keep some of the conflict and redistribute the remaining. But the belief given to $\{\theta_2\}$ is more than that to $\{\theta_1\}$ when using Dempster, DP, PCR6, cautious and the average rules, which indicates that these rules are not robust to the unreliable evidence. The obtained fused bba by the proposed rule assigns the largest mass to focal element $\{\theta_1\}$, which is consistent with the intuition. It keeps a certain level of global conflict, and at the same time reflects the superiority of $\{\theta_1\}$ compared with $\{\theta_2\}$. From the results we can see that only the LNS-CR rule can correctly elicit the major opinion.

The LNS-CR rule is a conjunctive based combination rule for mass functions with different reliability degrees. As mentioned before, the principle of the LNS-CR rule is similar that of Schubert's method [32]. Table II lists the results by Schubert's combination method with different values of k. As can be seen, the result by the use of the LNS-CR rule is similar to that by Schubert's method with a small value of threshold k. When k is set small, the discounting process in Schubert's method needs more steps. And in each step, the conjunctive rule should be evoked to calculate the falsity. It is more complex compared with the reliability estimation process of the LNS-CR rule in that sense.

[4]As the focal elements are singletons except Θ, parameter η has no effects on the final results when using LNS-CR rule.

TABLE II
The combination of six masses by Schubert's method with different values of k.

k	0.1	0.2	0.3	0.4	0.5
Ø	0.09776	0.19471	0.28680	0.37803	0.46444
$\{\theta_1\}$	0.32187	0.26219	0.19350	0.12081	0.04980
$\{\theta_2\}$	0.13521	0.23145	0.31033	0.37979	0.43871
$\{\theta_1,\theta_2\}$	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_1,\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_2,\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000
Θ	0.44516	0.31165	0.20937	0.12137	0.04704

TABLE III
The combination of six masses by Martin's method with different values of λ.

λ	0.1	0.5	1	1.5	2
Ø	0.00000	0.00350	0.10485	0.23330	0.31956
$\{\theta_1\}$	0.00000	0.21206	0.34700	0.26789	0.19410
$\{\theta_2\}$	0.00000	0.01272	0.12719	0.23219	0.30256
$\{\theta_1,\theta_2\}$	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_1,\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000
$\{\theta_2,\theta_3\}$	0.00000	0.00000	0.00000	0.00000	0.00000
Θ	1.00000	0.77172	0.42096	0.26661	0.18378

We also compare with another reliability discounting based combination method proposed by Martin et al. [10]. Same as Schubert's method, after the reliability degree of each source is estimated, the bbas are discounted following with a conjunctive combination. There is a parameter λ in the method to adjust the discounting factor. The results varying with different values of λ are shown in Table III. We can see this rule is similar to LNS-CR rule when λ is set to be around 1. When λ is not well set, the results are not good. Moreover, in this method, the distance between bbas should be calculated first. Consequently, it increases the complexity and makes the method not feasible for combining a large number of sources.

EXPERIMENT 2 (The discounting mechanism). In this experiment, we will discuss the reliability discounting mechanism of the LNS-CR rule. Two reliability discounting methods proposed by Shubert [32] and Martin et al. [10] will be used to compare. Same as the LNS-CR rule, after the discounting process by these two methods, the conjunctive rule is adopted to combine the new mass functions. For simplicity, here we call the combination rule, where the Schubert's discounting method (or Martin's discounting method) is first evoked and then the conjunctive combination rule is used, "Schubert's method" (Martin's method, correspondingly). A set of $3 * x$ bbas on a frame of discernment $\Theta = \{\theta_1,\theta_2\}$ are generated, x of them are unreliable while $2 * x$ are reliable. The reliable sources assign a large mass to the singleton $\{\theta_1\}$. The unreliable sources assign a large mass to the singleton $\{\theta_2\}$. The gain factor for sequential discounting in Schubert's method is set to be 0.1 here. Schubert and Martin's methods are evoked with different values of k and λ respectively. Let $x = 10$, the fused bbas by the use of different rules are listed in Table IV.

From the table we can see, the behavior of Martin's discounting method is similar to that of LNS-CR rule when λ is set around 0.4. The conjunctive combination based on Schubert's discounting does not give any belief to $\{\theta_2\}$ and $\Theta = \{\theta_1,\theta_2\}$ at all although there are 1/3 of sources supporting $\{\theta_2\}$. Moreover, when k is larger, most of the mass is assigned to the empty set in this rule. From these results we can see that only LNS-CR rule can give more belief on $\{\theta_1\}$ which can be regarded as the major opinion. The time elapsed for Schubert's method with different values of threshold k is listed in Table V. The smaller the value of k is, the more discounting steps are required in Schubert's method. Consequently, the time consumption becomes larger. The running time for both LNS-CR rule and Martin's method is less than one second. Schubert's method is much more time-consuming.

TABLE IV
The combination results by different rules.

	Schubert's method				Martin's method				LNS-CR
	$k = 0.2$	$k = 0.3$	$k = 0.5$	$k = 0.7$	$\lambda = 0.3$	$\lambda = 0.4$	$\lambda = 0.6$	$\lambda = 1$	
Ø	0.19949	0.29860	0.49704	0.69306	0.00248	0.10019	0.60681	0.98649	0.15060
$\{\theta_1\}$	0.80051	0.70140	0.50296	0.30694	0.16901	0.56713	0.38729	0.01351	0.48612
$\{\theta_2\}$	0.00000	0.00000	0.00000	0.00000	0.01200	0.04995	0.00360	0.00000	0.08593
Θ	0.00000	0.00000	0.00000	0.00000	0.81650	0.28274	0.00230	0.00000	0.27735

TABLE V
Time elapsed for Schubert's method with different values of k.

	1	2	3	4	5	6	7	8	9
k	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
Time Elapsed (s)	46.81	21.64	13.46	9.28	6.64	4.88	3.67	2.73	1.79

We have also tested the combination methods based on the discounting factors proposed by Schubert [32] and Martin et al. [10] on some simple support mass functions with arbitrary focal elements. The results are not shown here as we can get similar conclusions from the results: The reliability estimation process of these methods takes more time compared with that of LNS-CR rule. The behavior of these two methods is similar to that of LNS-CR rule when the parameter k or λ is set to be in a fixed range. But they are much more time-consuming compared with LNS-CR rule. This confirms that the reliability discounting method in LNS-CR rule is effective for the following conjunctive combination.

EXPERIMENT 3 (The influence of parameter η). We test here the influence of parameter η in the LNS-CR rule. Simple support mass functions are utilized in this experiment. Suppose that the discernment frame under consideration is $\Theta = \{\theta_1, \theta_2, \theta_3\}$. Three types of SSFs are adopted. First $s_1 = 60$ and $s_2 = 50$ SSFs with focal elements $\{\theta_1\}$ and $\{\theta_2\}$ respectively (the other focal element is Θ) are uniformly generated, and then $s_3 = 50$ SSFs with focal element $\theta_{23} \triangleq \{\theta_2, \theta_3\}$ are generated. The value of masses are randomly generated. Different values of η (see Eq. (32)) ranging from 0 to 6 are used to test. The mass values in the fused bba by LNS-CR varying with η are displayed in Figure 1(a), and the corresponding pignistic probabilities are shown in Figure 1(b).

From these figures, we can see that η can have some effects on the final decision. Figure 1.a shows that with the increasing of η, the mass assigned to the singleton focal elements increases. On the contrary, the mass given to the focal element whose cardinality is bigger than one decreases. In fact parameter η in LNS-CR aims at weakening the imprecise evidence which gives only positive mass to focal elements with high cardinality, and the exponent η allows to control the degree of discounting. If η is larger, more weight is given to the sources of evidence whose focal elements are more specific, and more discount will be committed to the imprecise evidence. As a result, in the experiment when η is larger than 1.2, $\text{BetP}(\theta_1) > \text{BetP}(\theta_2)$ (Figure 1(b)). At this time the mass functions with focal element $\{\theta_2, \theta_3\}$ make little contribution to the fusion process, while the final decision mainly depends on the other two types of simple support mass functions with singletons as focal elements.

In real applications, η could be determined based on specific requirement. This work is not specially focusing on how to determine η, thus in the following experiment we will set $\eta = 1$ as default.

EXPERIMENT 4 (The principle for the global conflict). The goal of this experiment is to show how Dempster's degree of conflict is dealt with by most of rules when combining a large number of conflicting sources.

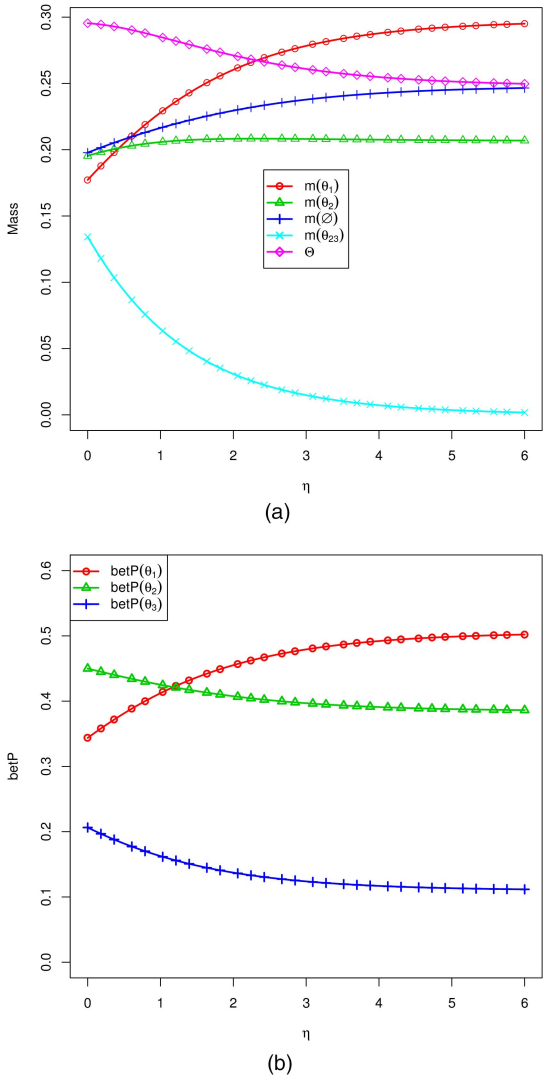


Fig. 1. Combination results for three types of SSFs using LNS-CR rule. The mass functions are generated randomly, and LNS-CR rule is evoked with different values of η ranging from 0 to 6. (a) bba. (b) Pignistic probability.

In this experiment, the frame of discernment is set to $\Theta = \{\theta_1, \theta_2\}$. Assume that there are only 2 focal elements on each bba. One is the whole frame Θ, and the other is any of the singletons ($\{\theta_1\}$ or $\{\theta_2\}$). The number of bbas which have the focal element $\{\theta_1\}$ is denoted by s_1, while that with $\{\theta_2\}$ is s_2. We first fix the value of s_2, and let $s_1 = t * s_2$, with t a positive integer. We generate $S = s_1 + s_2$ such kind of bbas randomly, but only withholding the bbas for which the mass value assigned to $\{\theta_1\}$ or $\{\theta_2\}$ is greater than 0.5.

Four values of t are considered here: $t = 1, 2, 3, 4$. If $t = 1$, $s_1 = s_2 = S/2$. If $t = 2$, the number of mass functions supporting $\{\theta_1\}$ is two times of that supporting $\{\theta_2\}$, and so on. The global conflict (mass given to the empty set) after the combination with different values of s_2 for the four cases is displayed in Figures 2–5 respectively. The mass assigned to the focal element $\{\theta_1\}$ with different combination approaches is shown in Figures 6–9.

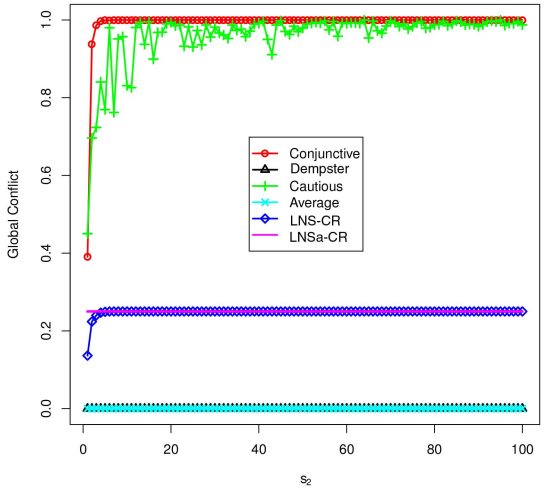


Fig. 2. The global conflict after the combination with s_2 ranging from $[0, 100]$ and $s_1 = s_2$.

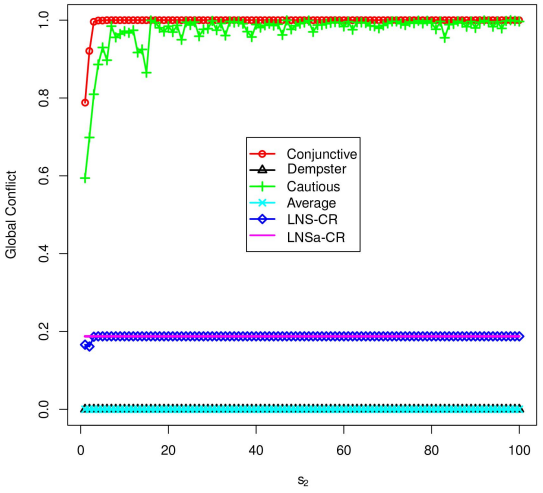


Fig. 4. The global conflict after the combination with s_2 ranging from $[0, 100]$ and $s_1 = 3 * s_2$.

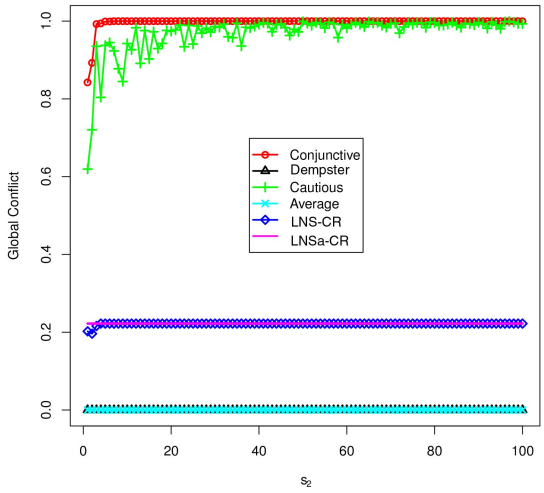


Fig. 3. The global conflict after the combination with s_2 ranging from $[0, 100]$ and $s_1 = 2 * s_2$.

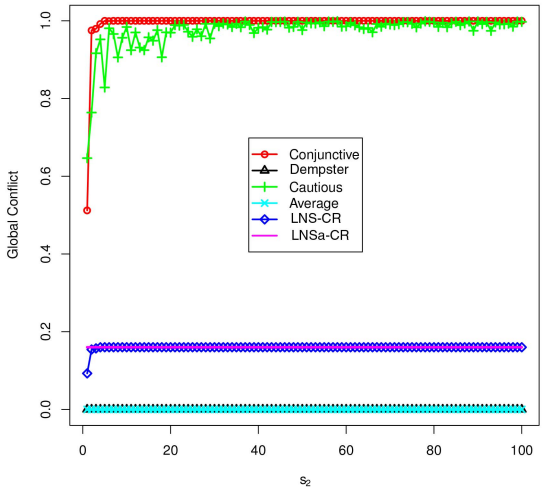


Fig. 5. The global conflict after the combination with s_2 ranging from $[0, 100]$ and $s_1 = 4 * s_2$.

It is intuitive that when t becomes larger, the global conflict should be smaller and we should give more belief to the focal element $\{\theta_1\}$. From Figures 2–9 we can see that only the results by LNS-CR rule are in accordance with this common sense. The simple average rule assigns larger bba to $\{\theta_1\}$, but it does not keep any conflict. In Figures 6–9, the mass given to $\{\theta_1\}$ by Dempster rule cannot be displayed when S is large (and also for some small S), because in these cases the global conflict is 1 and the normalization could not be processed. As we can see, Dempster rule could not work at all when s_2 is larger than 20. Although the conjunctive rule and cautious rule could work when combining a larger number of mass functions, the obtained fused mass function is $m(\emptyset) \approx 1$, which is useless for decision in practical situations.

The results also confirm the equivalent of the LNS-CR rule and LNSa-CR rule when the number of sources is large, although the results provided by the two rules are not the same when there are not many mass functions to combine. From Figures 2–5 we can see a kind of limit of the global conflict for the LNS-CR rule. In fact, the mass on the empty set for this rule depends on the size of the frame of discernment and more directly on the number of groups created in the first step of the rule. The limit value of the global conflict will tend to 1 with the increase of the size of discernment when considering only categorical bbas on different singletons.

EXPERIMENT 5 (The complexity). In this experiment, the complexity of LNS-CR rule will be compared with other combination rules in terms of time consumption. Simple support mass functions defined on a frame of discernment with eight elements are considered first. The focal elements of each bba are set to be a random subset of Θ and Θ itself. The time elapsed (and also the log value of the time elapsed) with the number of sources S varying from 10,000 to 100,000 is shown

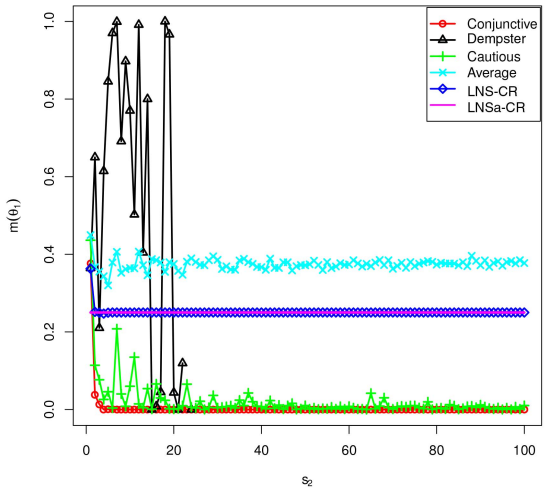


Fig. 6. The mass on $\{\theta_1\}$ after the combination with s_2 ranging from $[0,100]$ and $s_1 = s_2$.

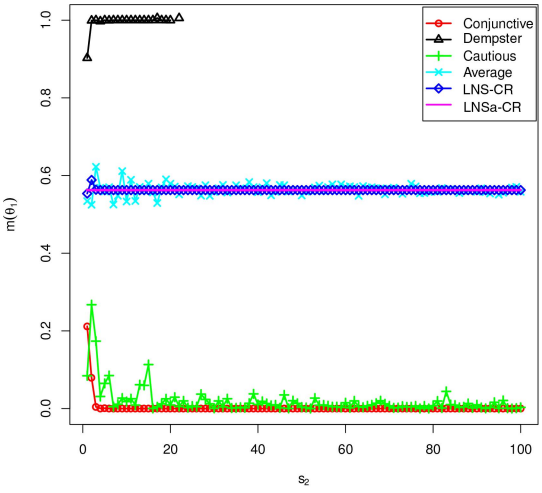


Fig. 8. The mass on $\{\theta_1\}$ after the combination with s_2 ranging from $[0,100]$ and $s_1 = 3 * s_2$.

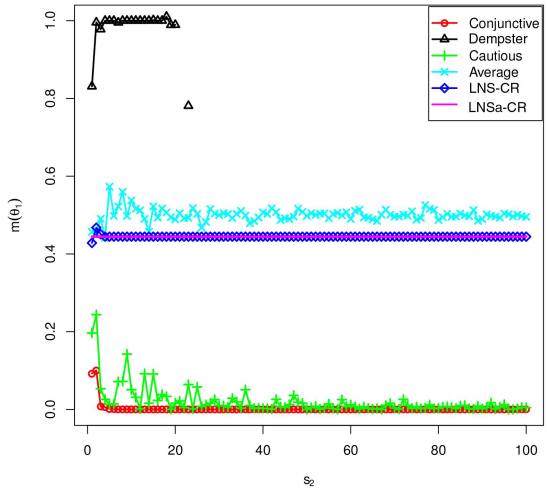


Fig. 7. The mass on $\{\theta_1\}$ after the combination with s_2 ranging from $[0,100]$ and $s_1 = 2 * s_2$.

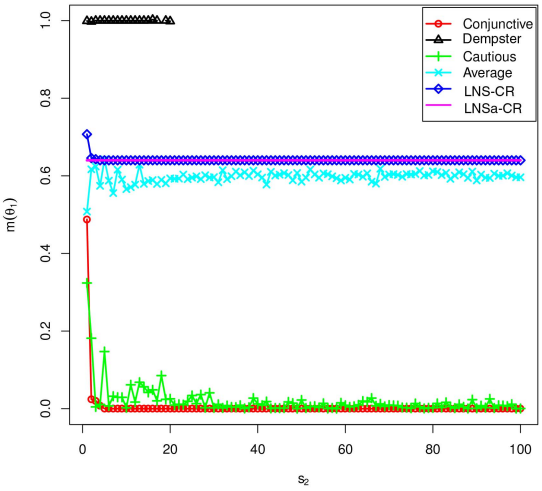


Fig. 9. The mass on $\{\theta_1\}$ after the combination with s_2 ranging from $[0,100]$ and $s_1 = 4 * s_2$.

in Figure 10.[5] We can see that the running time of LNS-CR is much smaller than that of the conjunctive rule. LNSa-CR rule takes almost the same time as cautious rule. Average rule is the best among the five rules. As S increases, the application of LNSa-CR rule can save more time compared with the use of LNS-CR rule. The increment of time consumption with respect to S is moderate. This tends to show that LNS-CR rule is suitable for combining a large number of SSFs. Remark that the decomposition process is not required when the cautious rule or LNS-CR(a) rule is adopted for combining SSFs.

As mentioned before, for the combination of general separable mass functions (not SSFs), LNS-CR needs four steps: decomposition, inner-group combination, discounting and global combination. The difference between the combination of any kind of separable bbas and of SSFs is the decomposition process, which is not necessary for the latter. We have designed another experiment on consonant bbas[6] over a frame of discernment with eight elements, and the number of focal elements is set to 5. The focal elements are randomly set to five nested subsets of Θ, and the mass values are generated uniformly. The average running time (and the log value of the running time) of 10 trials by the use of different combination rules with different number of sources S is displayed in Figure 11(a) (and Figure 11(b)).[7] In order to show the complexity of LNS-CR rule more clearly, the elapsed time in each of the four steps is shown in Figure 12.

As we can see from these figures, the time consumption of LNS-CR is significantly smaller than the cau-

[5]The result of Dempster rule is the same as that of conjunctive rule.

[6]All consonant bbas are separable.

[7]The result of cautious rule is not displayed for large S, as it has been already shown that cautious rule is significantly worse than the other rules in terms of time consumption when S is small.

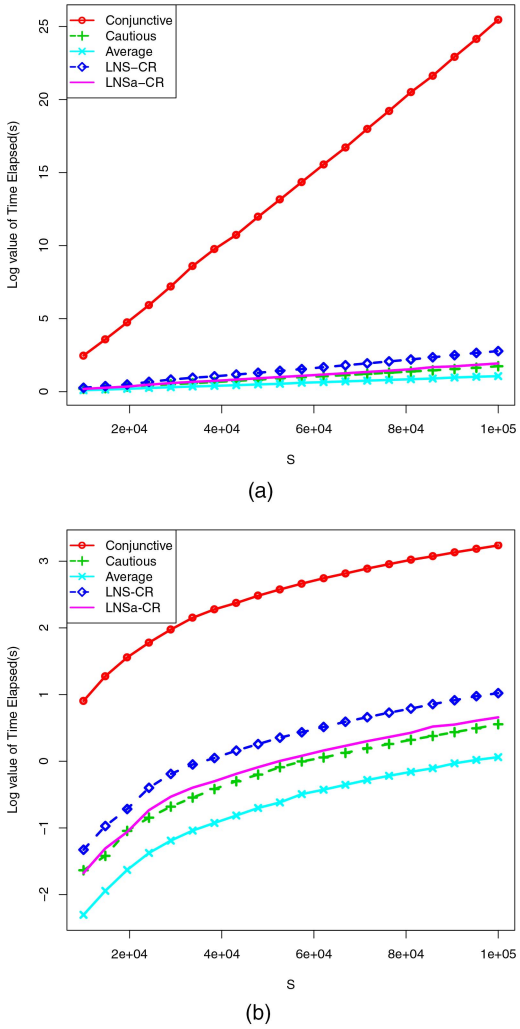


Fig. 10. Time lapse for combining SSFs. (a) Time lapse by five different rules. (b) The log value of Time lapse by five different rules.

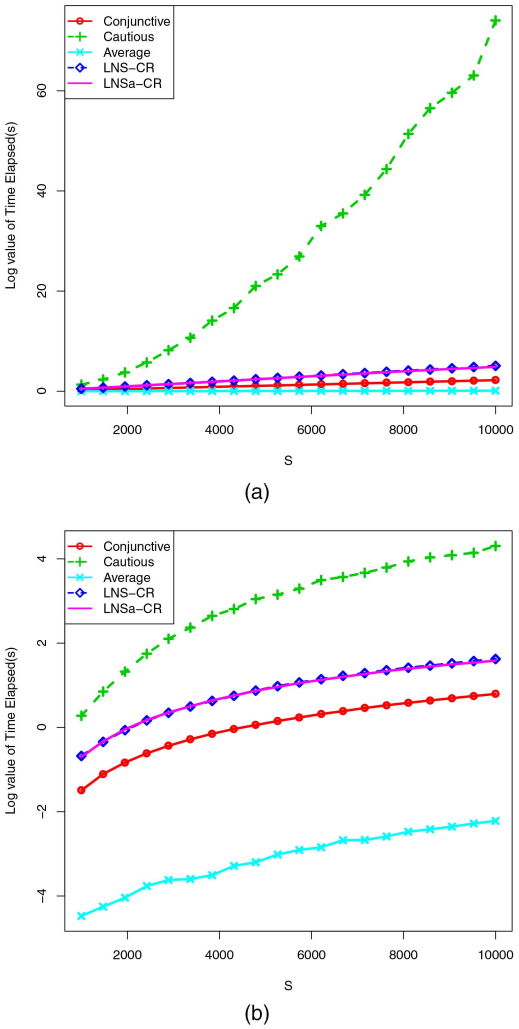


Fig. 11. Time lapse for combining consonant bbas. (a) Time lapse by five different rules. (b) The log value of Time lapse by five different rules.

tious rule, but a little worse than the conjunctive rule and the average rule. Although the complexity of cautious rule is the same as LNS-CR rule and both of them require a decomposition process, it takes more running time than LNS-CR rule. The reason may be the different combination approach for the mass functions in the same group. The complexity of that process by cautious rule is $O(S2^n)$ (The calculation is to find the minimum of each row in a $S \times 2^n$ matrix), while for LNS-CR is $O(S)$. LNSa-CR is faster than LNS-CR when S is large. Figure 12 shows that the most time-consuming step in LNS-CR rule is the decomposition. Moreover as S increases, the increase of time lapse for the inner-group combination, discount, and global combination is limited. This is compliant with the complexity analysis of each step for LNS-CR rule in Section III-C. In many applications the mass functions are directly SSFs in which case there is no need to perform the decomposition, and LNS-CR is the best choice to fuse a large number of bbas.

V. PERSPECTIVE ON APPLICATIONS

Pattern recognition is a class of problems where the theory of belief functions has proved to allow increased performances [2]. In such problems we can be facing many bbas to combine. Denœux [2] proposed Evidential KNN method (EKNN) as an extension of KNN in the framework of the theory of belief functions to better model the uncertainty in neighbor point interactions. The Dempster rule is adopted to combine the mass evidence from K neighbors in EKNN.

The problem considered here is to classify an input pattern $\mathbf{x}$ into n categories or classes, denoted by $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$. The available information is assumed to consist of a training set $\mathcal{L} = \{(\mathbf{x}^{(1)}, \theta^{(1)}), (\mathbf{x}^{(2)}, \theta^{(2)}), \ldots, (\mathbf{x}^{(N)}, \theta^{(N)})\}$ of N patterns $\mathbf{x}^{(i)}$ $i = 1, 2, \ldots, N$ with known class labels $\theta^{(i)} \in \Theta$. To classify pattern $\mathbf{x}$, each pair $(\mathbf{x}^{(i)}, \theta^{(i)})$ constitutes a distinct item of evidence regarding the class membership of $\mathbf{x}$. If the K nearest neighbors according to the distance measure are considered, K items of evidence can be obtained. These bbas can

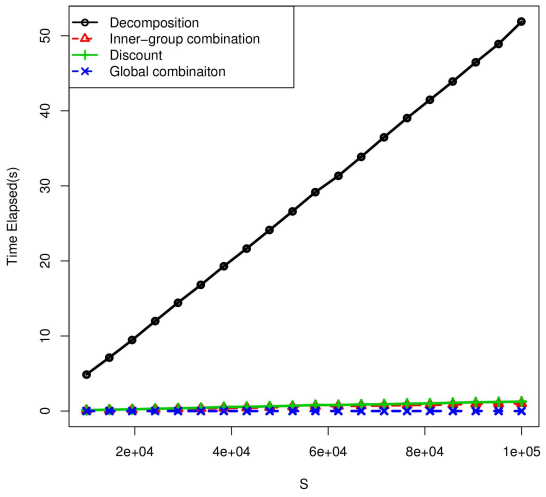


Fig. 12. Time lapse of each step using LNS combination rule with S varying from 10,000 to 100,000.

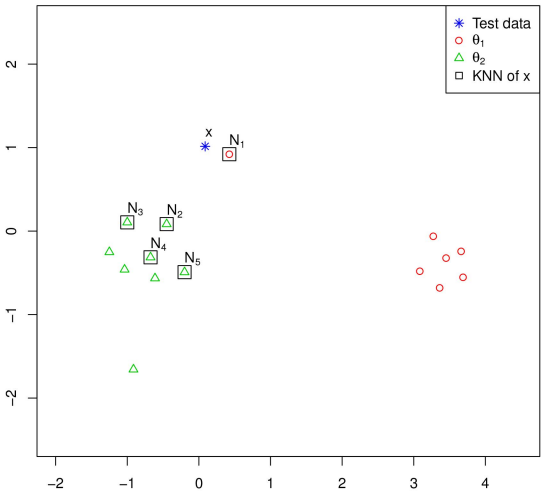


Fig. 13. A small data set.

be constructed according to a relevant metric between pattern $\mathbf{x}$ and its jth neighbor $\mathbf{x}^{(i)}$

$$\begin{aligned} m_i(\{\theta_q\}) &= \alpha\phi(d^{(i)}), \\ m_i(\Theta) &= 1-\alpha\phi(d^{(i)}), \\ m_i(A) &= 0 \quad \forall A \in 2^\Theta \setminus \{\{\theta_q\},\Theta\}, \end{aligned} \tag{41}$$

where $d^{(i)}$ is the (Euclidean) distance between $\mathbf{x}$ and its jth neighbor $\mathbf{x}^{(i)}$ with class label $\theta^{(i)} = \theta_q$, α is a discounting parameter and $\phi(\cdot)$ is a decreasing function on $\mathbb{R}^+$ defined as

$$\phi(d^{(i)}) = \exp(-\gamma_q(d^{(i)})^2) \tag{42}$$

with γ_q being a positive parameter associated to class θ_q. It can be heuristically set to the inverse of the mean Euclidean distance between training data belonging to class θ_q. In EKNN, the K bbas for each neighbor are aggregated using the Dempster rule to form a resulting bba. A decision has to be made regarding the assignment of sample $\mathbf{x}$ to one individual class. The maximum of pignistic probability can be used for decision-making.

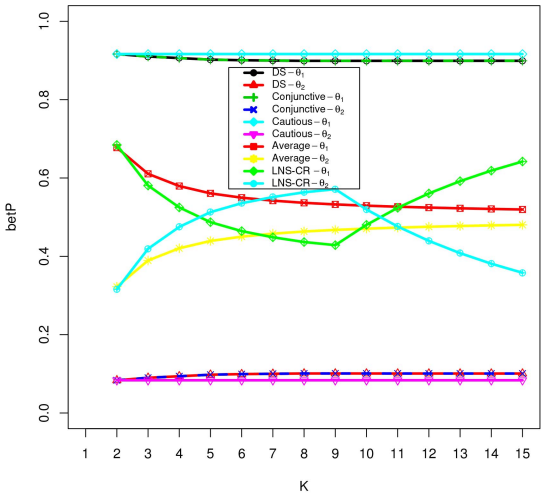


Fig. 14. Pignistic probability.

TABLE VI
The fused bba by different combination rules ($K = 4$).

	Conjunctive	Dempster	Cautious	Average	LNS-CR
Ø	0.2009	0.0000	0.1473	0.0000	0.0377
$\{\theta_1\}$	0.6771	0.8473	0.7307	0.2195	0.1818
$\{\theta_2\}$	0.0279	0.0349	0.0205	0.0606	0.1339
Θ	0.0941	0.1177	0.1015	0.7199	0.6466

TABLE VII
The fused bba by different combination rules ($K = 5$).

	Conjunctive	Dempster	Cautious	Average	LNS-CR
Ø	0.2198	0.0000	0.1473	0.0000	0.0352
$\{\theta_1\}$	0.6582	0.8436	0.7307	0.1756	0.1404
$\{\theta_2\}$	0.0305	0.0391	0.0205	0.0541	0.1651
Θ	0.0915	0.1172	0.1015	0.7703	0.6593

A. A small data set with noisy training sample

Figure 13 illustrates a simple two-class (red circle and green triangle) data set, where there are seven objects in each class. The pattern $\mathbf{x}$ marked by blue star is the sample data to be classified. The K bbas using the distance to its neighbor could be constructed by Eq. (41), and the five nearest neighbors are denoted by N_i orderly in the figure. Set $\alpha = 0.95$ and γ_i is the inverse of the average distance between the points in class θ_i, $i = 1,2$. The fused mass function by different combination rules with $K = 4$ and $K = 5$ are listed in Table VI and VII respectively.

As we can see from Figure 13, pattern $\mathbf{x}$ is closer to class θ_2. Among pattern $\mathbf{x}$s five nearest neighbor N_j, $j = 1,2,\ldots,5$, four belong to class θ_2 while only 1 to class θ_1. The real class of object N_1 is θ_1, but it is located in the boundary of the class and far from the other data points in the class. It may be a noisy item

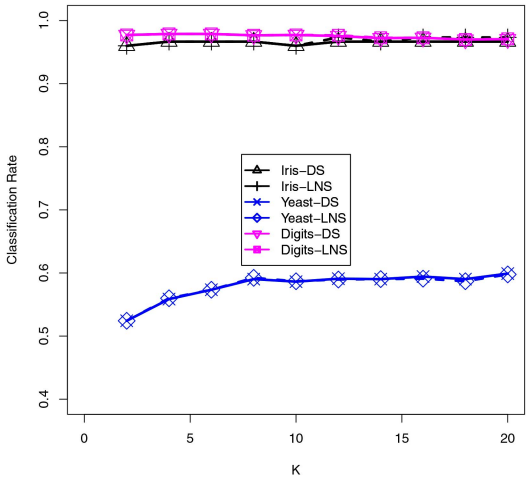


Fig. 15. Classification results with different values of K on UCI data set. In the figure, the legend "Iris-DS" means it is the classification rates on Iris data set using DS combination rule. Same as the other legends.

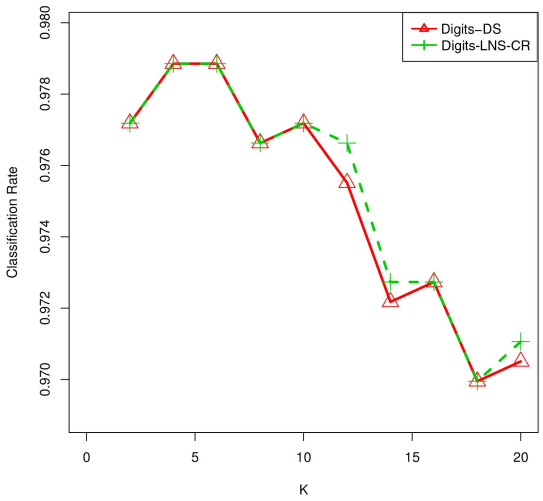


Fig. 16. Classification rates on Digits data set.

of θ_1. The standard KNN rule can correctly classify object x to θ_2 when $K > 3$. However, if the evidential KNN model is applied, due to the existence of a such neighbor, the behavior of the combination rules has been affected. From Table VI we can see, when $K = 4$, the fused bbas by all combination rules all assign more mass to θ_1 than to θ_2. Consequently, pattern $\mathbf{x}$ will be classified into class θ_1 if the pignistic probability is considered for making decision. The same phenomenon also occurs when K is smaller than 4 (see Figure 14). When $K = 5$ (Table VII), only the LNS-CR rule could partition pattern $\mathbf{x}$ into class θ_2, which seems more reasonable. The pignistic probabilities (Figure 14) by the Dempster, conjunctive, cautious and average rules for class θ_1 are significantly higher than those for class θ_2, even when K is large. These rules are not robust to the noisy training data. Pattern $\mathbf{x}$ could be correctly classified to θ_2 by LNS-CR rule when K is between 5 and 10.

It is indicated that when there are some noisy data in the training data set, the performance of the combination rule may become worse with small K. We should increase K moderately to improve the performance of the classifier. But as we analyzed before, the existing combination rules do not work well for aggregating a large number of mass functions. This is a limit of the use of evidential classifier.

B. Real data sets

In this section, we consider some well known real data sets from the UCI repository[8] summarized in Table VIII. The classification rates by using different combination rules in evidential KNN model are displayed in Figure 15. Note that the "leave-one-out" method is adopted here to test the classifier.

[8]http://archive.ics.uci.edu/ml/datasets.html.

TABLE VIII
A summary of UCI data sets.

Data set	No. of objects	No. of cluster	No. of attributes
Iris	150	3	4
Yeast	1484	10	8
Digits	5620	10	64

As we can see from Figure 15, for all the three data sets, the performance is almost the same for the two combination rules, LNS-CR and DS, in terms of classification rates. But there is a little improvement by the use of LNS-CR rule when K is large. To make it clear, we specially depict the results on Digits data set in Figure 16. It is shown that when $K > 12$, the classification rates by the use LNS-CR rule are a little larger than those through DS rule. We show the mass given to the empty set (global conflict) after the combination using conjunctive rule and LNS-CR rule with different values of K in Figure 17. The y-axis is the maximal assignment to $\emptyset$ among all the mass functions for the test data. As we can see, the global conflict tends to 1 quickly as K increases, while LNS-CR rule keeps a moderate degree of global conflict. As DS rule is a normalized conjunctive rule, there is not sense to normalize a mass assignment with high global conflict.

C. Perspective

The above two examples are just two perspectives on the application of LNS-CR rule. In the first example, there are some special noisy data in the training data set. At this time, the sources should not be considered with equal reliability. In this situation, using the DS rule or the conjunctive rule in EKNN model could not get good results. In the second example, it is shown that the global conflict may tend to one quickly as K increases. Sometimes we even could not do the normalization process for DS rule because of the machine precision.

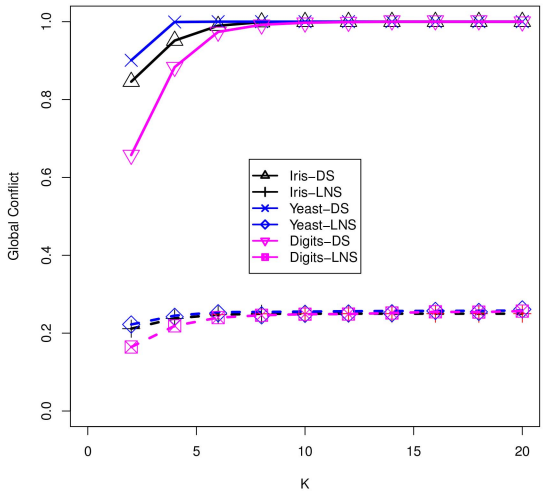


Fig. 17. Global conflict using conjunctive rule and LNS-CR rule varying with different values of K. In the figure, the legend "Iris-DS" means it is the conflict on Iris data set using DS combination rule. Same as the other legends.

In real world social networks, the available information can be uncertain, or even noisy. At this time, if we want to do a classification task such as for recommendation, the conjunctive rule could not be applied as the sources are not all reliable. Even if the sources are reliable, the global conflict may tend to 1 quickly if the bbas are not consistent. At this time, LNS-CR rule can be an alternative choice. In the future work, we will study how Dempster's degree of conflict is distributed in the feature space, and to study what special information contained in the moderate degree of global conflict kept by LNS-CR rule.

VI. CONCLUSION

Uncertainty in big data applications has attracted more and more attention. The theory of belief functions is one of the uncertainty theories allowing a model to deal with imprecise and uncertain information. This theory is also well designed for information fusion. However, despite that a lot of combination rules have been proposed in recent years in this framework, they are not able to combine a large number of sources because of the complexity or the absorbing element.

In this paper, a new combination rule, named LNS-CR rule, preserving the principle of the conjunctive rule is proposed. This rule considers the mass functions given by the sources and groups them according to their set of focal elements (without auto-conflict). The mass functions of each group can be summarized by one mass function after combination. The reliability of the source is estimated by the proportion of bbas in one group. Therefore, after discounting the mass function of each group by the reliability factor, the final combination can be proceeded by the conjunctive rule (or another rule according to the application). If the number of sources in each group is high enough, an approximation method is presented.

The LNS-CR rule is able to combine a large number of sources. The only existing method allowing to combine a large number of mass functions is the average rule. However, that rule may give more importance to few sources with a high belief (even if the source is not reliable) and cannot capture the conflict between the sources. The proposed rule with a reasonable complexity (lower than the DP and PCR6 rules) can provide good combination results.

Overall, this work provides a perspective for the application of belief functions on big data. We will study how to apply LNS-CR rule on the problems of social network and crowdsourcing in the future research work.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (Nos. 61701409, 61135001, 61403310, 61672431), the Natural Science Basic Research Plan in Shaanxi Province of China (No. 2018JQ6005), and the Fundamental Research Funds for the Central Universities of China (No. 3102016QD088).

REFERENCES

[1] K. Zhou, A. Martin, and Q. Pan
"Evidence combination for a large number of sources,"
in *20th International Conference on Information Fusion*. IEEE, 2017, pp. 1–8.

[2] T. Denœux
"A k-nearest neighbor classification rule based on dempster-shafer theory,"
Systems, Man and Cybernetics, IEEE Transactions on, vol. 25, no. 5, pp. 804–813, 1995.

[3] X. Deng, Q. Liu, Y. Deng, and S. Mahadevan
"An improved method to construct basic probability assignment based on the confusion matrix for classification problem,"
Information Sciences, vol. 340, pp. 250–261, 2016.

[4] M.-H. Masson and T. Denœux
"ECM: An evidential version of the fuzzy c-means algorithm,"
Pattern Recognition, vol. 41, no. 4, pp. 1384–1397, 2008.

[5] K. Zhou, A. Martin, Q. Pan, and Z.-G. Liu
"Ecmdd: Evidential c-medoids clustering with multiple prototypes,"
Pattern Recognition, vol. 60, pp. 239–257, 2016.

[6] K. Zhou, A. Martin, Q. Pan, and Z.-g. Liu
"Median evidential c-means algorithm and its application to community detection,"
Knowledge-Based Systems, vol. 74, pp. 69–88, 2015.

[7] P. Smets
"Analyzing the combination of conflicting belief functions,"
Information Fusion, vol. 8, pp. 387–412, 2007.

[8] ———
"The combination of evidence in the transferable belief model,"
Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 12, no. 5, pp. 447–458, 1990.

[9] P. Smets and R. Kennes
"The transferable belief model,"
Artificial intelligence, vol. 66, no. 2, pp. 191–234, 1994.

[10] A. Martin, A.-L. Jousselme, and C. Osswald
"Conflict measure for the discounting operation on belief functions,"
in *Information Fusion, 2008 11th International Conference on*. IEEE, 2008, pp. 1–8.

[11] W. Liu
"Analyzing the degree of conflict among belief functions,"
Artificial Intelligence, vol. 170, no. 11, pp. 909–924, 2006.

[12] S. Destercke and T. Burger
"Toward an axiomatic definition of conflict between belief functions,"
Cybernetics, IEEE Transactions on, vol. 43, no. 2, pp. 585–596, 2013.

[13] A. Martin and C. Osswald
"Human experts fusion for image classification,"
Information & Security: An International Journal, Special issue on Fusing Uncertain, Imprecise and Paradoxist Information (DSmT), vol. 20, pp. 122–143, 2006.

[14] E. Lefèvre and Z. Elouedi
"How to preserve the conflict as an alarm in the combination of belief functions,"
Decision Support Systems, vol. 56, pp. 326–333, 2013.

[15] R. R. Yager
"On the dempster-shafer framework and new combination rules,"
Information sciences, vol. 41, no. 2, pp. 93–137, 1987.

[16] D. Dubois and H. Prade
"Representation and combination of uncertainty with belief functions and possibility measures,"
Computational Intelligence, vol. 4, no. 3, pp. 244–264, 1988.

[17] R. Ilin and E. Blasch
"Information fusion with belief functions: A comparison of proportional conflict redistribution PCR5 and PCR6 rules for networked sensors,"
in *18th International Conference on Information Fusion*. IEEE, 2015, pp. 2084–2091.

[18] A. Martin, C. Osswald, J. Dezert, and F. Smarandache
"General combination rules for qualitative and quantitative beliefs,"
Journal of Advances in Information Fusion, vol. 3, no. 2, pp. 67–89, 2008.

[19] Y. Zhao, R. Jia, and P. Shi
"A novel combination method for conflicting evidence based on inconsistent measurements,"
Information Sciences, vol. 367–368, pp. 125–142, 2016.

[20] P. Orponen
"Dempster's rule of combination is #*P*-complete,"
Artificial Intelligence, vol. 44, pp. 245–253, 1990.

[21] W. T. Da Silva and R. L. Milidiú
"Algorithms for combining belief functions,"
International Journal of Approximate Reasoning, vol. 7, no. 1–2, pp. 73–94, 1992.

[22] A. Martin and C. Osswald
"Toward a combination rule to deal with partial conflict and specificity in belief functions theory,"
in *10th International Conference on Information Fusion*. IEEE, 2007, pp. 1–8.

[23] P. Smets
"The α-junctions: the commutative and associative non interactive combination operators applicable to belief function,"
in *1st International Joint Conference on Qualitative and Quantitative Practical Reasoning*, 1997, pp. 131–153.

[24] Y. Leung, N.-N. Ji, and J.-H. Ma
"An integrated information fusion approach based on the theory of evidence and group decision-making,"
Information Fusion, vol. 14, no. 4, pp. 410–422, 2013.

[25] G. Shafer
A mathematical theory of evidence.
Princeton University Press, 1976.

[26] P. Smets
"The application of the matrix calculus to belief functions,"
International Journal of Approximate Reasoning, vol. 31, no. 1, pp. 1–30, 2002.

[27] ———
"Decision making in the TBM: the necessity of the pignistic transformation,"
International Journal of Approximate Reasoning, vol. 38, no. 2, pp. 133–147, 2005.

[28] K. Sentz and S. Ferson
"Combination of evidence in dempster-shafer theory,"
Sandia National Laboratorie, Tech. Rep., 2002.

[29] P. Smets
"Belief Functions: the Disjunctive Rule of Combination and the Generalized Bayesian Theorem,"
International Journal of Approximate Reasoning, vol. 9, pp. 1–35, 1993.

[30] A. Martin
"Comparative study of information fusion methods for sonar images classification,"
in *Information Fusion, 2005 8th International Conference on*, vol. 2. IEEE, 2005, pp. 7–pp.

[31] A. Samet, E. Lefevre, and S. Ben Yahia
"Reliability estimation with extrinsic and intrinsic measure in belief function theory,"
in *5th International Conference on Modeling, Simulation and Applied Optimization*. IEEE, 2013, pp. 1–6.

[32] J. Schubert
"Conflict management in dempster-shafer theory using the degree of falsity,"
International Journal of Approximate Reasoning, vol. 52, no. 3, pp. 449–460, 2011.

[33] Z. Elouedi, K. Mellouli, and P. Smets
"The evaluation of sensors' reliability and their tuning for multisensor data fusion within the transferable belief model,"
in *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*. Springer, 2001, pp. 350–361.

[34] A. Samet, E. Lefèvre, I. Hammami, and S. Ben Yahia
"Reliability estimation measure: Generic discounting approach,"
International Journal of Pattern Recognition and Artificial Intelligence, vol. 29, no. 07, p. 1559011, 2015.

[35] Y. Yang, D. Han, and C. Han
"Discounted combination of unreliable evidence using degree of disagreement,"
International Journal of Approximate Reasoning, vol. 54, no. 8, pp. 1197–1216, 2013.

[36] J. Klein and O. Colot
"Singular sources mining using evidential conflict analysis,"
International Journal of Approximate Reasoning, vol. 52, no. 9, pp. 1433–1451, 2011.

[37] P. Smets
"The canonical decomposition of a weighted belief,"
in *14th International Joint Conference on Artificial Intelligence*, vol. 95, 1995, pp. 1896–1901.

[38] T. Denœux
"Conjunctive and disjunctive combination of belief functions induced by nondistinct bodies of evidence,"
Artificial Intelligence, vol. 172, no. 2, pp. 234–264, 2008.

[39] X. Ke, L. Ma, and Y. Wang
"Some notes on canonical decomposition and separability of a belief function,"
in *Belief Functions: Theory and Applications*, ser. Lecture Notes in Computer Science, F. Cuzzolin, Ed., vol. 8764. Springer International Publishing, 2014, pp. 153–160.

[40] R. Kennes
"Computational aspects of the möbius transformation of graphs,"
Systems, Man and Cybernetics, IEEE Transactions on, vol. 22, no. 2, pp. 201–223, 1992.

[41] A. P. Dempster
"Upper and lower probabilities induced by a multivalued mapping,"
The annals of mathematical statistics, pp. 325–339, 1967.

[42] C. Osswald and A. Martin
"Understanding the large family of dempster-shafer theory's fusion operators—a decision-based measure,"
in *9th International Conference on Information Fusion*. IEEE, 2006, pp. 1–7.

[43] P. Smets
"Belief functions: the disjunctive rule of combination and the generalized bayesian theorem,"
International Journal of approximate reasoning, vol. 9, no. 1, pp. 1–35, 1993.

[44] N. Wilson
"Algorithms for Dempster-Shafer theory,"
in *Hanbook of defeasible reasoning and uncertainty management*, D. Gabbay and P. Smets, Eds. Boston: Kluwer Academic Publisher, 2000, vol. 5: Algorithms for uncertainty and Defeasible Reasoning, pp. 421–475.

[45] T. Denœux and A. Ben Yaghlane
"Approximating the combination of belief functions using the fast Möbius transform in a coarsened frame,"
International Journal of Approximate Reasoning, vol. 30, no. 1–2, pp. 77–101, 2002.

[46] A. Martin
"Implementing general belief function framework with a practical codification for low complexity,"
in *Advances and Applications of DSmT for Information Fusion*, F. Smarandache and J. Dezert, Eds. American Research Press Rehoboth, 2009, vol. 3, ch. 7, pp. 217–274.

[47] T. Denœux
"The cautious rule of combination for belief functions and some extensions,"
in *9th International Conference on Information Fusion*. IEEE, 2006, pp. 1–8.

[48] K.-S. Chin and C. Fu
"Weighted cautious conjunctive rule for belief functions combination,"
Information Sciences, vol. 325, pp. 70–86, 2015.

[49] C. K. Murphy
"Combining belief functions when evidence conflicts,"
Decision support systems, vol. 29, no. 1, pp. 1–9, 2000.

[50] F. Smarandache and J. Dezert
Advances and Applications of DSmT for Information Fusion. American Research Press, Rehoboth, 2004–2009, vol. 1–3.

[51] D. Dubois and H. Prade
"Consonant approximation of belief functions,"
International Journal of Approximate Reasoning, vol. 4, no. 5–6, pp. 419–449, 1990.

[52] A. Aregui and T. Denœux
"Constructing consonant belief functions from sample data using confidence sets of pignistic probabilities,"
International Journal of Approximate Reasoning, vol. 49, no. 3, pp. 575–594, 2008.

[53] K. Zhou and A. Martin
ibelief: Belief Function Implementation,
2015, r package version 1.2. [Online]. Available: http://CRAN.R-project.org/package=ibelief.

[54] T. Burger and S. Destercke
"How to randomly generate mass functions,"
International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 21, no. 05, pp. 645–673, 2013.

Kuang Zhou was born in China in 1987. He received the Bachelor degree in Chang'an University in 2010, the master in Northwestern Polytechnical University (NPU) in 2013, and the doctor degree in University of Rennes1 in 2016. Now he is an assistant professor in School of Sciences, NPU. His research work focuses on statistical machine learning and data mining.

Arnaud Martin was born in France in 1974. He received a HDR (French ability to supervised research) in computer sciences (2009), a PhD degree in Signal Processing (2001), and Master in Probability (1998) from University of Rennes1, France. Since September 2010, he is a professor in the laboratory IRISA in University of Rennes1. His research interests are mainly related to belief function theory for the classification and include data fusion, data mining.

Quan Pan was born in China in 1961. He received the Bachelor degree in Huazhong University of Science and Technology, and he received the master and doctor degrees in Northwestern Polytechnical University (NPU) in 1991 and 1997 respectively. He has been a professor since 1998 in NPU. His current research interests are data classification and information fusion.

Evaluation of Fusion Algorithms for Passive Localization of Multiple Transient Emitters

WENBO DOU
JEMIN GEORGE
LANCE M. KAPLAN
RICHARD W. OSBORNE, III
YAAKOV BAR-SHALOM

The problem of localizing an unknown number of stationary transient emitters using passive sensors in the presence of missed detections and false alarms is investigated. Each measurement is based on one detection by a passive sensor and consists of a time of arrival and a bearing. It is assumed that measurements within a short time interval have to be associated before estimation. Both a Bernoulli measurement model and a Poisson measurement model are considered for each target. These two measurement models lead to two different proposed problem formulations: one is an *S*-dimensional (*S*-D) assignment problem and the other is a cardinality selection problem. The former can be solved by the Lagrangian relaxation algorithm reliably when the number of sensors is small. The sequential *m*-best 2-D (SEQ[*m*(2-D)]) assignment algorithm, which is resistant to the ghosting problem due to the estimation of the emitter signal's emission time, is developed to solve the problem when the number of sensors becomes large. Simulation results show that the SEQ[*m*(2-D)] assignment algorithm is efficient for real-time processing with reliable associations and estimates. In the cardinality selection formulation, a list of measurements is modeled as either realizations of a random variable with a uniform–Gaussian mixture (UGM) density or a Poisson point process (PPP). Because of an efficient way of incorporating false alarm rate, the UGM formulation is shown to be a useful alternative to the PPP formulation. Simulation studies show that both UGM and PPP formulations, which are based on the expectation–maximization algorithm, require the right initial estimates to yield reliable localization results.

Manuscript received March 20, 2016; revised August 19, 2016 and January 18, 2018; released for publication July 8, 2019.

Refereeing of this contribution was handled by Ramona Georgescu.

Authors' addresses: W. Dou, R. W. Osborne, III, and Y. Bar-Shalom are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269, USA (E-mail: wenbo.dou@uconn.edu, richard.osborne@engineer.uconn.edu, ybs@engr.uconn.edu).

J. George and L. M. Kaplan are with the U.S. Army Research Laboratory, 2800 Powder Mill Rd., Adelphi, MD 20783, USA (E-mail: jemin.george.civ, lance.m.kaplan.civ@mail.mil).

This research was supported by ARO Grant W991NF-10-1-0369.

1557-6418/19/$17.00 © 2019 JAIF

I. INTRODUCTION

This paper considers the problem of multiple transient emitter (target) localization using a group of passive sensors. One particular application is to utilize a network of acoustic gunfire detection systems on a group of soldiers to localize adversaries in a battlefield [12], [20]. It is assumed that the targets are stationary during the time window of interest but the number of targets is unknown. The sensors can measure line of sight (LOS) angles to the targets by detecting their emitted acoustic signals and record the times of arrival of the detected signals. Missed detections and false alarms are present due to the imperfection of the sensors. Furthermore, the association between the measurements and the targets is unknown; that is, each sensor does not know from which target (or clutter) a particular measurement originates. Before estimating the position of any target, one has to associate the measurements from all the sensors. Therefore, the quality of data association is critical to the overall localization performance.

The problem of data association has been studied extensively in tracking multiple targets. Methods including multiple hypothesis tracking [6], joint probabilistic data association filter [11], and probability hypothesis density filter [16] are recursive algorithms that require persistent measurements and provide solutions to a dynamic data association problem. Therefore, they cannot be employed to solve the static data association problem considered in the situation of multiple transient emitter localization.

There are two different philosophies—hard data association and soft data association (see [4, Sec. 2.4.3])—in solving the static data association problem considered in this paper. Hard data association either assigns a measurement to one and only one target or condemns it as a false alarm; in other words, the probability of a measurement coming from a target is either 0 or 1 (discrete). In contrast, soft data association assigns the event that a measurement originates from a target to a (continuous) probability, which can be any value between 0 and 1.

The hard data association for *S* lists of measurements with one list from each sensor,[1] assuming a Bernoulli measurement model that the number of measurements from each target received at each sensor is a Bernoulli random variable with parameter equal to the probability of detection, leads to an *S*-dimensional (*S*-D) assignment problem, which can be formulated as a discrete

[1] In a multisensor localization application, as in this paper, the number of lists is the same as the number of sensors.

constrained optimization problem aiming to find out the set of S-tuples of measurements that minimizes the overall association cost. The number of possible S-tuple sets for T targets and S sensors in the absence of missed detections and false alarms is $(T!)^{S-1}$, from which it can be seen that S-D assignment problem is nondeterministic polynomial-time (NP) hard with $S \geq 3$. Therefore, it is of great interest and importance to find robust suboptimal algorithms.

The Lagrangian relaxation-based approach [8], which is termed as the S-D algorithm in this paper, provides a measure of how close the final solution is to the (unknown) optimal solution in terms of the association cost. The application of the S-D assignment algorithm on a multiple shooter localization problem using a small number of sensors was presented in [19]. Although it does not explore the entire space of the S-tuple sets, it needs to calculate the cost of candidate S-tuples. The cost calculation involves finding the maximum likelihood (ML) estimate of the target locations and can take most of the computational time. The number of candidate S-tuples for T targets and S sensors in the absence of missed detections and false alarms is T^S, which increases exponentially with the number of sensors. Since more sensors generate more accurate estimates in the fusion center, computationally efficient algorithms are required when a large number of sensors are deployed.

The S_0-D + SEQ(2-D) algorithm [23], which performs the S-D assignment algorithm on S_0 lists of measurements before applying the modified auction algorithm [21] for 2-D assignments on the remaining lists sequentially $S - S_0$ times, is a more efficient algorithm than the S-D assignment. The number of candidate associations increases quadratically (rather than combinatorially/exponentially) with the number of sensors. Because of the ghosting problem [4], the S_0-D step requires, in general, at least three lists to achieve reliable association. However, since in the present problem one also has arrival times, one can use $S_0 = 2$.

The problem of multiple shooter localization using a single sensor [13] or using multiple sensors [14] is formulated as a cardinality (number of targets) selection problem that assumes a Poisson measurement model that the number of measurements from each target received at each sensor is a Poisson random variable with parameter equal to the probability of detection. The measurements at a single sensor from all targets and the clutter are modeled as a Poisson point process (PPP) [7]. For each possible selected cardinality, one solves a subproblem based on the learning expectation–maximization (EM) algorithm [9] to select the best cardinality based on an information criterion [1], [22]. During every iteration of the EM algorithm, each measurement will be assigned a probability of having originated from a target, which is an example of the soft data association.

In this paper, we discuss two classes of algorithms, each for a specific measurement model in the multiple passive transient emitter localization problem. For the Bernoulli measurement model, the SEQ[m(2-D)] algorithm [2], the m-best version of the fastest sequential algorithm SEQ(2-D), is shown to be able to yield associations as good as the S-D assignment. The ghosting effect for a pair of sensors is no longer present due to the estimation of the signal emission time, which makes SEQ[m(2-D)] practical. For the Poisson measurement model, we discuss both uniform–Gaussian mixture (UGM) [5] and PPP modeling of the lists of measurements for the cardinality selection formulation. In the previous work on PPP [14], both the range and bearing measurements are assumed available and the initialization in the EM-based algorithm uses a finite set including target locations that are close to the truth. Since the range measurement and prior information for a "good" initialization is not always available in the real world, this paper considers bearing and time of arrival measurements and presents some measurement-driven initialization approaches for the EM-based algorithms. In the UGM formulation, the probability of detection (assumed not known) and the expected number of false alarms per sensor (which can be known or unknown) are incorporated into the mixture coefficients and the maximization step in the EM algorithm is developed such that the constraint that the resulting probability of detection is not larger than unity is always satisfied.

The remaining sections of this paper are organized as follows. Section II describes the problem of localizing an unknown number of transient emitters. Section III assumes a Bernoulli measurement model for each target, formulates an S-D assignment problem, and presents two assignment algorithms. Sections IV and V present the UGM and PPP formulations both of which assume a Poisson measurement model for each target. Simulation results are shown and analyzed in Section VI and the conclusions are drawn in Section VII. For the convenience of the reader, the list of notations used in this paper is given in Table I.

II. PROBLEM DESCRIPTION

Consider a scenario where there are N targets located in $\mathbb{R}^2$. The target locations (fixed) are denoted as

$$\mathbf{T} = (\mathrm{T}_1,\ \mathrm{T}_2,\ \ldots,\ \mathrm{T}_N) = \left(\begin{bmatrix} T_{x_1} \\ T_{y_1} \end{bmatrix},\ \begin{bmatrix} T_{x_2} \\ T_{y_2} \end{bmatrix},\ \ldots,\ \begin{bmatrix} T_{x_N} \\ T_{y_N} \end{bmatrix}\right) \tag{1}$$

and the emission times are denoted as

$$\mathbf{t}^{\mathrm{e}} = (t_1^{\mathrm{e}},\ t_2^{\mathrm{e}},\ \ldots,\ t_N^{\mathrm{e}}). \tag{2}$$

The number of targets and their locations are unknown quantities of interest, to be estimated. A total number of N_{s} stationary sensors with known locations at

$$\mathbf{S} = (\mathrm{S}_1,\ \mathrm{S}_2,\ \ldots,\ \mathrm{S}_{N_{\mathrm{s}}}) = \left(\begin{bmatrix} S_{x_1} \\ S_{y_1} \end{bmatrix},\ \begin{bmatrix} S_{x_2} \\ S_{y_2} \end{bmatrix},\ \ldots,\ \begin{bmatrix} S_{x_{N_{\mathrm{s}}}} \\ S_{y_{N_{\mathrm{s}}}} \end{bmatrix}\right) \tag{3}$$

are able to observe transient acoustic events that occurred at target locations at the emission times and

TABLE I
List of Notations

Notation	Definition
S	Dimension of the assignment problem
$\mathbf{T}$	Set of target position vectors
T_i	Position vector of target i
$\mathbf{t}_\mathrm{e}$	Set of emission times
t_i^e	Signal emission time of target i
S_ℓ	Position vector of sensor ℓ
n_ℓ	Number of measurements at sensor ℓ
$\mathbf{z}_{\ell j}$	jth measurement at sensor ℓ
N	Number of targets
N_s	Number of sensors
N_fa	Expected number of false alarms per sensor
T_0	The clutter
Φ	The range of the sensor field view
$\mathbf{Z}_\ell$	Augmented measurement list at sensor ℓ
$\mathbf{z}_{\ell 0}$	Dummy measurement at sensor ℓ
$Z_{j_1 j_2 \cdots j_{N_\mathrm{s}}}$	An N_s-tuple of measurements, one from each sensor
$c_{j_1 j_2 \cdots j_{N_\mathrm{s}}}$	Cost of associating $Z_{j_1 j_2 \cdots j_{N_\mathrm{s}}}$ with a target
$\rho_{j_1 j_2 \cdots j_{N_\mathrm{s}}}$	Binary variable denoting whether $Z_{j_1 j_2 \cdots j_{N_\mathrm{s}}}$ is an association in the final assignment
$p_{\mathrm{d}\ell}$	Detection probability for sensor ℓ
m	Number of top solutions to be kept in the SEQ[m(2-D)] assignment algorithm
$\mathbf{K}$	Set of all $\mathbf{k}_{\ell j}$
$\mathbf{k}_{\ell j}$	Association variable of $\mathbf{z}_{\ell j}$ in the UGM formulation
π_i	Mixing coefficient of the UGM
$\boldsymbol{\kappa}$	Set of all $\kappa_{\ell j}$
$\kappa_{\ell j}$	Association variable of $\mathbf{z}_{\ell j}$ in the PPP formulation

measure the bearings to these targets and the time of arrival of the observed acoustic signals. For events and measurements that are separated significantly in time, there is no data association ambiguity, so it is assumed that only measurements falling within a certain time window of interest need to be associated. Let n_ℓ denote the number of such measurements (one measurement is defined as a vector consisting of both a bearing and a time of arrival due to one acoustic signal in this context) obtained by the ℓth sensor within the time window.

The jth measurement (a direction of arrival and time of arrival) received by the ℓth sensor, if it corresponds to the event at t_i^e from the ith target, is

$$\mathbf{z}_{\ell j}(\mathrm{T}_i, t_i^\mathrm{e}) = \mathbf{h}_\ell\,(\mathrm{T}_i, t_i^\mathrm{e}) + \mathbf{w}_{\ell j}, \quad i = 1, \ldots, N; \quad \ell = 1, \ldots, N_\mathrm{s}; \quad j = 1, \ldots, n_\ell \tag{4}$$

where $\mathbf{w}_{\ell j}$ is a zero-mean white Gaussian measurement noise with known covariance matrix R_ℓ and

$$\mathbf{h}_\ell\,(\mathrm{T}_i, t_i^\mathrm{e}) = \begin{bmatrix}\theta_{\ell i} \\ t_{\ell i}\end{bmatrix} = \begin{bmatrix} \arctan\left[\frac{T_{y_i} - S_{y_\ell}}{T_{x_i} - S_{x_\ell}}\right] \\ t_i^\mathrm{e} + \dfrac{\sqrt{(T_{x_i} - S_{x_\ell})^2 + (T_{y_i} - S_{y_\ell})^2}}{c}\end{bmatrix} \tag{5}$$

where t_i^e is the unknown emission time of the acoustic signal from T_i and c is the known speed of sound.

To incorporate false alarms, we denote a clutter target (with index 0) as T_0. A false measurement detected by the ℓth sensor consists of a bearing θ_0, which is uniformly distributed in the field of view of the ℓth sensor, and its arrival time t_0, which is uniformly distributed in the interval $[0, W]$. The number of false alarms from each sensor is assumed to be a Poisson random variable[2] with mean

$$N_\mathrm{fa} = \lambda_\mathrm{fa}\Phi W \tag{6}$$

where Φ is the range of field of view and is assumed to be the same for each sensor and λ_fa can be interpreted as the temporal–spatial density.

The probability density function (pdf) of measurement j from sensor ℓ—the likelihood function [3] of the target location and its emission time based on the measurement[3]—is

$$p(\mathbf{z}_{\ell j}|\mathrm{T}_0) = p(\theta_0)p(t_0) = \frac{1}{\Phi W} \triangleq \Lambda(\mathrm{T}_0; \mathbf{z}_{\ell j}) \tag{7}$$

$$p(\mathbf{z}_{\ell j}|\mathrm{T}_i, t_i^\mathrm{e}) = |2\pi R_\ell|^{-\frac{1}{2}} \cdot \exp\left\{-\frac{1}{2}\left[\mathbf{z}_{\ell j} - \mathbf{h}_\ell(\mathrm{T}_i, t_i^\mathrm{e})\right]' R_\ell^{-1}\left[\mathbf{z}_{\ell j} - \mathbf{h}_\ell(\mathrm{T}_i, t_i^\mathrm{e})\right]\right\} \triangleq \Lambda(\mathrm{T}_i, t_i^\mathrm{e}; \mathbf{z}_{\ell j}), \qquad i = 1, \ldots, N \tag{8}$$

where (7) is the pdf of a measurement from the clutter (a false alarm) and (8) is the pdf of a measurement from a true target.

The problem is to estimate N and $\mathbf{T} = \{\mathrm{T}_i, i = 1, \ldots, N\}$ given the complete set of observations $\mathbf{Z} = \{\mathbf{z}_{\ell j}, \ell = 1, \ldots, N_\mathrm{s}; j = 1, \ldots, n_\ell\}$ in the presence of missed detections and false alarms and without the knowledge of the true data association.

III. THE S-D ASSIGNMENT ALGORITHM

A. Formulation

The S-D assignment problem formulation assumes a Bernoulli measurement model that the number of measurements from a real target received by a sensor is a Bernoulli random variable. Note that the number of false alarms is modeled as a Poisson random variable.

An augmented list of measurements at the ℓth sensor is defined as

$$\mathbf{Z}_\ell \triangleq \left\{\mathbf{z}_{\ell 0}, \ldots, \mathbf{z}_{\ell n_\ell}\right\} \tag{9}$$

where $\mathbf{z}_{\ell 0}$ is a dummy measurement[4] representing missed detections. An association of N_s measurements (N_s-tuple) consisting of one measurement from each augmented list will be denoted as

$$Z_{j_1 j_2 \ldots j_{N_\mathrm{s}}} = \left\{\mathbf{z}_{1 j_1}, \mathbf{z}_{2 j_2}, \ldots, \mathbf{z}_{N_\mathrm{s} j_{N_\mathrm{s}}}\right\} \tag{10}$$

where $j_\ell \in \{0, 1, \ldots, n_\ell\}$ represents the index of the measurement from the augmented list $\mathbf{Z}_\ell$, which is included in the association.[5]

[2]While for targets we consider two measurement models (Bernoulli and Poisson), for clutter only a Poisson model is considered.
[3]If the source is clutter, it has no emission time, only an arrival time.
[4]Please see [20, Fig. 2] for the illustration of dummy measurement and N_s-tuple.
[5]Recall that $j_\ell = 0$ represents the dummy measurement, so (10) need not contain N_s "real" measurements; i.e., missed detections are allowed in the association.

Assuming that the measurements in $Z_{j_1 j_2 \dots j_{N_s}}$ originated from the same target at the location T_i and emission time t_i^{e}, the cost of this association will be given by the (physically dimensionless) negative log-likelihood ratio

$$c_{j_1 j_2 \dots j_{N_s}} = -\ln \frac{\Lambda(\mathrm{T}_i, t_i^{\mathrm{e}}; Z_{j_1 j_2 \dots j_{N_s}})}{\Lambda(\mathrm{T}_0; Z_{j_1 j_2 \dots j_{N_s}})} \tag{11}$$

where the numerator is calculated based on (8) and the denominator (the likelihood that they are all false) is calculated using (7).

Assuming the measurements are (conditioned on the true target locations) independent across the sensors, i.e., uncorrelated measurement noises, the likelihood function that the measurements in $Z_{j_1 j_2 \dots j_{N_s}}$ originated from the same target at the location T_i and emission time t_i^{e} is

$$\Lambda(\mathrm{T}_i, t_i^{\mathrm{e}}; Z_{j_1 j_2 \dots j_{N_s}}) = \prod_{\ell=1}^{N_s} (1 - p_{\mathrm{d}\ell})^{1-u(j_\ell)} \cdot (p_{\mathrm{d}\ell}\, p(\mathbf{z}_{\ell j_\ell} | \mathrm{T}_i, t_i^{\mathrm{e}}))^{u(j_\ell)} \tag{12}$$

where $p_{\mathrm{d}\ell}$ is the probability of detection for the ℓth sensor (assumed the same for each real target) and the indicator function $u(j_\ell)$ is

$$u(j_\ell) \triangleq \begin{cases} 0, & \text{if } j_\ell = 0 \\ 1, & \text{otherwise} \end{cases}. \tag{13}$$

Since the target location T_i and the emission time t_i^{e} are unknown, we replace them by their ML estimates $\hat{\mathrm{T}}_i$ and $\hat{t}_i^{\mathrm{e}}$ that are obtained by maximizing (12), that is,

$$\hat{\mathrm{T}}_i, \hat{t}_i^{\mathrm{e}} = \arg\max_{\mathrm{T}_i, t_i^{\mathrm{e}}} \Lambda(\mathrm{T}_i, t_i^{\mathrm{e}}; Z_{j_1 j_2 \dots j_{N_s}}). \tag{14}$$

Therefore, (11) is modified to a *generalized* negative log-likelihood ratio given by

$$c_{j_1 j_2 \dots j_{N_s}} = -\ln \frac{\Lambda(\hat{\mathrm{T}}_i, \hat{t}_i^{\mathrm{e}}; Z_{j_1 j_2 \dots j_{N_s}})}{\Lambda(\mathrm{T}_0; Z_{j_1 j_2 \dots j_{N_s}})}. \tag{15}$$

The likelihood that all the measurements in $Z_{j_1 j_2 \dots j_{N_s}}$ are false alarms is

$$\Lambda(\mathrm{T}_0; Z_{j_1 j_2 \dots j_{N_s}}) = \prod_{\ell=1}^{N_s} \left(\frac{1}{\Phi W}\right)^{u(j_\ell)}. \tag{16}$$

The assignment problem is formulated as

$$\min_{\rho_{j_1 j_2 \cdots j_{N_s}}} \sum_{j_1=0}^{n_1} \sum_{j_2=0}^{n_2} \cdots \sum_{j_{N_s}=0}^{n_{N_s}} c_{j_1 j_2 \dots j_{N_s}} \rho_{j_1 j_2 \dots j_{N_s}} \tag{17}$$

subject to

$$\sum_{j_2=0}^{n_2} \sum_{j_3=0}^{n_3} \cdots \sum_{j_{N_s}=0}^{n_{N_s}} \rho_{j_1 j_2 \dots j_{N_s}} = 1, \quad j_1 = 1, 2, \dots, n_1 \tag{18}$$

$$\sum_{j_1=0}^{n_1} \sum_{j_3=0}^{n_3} \cdots \sum_{j_{N_s}=0}^{n_{N_s}} \rho_{j_1 j_2 \dots j_{N_s}} = 1, \quad j_2 = 1, 2, \dots, n_2 \tag{19}$$

$$\vdots \qquad\qquad \vdots$$

$$\sum_{j_1=0}^{n_1} \sum_{j_2=0}^{n_2} \cdots \sum_{j_{N_s-1}=0}^{n_{N_s-1}} \rho_{j_1 j_2 \dots j_{N_s}} = 1, \quad j_{N_s} = 1, 2, \dots, n_{N_s} \tag{20}$$

where $\rho_{j_1 j_2 \dots j_{N_s}} \in \{0, 1\}$ and $\rho_{j_1 j_2 \dots j_{N_s}} = 1(0)$ means $Z_{j_1 j_2 \dots j_{N_s}}$ is (not) an association in the final assignment.

Note that if $c_{j_1 j_2 \dots j_{N_s}} > 0$, then $Z_{j_1 j_2 \dots j_{N_s}}$ will not be an association in the final assignment since the overall cost will be smaller for the decision that all the real measurements in $Z_{j_1 j_2 \dots j_{N_s}}$ are false (cost = 0) than for the decision that they are from the same real target.

The ℓth constraint set in (18)–(20)

$$\sum_{j_1=0}^{n_1} \cdots \sum_{j_{\ell-1}=0}^{n_{\ell-1}} \sum_{j_{\ell+1}=0}^{n_{\ell+1}} \cdots \sum_{j_{N_s}=0}^{n_{N_s}} \rho_{j_1 j_2 \dots j_{N_s}} = 1, \quad j_\ell = 1, 2, \dots, n_\ell \tag{21}$$

enforces that each measurement (except the dummy) is associated with a single measurement from each other list, yielding a "target." Once the minimization problem (17) is solved, based on the assumption that each target is associated with one and only one measurement in each sensor list (including the dummy measurement), the number of associations will be equal to the number of targets (some will be real and some false). Associations with less than τ real measurements will be considered as from the clutter. The remaining associations will be deemed from real targets. The corresponding locations and emission times will be the ML estimates as obtained in (14).

B. The Optimization via Lagrangian Relaxation

The optimization problem (17) is NP hard when $N_s \geq 3$. One suboptimal algorithm is the Lagrangian relaxation-based *S*-D assignment algorithm as shown in Fig. 1, which solves the original problem as a series of relaxed 2-D subproblems. The rth ($r = N_s, N_s - 1, \dots, 3$) constraint set is successively relaxed and appended to the cost with Lagrange multipliers $\mathbf{u}_r$. At stage $r = 3$, one has a 2-D problem, which can be optimally[6] solved using the modified auction algorithm.

The constraint sets are then reimposed one at a time ($r = 3, 4, \dots, N_s$), and the corresponding Lagrange multipliers are updated to $\mathbf{u}_r^{\mathrm{new}}$; at each stage, the cost J_r of the resulting feasible solution is computed, until all constraint sets are met. The duality gap—difference between the cost J_2^* from the maximally relaxed problem

[6] Up to the rounding error, i.e., quasi-optimally.

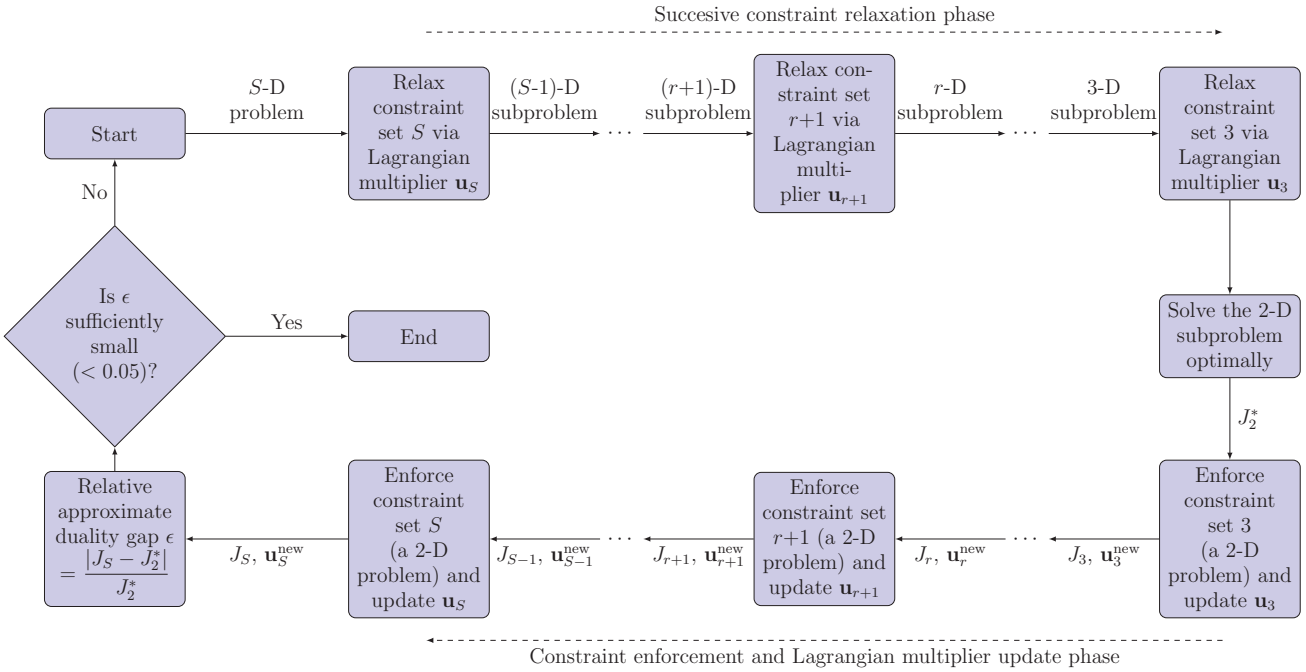


Fig. 1. Flow chart of the Lagrangian relaxation-based S-D assignment algorithm.

and J_S from the fully constrained one—is calculated and the iterations continue until this gap is small enough (usually 5% of the cost from the fully constrained one). See [8] and [21] for the detailed description.

C. The SEQ[m(2-D)] Assignment Algorithm

When $N_s = 2$, (17) becomes a 2-D assignment problem. By using Murty's ranking algorithm [18], one can find the top m best assignments instead of only the best one. The SEQ[m(2-D)] assignment algorithm can be described as follows. Initially, one selects two lists of measurements and obtains the top m best 2-D assignments with each assignment being a set of 2-tuples. Next, for each of these one continues to solve an m-best 2-D assignment, which yields a set of 3-tuples, between any one of the previous m association results and a third list of measurement. After this second step, one has m^2 assignments available, out of which the top m solutions in terms of the association cost will be selected for the next step. This procedure (shown in Fig. 2) is repeated until all the N_s lists of measurements are processed and the final assignment will be a set of N_s-tuples.

Note that it is possible to have an association $Z_{j_1 j_2 \cdots j_{N_s}}$ with $c_{j_1 j_2 \cdots j_{N_s}} > 0$ in the final assignment once the SEQ[m(2-D)] algorithm terminates. Such associations will be discarded before any association with less than τ real measurements is removed.

IV. UGM FORMULATION

If one assumes a Poisson measurement model that the number of measurements from a real target received by a sensor is a Poisson random variable, then one can model a list of measurements as realizations of a random variable with a UGM density [5] or a PPP. The UGM formulation is presented in this section and the PPP formulation will be presented in Section V.

A. Formulation

Assume (temporarily) the number of targets, N, is given. Since the association between a measurement $\mathbf{z}$ (without subscript, for simplicity) and the targets is unknown, we introduce an $(N+1)$-dimensional random binary-valued association vector

$$\mathbf{k} = [k_0,\ k_1,\ \ldots,\ k_N] \tag{22}$$

to indicate the target from which the measurement $\mathbf{z}$ originates. In our formulation, the random variable $\mathbf{z}$ is observed. The random variable $\mathbf{k}$ is not observed, thus is called a latent variable.[7] The entries k_i of the vector $\mathbf{k}$ satisfy the following conditions:

$$\sum_{i=0}^{N} k_i = 1 \tag{23}$$

$$k_i \in \{0,\ 1\} \qquad \forall\, i \tag{24}$$

that is, there are $N+1$ possible values for the vector $\mathbf{k}$. Let us use $\mathbf{e}_i$ to denote the $(N+1)$-dimensional vector with 1 in its ith entry and zeros elsewhere. The event $\{\mathbf{k} = \mathbf{e}_1\}$, which is the same as the event $\{k_0 = 1\}$, means that $\mathbf{z}$ is

[7]Latent variables are random variables whose values we do not observe or measure.

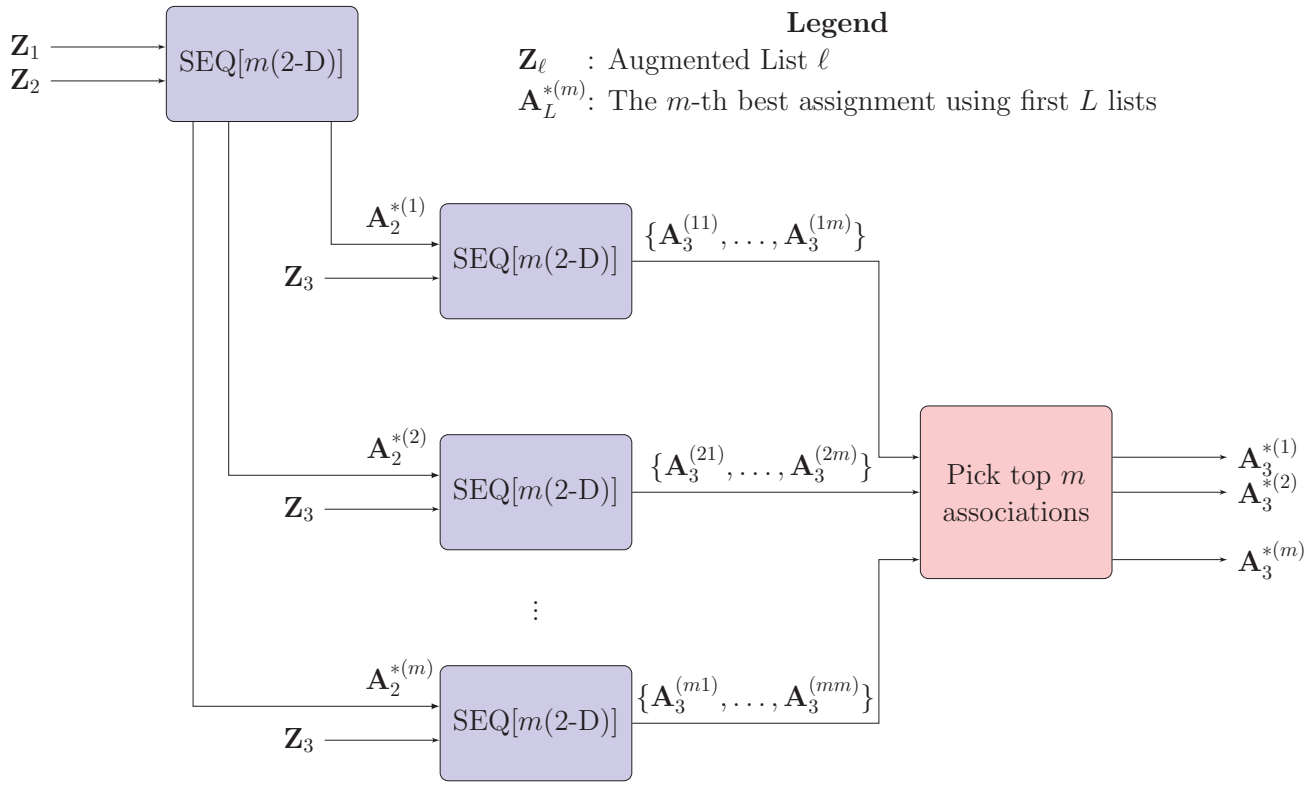


Fig. 2. Initial iteration of the SEQ[m(2-D)] assignment algorithm.

a clutter-originated measurement. The event $\{\mathbf{k} = \mathbf{e}_{i+1}\}$ with $i > 0$, which is the same as $\{k_i = 1\}$, means that the measurement $\mathbf{z}$ originated from the ith target.

The prior probability that $\mathbf{z}$ originated from the ith target given that the acoustic signal has been detected (assuming that a detected acoustic signal originates equally likely from all the targets) is

$$p(\mathbf{k} = \mathbf{e}_{i+1}) = p(k_i = 1) \triangleq \pi_i = \frac{p_{\mathrm{d}}(\mathrm{T}_i)}{\sum_{i=0}^{N} p_{\mathrm{d}}(\mathrm{T}_i)}, \quad i = 0, 1, \ldots, N \tag{25}$$

where $p_{\mathrm{d}}(\mathrm{T}_i)$ is the probability of detection for the real target i ($i \neq 0$) and is assumed to be the same at each sensor and

$$p_{\mathrm{d}}(\mathrm{T}_0) = N_{\mathrm{fa}}. \tag{26}$$

With abuse of notation, (26) is the expected number of false alarms at each sensor. The probabilities π_i, therefore, satisfy the following two conditions:

$$0 \leq \pi_i \leq 1 \tag{27}$$

$$\sum_{i=0}^{N} \pi_i = 1. \tag{28}$$

Because of (23) and (24), the prior probability in (25) can be equivalently expressed, in the form of a probability mass function, as

$$p(\mathbf{k} = \mathbf{e}_{i+1}) = p(k_i = 1) = \pi_i = \prod_{i=0}^{N} \pi_i^{k_i} \tag{29}$$

where the last equality holds because only the exponent k_i is equal to 1 while all other exponents are equal to 0, and thus do not affect the product.

From (7) and (8), the conditional pdf of a measurement $\mathbf{z}$ (without subscript, for simplicity) obtained by the ℓth sensor given that it is associated with the ith target is

$$p_\ell(\mathbf{z}|k_0 = 1, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \frac{1}{\Phi W} \tag{30}$$

$$p_\ell(\mathbf{z}|k_i = 1, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \mathcal{N}\left(\mathbf{z}; \mathbf{h}_\ell(\mathrm{T}_i, t_i^{\mathrm{e}}), R_\ell\right), \quad i = 1, \ldots, N. \tag{31}$$

For notational simplicity, let us denote[8]

$$g_{\ell i}(\mathbf{z}) = p_\ell(\mathbf{z}|k_i = 1, \mathbf{T}, \mathbf{t}^{\mathrm{e}}), \quad i = 0, 1, \ldots, N. \tag{32}$$

In a similar way as we derived (29), using (23) and (24) we have

$$p_\ell(\mathbf{z}|\mathbf{k} = \mathbf{e}_{i+1}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = p_\ell(\mathbf{z}|k_i = 1, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = g_{\ell i}(\mathbf{z}) = \prod_{i=0}^{N} \left(g_{\ell i}(\mathbf{z})\right)^{k_i}. \tag{33}$$

[8] $g_{\ell i}(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})$ will be used when the conditioning needs to be explicitly indicated.

The joint density of a measurement $\mathbf{z}$ from sensor ℓ and its association vector $\mathbf{k}$ is therefore

$$p_\ell(\mathbf{z}, \mathbf{k} = \mathbf{e}_{i+1}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = p_\ell(\mathbf{z}|\mathbf{k} = \mathbf{e}_{i+1}, \mathbf{T}, \mathbf{t}^{\mathrm{e}})p(\mathbf{k} = \mathbf{e}_{i+1}) = \prod_{i=0}^{N} (\pi_i\, g_{\ell i}(\mathbf{z}))^{k_i} = \pi_i\, g_{\ell i}(\mathbf{z}) \tag{34}$$

where the first equality holds because of the conditional probability definition, the second equality holds as a result of direct substitutions of $p(\mathbf{k} = \mathbf{e}_{i+1})$ from (29) and $p_\ell(\mathbf{z}|\mathbf{k} = \mathbf{e}_{i+1}, \mathbf{T}, \mathbf{t}^{\mathrm{e}})$ from (33), and the last equality holds because only the exponent k_i is equal to 1 and other exponents are zero. The marginal density of $\mathbf{z}$ is then obtained by summing the joint density over all the $N + 1$ values of $\mathbf{k}$ as

$$p_\ell(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \sum_{i=0}^{N} p_\ell(\mathbf{z}, \mathbf{k} = \mathbf{e}_{i+1}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \sum_{i=0}^{N} \pi_i\, g_{\ell i}(\mathbf{z}) \tag{35}$$

where the first equality holds because of the total probability theorem and the second equality holds as a result of substitution of $p_\ell(\mathbf{z}, \mathbf{k} = \mathbf{e}_{i+1}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})$ from (34). Therefore, the marginal density of one measurement is a mixture (termed as "uniform–Gaussian" mixture in this paper) of one uniform density and N Gaussian densities with the parameters π_i being the mixing coefficients. The conditional density of $\mathbf{k}$ given $\mathbf{z}$ is obtained using Bayes' theorem as

$$p_\ell(\mathbf{k}|\mathbf{z}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \frac{p_\ell(\mathbf{z}, \mathbf{k}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})}{p_\ell(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})} = \frac{\prod_{i=0}^{N} (\pi_i\, g_{\ell i}(\mathbf{z}))^{k_i}}{\sum_{i=0}^{N} \pi_i\, g_{\ell i}(\mathbf{z})} \tag{36}$$

which is equivalent to

$$P(k_i = 1|\mathbf{z}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \frac{\pi_i\, g_{\ell i}(\mathbf{z})}{\sum_{i=0}^{N} \pi_i\, g_{\ell i}(\mathbf{z})}. \tag{37}$$

Let (with $\mathbf{k}$ indexed as in (4))

$$\mathbf{K} = \{\mathbf{k}_{\ell j}, \ell = 1, 2, \ldots, N_{\mathrm{s}};\ j = 1, 2, \ldots, n_\ell\} \tag{38}$$

be the corresponding set of association vectors (or latent variables) for $\mathbf{Z}$ and

$$\boldsymbol{\pi} = \{\pi_i, i = 0,\ 1,\ \ldots,\ N\}. \tag{39}$$

From (34), the conditional independence of measurements across all the sensors yields the joint density of $\mathbf{Z}$ and $\mathbf{K}$

$$p(\mathbf{Z}, \mathbf{K}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \prod_{\ell=1}^{N_{\mathrm{s}}} \prod_{j=1}^{n_\ell} \prod_{i=0}^{N} (\pi_i\, g_{\ell i}(\mathbf{z}_{\ell j}))^{[k_{\ell j}]_i} \tag{40}$$

where $\left[k_{\ell j}\right]_i$ is the ith component of the association vector $\mathbf{k}_{\ell j}$. The marginal density of $\mathbf{Z}$ is obtained by summing the joint density (40) over all possible values of $\mathbf{K}$ as

$$p(\mathbf{Z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \prod_{\ell=1}^{N_{\mathrm{s}}} \prod_{j=1}^{n_\ell} \left(\sum_{i=0}^{N} \pi_i\, g_{\ell i}(\mathbf{z}_{\ell j}) \right) \tag{41}$$

and the posterior density (actually probability mass function (pmf) since $\mathbf{K}$ is discrete) of $\mathbf{K}$ conditioned on $\mathbf{Z}$ is

$$p(\mathbf{K}|\mathbf{Z}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \prod_{\ell=1}^{N_{\mathrm{s}}} \prod_{j=1}^{n_\ell} \frac{\prod_{i=0}^{N} (\pi_i g_{\ell i}(\mathbf{z}_{\ell j}))^{[k_{\ell j}]_i}}{\sum_{i=0}^{N} \pi_i\, g_{\ell i}(\mathbf{z}_{\ell j})}. \tag{42}$$

B. The EM Algorithm

We are interested in finding the ML estimates of $\mathbf{T}$ and $\mathbf{t}^{\mathrm{e}}$ that maximize $p(\mathbf{Z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})$ or $\ln p(\mathbf{Z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})$. However, it is difficult to obtain these estimates since the data association between $\mathbf{Z}$ and $\mathbf{T}$ is unknown; that is, $\mathbf{K}$ is not observed. Mathematically, setting the derivatives of $\ln p(\mathbf{Z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})$ with respect to $\mathbf{T}$ and $\mathbf{t}^{\mathrm{e}}$ does not lead to a closed-form solution, which suggests an iterative approach.

The EM algorithm [9] is a two-step iterative optimization technique to find the ML estimate from incomplete data. In this context, $\{\mathbf{Z}, \mathbf{K}\}$ are the complete data set and the observed data $\mathbf{Z}$ are the incomplete data available since the association variables in $\mathbf{K}$ are unknown.

Each iteration of the EM algorithm has an expectation step (E step) and a maximization step (M step). In the E step, we use temporary estimates of $\mathbf{T}$ and $\mathbf{t}^{\mathrm{e}}$ to find the posterior distribution in (42) to "learn" about $\mathbf{K}$. We then use this posterior distribution of $\mathbf{K}$ to find the expectation of the joint density of $\mathbf{Z}$ and $\mathbf{K}$ in (40). In the M step, we maximize the expectation obtained in the E step to obtain updated estimates of $\mathbf{T}$ and $\mathbf{t}^{\mathrm{e}}$ [5].

C. Optimization

1) Initialization: The EM algorithm is an iterative method. The first step is to initialize the parameters $\mathbf{T}$, $\mathbf{t}^{\mathrm{e}}$, and $\boldsymbol{\pi}$. Here, we assume that the mixing coefficients π_i are scalar quantities that need to be estimated along with T_i and t_i^{e}.

The EM algorithm guarantees that $p(\mathbf{Z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})$ increases at each iteration. However, a poor initialization can cause convergence to a local maximum as opposed to the global one. As shown later, because of the relationship between π_i and $p_{\mathrm{d}}(\mathrm{T}_i)$ in (25), the iterative procedures depend on whether N_{fa} is known. In either case, $p_{\mathrm{d}}(\mathrm{T}_i)$ is initialized to be 1 and the initial values of π_i will be calculated according to (25). In this paper, three initialization approaches for $\mathbf{T}$ and $\mathbf{t}^{\mathrm{e}}$ are considered and will be discussed in Section VI.D.

2) E Step: Let $\mathbf{T}^{(n-1)}$, $\mathbf{t}^{\mathrm{e},(n-1)}$, and $\boldsymbol{\pi}^{(n-1)}$ denote the estimates from the previous step. In the expectation step, we compute $p(\mathbf{K}|\mathbf{Z}, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ and evaluate the expected value of $\ln p(\mathbf{Z}, \mathbf{K}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})$ conditioned on $p(\mathbf{K}|\mathbf{Z}, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$, which is given by

$$\begin{aligned} &Q(\mathbf{T}, \mathbf{t}^{\mathrm{e}}\big|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \\ &\quad = \mathbb{E}[\ln(p(\mathbf{Z}, \mathbf{K}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}))|p(\mathbf{K}|\mathbf{Z}, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})] \\ &\quad = Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) + Q_\pi(\boldsymbol{\pi}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \end{aligned} \tag{43}$$

where

$$Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \sum_{i=0}^{N} \ln\left(g_{\ell i}(\mathbf{z}_{\ell j})\right) w_{\ell ji}^{(n-1)} \tag{44}$$

$$Q_\pi(\boldsymbol{\pi} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \sum_{i=0}^{N} \ln(\pi_i)\, w_{\ell ji}^{(n-1)} \tag{45}$$

where

$$w_{\ell ji}^{(n-1)} = \frac{\pi_i^{(n-1)} g_{\ell i}^{(n-1)}(\mathbf{z}_{\ell j})}{\sum_{i=0}^{N} \pi_i^{(n-1)} g_{\ell i}^{(n-1)}(\mathbf{z}_{\ell j})} \tag{46}$$

$$g_{\ell i}^{(n-1)}(\mathbf{z}_{\ell j}) = p_\ell(\mathbf{z}_{\ell j} | k_i = 1, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \tag{47}$$

$w_{\ell ji}^{(n-1)}$ is the posterior probability that the measurement $\mathbf{z}_{\ell j}$ originates from the ith target, given that the target locations are $\mathbf{T}^{(n-1)}$ and the emission times are $\mathbf{t}^{\mathrm{e},(n-1)}$.

3) M Step: In the maximization step, we maximize $Q(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ over all feasible $\mathbf{T}$, $\mathbf{t}^{\mathrm{e}}$, and $\boldsymbol{\pi}$. Inspection of (43) reveals that $Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ depends only on the locations $\mathbf{T}$ and $Q_\pi(\boldsymbol{\pi} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ depends only on detection probabilities through mixing coefficients. Therefore, maximization of $Q(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ can be done by maximizing $Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ and $Q_\pi(\boldsymbol{\pi} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ separately.

We define $Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ as

$$Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{i=0}^{N} Q_{\mathrm{T}_i}(\mathrm{T}_i | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \tag{48}$$

where

$$Q_{\mathrm{T}_i}(\mathrm{T}_i | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \ln\left(g_{\ell i}(\mathbf{z}_{\ell j})\right) w_{\ell ji}^{(n-1)}. \tag{49}$$

Note that $Q_{\mathrm{T}_0}(\mathrm{T}_0 | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ is a constant, and there is no functional relation between T_{i_1} and T_{i_2} for $i_1 \neq i_2$ Therefore, each target location T_i can be obtained separately by maximizing $Q_{\mathrm{T}_i}(\mathrm{T}_i | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$.

Next we maximize $Q_\pi(\boldsymbol{\pi} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ with respect to π_i, while accounting for the constraint that the mixing coefficients sum to 1. This can be achieved using a Lagrange multiplier λ and maximizing the following quantity:

$$Q_\pi^L(\boldsymbol{\pi} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \sum_{i=0}^{N} \ln(\pi_i)\, w_{\ell ji}^{(n-1)} + \lambda \left(\sum_{i=0}^{N} \pi_i - 1 \right) \tag{50}$$

which gives

$$\pi_i^{(n)} = \frac{\sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} w_{\ell ji}^{(n-1)}}{\sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \sum_{i=0}^{N} w_{\ell ji}^{(n-1)}}. \tag{51}$$

When N_{fa} is unknown, one can set $p_{\mathrm{d}}(\mathrm{T}_i)$ and N_{fa} based on (51) as follows:

$$j = \arg\max_{i, i \neq 0} \pi_i \tag{52}$$

$$p_{\mathrm{d}}(\mathrm{T}_i) = \frac{\pi_i}{\pi_j}, \quad i \neq j \tag{53}$$

$$N_{\mathrm{fa}} = \frac{\pi_0}{\pi_j} \tag{54}$$

which guarantees the constraints

$$p_{\mathrm{d}}(\mathrm{T}_i) \leq 1, \quad i > 0. \tag{55}$$

However, when N_{fa} is known, it is not always possible to find $p_{\mathrm{d}}(\mathrm{T}_i) \leq 1$ such that (51) holds. For instance, the following may not hold for π_0 from (51) and a given N_{fa}:

$$\pi_0 = \frac{N_{\mathrm{fa}}}{\sum_{i=1}^{N} p_{\mathrm{d}}(\mathrm{T}_i) + N_{\mathrm{fa}}} \geq \frac{N_{\mathrm{fa}}}{N + N_{\mathrm{fa}}}. \tag{56}$$

We need to maximize $Q_\pi(\boldsymbol{\pi} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ with respect to $p_{\mathrm{d}}(\mathrm{T}_i)$ subject to (55).

The Karush–Kuhn–Tucker (KKT) conditions [15] give rise to the following proposition (see Appendix A for proof):

Proposition 1 *Let*

$$\mathcal{S} = \left\{ i \,\middle|\, \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} w_{\ell ji}^{(n-1)} N_{\mathrm{fa}} > \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} w_{\ell j0}^{(n-1)} \right\} \tag{57}$$

which can be an empty set, and its cardinality is denoted by $|\mathcal{S}|$. The optimal values of $p_{\mathrm{d}}(\mathrm{T}_i)$ are given by

$$p_{\mathrm{d}}^{(n)}(\mathrm{T}_i) = \begin{cases} 1, & \text{if } i \in \mathcal{S} \\ \dfrac{\sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} w_{\ell ji}^{(n-1)} (|\mathcal{S}| + N_{\mathrm{fa}})}{\sum_{k>0, k \in \mathcal{S}} \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} w_{\ell jk}^{(n-1)} + \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} w_{\ell j0}^{(n-1)}}, & \text{if } i \notin \mathcal{S} \end{cases}. \tag{58}$$

Therefore, by (25)

$$\pi_i^{(n)} = \frac{p_{\mathrm{d}}^{(n)}(\mathrm{T}_i)}{\sum_{i=0}^{N} p_{\mathrm{d}}^{(n)}(\mathrm{T}_i)} \tag{59}$$

which can be verified to be identical to (51) when the set $\mathcal{S}$ is empty.

The EM algorithm is terminated when the likelihood function (41) converges, that is,

$$\left| \ln p\left(\mathbf{Z} | \mathbf{T}^{(n)}, \mathbf{t}^{\mathrm{e}(n)}\right) - \ln p\left(\mathbf{Z} \,\middle|\, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right) \right| \leq \epsilon \tag{60}$$

where ϵ is a small number (e.g., 10^{-3}).

One can use a fixed $\boldsymbol{\pi}$ throughout the EM iterations and skip the update process in (51) under the assumption

that both the detection probabilities and the false alarm density are known.

D. Use of the Information Criterion for Cardinality Selection

So far, we have assumed that the number of targets is given. Since the number of targets is unknown, we can use the above described procedure to estimate the parameters $\boldsymbol{\pi}$ and $\mathbf{T}$ given a specific cardinality, i.e., the number of targets N. Now we are faced with a cardinality selection problem, or a model selection problem where the dimensionality of a model is the number of targets. One of the most widely used criteria for model selection problems is the Bayesian information criterion (BIC) [22].

Let $M^k = \{\hat{\boldsymbol{\pi}}^k, \hat{\mathbf{T}}^k, \hat{N}^k\}$ denote the set of estimated parameters based on the kth cardinality. According to BIC, we choose the model for which the following is largest:

$$\ln p\left(\mathbf{Z}\,|M^k\right) - \frac{1}{2}d^k \ln(N_{\mathbf{z}}) \tag{61}$$

where from (41)

$$p\left(\mathbf{Z}\,|M^k\right) = \prod_{\ell=1}^{N_s}\prod_{j=1}^{n_\ell}\left(\sum_{i=0}^{\hat{N}^k}\hat{\pi}_i^k\, g_{\ell i}(\mathbf{z}_{\ell j}|\hat{\mathbf{T}}^k)\right) \tag{62}$$

and $N_{\mathbf{z}}$ is the total number of measurements across all the sensors; d^k is the total number of parameters to be estimated based on the kth cardinality. In our case, d^k is $4\hat{N}^k + 1$ ($2\hat{N}^k$ position coordinates for a problem in 2-D, $\hat{N}^k$ emission times, $\hat{N}^k + 1$ UGM coefficients including the expected number of false alarms) if it is assumed the detection probabilities and the false alarm density are unknown. If the detection probabilities and the false alarm density are assumed to be known, then d^k is $3\hat{N}^k$.

V. PPP MODEL

A. Formulation

Assume the number of targets, N, is given. Let $\mathbf{w} = \{w_i, i = 1, \ldots, N\}$, where

$$w_i = p_{\mathrm{d}}(\mathrm{T}_i) \tag{63}$$

are the detection probabilities. The number of measurements n_ℓ and $\{\mathbf{z}_{\ell j},\, j = 1, 2, \ldots, n_\ell\}$ obtained at the ℓth sensor are jointly modeled as a realization of a PPP. The measurement set at the ℓth sensor is denoted as

$$\boldsymbol{\psi}_\ell = \left\{n_\ell,\, \mathbf{z}_{\ell 1},\, \mathbf{z}_{\ell 2},\, \ldots,\, \mathbf{z}_{\ell n_\ell}\right\}. \tag{64}$$

In this case, the points $\mathbf{z}_{\ell j}$ occur in the space $\mathbb{S} = \{(\theta, t) : \theta \in [-\pi, \pi), t \in [0, W]\}$ and their order is irrelevant. The PPP is fully parameterized by its spatial intensity function

$$\mu_\ell(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \sum_{i=0}^{N} p_{\mathrm{d}}(\mathrm{T}_i)\, g_{\ell i}(\mathbf{z}) \tag{65}$$

where, similarly to (26),

$$p_{\mathrm{d}}\left(\mathrm{T}_0\right) = N_{\mathrm{fa}}. \tag{66}$$

The number of points in the PPP is a Poisson random variable with rate $\int_{\mathbb{S}} \mu_\ell(\mathbf{z})d\mathbf{z}$; that is, the probability mass function of n_ℓ is

$$p(n_\ell) = \frac{\left(\int_{\mathbb{S}} \mu_\ell(\mathbf{z})d\mathbf{z}\right)^{n_\ell}}{n_\ell\,!}\exp\left\{-\int_{\mathbb{S}}\mu_\ell(\mathbf{z})d\mathbf{z}\right\}. \tag{67}$$

The n_ℓ points are defined as independent and identically distributed (i.i.d.) samples of a random variable with probability density function

$$p(\mathbf{z}) = \frac{\mu_\ell(\mathbf{z})}{\int_{\mathbb{S}}\mu_\ell(\mathbf{z})d\mathbf{z}} = \frac{\sum_{i=0}^{N} p_{\mathrm{d}}(\mathrm{T}_i)\, g_{\ell i}(\mathbf{z})}{\sum_{i=0}^{N} p_{\mathrm{d}}(\mathrm{T}_i)}. \tag{68}$$

The joint pmf–pdf of $\boldsymbol{\psi}^\ell$ is

$$p\left(\boldsymbol{\psi}^\ell\right) = \exp\left(-\int_{\mathbb{S}}\mu_\ell\left(\mathbf{z}\,|\mathbf{T}, \mathbf{t}^{\mathrm{e}}\right)d\mathbf{z}\right)\prod_{j=1}^{n_\ell}\mu_\ell(\mathbf{z}_{\ell j}\,|\mathbf{T}, \mathbf{t}^{\mathrm{e}}). \tag{69}$$

The factorial term $n_\ell\,!$ in (67) is canceled out because there are $n_\ell\,!$ permutations of an ordered list of measurements.

Let $\boldsymbol{\Psi}$ denote the set of all measurement sets (from the N_{s} sensors), i.e.,

$$\boldsymbol{\Psi} = \left\{\boldsymbol{\psi}_1,\, \boldsymbol{\psi}_2,\, \ldots,\, \boldsymbol{\psi}_{N_{\mathrm{s}}}\right\}. \tag{70}$$

The independence of the N_{s} measurement sets yields

$$p\left(\boldsymbol{\Psi}\,|\mathbf{T}, \mathbf{t}^{\mathrm{e}}\right) = \prod_{\ell=1}^{N_{\mathrm{s}}} p\left(\boldsymbol{\psi}_\ell\,|\mathbf{T}, \mathbf{t}^{\mathrm{e}}\right). \tag{71}$$

Since the intensity function is a mixture of uniform or Gaussian pdf and the association is unknown, we model the latent association variables as conditionally independent random variables

$$\kappa_{\ell j} \in \{0, 1, 2, \ldots, N\} \tag{72}$$

that identify which component spawned the jth measurement in the ℓth sensor. Here, $\kappa_{\ell j} = 0$ indicates that the measurement is generated by the clutter. The set of latent variables for the ℓth sensor is denoted as

$$\boldsymbol{\kappa}_\ell = \{\kappa_{\ell 1}, \ldots, \kappa_{\ell n_\ell}\} \tag{73}$$

such that the full set is

$$\boldsymbol{\kappa} = \{\boldsymbol{\kappa}_1, \ldots, \boldsymbol{\kappa}_{N_{\mathrm{s}}}\}. \tag{74}$$

The latent association variables may be regarded as "marks" associated with each of the points in the PPP. Define a mark space

$$M \triangleq \{0, 1, 2, \ldots, N\}. \tag{75}$$

Now

$$\boldsymbol{\psi}_\ell^{\mathrm{M}} = \left\{n_\ell, (\mathbf{z}_{\ell 1}, \kappa_{\ell 1}), \ldots, (\mathbf{z}_{\ell n_\ell}, \kappa_{\ell n_\ell})\right\} \tag{76}$$

denotes a realization of the marked PPP for the ℓth sensor, where "M" indicates that the associations are known

("marked"). Based on the marking theorem [17], the intensity function of $\boldsymbol{\psi}_\ell^{\mathrm{M}}$ is

$$\mu_\ell^{\mathrm{M}}(\mathbf{z}, \kappa|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \mu_\ell(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) p_\ell(\kappa|\mathbf{z}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) \tag{77}$$

where $p_\ell(\kappa|\mathbf{z}, \mathbf{T}, \mathbf{t}^{\mathrm{e}})$ denotes the conditional probability of κ given $\mathbf{z}$ (without subscript here, for simplicity) with $\kappa = i$ indicating the probability of $\mathbf{z}$ originating from target i. Using the same reasoning as for the derivation of (25), we have the prior probability of κ

$$p_\ell(\kappa|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \frac{w_\kappa}{\sum_{i=0}^{N} w_i}. \tag{78}$$

Given that a point $\mathbf{z}$ in the PPP is associated with the κth mixture component, the conditional intensity becomes

$$\mu_\ell(\mathbf{z}|\kappa) = w_\kappa \, g_{\ell\kappa}(\mathbf{z}) \tag{79}$$

and the conditional density of $\mathbf{z}$ given κ is

$$p_\ell(\mathbf{z}|\kappa, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = g_{\ell\kappa}(\mathbf{z}). \tag{80}$$

Using Bayes' theorem

$$p_\ell(\kappa|\mathbf{z}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \frac{p_\ell(\mathbf{z}|\kappa, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) p_\ell(\kappa|\mathbf{T}, \mathbf{t}^{\mathrm{e}})}{p_\ell(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})} = \frac{w_\kappa g_{\ell\kappa}(\mathbf{z})}{\mu_\ell(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})}. \tag{81}$$

Substituting (81) into (77) yields

$$\mu_\ell^{\mathrm{M}}(\mathbf{z}, \kappa|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) = w_\kappa g_{\ell\kappa}(\mathbf{z}). \tag{82}$$

The joint probability density function of $\boldsymbol{\psi}_\ell^{\mathrm{M}}$ is, similarly to (69), given by

$$p\left(\boldsymbol{\psi}_\ell^{\mathrm{M}}\right) = \exp\left(\sum_{\kappa=0}^{N} - \int_{\mathbb{S}} \mu_\ell^{\mathrm{M}}(\mathbf{z}, \kappa|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) \, d\mathbf{z}\right) \cdot \prod_{j=1}^{n_\ell} \mu_\ell^{\mathrm{M}}(\mathbf{z}_{\ell j}, \kappa_{\ell j}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}). \tag{83}$$

Now let the complete data from (76) be

$$\boldsymbol{\Psi}^{\mathrm{M}} = \left\{\boldsymbol{\psi}_1^{\mathrm{M}}, \, \boldsymbol{\psi}_2^{\mathrm{M}}, \, \ldots, \, \boldsymbol{\psi}_{N_{\mathrm{s}}}^{\mathrm{M}}\right\}. \tag{84}$$

The conditional independence of the N_{s} measurement sets yields the pmf–pdf for the complete data

$$p\left(\boldsymbol{\Psi}^{\mathrm{M}}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}\right) = \exp\left(-N_{\mathrm{s}} \sum_{i=0}^{N} w_i\right) \cdot \prod_{\ell=1}^{N_{\mathrm{s}}} \prod_{j=1}^{n_\ell} w_{\kappa_{\ell j}} g_{\ell\kappa_{\ell j}}(\mathbf{z}_{\ell j}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) \tag{85}$$

where we have used the fact

$$\sum_{\kappa=0}^{N} \left(\int w_\kappa g_{\ell\kappa}(\mathbf{z}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}) \, d\mathbf{z}\right) = \sum_{i=0}^{N} w_i. \tag{86}$$

Dividing (85) by (71) leads to the density of the marks conditioned on the observed measurements and the unknown parameters

$$p(\boldsymbol{\kappa}|\mathbf{Z}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}) = \prod_{\ell=1}^{N_{\mathrm{s}}} \prod_{j=1}^{n_\ell} p_\ell\left(\kappa_{\ell j} \,|\, \mathbf{z}_{\ell j}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}\right) \tag{87}$$

where

$$p_\ell\left(\kappa_{\ell j} \,|\, \mathbf{z}_{\ell j}, \mathbf{T}, \mathbf{t}^{\mathrm{e}}\right) = \frac{w_{\kappa_{\ell j}} g_{\ell\kappa_{\ell j}}(\mathbf{z}_{\ell j}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})}{\mu_\ell(\mathbf{z}_{\ell j}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})}. \tag{88}$$

In this PPP formulation, the goal is find the ML estimate of $\mathbf{T}$ by maximizing (71), which can also be solved using the EM algorithm.

B. Optimization

1) Initialization: In this paper, three initialization approaches are considered and will be discussed in Section VI.D.

2) E Step: Let $\mathbf{w}^{(n-1)}$, $\mathbf{T}^{(n-1)}$, and $\mathbf{t}^{\mathrm{e},(n-1)}$ denote the estimates from the previous step. In the expectation step, we use them to find $p(\boldsymbol{\kappa}|\mathbf{Z}, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$ and compute the expected value of $p\left(\boldsymbol{\Psi}^{\mathrm{M}}|\mathbf{T}, \mathbf{t}^{\mathrm{e}}\right)$ conditioned on $p(\boldsymbol{\kappa}|\mathbf{Z}, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})$; that is, we evaluate

$$\begin{aligned} &Q(\mathbf{T}, \mathbf{t}^{\mathrm{e}}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \\ &\quad = \mathbb{E}\left[\ln\left(p(\boldsymbol{\Psi}^{\mathrm{M}}|\mathbf{T}, \mathbf{t}^{\mathrm{e}})\right) \left| p(\boldsymbol{\kappa}\,|\mathbf{Z}, \, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})\right.\right] \\ &\quad = Q_w(\mathbf{w}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) + Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \end{aligned} \tag{89}$$

where

$$Q_w(\mathbf{w}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = -N_{\mathrm{s}} \sum_{i=0}^{N} w_i + \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \sum_{i=0}^{N} \ln(w_i) \, \alpha_{\ell ji}^{(n-1)} \tag{90}$$

$$Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \sum_{i=0}^{N} \ln\left(g_{\ell i}(\mathbf{z}_{\ell j})\right) \alpha_{\ell ji}^{(n-1)} \tag{91}$$

and

$$\begin{aligned} \alpha_{\ell ji}^{(n-1)} &= p_\ell(\kappa_{\ell j} = i \,|\, \mathbf{z}_{\ell j}, \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \\ &= \frac{w_i^{(n-1)} g_{\ell i}^{(n-1)}(\mathbf{z}_{\ell j})}{\sum_{i=0}^{N} w_i^{(n-1)} g_{\ell i}^{(n-1)}(\mathbf{z}_{\ell j})}. \end{aligned} \tag{92}$$

The weight $\alpha_{\ell ji}^{(n-1)}$ is the probability that the point $\mathbf{z}_{\ell j}$ is generated by the ith target given $\mathbf{T}^{(n-1)}$ and $\mathbf{t}^{\mathrm{e},(n-1)}$.

3) M Step: The M step maximizes $Q\left(\mathbf{T}, \mathbf{t}^{\mathrm{e}}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right)$ over all feasible values for $\mathbf{T}$ and $\mathbf{t}^{\mathrm{e}}$. Inspection of (90) reveals that $Q_w\left(\mathbf{w}\,|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right)$ depends only on the values of $p_{\mathrm{d}}(T_i)$ because $w_i = p_{\mathrm{d}}(T_i)$ for $i = 1, \ldots, N$. Likewise, $Q_T\left(\mathbf{T}, \mathbf{t}^{\mathrm{e}}|\mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right)$ depends only on the target locations and emission times through $g_{\ell i}(\mathbf{z}_{\ell j})$.

Therefore, maximization of $Q\left(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right)$ is accomplished by maximizing $Q_w\left(\mathbf{w} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right)$ and $Q_T\left(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right)$ separately.

The value of $Q_T\left(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}\right)$ in (91) decomposes as

$$Q_T(\mathbf{T}, \mathbf{t}^{\mathrm{e}} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{i=0}^{N} Q_{T_i}(T_i | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) \tag{93}$$

where

$$Q_{T_i}(T_i | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)}) = \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \ln\left(g_{\ell i}\left(\mathbf{z}_{\ell j}\right)\right) \alpha_{\ell j i}^{(n-1)}. \tag{94}$$

For $i = 0$, $Q_{T_i}\left(T_i | \mathbf{T}^{(n-1)}\right)$ is constant with respect to T_i since the density is assumed known for the clutter. When $i \neq 0$, $g_{\ell i}\left(\mathbf{z}_{\ell j}\right)$ depends only on T_i through $\mathbf{h}_\ell(T_i, t_i^{\mathrm{e}})$; thus, $T_i^{(n)}$ is determined by maximizing (94) separately for each value of i.

The values of $w_i^{(n)}$ are determined by maximizing (90) given the fact that $p_{\mathrm{d}}(T_i) = w_i$ for $k = 1, \ldots, N$ and the assumption that they are scalar quantities. The detection probabilities are also constrained to less than or equal to 1. By setting up the Lagrange multipliers, it is easy to see that the KKT conditions are satisfied when

$$w_i^{(n)} = \min\left\{1, \frac{1}{N_{\mathrm{s}}} \sum_{\ell=1}^{N_{\mathrm{s}}} \sum_{j=1}^{n_\ell} \alpha_{\ell j i}^{(n-1)}\right\}. \tag{95}$$

The EM algorithm is terminated when the likelihood function (71) converges, that is,

$$|\ln p(\boldsymbol{\Psi} | \mathbf{T}^{(n)}, \mathbf{t}^{\mathrm{e}(n)}) - \ln p(\boldsymbol{\Psi} | \mathbf{T}^{(n-1)}, \mathbf{t}^{\mathrm{e},(n-1)})| \leq \epsilon. \tag{96}$$

One can use a fixed $\mathbf{w}$ throughout the EM iterations and skip the update process in (95) under the assumption that both the detection probabilities and the false alarm density are known.

C. Use of the Information Criterion for Cardinality Selection

The EM algorithm will eventually converge to $\hat{\mathbf{w}}^k$ and $\hat{\mathbf{T}}^k$ given the number of targets $\hat{N}^k$. Let the set $M_p^k = \{\hat{\mathbf{w}}^k, \hat{\mathbf{T}}^k, \hat{N}^k\}$ denote the estimation result based on the kth cardinality. The BIC selects the set M_p^k that minimizes

$$-2 \ln p\left(\boldsymbol{\Psi} \left| M_p^k\right.\right) + d \ln N_{\mathbf{z}} \tag{97}$$

where from (71)

$$p\left(\boldsymbol{\Psi} | M_p^k\right) = \exp\left(-N_{\mathrm{s}} \sum_{i=0}^{\hat{N}^k} w_i^k\right) \prod_{\ell=1}^{N_{\mathrm{s}}} \prod_{j=1}^{n_\ell} \sum_{i=0}^{\hat{N}^k} w_i^k \, g_{\ell i}(\mathbf{z}_{\ell j} | \hat{\mathbf{T}}^k) \tag{98}$$

and d is the total number of parameters to be estimated.

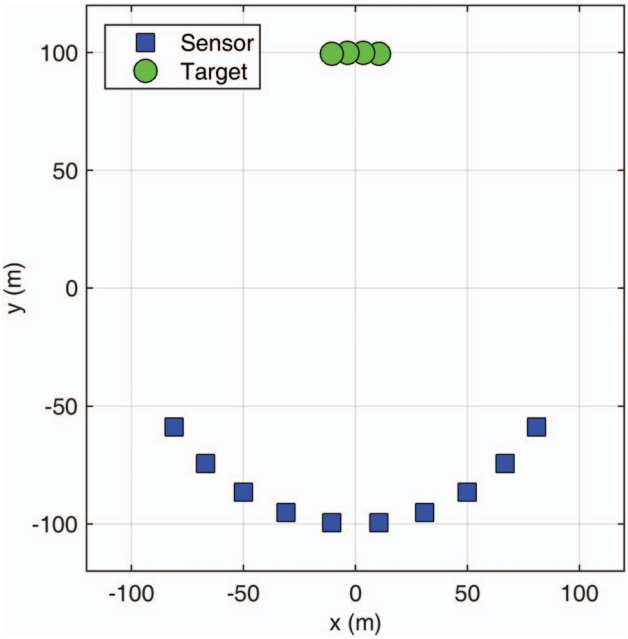


Fig. 3. Overhead view of a ten-sensor four-target scenario.

VI. SIMULATION RESULTS

A. Scenario

Assume there are four targets ($N = 4$). The emission times of the acoustic events at the target locations are 0.2, 0.25, 0.3, and 0.35 s, respectively. The speed of the acoustic signal is assumed to be 342 m/s. The measurement noise covariance matrix is

$$R_\ell = \begin{bmatrix} 7.6 \times 10^{-5} & 0 \\ 0 & 2.5 \times 10^{-5} \end{bmatrix} \tag{99}$$

i.e., the bearing standard deviation amounts to $\sqrt{76}$ mrad $= 0.5^{\mathrm{o}}$ and the time of arrival measurement standard deviation amounts to 5 ms, assumed to be the same for all targets. The time window W is chosen to be 1 s and the density of the false alarms is set to be 0.32 $\mathrm{s}^{-1}\mathrm{rad}^{-1}$ such that the expected number of false alarms at each sensor is 1. The field view of each sensor is from 0 to π. Fig. 3 shows one example using ten sensors to localize these four targets. In the simulation, the targets and the sensors are arranged in the way such that the angle between two LOS from two neighboring targets to any sensor is 2^{o}, which is four times the standard deviation of LOS measurement noise.

B. Performance Metrics

The performance metrics of interest for N_{MC} Monte Carlo runs include

1) φ_{over}: fraction of N_{MC} runs for which N (the number of targets) has been overestimated;
2) $\bar{M}_{\mathrm{over}}$: average magnitude of estimation error for N from $N_{\mathrm{MC}}\varphi_{\mathrm{over}}$ runs;

TABLE II
SEQ[m(2-D)] Assignment Performance Using Different m for Known p_{d}

p_{d}	0.7			0.8			0.9		
m	1	2	4	1	2	4	1	2	4
φ_{over}	1.6%	1.2%	0.9%	0.9%	0.8%	0.7%	0.6%	0.6%	0.4%
$\bar{M}_{\mathrm{over}}$	1	1	1	1	1	1	1	1	1
$T_{\mathrm{over}}^{\mathrm{RMSE}}$ (m)	2.86	5.13	3.01	1.97	1.63	1.72	2.19	2.19	1.98
$\theta_{\mathrm{over}}^{\mathrm{RMSE}}$ (°)	0.289	0.460	0.298	0.255	0.251	0.262	0.272	0.272	0.274
φ_{under}	3.2%	2.3%	2.1%	0.2%	0	0	0	0	0
$\bar{M}_{\mathrm{under}}$	1	1	1	1.5	N.A.	N.A.	N.A.	N.A.	N.A.
$T_{\mathrm{under}}^{\mathrm{RMSE}}$ (m)	2.58	2.69	2.69	4.50	N.A.	N.A.	N.A.	N.A.	N.A.
$\theta_{\mathrm{under}}^{\mathrm{RMSE}}$ (°)	0.296	0.299	0.298	0.411	N.A.	N.A.	N.A.	N.A.	N.A.
φ_{exact}	95.2%	96.5%	97.0%	98.9%	99.2%	99.3%	99.4%	99.4%	99.6%
$T_{\mathrm{exact}}^{\mathrm{RMSE}}$ (m)	2.92	2.81	2.76	3.23	2.59	2.59	1.99	1.96	1.96
$\theta_{\mathrm{exact}}^{\mathrm{RMSE}}$ (°)	0.329	0.320	0.313	0.329	0.282	0.282	0.239	0.236	0.236
$T_{\mathrm{all}}^{\mathrm{RMSE}}$ (m)	2.91	2.85	2.76	3.23	2.58	2.58	1.99	1.96	1.96
$\theta_{\mathrm{all}}^{\mathrm{RMSE}}$ (°)	0.328	0.322	0.313	0.328	0.282	0.282	0.239	0.236	0.236
t (s)	0.068	0.142	0.310	0.077	0.166	0.353	0.084	0.186	0.394

3) $T_{\mathrm{over}}^{\mathrm{RMSE}}$: root mean squared error (RMSE) of the target location estimate from $N_{\mathrm{MC}}\varphi_{\mathrm{over}}$ runs;
4) $\theta_{\mathrm{over}}^{\mathrm{RMSE}}$: RMSE of the bearing estimate from $N_{\mathrm{MC}}\varphi_{\mathrm{over}}$ runs;
5) φ_{under}: fraction of N_{MC} runs for which N has been underestimated;
6) $\bar{M}_{\mathrm{under}}$: average magnitude of estimation error for N from $N_{\mathrm{MC}}\varphi_{\mathrm{under}}$ runs;
7) $T_{\mathrm{under}}^{\mathrm{RMSE}}$: RMSE of the target location estimate from $N_{\mathrm{MC}}\varphi_{\mathrm{under}}$ runs;
8) $\theta_{\mathrm{under}}^{\mathrm{RMSE}}$: RMSE of the bearing estimate from $N_{\mathrm{MC}}\varphi_{\mathrm{under}}$ runs;
9) φ_{exact}: fraction of N_{MC} runs for which N has been correctly estimated;
10) $T_{\mathrm{exact}}^{\mathrm{RMSE}}$: RMSE of the target location estimate from $N_{\mathrm{MC}}\varphi_{\mathrm{exact}}$ runs;
11) $\theta_{\mathrm{exact}}^{\mathrm{RMSE}}$: RMSE of the bearing estimate from $N_{\mathrm{MC}}\varphi_{\mathrm{exact}}$ runs;
12) $T_{\mathrm{all}}^{\mathrm{RMSE}}$: RMSE of the target location estimate from all N_{MC} runs;
13) $\theta_{\mathrm{all}}^{\mathrm{RMSE}}$: RMSE of the bearing estimate from all N_{MC} runs.
14) t: average processing time in a single run.

For all the simulations in this paper, $N_{\mathrm{MC}} = 1000$ unless otherwise specified. In the overestimation cases, RMSE is calculated by mapping the best (yielding the minimum RMSE) subset of estimated targets to true targets. In the underestimation cases, RMSE is calculated by mapping the estimated targets to the best subset of true targets. The bearing estimate is examined here, because in some applications (for instance, shooter localization), bearing accuracy is more critical than location accuracy.

C. Assignment Algorithms

The multidimensional assignment problem (17), which is solved by the two assignment algorithms—the S-D assignment algorithm and the SEQ[m(2-D)] algorithm, assumes a Bernoulli measurement model that the number of measurements from a real target received by a sensor is a Bernoulli random variable whose parameter equal to the probability of detection p_{d}. For the evaluation of these two assignment algorithms in this section, the target measurements are generated according to this Bernoulli measurement model; specifically, one measurement from each target is generated for each sensor with a probability p_{d} or nothing with a probability $1 - p_{\mathrm{d}}$. The false alarms are generated for each sensor according to the Poisson model (6) and (7).

Note that the values of the probability of detection, p_{d}, and the expected number of false alarms, N_{fa}, are required to generate the target measurements. The assignment algorithms do not need to know the value of N_{fa} but need to know the value of p_{d}. However, the assignment algorithms are shown to be robust to incorrect p_{d} (see Table IV).

Table II shows the effects of the algorithm parameter m, the number of best assignments to be kept at each step, and p_{d}, probability of detection, on the performance of the SEQ[m(2-D)] assignment algorithm in a scenario using ten sensors to locate four targets. The true probability of detection is assumed to be known. Once the assignment algorithm is finished, any association with less than three ($\tau = 3$) real measurements is discarded. For each p_{d}, keeping more (larger m) top assignments at each 2-D step makes it more likely to find the best assignment; therefore, a larger m gives better estimates for the number of targets, the locations of the targets, and the directions to the targets (for counterfire). In fact, if the association between the measurements and

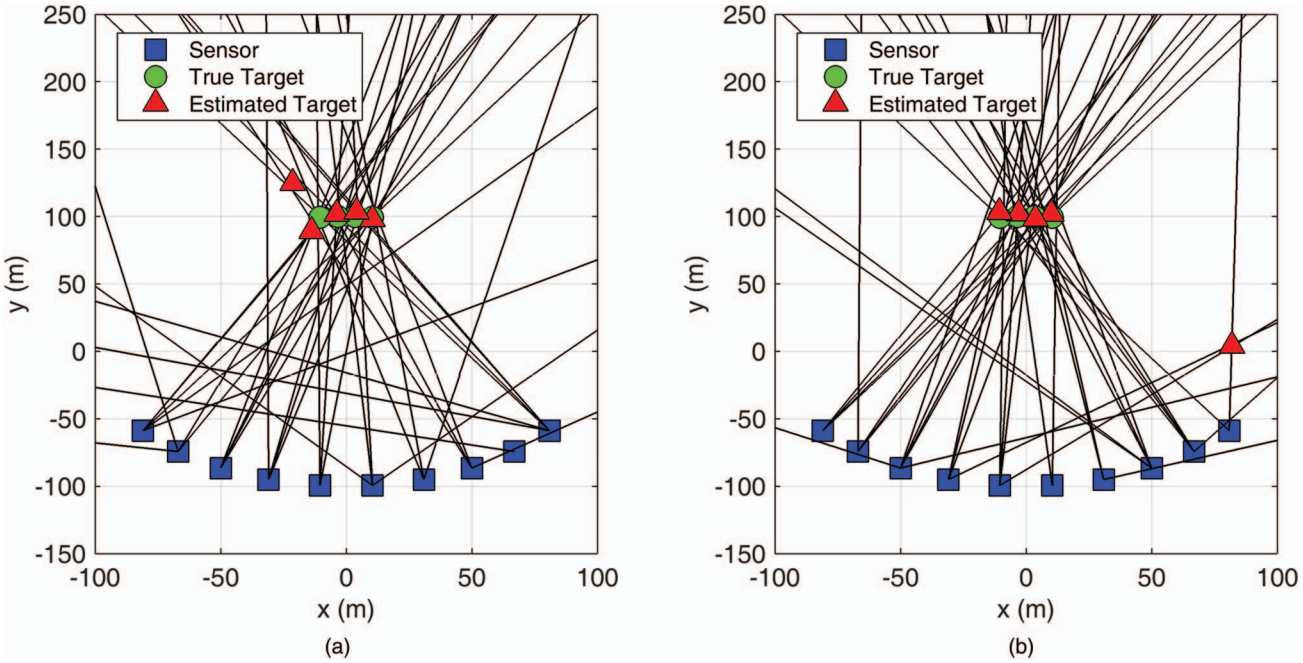


Fig. 4. Two cardinality overestimation situations.

the targets is known, the values of $\theta_{\text{all}}^{\text{RMSE}}$ would be 2.50, 2.15, and 1.91 m in the three scenarios with p_{d} varying from 0.7 to 0.9. The sequential algorithm with $m = 4$ yields very good association accuracy since the values of $\theta_{\text{all}}^{\text{RMSE}}$ are very close to these lower bounds. The performance gain is at the expense of a higher computational cost, which, however, is acceptable for real-time applications. For a fixed value of m, the algorithm performs better when the probability of detection increases. Lower p_{d} makes it more likely for the final association to have fewer real measurements, which could fail in the real threshold test; therefore, there are more cases when the number of targets is underestimated at $p_{\text{d}} = 0.7$. There are two situations when the number of targets is overestimated. The first one is "target splitting"; that is, one real target is perceived as two (or more) targets that are close to each other as shown in Fig. 4a, where the targets at location $(-10, 99)$ are split into the targets at locations $(-14, 89)$ and $(-21, 125)$. The second one is when three (or more) false measurements are perceived to be from a real target as shown in Fig. 4b at location $(82, 4)$. Both situations are more likely to occur at a lower p_{d}, so there are slightly fewer overestimation cases as p_{d} increases, as shown in Table II. When p_{d} is higher, there could be more real measurements available, which, if correctly associated, can lead to more accurate location and direction estimates. This is also at the expense of a slightly longer processing time.

Table III shows how the performance of the SEQ[m(2-D)] assignment algorithm with $m = 4$ varies with the number of sensors for three levels of known p_{d} in the scenario with four real targets. For the threshold test, any associations with less than three real measurements are discarded. For each p_{d}, it is observed that using a smaller number of sensors leads to more cardinality underestimation cases and fewer overestimation cases. This is so because the chance that only one sensor detects a real target is higher when a smaller number of sensors are used. On the other hand, a larger number of sensors in the presence of false alarms make it more likely to associate false measurements into a ghost like the situation in Fig. 4b. However, with a larger number of sensors, the decrease in the occurrence of overestimation cases is more significant than the increase in the underestimation cases. In addition, deploying more sensors could give rise to more measurements for a target, which in return generate more accurate location and direction estimates. In general, it is more beneficial to use a larger number of sensors.

The calculation of the cost (11) requires the knowledge of p_{d}. In practical scenarios, the actual value of p_{d} may not be available, in which case one has to use an estimated value of p_{d}. Table IV shows that the performance of the SEQ[m(2-D)] assignment algorithm is almost insensitive to the mismatch between the assumed value and the true value of p_{d} when they are close (up to 0.1 difference) in the scenario where ten sensors are used to locate four targets.

Fig. 5 shows the performance of the SEQ[m(2-D)] assignment algorithm for a wider range of true p_{d}. The quality of cardinality, location, and bearing estimates is almost independent of the assumed p_{d} value when the true p_{d} is from 0.6 to 0.9. When the true p_{d} is as low as 0.5, the quality of those estimates will vary with the assumed p_{d} value. A good location or bearing estimate requires at least three real measurements to be associated correctly; however, the probability that at least three out of ten sensors have detected the same target is only around

TABLE III
SEQ[m(2-D)] Assignment Performance Using Different N_s for Known p_d With $m = 4$

p_d	0.7			0.8			0.9		
N_s	6	8	10	6	8	10	6	8	10
φ_{over}	0.1%	0.4%	0.9%	0.2%	0.6%	0.7%	0.4%	0.5%	0.4%
$\bar{M}_{over}$	1	1	1	1	1	1	1	1	1
T_{over}^{RMSE} (m)	4.37	4.50	3.01	3.90	2.87	1.72	4.21	2.98	1.98
θ_{over}^{RMSE} (°)	0.453	0.386	0.298	0.361	0.294	0.262	0.302	0.289	0.274
φ_{under}	28.2%	7.8%	2.1%	7.8%	1%	0	0.4%	0	0
$\bar{M}_{under}$	1.12	1.01	1	1.04	1	N.A.	1	N.A.	N.A.
T_{under}^{RMSE} (m)	5.95	3.89	2.69	4.93	4.11	N.A.	4.64	N.A.	N.A.
θ_{under}^{RMSE} (°)	0.402	0.344	0.298	0.347	0.328	N.A.	0.330	N.A.	N.A.
φ_{exact}	71.7%	91.8%	97.0%	92%	98.4%	99.3%	99.2%	99.5%	99.6%
T_{exact}^{RMSE} (m)	7.26	4.37	2.76	5.66	3.41	2.59	4.42	2.76	1.96
θ_{exact}^{RMSE} (°)	1.03	0.442	0.313	0.369	0.310	0.282	0.314	0.266	0.236
T_{all}^{RMSE} (m)	6.99	4.35	2.76	5.62	3.42	2.58	4.42	2.76	1.96
θ_{all}^{RMSE} (°)	0.931	0.437	0.313	0.368	0.310	0.282	0.314	0.266	0.236
t (s)	0.132	0.212	0.310	0.158	0.274	0.353	0.183	0.277	0.394

TABLE IV
SEQ[m(2-D)] Assignment Performance Using Different Assumed Values for Unknown p_d

True p_d	0.7			0.8			0.9		
Assumed p_d	0.6	0.7	0.8	0.7	0.8	0.9	0.8	0.9	0.95
φ_{over}	1.0%	0.9%	0.8%	0.8%	0.7%	0.5%	0.5%	0.4%	0.2%
$\bar{M}_{over}$	1	1	1	1	1	1	1	1	1
T_{over}^{RMSE} (m)	3.12	3.01	2.14	2.40	1.72	1.71	1.97	1.98	2.12
θ_{over}^{RMSE} (°)	0.314	0.298	0.252	0.315	0.262	0.249	0.263	0.274	0.270
φ_{under}	1.7%	2.1%	2.2%	0	0	0.1%	0	0	0
$\bar{M}_{under}$	1	1	1	N.A.	N.A.	1	N.A.	N.A.	N.A.
T_{under}^{RMSE} (m)	2.80	2.69	2.79	N.A.	N.A.	2.91	N.A.	N.A.	N.A.
θ_{under}^{RMSE} (°)	0.311	0.298	0.317	N.A.	N.A.	0.325	N.A.	N.A.	N.A.
φ_{exact}	97.3%	97.0%	97.0%	99.2%	99.3%	99.4%	99.5%	99.6%	99.8%
T_{exact}^{RMSE} (m)	2.76	2.76	2.80	2.60	2.59	2.60	1.96	1.96	1.96
θ_{exact}^{RMSE} (°)	0.313	0.313	0.318	0.284	0.282	0.282	0.236	0.236	0.236
T_{all}^{RMSE} (m)	2.76	2.76	2.80	2.60	2.58	2.59	1.96	1.96	1.96
θ_{all}^{RMSE} (°)	0.313	0.313	0.317	0.284	0.282	0.282	0.236	0.236	0.236
t (s)	0.307	0.310	0.316	0.353	0.353	0.350	0.394	0.394	0.392

0.80 at true $p_d = 0.5$. In other words, at true $p_d = 0.5$ in about 200 simulation runs either there is at least one target missing or there is at least a false association, which could cause very different performances at different assumed p_d values. It gets worse at lower p_d values such as 0.4 and 0.3. Therefore, given a fixed number of sensors, there is a lower bound on the true p_d, below which it is very difficult to achieve good location and bearing estimates.

Fig. 6 shows the dependence of the performance of the SEQ[m(2-D)] assignment algorithm on the expected number of false alarms per sensor when ten sensors are used to localize four targets with known p_d values at 0.7, 0.8, and 0.9. In the final assignment, the least number of real measurements required to be associated with a real target is 3. As expected, the performance gets worse when the false alarm rate is higher. When the true p_d is 0.8 or 0.9, the localization results are reliable even for $N_{fa} = 4$ (the total expected number of false alarms is larger than the total expected number of real measurements). When the expected number of false alarms per sensor is very large ($N_{fa} = 8$), there are more false targets in the final assignment, leading to a worse performance. Setting a higher number for the required minimum number of real measurements reduces the number of false targets but will also miss real targets in a scenario, especially for a low p_d value.

Fig. 7a shows the processing time (averaged over 100 Monte Carlo runs) of the S-D assignment algorithm and the SEQ[m(2-D)] ($m = 4$) assignment algorithm in

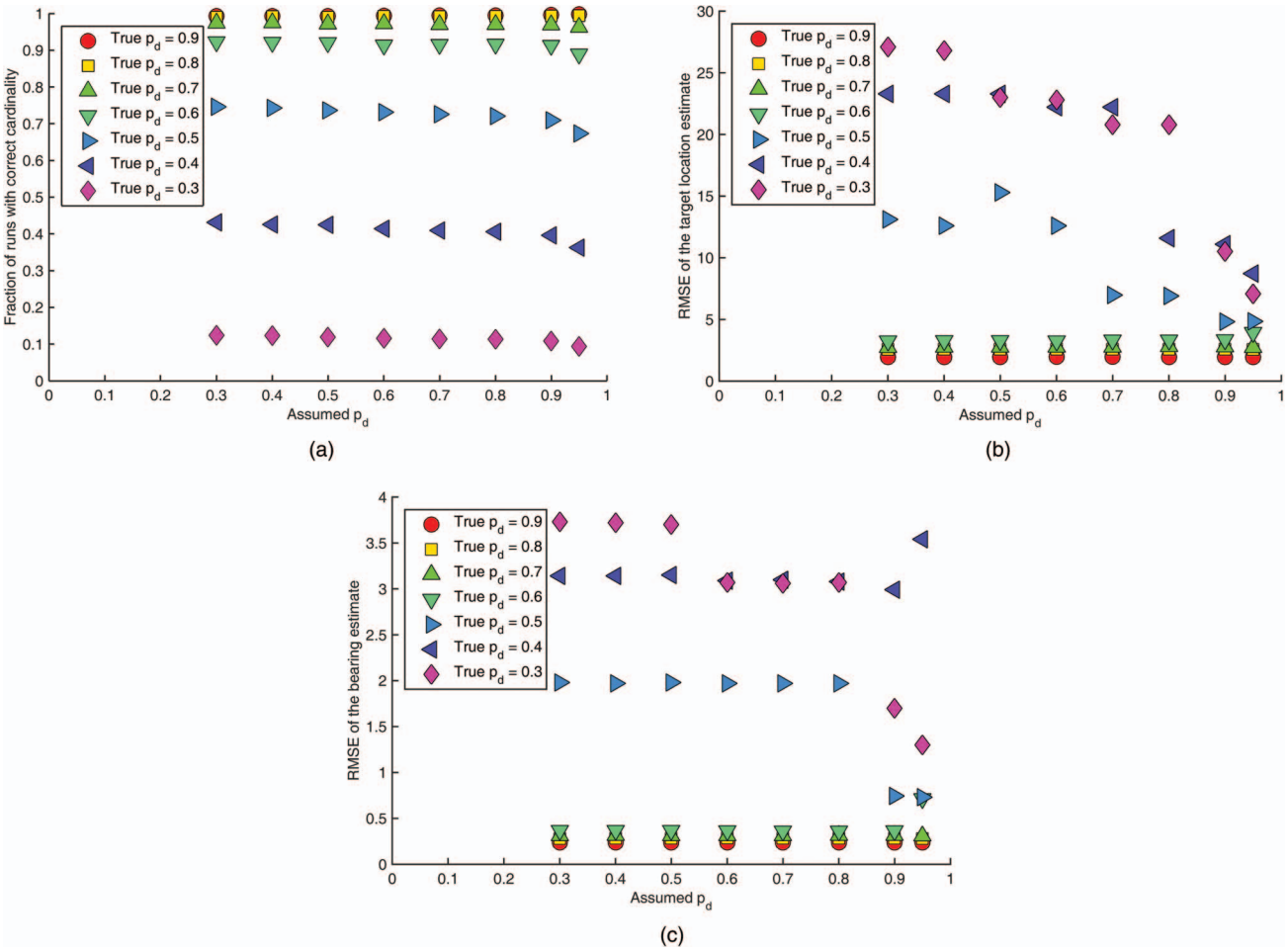


Fig. 5. The performance (in terms of φ_{exact}, $T_{\text{all}}^{\text{RMSE}}$, and $\theta_{\text{all}}^{\text{RMSE}}$) of the SEQ[$m$(2-D)] assignment algorithm using different assumed values of unknown p_{d} for true p_{d} values ranging from 0.3 to 0.9.

scenarios with the probability of detection $p_{\text{d}} = 0.9$ and the expected number of false alarms (per sensor) $N_{\text{fa}} = 1$. When the number of targets, N, is 4, the processing time using the S-D assignment algorithm with seven sensors is around 20 s, which is too long as far as real-time applications are concerned. This is due to the fact that the processing time of the S-D assignment algorithm scales exponentially with the number of sensors while the sequential processing time scales quadratically with the number of sensors in the worst case. In terms of the localization performance, the S-D assignment algorithm is also shown to be inferior to the sequential m-best assignment algorithm.

We must note that both algorithms are very different suboptimal solutions to the problem (17). The sequential m-best assignment algorithm makes efficient use of the modified auction algorithm in a sequential manner. As a greedy algorithm, it solves a locally optimal solution based on two lists of measurements followed by considering one more list of measurements at a time until a final solution is obtained for all lists of measurements. It uses a suitably large value of the parameter m with the hope that the optimal solution is kept in the search space at all times and the final solution is close to optimal.

In contrast, the S-D assignment algorithm is a much more sophisticated, iterative technique as shown in Fig. 1. However, it was shown in [8] that the S-D assignment algorithm could be underperforming in challenging localization scenarios where the association graph is strongly connected and the number of candidate associations is huge. In a scenario with $p_{\text{d}} = 0.9$ where five targets and seven sensors are placed such that the worst angular intertarget separations as seen by the seven sensors are 2, 3.5, 5, 6.5, 5, 3.5, and 2 standard deviations of the bearing measurement noise, the association accuracy of the S-D assignment algorithm is shown to be 74%. In our simulation scenario, the worst angular intertarget separation is four standard deviations for every sensor, which poses a similar challenging situation in terms of the density of the association graph (or the number of candidate associations). Therefore, the somewhat inferior localization performance of the S-D assignment algorithm at $p_{\text{d}} = 0.9$ is not unexpected.

As suggested in [8], the algorithmic parameters are selected such that the algorithm is terminated if the relative approximate duality gap is less than 5% or the number of iterations exceeds 100. Although the chance is very small, it is still possible that the algorithm already

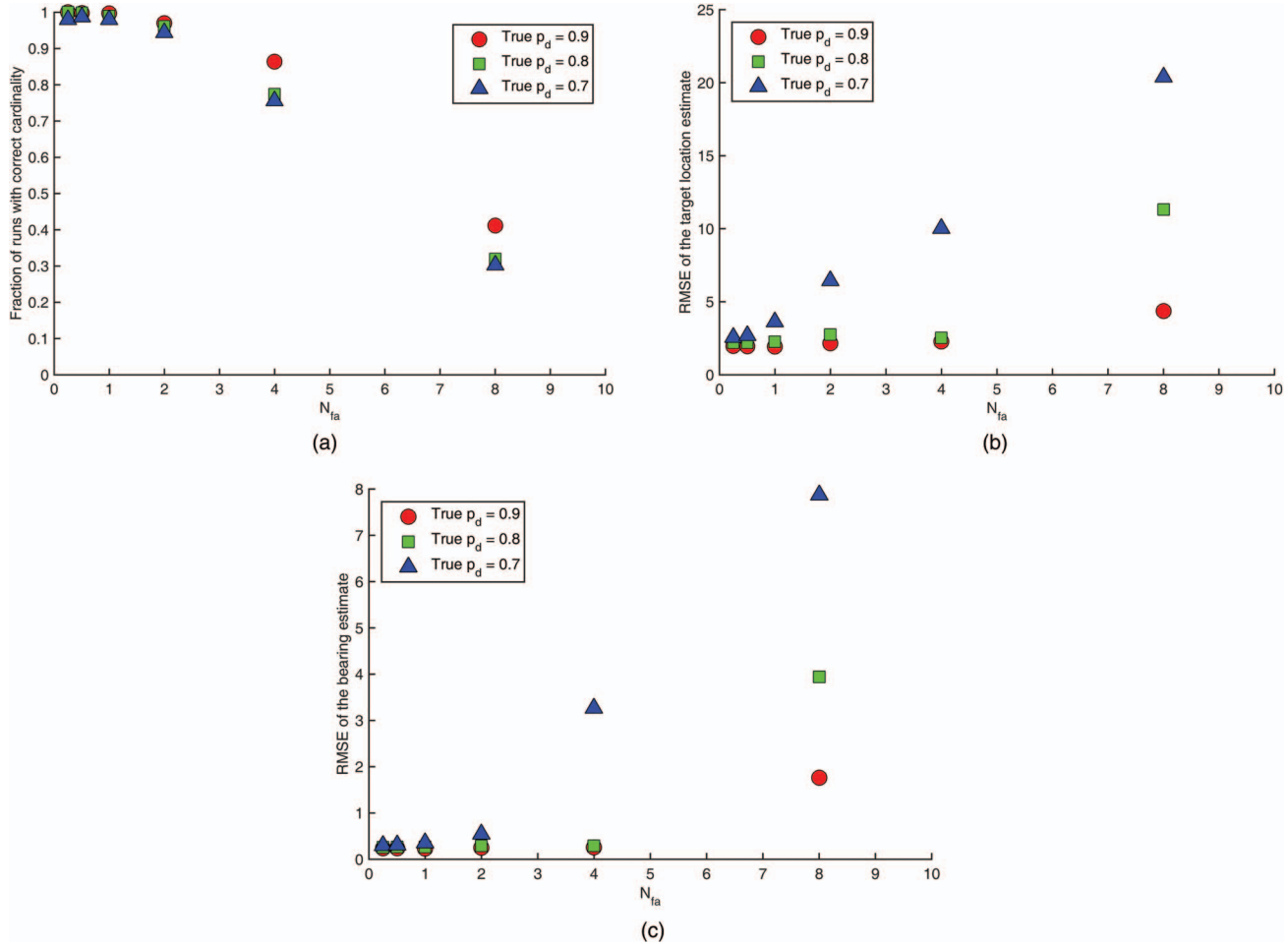


Fig. 6. The performance (in terms of φ_{exact}, $T_{\text{all}}^{\text{RMSE}}$, and $\theta_{\text{all}}^{\text{RMSE}}$) of the SEQ[$m$(2-D)] assignment algorithm in scenarios with different known expected number of false alarms (0.25, 0.5, 1, 2, 4, and 8) for known p_{d} values at 0.7, 0.8, and 0.9.

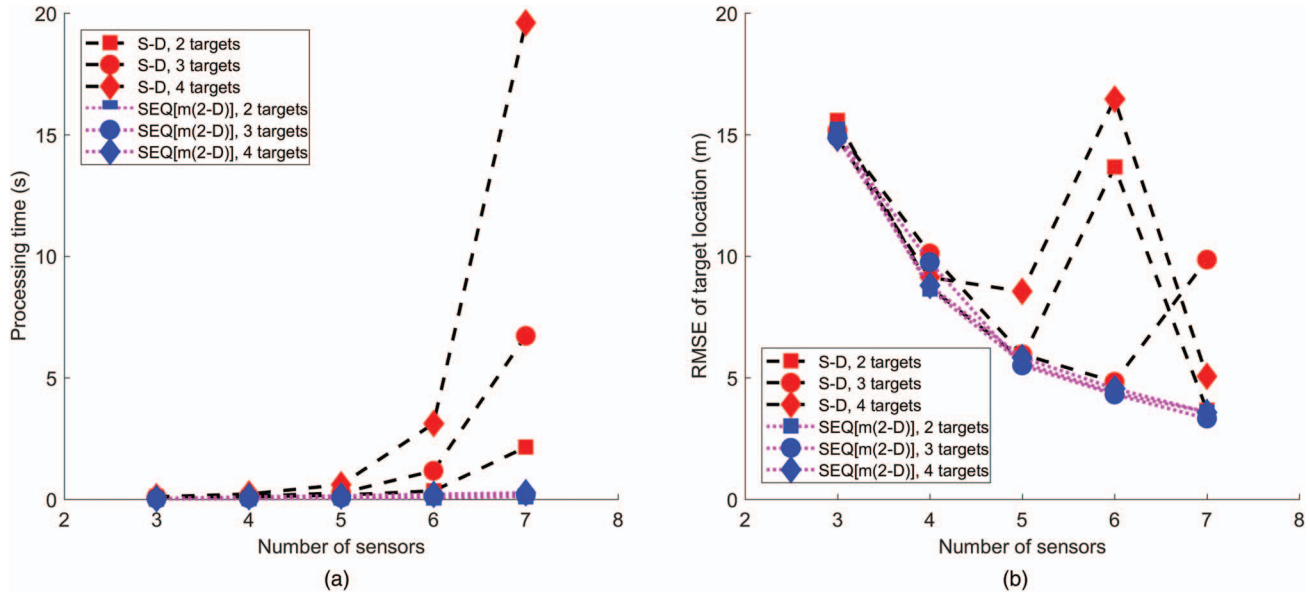


Fig. 7. Performance comparison between the S-D assignment algorithm and the SEQ[m(2-D)] assignment algorithm for $p_{\text{d}} = 0.9$ and $N_{\text{fa}} = 1$.

TABLE V
Comparison Between S-D Assignment and Sequential m-Best (SmB) 2-D Assignment Performance for $p_d = 0.9$ (Assumed Unknown)

	S-D			SmB($m=4$)		
Assumed p_d	0.8	0.9	0.99	0.8	0.9	0.99
φ_{over}	1.8%	2.0%	3.3%	0.4%	0.4%	0.2%
$\bar{M}_{over}$	1	1	1	1	1	1
T^{RMSE}_{over} (m)	14.88	14.24	6.93	3.68	3.55	3.64
θ^{RMSE}_{over} (°)	0.714	0.640	0.459	0.309	0.306	0.241
φ_{under}	9.7%	1.9%	0.8%	0.4%	0.3%	0.3%
$\bar{M}_{under}$	1.06	1	1	1	1	1
T^{RMSE}_{under} (m)	30.67	18.14	8.72	3.08	2.91	2.91
θ^{RMSE}_{under} (°)	1.17	0.792	0.518	0.272	0.286	0.286
φ_{exact}	88.5%	96.1%	95.9%	99.2%	99.3%	99.5%
T^{RMSE}_{exact} (m)	10.47	9.44	4.87	4.91	4.42	4.43
θ^{RMSE}_{exact} (°)	1.14	0.832	0.348	0.325	0.318	0.319
T^{RMSE}_{all} (m)	13.14	9.74	4.98	4.90	4.42	4.42
θ^{RMSE}_{all} (°)	1.14	0.828	0.354	0.325	0.318	0.319
t (s)	0.549	0.568	0.541	0.166	0.162	0.160

TABLE VI
Comparison Between S-D Assignment and Sequential m-Best (SmB) 2-D Assignment Performance for $p_d = 0.8$ (Assumed Unknown)

	S-D			SmB($m=4$)		
Assumed p_d	0.7	0.8	0.9	0.7	0.8	0.9
φ_{over}	0.7%	0.6%	0.9%	0.4%	0.4%	0.4%
$\bar{M}_{over}$	1	1	1	1	1	1
T^{RMSE}_{over} (m)	33.13	4.21	15.96	6.74	6.74	6.74
θ^{RMSE}_{over} (°)	0.882	0.302	0.829	0.421	0.421	0.421
φ_{under}	31.1%	25.4%	11.8%	7.6%	7.5%	7.5%
$\bar{M}_{under}$	1.24	1.23	1.03	1.03	1.03	1.03
T^{RMSE}_{under} (m)	31.40	30.12	20.52	5.46	5.49	5.88
θ^{RMSE}_{under} (°)	1.53	1.58	0.989	0.366	0.367	0.380
φ_{exact}	68.2%	74.0%	87.3%	92%	92.1%	92.1%
T^{RMSE}_{exact} (m)	12.69	12.86	18.51	5.11	5.11	5.11
θ^{RMSE}_{exact} (°)	1.10	1.06	1.10	0.359	0.359	0.359
T^{RMSE}_{all} (m)	19.08	17.49	18.67	5.14	5.14	5.16
θ^{RMSE}_{all} (°)	1.22	1.18	1.09	0.360	0.360	0.361
t (s)	0.415	0.420	0.413	0.142	0.141	0.150

terminates at the 100th iteration and the relative approximate duality gap has not been reduced to 5%. Even if the relative approximate duality gap is less than 5% long before the 100th iteration, it is still possible that the algorithm stops at a local minimum of the objective function (17) and, although it has a similar association cost to the optimal solution, it yields different target locations. In addition, the more false alarms the sensors detect, the more likely that the algorithm terminates at a local minimum.

As the number of sensors increases, the association graph becomes more dense and the number of candidate associations explodes combinatorially and it becomes more difficult for the S-D assignment algorithm to solve the association problem. Therefore, it is not practical to apply the S-D assignment algorithm directly[9] when the number of sensors is large. We suggest the use of the sequential m-best assignment algorithm in applications with a large number of sensors.

For additional comparison, the performance of both the S-D assignment and the sequential m-best assignment algorithms for different p_d in a scenario with four targets and six sensors is listed in Tables V–VII.

D. EM-Based Algorithms

In Sections IV and V, we have considered the Poisson measurement model for each target, which leads to either the UGM or the PPP formulation. Both are solved using the EM algorithm. For the evaluation of these two EM-based algorithms in this section, the target measurements are generated according to this Poisson measurement model; specifically, if a sensor has a certain p_d, the number of measurements originated from a target is a Poisson random variable with parameter p_d. The clutter follows a Poisson model with parameter N_{fa}.

Note that the values of the probability of detection, p_d, and the expected number of false alarms, N_{fa}, are required to generate the target measurements. However, these EM-based algorithms do not need to know the values of N_{fa} and p_d. They adapt to these values by "learning them."

The EM-based algorithm starts with an initialization, which determines whether the objective function can converge to the global maximum or a local maximum.

TABLE VII
Comparison Between S-D Assignment and Sequential m-Best (SmB) 2-D Assignment Performance for $p_d = 0.7$ (Assumed Unknown)

	S-D			SmB($m=4$)		
Assumed p_d	0.6	0.7	0.8	0.6	0.7	0.8
φ_{over}	0.5%	0.3%	0.2%	0	0	0
$\bar{M}_{over}$	1	1	1	N.A.	N.A.	N.A.
T^{RMSE}_{over} (m)	5.93	6.30	6.80	N.A.	N.A.	N.A.
θ^{RMSE}_{over} (°)	0.360	0.382	0.395	N.A.	N.A.	N.A.
φ_{under}	62.6%	57.4%	52.2%	27.5%	27.5%	27.5%
$\bar{M}_{under}$	1.45	1.46	1.39	1.14	1.14	1.14
T^{RMSE}_{under} (m)	35.39	35.24	34.48	6.93	6.97	6.59
θ^{RMSE}_{under} (°)	2.14	2.19	2.20	0.398	0.400	0.402
φ_{exact}	36.9%	42.3%	47.6%	72.5%	72.5%	72.5%
T^{RMSE}_{exact} (m)	14.78	14.90	14.92	6.96	6.96	6.96
θ^{RMSE}_{exact} (°)	0.941	0.911	0.910	0.860	0.860	0.860
T^{RMSE}_{all} (m)	27.41	26.29	24.99	6.95	6.96	6.88
θ^{RMSE}_{all} (°)	1.67	1.63	1.58	0.784	0.785	0.785
t (s)	0.318	0.318	0.322	0.131	0.127	0.131

[9]One possible practice is to use the S-D assignment algorithm on a subset of sensors followed by sequential processing as in [23].

TABLE VIII
UGM/EM and PPP/EM Performance Using Different Initialization ("I:") Approaches for Unknown $p_d = 0.7$

	I: Truth		I: Clustering		I: SmB($m = 1$)		I: SmB($m = 2$)	
	UGM/EM	PPP/EM	UGM/EM	PPP/EM	UGM/EM	PPP/EM	UGM/EM	PPP/EM
φ_{over}	0.1%	0	49.7%	50.5%	19.4%	19.3%	19.8%	19.7%
$\bar{M}_{\text{over}}$	1	N.A.	1.57	1.51	1.14	1.14	1.15	1.15
$T_{\text{over}}^{\text{RMSE}}$ (m)	4.96	N.A.	36.69	28.51	3.46	3.47	3.34	3.35
$\theta_{\text{over}}^{\text{RMSE}}$ (°)	0.535	N.A.	2.23	2.37	0.493	0.494	0.449	0.451
φ_{under}	0.6%	0.7%	29.7%	30.5%	23.4%	23.4%	23%	22.9%
$\bar{M}_{\text{under}}$	1	1	1.52	1.48	1.10	1.10	1.09	1.08
$T_{\text{under}}^{\text{RMSE}}$ (m)	3.06	2.57	34.21	31.8	8.28	8.30	2.88	3.04
$\theta_{\text{under}}^{\text{RMSE}}$ (°)	0.403	0.352	3.66	3.30	0.442	0.449	0.347	0.357
φ_{exact}	99.3%	99.3%	20.6%	19%	57.2%	57.3%	57.2%	57.4%
$T_{\text{exact}}^{\text{RMSE}}$ (m)	3.05	3.01	70.95	52.96	5.56	5.54	5.53	5.55
$\theta_{\text{exact}}^{\text{RMSE}}$ (°)	0.351	0.347	5.38	5.27	0.615	0.615	0.615	0.620
$T_{\text{all}}^{\text{RMSE}}$ (m)	3.05	3.01	46.37	35.69	5.81	5.81	4.74	4.78
$\theta_{\text{all}}^{\text{RMSE}}$ (°)	0.351	0.347	3.49	3.38	0.563	0.564	0.543	0.548
t (s)	2.52	2.40	4.72	4.32	1.23	1.21	1.31	1.31

Three initialization approaches are considered in this paper.

The first approach is to initialize the target locations and the emission times using their true values. This initialization approach works well as shown later; however, it critically depends on the truth, which is not available in the real world. Nevertheless, it provides a benchmark on how well the EM-based algorithms can perform. Since the number of targets N is unknown, one needs to evaluate a range of values for N and the algorithm selects the best N based on BIC. Such possible values for N can be selected based on the number of measurements obtained at each sensor; five values (2–6) are chosen for the four-target scenario considered here. When the evaluated number of targets is less than the true value, a subset of the true targets is used for initialization. When the evaluated number of targets is more than the true value, auxiliary targets in addition to the true ones are used for initialization.

The second approach is based on the k-means clustering algorithm. Any two bearing (or LOS) measurements from two different sensors can lead to a potential target. In the absence of measurement noise, the LOS measurements coming from the same target intersect at a single point. In the presence of measurement noise, the LOS measurements originating from the same target should intersect with each other in a close neighborhood. Therefore, the points of intersection from the LOS measurements of any two sensors are clustered and the centroids of each cluster are used to initialize the target locations. The emission times are initialized in the same way. As in the first approach, five values are evaluated for N.

The third approach is based on the SEQ[m(2-D)] assignment algorithm. The associations with more than two real measurements correspond to potential targets. Let $N_{\max}$ denote the number of such associations. These associations are ranked in terms of the association cost. A range of values from 1 to $N_{\max}$ will be evaluated for N, and the top N associations will be used to initialize the EM-based algorithm.

Tables VIII–X present the performance of the EM-based algorithm with both UGM and PPP formulations (UGM/EM and PPP/EM) using different initialization approaches at three levels[10] of p_d with a known false alarm rate ($N_{fa} = 1$) in a scenario where ten sensors are used to locate four targets.

Initialization at the truth enables the EM-based algorithm to estimate the number of targets, target locations, and target directions accurately and the estimation becomes more accurate as p_d increases. In this case, the global maximum is attained.

The clustering-based initialization is very prone to ghosting and therefore results in very large errors in terms of the number of targets, target locations, and target directions. It also takes a longer processing time with such a poor initialization. In this case, the algorithm terminates at a local maximum.

The assignment-based initialization overcomes the ghosting problem. With $m = 2$ in the SEQ[m(2-D)] assignment, the target direction errors are less than the standard deviation of the bearing measurement noise and the target location errors are close to those obtained using initialization at the truth. At a higher p_d, the number of overestimation cases increases. This is due to double counting of the same target by the assignment algorithm when two acoustic events occur at the same location. The assignment algorithm does not differentiate the acoustic events that occurred at the same location.

[10]The probability of detection is set to be the same for each target in the simulation studies only for simplicity; the EM-based algorithms can deal with the case that the probabilities of detection for different targets are distinct.

TABLE IX
UGM/EM and PPP/EM Performance Using Different Initialization ("I:") Approaches for Unknown $p_d = 0.8$

	I: Truth		I: Clustering		I: SmB($m = 1$)		I: SmB($m = 2$)	
	UGM/EM	PPP/EM	UGM/EM	PPP/EM	UGM/EM	PPP/EM	UGM/EM	PPP/EM
φ_{over}	0	0	51.5%	54.8%	26.9%	27.1%	27.5%	27.8%
$\bar{M}_{\text{over}}$	N.A.	N.A.	1.53	1.52	1.21	1.21	1.23	1.23
$T_{\text{over}}^{\text{RMSE}}$ (m)	N.A.	N.A.	30.97	40.07	2.84	2.87	2.85	2.88
$\theta_{\text{over}}^{\text{RMSE}}$ (°)	N.A.	N.A.	2.14	2.17	0.391	0.405	0.394	0.409
φ_{under}	0.1%	0.1%	29.6%	27.1%	14.2%	14.0%	14.1%	13.8%
$\bar{M}_{\text{under}}$	1	1	1.50	1.46	1.09	1.09	1.10	1.09
$T_{\text{under}}^{\text{RMSE}}$ (m)	0.69	0.74	33.60	29.94	2.58	2.57	2.57	2.57
$\theta_{\text{under}}^{\text{RMSE}}$ (°)	0.0886	0.0903	3.28	3.09	0.317	0.316	0.317	0.316
φ_{exact}	99.9%	99.9%	18.9%	18.1%	58.9%	58.9%	58.4%	58.4%
$T_{\text{exact}}^{\text{RMSE}}$ (m)	2.55	2.52	46.88	47.86	3.06	3.06	2.97	2.98
$\theta_{\text{exact}}^{\text{RMSE}}$ (°)	0.306	0.303	4.60	4.64	0.431	0.430	0.429	0.428
$T_{\text{all}}^{\text{RMSE}}$ (m)	2.55	2.52	35.44	40.07	2.95	2.96	2.89	2.91
$\theta_{\text{all}}^{\text{RMSE}}$ (°)	0.306	0.303	3.06	3.00	0.409	0.412	0.408	0.412
t (s)	2.17	2.16	4.51	4.62	1.69	1.71	1.85	1.84

TABLE X
UGM/EM and PPP/EM Performance Using Different Initialization ("I:") Approaches for Unknown $p_d = 0.9$

	I: Truth		I: Clustering		I: SmB($m = 1$)		I: SmB($m = 2$)	
	UGM/EM	PPP/EM	UGM/EM	PPP/EM	UGM/EM	PPP/EM	UGM/EM	PPP/EM
φ_{over}	0	0	52.1%	53.8%	33.6%	33.6%	34.7%	34.7%
$\bar{M}_{\text{over}}$	N.A.	N.A.	1.60	1.59	1.26	1.27	1.26	1.27
$T_{\text{over}}^{\text{RMSE}}$ (m)	N.A.	N.A.	37.92	29.36	2.44	2.46	2.49	2.50
$\theta_{\text{over}}^{\text{RMSE}}$ (°)	N.A.	N.A.	1.99	1.89	0.321	0.321	0.331	0.331
φ_{under}	0.3%	0.3%	28%	27.2%	9.9%	9.9%	8.9%	8.9%
$\bar{M}_{\text{under}}$	1	1	1.48	1.48	1.03	1.03	1.03	1.03
$T_{\text{under}}^{\text{RMSE}}$ (m)	1.41	1.40	27.82	25.97	2.55	2.58	2.29	2.36
$\theta_{\text{under}}^{\text{RMSE}}$ (°)	0.245	0.244	2.68	2.52	0.281	0.284	0.263	0.269
φ_{exact}	99.7%	99.7%	19.9%	19%	56.5%	56.5%	56.4%	56.4%
$T_{\text{exact}}^{\text{RMSE}}$ (m)	2.23	2.22	43.77	52.87	2.95	2.96	2.99	3.00
$\theta_{\text{exact}}^{\text{RMSE}}$ (°)	0.275	0.273	4.39	4.35	0.377	0.380	0.388	0.391
$T_{\text{all}}^{\text{RMSE}}$ (m)	2.23	2.21	37.59	35.14	2.76	2.77	2.77	2.79
$\theta_{\text{all}}^{\text{RMSE}}$ (°)	0.275	0.273	2.83	2.71	0.352	0.354	0.361	0.364
t (s)	2.22	2.22	4.88	4.92	1.97	1.95	2.21	2.19

Although it indicates that there are more targets than the truth, all real targets have actually been identified.

Fig. 8 compares the performance of the EM-based algorithm with both UGM and PPP formulations (UGM/EM and PPP/EM) with initialization at the truth for different known expected numbers of false alarms (or false alarm rate) in a scenario where ten sensors are used to locate four targets. The quality of the cardinality, location, and bearing estimates using both formulations is almost identical for the same p_d value, which demonstrates the effectiveness of the UGM formulation to incorporate the false alarm rate when it is known.

E. Assignment Algorithms and EM-Based Algorithms

In Sections VI-C and VI-D, the two types of algorithms—the assignment algorithms and the EM-based algorithms—were evaluated separately according to their assumed target-originated measurement models (Bernoulli and Poisson, respectively). Since the Bernoulli measurement model is the more realistic one, the target measurements are generated in the next evaluation according to this Bernoulli measurement model for comparing all the algorithms. Therefore, there is no measurement model mismatch for the assignment algorithms, but there is a measurement model mismatch for the EM-based algorithms.

Tables XI–XIII compare the assignment algorithms and EM-based algorithms with assignment-based initialization. In this case, one may consider the EM-based algorithms as postprocessing procedures. Such a processing increases the entire processing time and only leads to an insignificant improvement of the estimation accuracies. However, it reflects the capability of the EM-based

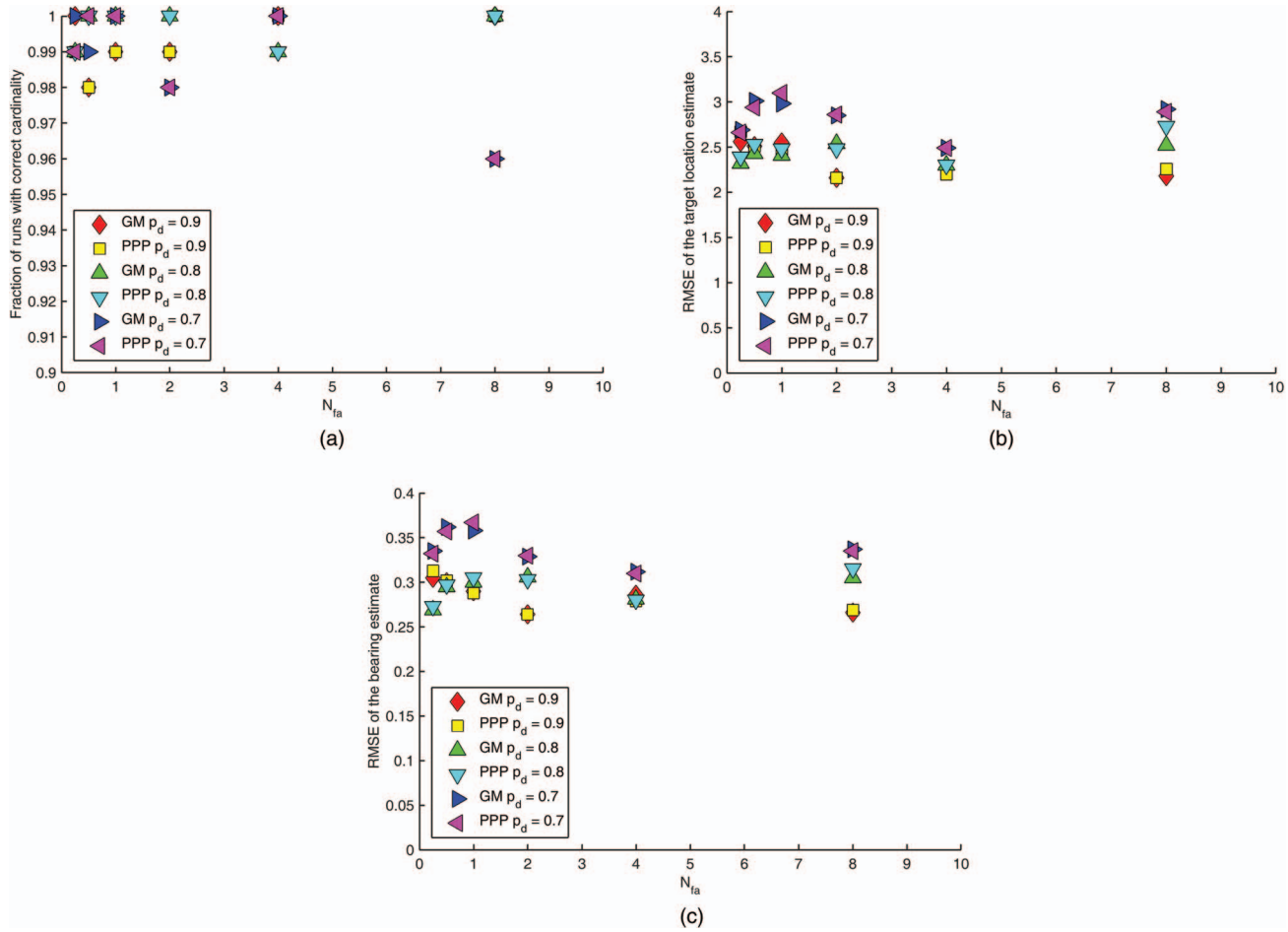


Fig. 8. The performance (in terms of φ_{exact}, $T_{\text{all}}^{\text{RMSE}}$, and $\theta_{\text{all}}^{\text{RMSE}}$) of UGM/EM and PPP/EM in scenarios with different known expected number of false alarms (0.25, 0.5, 1, 2, 4, and 8) for true p_{d} values at 0.7, 0.8, and 0.9.

TABLE XI
Performance Comparison Among *S*-D Assignment, Sequential *m*-Best (SmB) 2-D Assignment, and EM-Based Algorithms (With Different Initializations "I:") for $p_{\text{d}} = 0.9$ ($N_{\text{s}} = 6$)

	Assignment			I: SmB($m = 2$)		I: SmB($m = 4$)	
	S-D	SmB($m = 2$)	SmB($m = 4$)	UGM/EM	PPP/EM	UGM/EM	PPP/EM
φ_{over}	2.0%	0.4%	0.4%	0.4%	0.4%	0.4%	0.4%
$\bar{M}_{\text{over}}$	1	1	1	1	1	1	1
$T_{\text{over}}^{\text{RMSE}}$ (m)	14.24	3.55	3.55	3.54	3.54	3.54	3.54
$\theta_{\text{over}}^{\text{RMSE}}$ (°)	0.640	0.306	0.306	0.307	0.307	0.307	0.307
φ_{under}	1.9%	0.4%	0.3%	0.4%	0.4%	0.3%	0.3%
$\bar{M}_{\text{under}}$	1	1	1	1	1	1	1
$T_{\text{under}}^{\text{RMSE}}$ (m)	18.14	2.86	2.91	2.86	2.86	2.91	2.91
$\theta_{\text{under}}^{\text{RMSE}}$ (°)	0.792	0.279	0.286	0.279	0.279	0.286	0.286
φ_{exact}	96.1%	99.2%	99.3%	99.2%	99.2%	99.3%	99.3%
$T_{\text{exact}}^{\text{RMSE}}$ (m)	9.44	4.42	4.42	4.41	4.41	4.40	4.40
$\theta_{\text{exact}}^{\text{RMSE}}$ (°)	0.832	0.318	0.318	0.318	0.318	0.318	0.318
$T_{\text{all}}^{\text{RMSE}}$ (m)	9.74	4.41	4.42	4.40	4.40	4.40	4.40
$\theta_{\text{all}}^{\text{RMSE}}$ (°)	0.828	0.318	0.318	0.318	0.318	0.318	0.318
t (s)	0.568	0.096	0.162	0.181	0.190	0.249	0.247

TABLE XII
Performance Comparison Among S-D Assignment, Sequential m-Best (SmB) 2-D Assignment, and EM-Based Algorithms (With Different Initializations "I:") for $p_d = 0.8$ ($N_s = 6$)

	Assignment			I: SmB($m = 2$)		I: SmB($m = 4$)	
	S-D	SmB($m = 2$)	SmB($m = 4$)	UGM/EM	PPP/EM	UGM/EM	PPP/EM
φ_{over}	0.6%	0.5%	0.4%	0.4%	0.3%	0.3%	0.3%
$\bar{M}_{\text{over}}$	1	1	1	1	1	1	1
$T^{\text{RMSE}}_{\text{over}}$ (m)	4.21	6.75	6.74	5.69	6.27	6.26	6.26
$\theta^{\text{RMSE}}_{\text{over}}$ (°)	0.302	0.423	0.421	0.367	0.395	0.396	0.395
φ_{under}	25.4%	7.2%	7.5%	7.2%	7.2%	7.5%	7.5%
$\bar{M}_{\text{under}}$	1.23	1.03	1.03	1.03	1.03	1.03	1.03
$T^{\text{RMSE}}_{\text{under}}$ (m)	30.12	5.55	5.49	5.19	5.19	5.14	5.14
$\theta^{\text{RMSE}}_{\text{under}}$ (°)	1.58	0.369	0.367	0.359	0.359	0.358	0.358
φ_{exact}	74.0%	92.3%	92.1%	92.4%	92.5%	92.2%	92.2%
$T^{\text{RMSE}}_{\text{exact}}$ (m)	12.86	5.15	5.11	5.08	5.08	5.05	5.05
$\theta^{\text{RMSE}}_{\text{exact}}$ (°)	1.06	0.362	0.359	0.358	0.358	0.356	0.356
$T^{\text{RMSE}}_{\text{all}}$ (m)	17.49	5.18	5.14	5.09	5.09	5.06	5.06
$\theta^{\text{RMSE}}_{\text{all}}$ (°)	1.18	0.362	0.360	0.358	0.359	0.356	0.356
t (s)	0.420	0.070	0.141	0.151	0.148	0.231	0.237

TABLE XIII
Performance Comparison Among S-D Assignment, Sequential m-Best (SmB) 2-D Assignment, and EM-Based Algorithms (With Different Initializations "I:") for $p_d = 0.7$ ($N_s = 6$)

	Assignment			I: SmB($m = 2$)		I: SmB($m = 4$)	
	S-D	SmB($m = 2$)	SmB($m = 4$)	UGM/EM	PPP/EM	UGM/EM	PPP/EM
φ_{over}	0.3%	0	0	0	0	0	0
$\bar{M}_{\text{over}}$	1	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
$T^{\text{RMSE}}_{\text{over}}$ (m)	6.30	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
$\theta^{\text{RMSE}}_{\text{over}}$ (°)	0.382	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
φ_{under}	57.4%	27.6%	27.5%	27.6%	27.6%	27.5%	27.5%
$\bar{M}_{\text{under}}$	1.46	1.14	1.14	1.14	1.14	1.14	1.14
$T^{\text{RMSE}}_{\text{under}}$ (m)	35.24	7.10	6.96	5.69	5.68	5.65	5.65
$\theta^{\text{RMSE}}_{\text{under}}$ (°)	2.19	0.402	0.400	0.383	0.383	0.381	0.382
φ_{exact}	42.3%	72.4%	72.5%	72.4%	72.4%	72.5%	72.5%
$T^{\text{RMSE}}_{\text{exact}}$ (m)	14.90	7.44	6.96	7.27	7.27	6.88	6.86
$\theta^{\text{RMSE}}_{\text{exact}}$ (°)	0.911	0.872	0.860	0.864	0.864	0.851	0.851
$T^{\text{RMSE}}_{\text{all}}$ (m)	26.29	7.37	6.96	6.96	6.96	6.63	6.62
$\theta^{\text{RMSE}}_{\text{all}}$ (°)	1.63	0.795	0.785	0.786	0.786	0.775	0.775
t (s)	0.318	0.058	0.127	0.126	0.133	0.191	0.196

algorithm to solve the data association problem. Associating the measurements with a good degree of accuracy requires a good initialization, such as the assignment approach. In this case, although there is a mismatch in the measurement model, the EM-based algorithms estimate the number of targets, target locations, and target directions quite accurately due to the fact that the initialization by the SEQ[m(2-D)] assignment algorithm is close to the truth.

VII. CONCLUSION

This paper considers the problem of multiple transient emitter localization using a network of passive sensors. It is assumed that the number of targets as well as the association between measurements and targets is unknown and in the presence of missed detections and false alarms. Two different measurement models—the Bernoulli measurement model and the Poisson measurement model—are considered for each target and two types of algorithms—assignment- and EM-based—are presented, one for each measurement model. Simulation studies show that the SEQ[m(2-D)] assignment algorithm has very promising performance and can be employed in real-time applications. While the EM-based algorithms have the capability of solving the data association problem, simulation results suggest that they require the right initial estimates to provide reliable

localization results and the processing time could be longer than required. The fusion algorithms discussed in this paper assume that there is a fusion center to which each sensor can communicate. Fusion algorithms, which assume that no such fusion center exists and allow distributed processing and only single-hop communication, are developed in [10].

APPENDIX A PROOF OF PROPOSITION 1

For notational simplicity, let us denote

$$a_i = \sum_{\ell=1}^{N_s} \sum_{j=1}^{n_\ell} w_{\ell ji}^{(n-1)}, \quad i = 0, 1, \ldots, N \tag{100}$$

$$p_i = p_d(T_i), \quad i = 1, 2, \ldots, N \tag{101}$$

$$\mathbf{p} = [p_1, p_2, \ldots, p_N] \tag{102}$$

$$h_i(\mathbf{p}) = p_i - 1. \tag{103}$$

Substituting (25) into (45), the problem becomes

$$\begin{aligned} \underset{\mathbf{p}}{\text{maximize}} \quad & f(\mathbf{p}) = a_0 \ln N_{fa} + \sum_{i=1}^{N} a_i \ln p_i \\ & - \left(\sum_{i=0}^{N} a_i \right) \ln \left(\sum_{i=1}^{N} p_i + N_{fa} \right) \\ \text{subject to} \quad & h_i(\mathbf{p}) \le 0, \quad i = 1, \ldots, N. \end{aligned} \tag{104}$$

Let μ_i be a Lagrange multiplier corresponding to $p_i \le 1$ and $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_N]$. The Lagrangian is

$$L(\mathbf{p}, \boldsymbol{\mu}) = f(\mathbf{p}) + \sum_{i=1}^{N} \mu_i (0 - h_i(\mathbf{p})). \tag{105}$$

From the KKT conditions, the optimal values of $\mathbf{p}$ and $\boldsymbol{\mu}$ satisfy the following system of equations and inequalities:

$$\begin{cases} 0 = \dfrac{\partial f}{\partial p_i} - \mu_i \dfrac{\partial h_i}{p_i} = \dfrac{a_i}{p_i} - \dfrac{\sum_{i=0}^{N} a_i}{\sum_{i=1}^{N} p_i + N_{fa}} - \mu_i, \\ i = 1, 2, \ldots, N \end{cases} \tag{106}$$

$$0 = \mu_i h_i(\mathbf{p}) = \mu_i (p_i - 1), \quad i = 1, 2, \ldots, N \tag{107}$$

$$0 \le \mu_i, \quad i = 1, 2, \ldots, N. \tag{108}$$

We need to break the analysis into cases according to (107).

Case 1: If

$$\mu_i = 0, \quad i = 1, 2, \ldots, N \tag{109}$$

then (106) is simplified to

$$p_i = \frac{a_i \left(\sum_{k=1}^{N} p_k + N_{fa} \right)}{\sum_{i=0}^{N} a_i}. \tag{110}$$

Summing over i from 1 to N, we have

$$\sum_{i=1}^{N} p_i = \sum_{k=1}^{N} p_k = \frac{\sum_{i=1}^{N} a_i \left(\sum_{k=1}^{N} p_k + N_{fa} \right)}{\sum_{i=0}^{N} a_i} \tag{111}$$

which can be simplified to

$$\sum_{k=1}^{N} p_k = \frac{\sum_{i=1}^{N} a_i N_{fa}}{a_0}. \tag{112}$$

Substituting (112) into (110), we have

$$p_i = \frac{a_i N_{fa}}{a_0}. \tag{113}$$

The feasibility of this solution depends on whether $a_i N_{fa}/a_0$ is greater than 1. Let

$$\mathcal{S} = \{ i \,|\, a_i N_{fa} > a_0 \}. \tag{114}$$

If the set $\mathcal{S}$ is empty, (113) will be the optimal solution for $\mathbf{p}$. If the set $\mathcal{S}$ is not empty, then we must have

$$\begin{cases} 0 = \mu_i h_i(\mathbf{p}) = \mu_i (p_i - 1), \quad i \notin \mathcal{S} & (115) \\ 0 \le \mu_i, \quad i \notin \mathcal{S} & (116) \\ 0 = p_i - 1, \quad i \in \mathcal{S} & (117) \\ 0 < \mu_i, \quad i \in \mathcal{S}. & (118) \end{cases}$$

Case 2: If

$$\begin{cases} 0 = \mu_i, \quad i \notin \mathcal{S} & (119) \\ 1 = p_i, \quad i \in \mathcal{S} & (120) \end{cases}$$

then (106) is simplified to

$$p_i = \frac{a_i \left(\sum_{k \notin \mathcal{S}} p_k + |\mathcal{S}| + N_{fa} \right)}{\sum_{i=0}^{N} a_i}, \quad i \notin \mathcal{S}. \tag{121}$$

Summing over i that is not in the set $\mathcal{S}$ and solving for $\sum_{i \notin \mathcal{S}} p_i$,

$$\sum_{i \notin \mathcal{S}} p_i = \frac{\sum_{i>0, i \notin \mathcal{S}} a_i (|\mathcal{S}| + N_{fa})}{\sum_{i>0, i \in \mathcal{S}} a_i + a_0}. \tag{122}$$

Substituting (122) into (121), we have

$$p_i = \frac{a_i (|\mathcal{S}| + N_{fa})}{\sum_{i>0, i \in \mathcal{S}} a_i + a_0}, \quad i \notin \mathcal{S}. \tag{123}$$

Since

$$a_i N_{fa} > a_0, \quad i \in \mathcal{S} \tag{124}$$

we have

$$\sum_{i>0, i \in \mathcal{S}} a_i N_{fa} > |\mathcal{S}| a_0 \tag{125}$$

$$\sum_{i>0, i \in \mathcal{S}} a_i N_{fa} + a_0 N_{fa} > |\mathcal{S}| a_0 + a_0 N_{fa} \tag{126}$$

$$\frac{N_{\text{fa}}}{a_0} > \frac{|\mathcal{S}| + N_{\text{fa}}}{\sum_{i>0, i\in\mathcal{S}} a_i + a_0}. \tag{127}$$

Since

$$a_i N_{\text{fa}} \le a_0, \quad i \notin \mathcal{S} \tag{128}$$

we have

$$p_i = \frac{a_i(|\mathcal{S}| + N_{\text{fa}})}{\sum_{i>0, i\in\mathcal{S}} a_i + a_0} < \frac{a_i N_{\text{fa}}}{a_0} \le 1, \quad i \notin \mathcal{S} \tag{129}$$

which verifies the feasibility of the solution consisting of (120) and (123). One can summarize the two cases as follows:

$$p_i = \begin{cases} 1, & \text{if } i \in \mathcal{S} \\ \dfrac{a_i(|\mathcal{S}| + N_{\text{fa}})}{\sum_{i>0, i\in\mathcal{S}} a_i + a_0}, & \text{if } i \notin \mathcal{S} \end{cases} \tag{130}$$

which is equivalent to (58) because of (100) and (101).

REFERENCES

[1] H. Akaike
"A new look at the statistical model identification,"
IEEE Trans. Autom. Control, vol. 19, no. 6, pp. 716–723, Dec. 1974.

[2] J. Areta, Y. Bar-Shalom, M. Levedahl, and K. Pattipati
"Hierarchical track association and fusion for a networked surveillance system,"
J. Adv. Inf. Fusion, vol. 1, no. 2, pp. 140–157, Dec. 2006.

[3] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan
Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software, Hoboken, NJ: Wiley, 2001.

[4] Y. Bar-Shalom, P. Willett, and X. Tian
Tracking and Data Fusion, Storrs, CT: YBS Publishing, 2011.

[5] C. M. Bishop
Pattern Recognition and Machine Learning, New York: Springer, 2006.

[6] S. Blackman
"Multiple hypothesis tracking for multiple target tracking,"
IEEE Aerosp. Electron. Syst. Mag., vol. 19, no. 1, pp. 5–18, Jan. 2004.

[7] D. Daley and D. Vere-Jones
An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods, New York: Springer, 2003.

[8] S. Deb, M. Yeddanapudi, K. Pattipati, and Y. Bar-Shalom
"A generalized $\mathcal{S}$-D assignment algorithm for multisensor–multitarget state estimation,"
IEEE Trans. Aerosp. Electron. Syst., vol. 33, no. 2, pp. 523–538, Apr. 1997.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin
"Maximum likelihood from incomplete data via the EM algorithm,"
J. R. Stat. Soc., Ser. B, vol. 39, no. 1, pp. 1–38, 1977.

[10] W. Dou, Y. Bar-Shalom, L. Kaplan, and J. George
"Distributed fusion algorithm for passive localization of multiple transient emitters,"
J. Adv. Inf. Fusion, vol. 13, no. 1, pp. 15–35, Jun. 2018.

[11] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe
"Sonar tracking of multiple targets using joint probabilistic data association,"
IEEE J. Ocean. Eng., vol. 8, no. 3, pp. 173–184, Jul. 1983.

[12] J. George and L. Kaplan
"Shooter localization using a wireless sensor network of soldier-worn gunfire detection systems,"
J. Adv. Inf. Fusion, vol. 8, no. 1, pp. 15–32, Jun. 2013.

[13] J. George, L. Kaplan, S. Deligeorges, and G. Cakiades
"Multi-shooter localization using finite point process,
Proc. 17th Int. Conf. Inf. Fusion, Jul. 2014.

[14] J. George and L. Kaplan
"A finite point process approach to multi-agent localization using transient measurements,"
J. Inf. Fusion, vol. 32, no. PA, pp. 62–74, Nov. 2016.

[15] H. W. Kuhn and A. W. Tucker
"Nonlinear programming,"
Proc. 2nd Berkeley Symp. Math. Stat. Probability, 1951, pp. 481–492.

[16] R. Mahler
"Multitarget Bayes filtering via first-order multitarget moments,"
IEEE Trans. Aerosp. Electron. Syst., vol. 39, no. 4, pp. 1152–1178, Oct. 2003.

[17] J. Moller and R. P. Waagepetersen
Statistical Inference and Simulation for Spatial Point Processes, London, UK: Chapman & Hall/CRC, 2003.

[18] K. G. Murty
"An algorithm for ranking all the assignments in order of increasing cost,"
Oper. Res., vol. 16, no. 3, pp. 682–687, 1968.

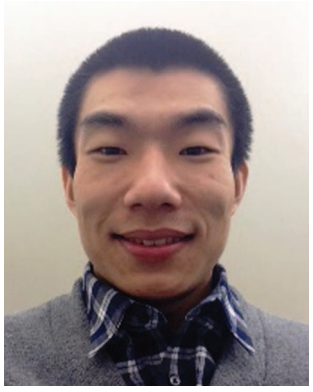
[19] R. W. Osborne, Y. Bar-Shalom, J. George, and L. Kaplan
"Data fusion from multiple passive sensors for multiple shooter localization via assignment,"
in *Proc. 17th Int. Conf. Inf. Fusion*, Jul. 2014.

[20] R. W. Osborne, Y. Bar-Shalom, J. George, and L. Kaplan
"Statistical efficiency of target localization from angle and shockwave measurements,"
J. Adv. Inf. Fusion, vol. 9, no. 2, pp. 75–89, Dec. 2014.

[21] K. R. Pattipati, S. Deb, Y. Bar-Shalom, and R. B. Washburn
"A new relaxation algorithm and passive sensor data association,"
IEEE Trans. Autom. Control, vol. 37, no. 2, pp. 198–213, Feb. 1992.

[22] G. Schwarz
"Estimating the dimension of a model,"
Ann. Stat., vol. 6, no. 2, pp. 461–464, 1978.

[23] S. Zhang and Y. Bar-Shalom
"Practical data association for passive sensors in 3D,"
J. Adv. Inf. Fusion, vol. 9, no. 1, pp. 38–46, Jun. 2014.



Wenbo Dou received the Ph.D. degree in electrical engineering from the University of Connecticut, Storrs, CT, USA, in 2017. His current research interests include bias estimation, multitarget localization and tracking, data fusion, and machine learning.

Jemin George received the M.S. and Ph.D. degrees in aerospace engineering from the State University of New York at Buffalo, Buffalo, NY, USA, in 2007 and 2010, respectively. In 2008, he was a Summer Research Scholar with the U.S. Air Force Research Laboratory's Space Vehicles Directorate, and in 2009, he was a National Aeronautics and Space Administration Langley Aerospace Research Summer Scholar. From 2009 to 2010, he was a Research Fellow with the Department of Mathematics, Technische Universität Darmstadt, Darmstadt, Germany. Since 2010, he has been with the U.S. Army Research Laboratory, Adelphi, MD, USA. His principal research interests include stochastic systems, control theory, nonlinear filtering, information fusion, adaptive networks, and distributed sensing and estimation.

Lance M. Kaplan received the B.S. degree with distinction from Duke University, Durham, NC, USA, in 1989, and the M.S. and Ph.D. degrees from the University of Southern California, Los Angeles, CA, USA, in 1991 and 1994, respectively, all in electrical engineering. From 1987 to 1990, he was a Technical Assistant with the Georgia Tech Research Institute. He held a National Science Foundation Graduate Fellowship and a USC Dean's Merit Fellowship from 1990 to 1993, and was a Research Assistant with the Signal and Image Processing Institute, University of Southern California, from 1993 to 1994. Then, he was on staff with the Reconnaissance Systems Department, Hughes Aircraft Company, from 1994 to 1996. From 1996 to 2004, he was a member of the faculty with the Department of Engineering and a senior investigator with the Center of Theoretical Studies of Physical Systems, Clark Atlanta University, Atlanta, GA, USA. He is currently a Researcher with the Networked Sensing and Fusion branch of the U.S. Army Research Laboratory (ARL), Adelphi, MD, USA. Dr. Kaplan is on the Board of Governors for the IEEE Aerospace and Electronic Systems Society (2008–2013, 2018–present) and a VP of Conferences for the International Society of Information Fusion (ISIF) (2014–present). Previously, he was Editor-in-Chief for the *IEEE Transactions on Aerospace and Electronic Systems* (2012–2017) and on the Board of Directors of ISIF (2012–2014). He is a three-time recipient of the Clark Atlanta University Electrical Engineering Instructional Excellence Award from 1999 to 2001. He is a Fellow of IEEE and of ARL. His current research interests include information/data fusion, reasoning under uncertainty, network science, resource management, and signal and image processing.

Richard W. Osborne, III, received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Connecticut, Storrs, CT, USA, in 2004, 2007, and 2012, respectively. From 2012 to 2014, he was an Assistant Research Professor with the Electrical Engineering Department, University of Connecticut. From 2014 to 2015, he was a Senior Research Engineer with BAE Systems, Inc., Burlington, MA, USA, and since 2015, he has been a Senior Research Engineer with United Technologies Research Center, East Hartford, CT, USA. His academic interests include adaptive target tracking, information/sensor fusion, perception/computer vision, autonomous systems, and other aspects of estimation.

Yaakov Bar-Shalom received the B.S. and M.S. degrees from the Technion, Haifa, Israel, in 1963 and 1967, respectively, and the Ph.D. degree from Princeton University, Princeton, NJ, USA, in 1970, all in electrical engineering. From 1970 to 1976, he was with Systems Control, Inc., Palo Alto, CA, USA. He is currently a Board of Trustees Distinguished Professor with the Department of Electrical and Computer Engineering and Marianne E. Klewin Professor in Engineering with the University of Connecticut, Storrs, CT, USA. His current research interests include estimation theory, target tracking, and data fusion. He has authored or coauthored more than 550 papers and book chapters, and coauthored/edited 8 books, including *Tracking and Data Fusion* (YBS Publishing, 2011). He was elected Fellow of IEEE for contributions to the theory of stochastic systems and of multitarget tracking. He was an Associate Editor of the *IEEE Transactions on Automatic Control* and *Automatica*. He was General Chairman of the 1985 ACC, Chairman of the Conference Activities Board of the IEEE CSS and member of its Board of Governors, General Chairman of FUSION 2000, President of ISIF in 2000 and 2002, and Vice President for Publications during 2004–2013. In 1987, he was the recipient of the IEEE CSS Distinguished Member Award. Since 1995, he is a Distinguished Lecturer of the IEEE Aerospace and Electronic Systems Society. He was the corecipient of the M. Barry Carlton Award for the best paper in the *IEEE Transactions on Aerospace and Electronic Systems* in 1995 and 2000. In 2002, he was the recipient of the J. Mignona Data Fusion Award from the DoD JDL Data Fusion Group. He is a member of the Connecticut Academy of Science and Engineering. In 2008, he was the recipient of the IEEE Dennis J. Picard Medal for Radar Technologies and Applications, and in 2012 the Connecticut Medal of Technology. He has been listed by academic.research.microsoft (top authors in engineering) as #1 among the researchers in aerospace engineering based on the citations of his work. He was the recipient of the 2015 ISIF Award for a Lifetime of Excellence in Information Fusion. This award has been renamed in 2016 as the Yaakov Bar-Shalom Award for a Lifetime of Excellence in Information Fusion.

Arabic Dynamic Gesture Recognition Using Classifier Fusion

BASMA HISHAM
ALAA HAMOUDA

Sign language is a visual language that is the primary way used by hearing-impaired people in order to connect and communicate with each other and with their societies. Some studies have been conducted on Arabic sign language (ArSL) recognition systems, but a practically deployable system for real-time use is still a challenge. The main objective of this paper is to develop a novel model that is able to recognize the ArSL using Microsoft's Kinect V2. This paper works on the dynamic gestures that are performed by both hands and body parts, and introduces an effective way of capturing and detecting the hand and skeleton joints from the depth image that is provided by Kinect. The model used two supervised machine learning algorithms, support vector machine (SVM) and *K*-nearest neighbors (KNN), and then applied Dezert–Smarandache theory (DSmT) as a fusion technique in order to combine their results. We compared the results of the proposed model with the Ada-Boosting technique and finally applied two most widely used methods that are used with dynamic gesture recognition, dynamic time warping (DTW) and hidden Markov model (HMM), to compare their results with the previous classifier fusion. Finally, we applied the model on ArSL dataset that is composed of 40 Arabic medical signs to ease the communication between hearing-impaired patients and their doctor. The accuracy of the model is improved when the classifier fusion is applied compared to using each classifier separately. The overall accuracies for SVM, KNN, DSmT fusion, and Ada-Boosting are 79%, 89%, 91.5%, and 90.2%, respectively. Also, DTW and HMM achieved overall accuracies of 82.6% and 79.5%, respectively.

Manuscript received February 6, 2019; released for publication July 8, 2019.

The authors are with the Computer Engineering Department, Al-Azhar University, Cairo, Egypt.
(Corresponding author: Basma Hisham) basmahisham.2015@gmail.com

1557-6418/19/$17.00

I. INTRODUCTION

Sign language is the most basic way for hearing-impaired people to connect and interact with each other and integrate with their societies. The main problem is that most of the normal people do not understand sign language [1]. Therefore, the need to develop an automatic system that is capable of translating sign languages into understandable words and sentences is becoming very necessary. There are two main approaches to sign language recognition systems: vision-based approach and sensor-based approach. The main advantage of the vision-based system is that there is no need to use complex devices, so it has low cost and does not need pre-setup, but this approach requires extra calculations in the preprocessing stage, image processing, and artificial intelligence to recognize and interpret signs. Also, it suffers from the background problems because it needs subtraction techniques to subtract the signer from the background and it may fail if the background changes. Sensor-based systems provide robust, reliable, and more accurate data, but they are not user-friendly like vision-based systems because they require extra equipment like data gloves. The user is required to wear the gloves in order to collect the data, so this approach is not practical [2]. Microsoft Kinect is a motion sensing input device that is developed by Microsoft. It provides live streams of depth information about the skeleton joints and body motion. This information is essential to construct the three-dimensional (3-D) view of the tracked objects. It used to track standing skeleton with high-depth fidelity, so compared with other depth sensors, Kinect is the best choice in short-range environment [3]. Kinect also has an RGB camera, voice recognition capability, face-tracking capabilities, and access to the raw sensor records. Once the data have been collected from the user, the recognition system, whether it is sensor-based or image-based, must use these data for processing to recognize the signs [4]. Several approaches have been proposed for sign recognition, the most important and effective approach being that using machine learning algorithms. It can handle the complexity and the differentiation of sign language gestures [5]. Also, it can handle the different manners in which the people repeat different signs [6]. Machine learning algorithms such as neural networks, support vector machines (SVMs), *K*-nearest neighbors (KNN), decision tree models, etc. have been focused on the classification stage of recognizing a gesture captured from the signer. The single sign classifier assumes that signs are presegmented, and it recognizes sign by sign rather than continuous sentences. It is supposed to automate the process of splitting a sentence into words, which is called segmentation. Segmentation is one of the major issues of information processing in sign languages. Motion speed during capturing of continuous sentences may be used as a segmenter. It is noticeable that the motion speed is changed while performing the signs, and when the transition from

one sign to another occurs, the motion speed is slowed down.

The main aim of this paper is to develop an Arabic sign language (ArSL) recognition system that identifies the Arabic signs captured by Microsoft Kinect based on the data that represent body and hand motion. These data will be excluded from the depth image information obtained from the Kinect sensor. However, Kinect cannot accurately detect the hand movement and also the details of fingers, but we overcame such limitations and introduced an effective and simple method for hand detection. We used two machine learning algorithms, KNN and SVM, and introduced an effective fusion method based on Dezert–Smarandache theory (DSmT) to combine the classifier results and enhance the accuracy. Also, we applied two direct matching algorithms, dynamic time wrapping (DTW) and hidden Markov model (HMM), to compare the results of fusion with other dynamic gesture recognition techniques. The structure of this paper is organized as follows. Section II presents the related work of the sign language recognition. The methodology is presented in Section III. The experimental setup and results are presented in Section IV. Section V contains conclusion and future work.

II. RELATED WORK

Sign language is a combination of words that are represented by using movements of different body parts such as head, shoulders, elbow, wrist, etc. and finally added to the hand signs to create a meaning [7]–[11]. Many researchers aimed to build an automated system in order to translate ArSL to Arabic text or voice. ArSL research works faced several difficulties; for example, it is not defined well and the works in it started in the last decade. However, in ArSL, there are more than 9000 signs and it uses 26 hand postures and 5 dynamic gestures in order to represent the Arabic alphabet. There is a variation in ArSL among the different Arabic countries. Some Arabic countries have their own sign language, such as Tunis, Gulf countries, Egypt, etc. We are concerned with ArSL in Egypt. The organization of ArSL in Egypt has started in 1983, and there are 7 million hearing-impaired persons till the last studies performed by United Nations. This is a large number, so they need to be merged with their societies as any normal person. As explained earlier, there are mainly two approaches for sign recognition: 1) vision-based approach and 2) sensor-based approach. For vision-based approach, there are several ArSL recognition works, such as an Arabic sign recognition model developed by Mohandes *et al.* for Arabic alphabet recognition, using SVM as a classifier with an accuracy of 87% [12]. Ahmed *et al.* also proposed a model for sign language recognition. Several statistical analyses were performed on the data extracted from the collected images to create the feature vector that is input to an SVM. The model was tested on ten letters and the accuracy was 83%. They suggested building a real-time system that is able to work on both dynamic and static signs [13]. Maraqa and Abu-Zaiter developed a static and dynamic ArSL recognition system by applying feedforward and recurrent neural networks on the features extracted from the captured images. They tested the proposed system on 30 gestures and achieved an accuracy of 95% [14]. Assaleh and Al-Rousan developed ArSL recognition for alphabet signs based on polynomial classifiers; they compared the results of the system with the previously published results using Artificial Neural Network Fuzzy Inference System (ANFIS)-based classification on the same dataset and feature extraction method. The comparison showed significant improvement, and the misclassified patterns were reduced by 36% on the training set and by 57% on the test set [15]. Al-Jarrah and Al-Omari developed an automatic system for Arabic alphabet recognition with an accuracy of 97.5% [16]. Also, El-Bendary *et al.* proposed a sign language recognition system for the Arabic alphabets, which dealt with the images of bare hands that allow the user to interact naturally with the system and achieved an accuracy of 91.3% [17]. For a sensor-based approach, Assaleh *et al.* proposed a low-complexity word-based classification for ArSL recognition system using two DG5-VHand data gloves, and the recognition rates were 92.5% and 95.3% for user-independent and user-dependent modes, respectively [18]. Mohandes proposed an ArSL recognition system using CyberGlove in order to track 100 two-handed signs with 20 samples and achieved an accuracy of 99.6% [19]. Sadek *et al.* proposed a hand gesture recognition system using a smart glove that was designed from a set of sensors; the recognition was based on a statistical analysis of the hand shape while performing the 1300 words of the ArSL [20]. Hemayed and Hassanien (2010) presented Arabic alphabets recognition system, the alphabets signs were converted into speech but the system cannot operate in real time, the model was built based on the vision-based, the system take the colored images as input also they used Prewitt edge detector in order to detect the hand shape. KNN (K-Nearest Neighbour) was used in the classification phase [21]. Recently, some researchers used active devices such as Microsoft Kinect and Leap Motion controller (LMC). Almasre and Al-Nuaim proposed hand gesture recognition systems using supervised machine learning that predicts the hand pose using two sensors, Microsoft Kinect and LMC, depending on the depth images. They collected data regarding 28 letters from different signers and the results achieved about 100% detection rate in recognizing 22 letters from 28 Arabic letters [22]. ElBadawy *et al.* proposed a system that integrates a set of different types of sensors to capture all sign features. They used LMC in order to capture hands with finger movements, and also used two digital cameras to capture face features and body movement. They applied their system on 20 dynamic signs and the system achieved an accuracy of

95% [23]. Aliyu *et al.* proposed a Kinect-based system for ArSL recognition; the system was applied to 20 signs, and they used linear discriminant analysis for feature reduction and sign classification. Furthermore, fusion from RGB and depth sensor was carried out at feature and the decision level and it achieved an overall accuracy of 99.8% [24]. Jmaa *et al.* proposed a new approach based on hand gesture analysis for ArSL alphabet recognition by extracting a histogram of oriented gradient features from a hand image and then using them to train SVM models. Their approach involved three steps: 1) hand detection and localization using a Microsoft Kinect camera; 2) hand segmentation; and 3) feature extraction using Arabic alphabet recognition. The results showed an accuracy of about 90% [25]. Mohandes *et al.* developed a new model for ArSL recognition in order to detect and track at least one hand and one finger; two different sensors in two different locations in a room generate 3-D interaction space. They used a classifier integrated with two different sensors, LMC and Microsoft Kinect, and 28 Arabic alphabet signs were performed in the interaction space [26]. Almasre and Al-Nuaim proposed a model to recognize the hand gestures of ArSL words using two depth sensors. They examined 143 signs gestured by 10 users for 5 ArSL words. The sensors captured depth images of the upper human body, from which 235 angles (features) were extracted for each joint and between each pair of bones. The dataset was divided into a training set (109 observations) and a testing set (34 observations). They used an SVM classifier with different parameters in order to proceed with four SVM models, with linear kernel (SVMLD and SVMLT) and radial kernel (SVMRD and SVMRT) functions. The accuracy of the words in the training set for the SVMLD, SVMLT, SVMRD, and SVMRT models was 88.92%, 88.92%, 90.88%, and 90.884%, respectively. The accuracy of the testing set for SVMLD, SVMLT, SVMRD, and SVMRT was 97.059%, 97.059%, 94.118%, and 97.059%, respectively [10]. Several sign language recognition research works were performed based on data fusion. Rashid *et al.* developed a multimodal system in order to combine both gestures and postures for recognizing alphabets and numbers; the fusion was done on the decision level. The gesture recognition system was trained using HMM and was concerned with the dynamic motion. The posture recognition system was trained using SVM and was concerned with the static hand at the same time. They applied Gaussian distribution on the captured 3-D depth information to detect and segment gestures and postures. Then, feature vectors were constructed and extracted from spatial and temporal hand properties. Finally, they used the rule of AND/OR combination to state the decision; the model achieved an accuracy of 98% for alphabet and number gestures [27]. Song *et al.* introduced a model of gesture recognition using Microsoft Kinect. The 3-D position data regarding all body skeleton joints were captured using Kinect, and then the features of interest for each gesture were extracted. They segmented the gestures in real time and finally applied the data fusion approach on the decision level by combining the decision of the trained Gaussian mixture model and HMM. They applied their model on seven common gestures and achieved an accuracy of 94.36% [28]. Kishore and Rajesh Kumar presented an Indian sign language recognition system. They extracted the features from the captured video using two algorithms, Fourier descriptions and principal component analysis, and finally performed fusion on the level of features and applied a Sugeno-type fuzzy inference system. The system was applied on 80 common Indian signs and achieved an accuracy of 96% [29]. Penelle and Debeir proposed a data fusion system using Leap Motion and Microsoft Kinect sensors to improve hand recognition accuracy [30]. ElBadawy *et al.* proposed a hybrid system using LMC and two digital cameras. They used LMC for finger tracking and the digital cameras for body movement tracking with facial emotions. The proposed system was applied by a neural network on 20 Arabic signs with an accuracy of 95% [23]. Marin *et al.* proposed a framework to recognize static American signs. They used LMC for fingers and captured features based on distance, while Kinect was used for capturing features based on body and correlation. The proposed system applied SVM with an accuracy of 91% [31]. Fok *et al.* proposed a data fusion system based on two devices. Kalman filter was used for fusion and HMM was used for sign recognition. They applied the system on ten American digits [32]. Yang *et al.* proposed an optimized framework based on a tree structure classification model using three sensors, sEMG, ACC, and GYRO, to get the best performance as a single sensor, two-sensor fusion, and three-sensor fusion. The final recognition rates were 94.31% and 87.02% for 150 Chinese sign language subwords by two test scenarios [33]. Sadek *et al.* proposed a hand gesture recognition using a smart glove that was designed from a set of sensors; the recognition was based on a statistical analysis of the hand shape while performing the 1300 words of the ArSL [20]. Kumar *et al.* proposed a multisensor fusion framework for sign language recognition using a coupled HMM. They used Microsoft Kinect and LMC [34]. Sun *et al.* proposed a weighted fusion method based on Dempster–Shafer evidence theory (DST). The proposed recognition method depends on Kinect and sEMG signal. The average recognition rate was about 87% [35]. Mohandes *et al.* proposed ArSL recognition for the Arabic alphabets using two LMCs and applied DST. They tested the system using ten cross-validations. The first LMC achieved an accuracy of 93.077% and the second LMC achieved an accuracy of 89.907%. Then, they applied the DST on the feature level and on the decision level. The achieved accuracy reached 97.686% and 97.053%, respectively [26]. The main contribution of our proposed model is applying the data fusion on the decision level by combining the results of the two classifiers, KNN and SVM, using an effective fusion technique (DSmT). This

combination enhanced the accuracy of the system and made it robust rather than depending on single classifier. We have to mention that ArSL is not a unified language and varies from one country to another, so we focused on the Egyptian ArSL that is most generally comprehended by Arabs. The works of data fusion in sign language recognition, especially the ArSL, are very rare, so our research introduces a way for improving the existing sign language recognition systems by applying the concept of data fusion techniques that make the system robust and more reliable.

III. METHODOLOGY

In this section, we will discuss the structure of our proposed system for ArSL recognition using Microsoft Kinect. We describe the various phases of our system from capturing the gestures using Kinect till the gesture recognition. The structure of the proposed model is shown in Fig. 1. The first step in the proposed model is the data acquisition phase that occurs when the Kinect depth camera starts capturing the skeleton of standing signer in front of the Kinect camera and infers his/her skeleton positions; the system receives joint information such as type and coordinates, bone orientation, and motion velocity as a stream of frames. The preprocessing phase includes extracting the features of interest for both signer skeleton joints and signer hands. Normalization is applied on the collected frames to overcome mainly two problems: first, the variation of user position; and second, the variation of users' sizes. Feature integration is used for fusing the hand features with the skeleton features in order to form the final feature vector. Two classifiers, SVM and KNN, are applied and each one works separately on the feature vector; these classifiers work as two sources of information and the results of each classifier can be considered as the basic belief assignment (BBA). The late fusion is applied using DSmT to combine the BBA of both SVM and KNN; this includes applying the BBA fusion, applying proportional conflict redistribution 5 (PCR5) rule, calculating the pignistic probability, and finally recognizing the performed sign according to the pignistic probability. We compared the results of the DSmT with the Ad-Boosting algorithm and also applied the direct matching techniques (DTW and HMM) instead of the fusion model as an alternative method for recognition in order to compare between the fusion model and other dynamic gesture recognition techniques.

A. Data Acquisition

In this step, we used Microsoft's Kinect Version 2.0 to track the skeleton joints of the standing signer. Kinect provides the information regarding color, depth, and

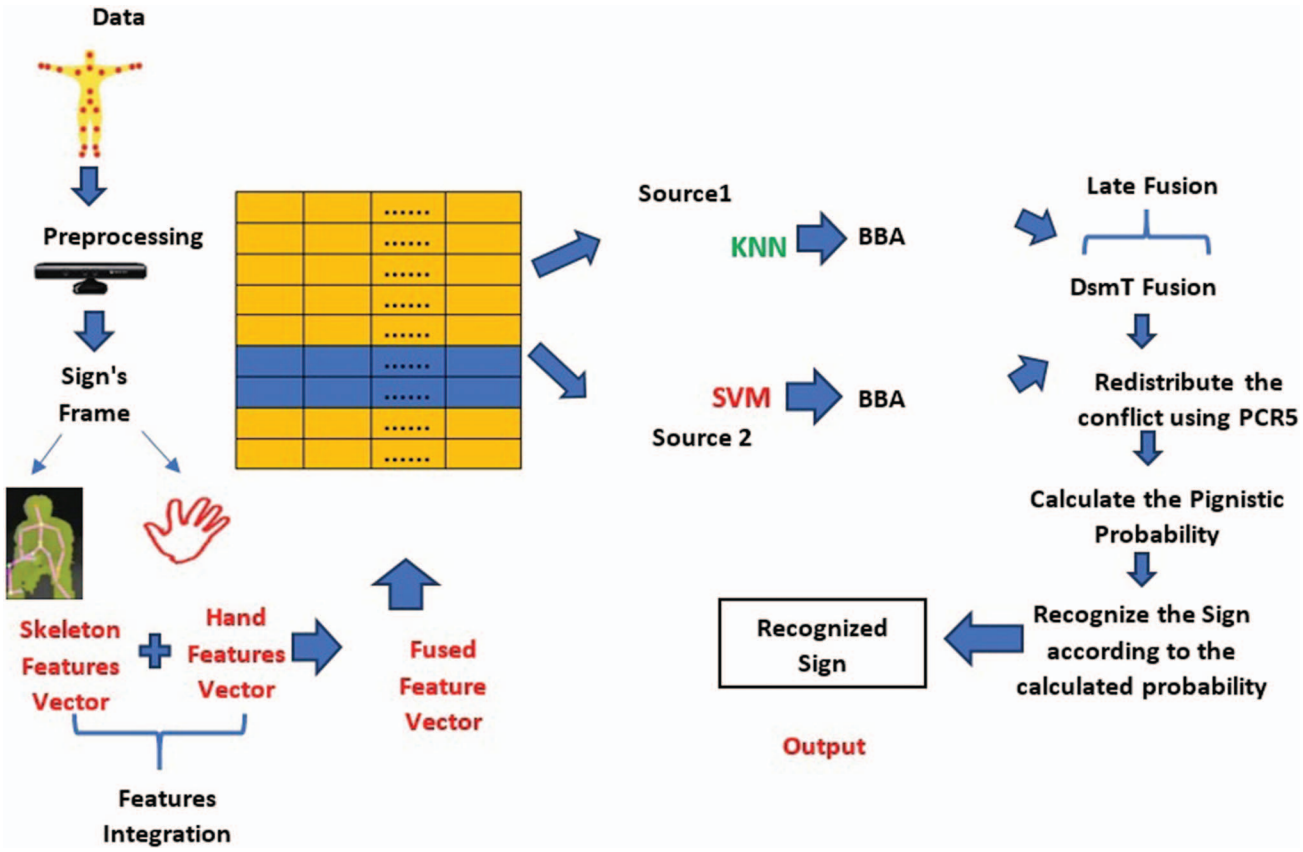


Fig. 1. Proposed model structure.

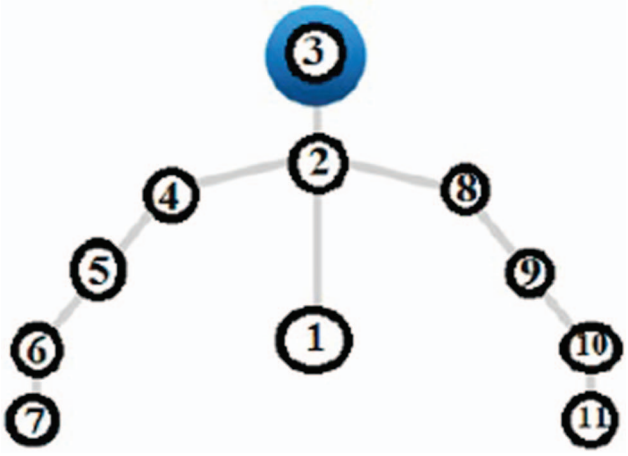


Fig. 2. Points of the upper human body joint: 1—spine, 2—shoulder center, 3—head, 4—right shoulder, 5—right elbow, 6—right wrist, 7—right hand, 8—left shoulder, 9—left elbow, 10—left wrist, and 11—left hand.

joint coordinates using its open-source SDK. The depth information is captured frame by frame. So, when Kinect depth camera starts, we capture the coordinates of 20 skeleton joints with a rate of 30 frames per second. In our system, we are interested in the upper human joint points as shown in Fig. 2.

B. Preprocessing

In this step, we are concerned with the feature extraction, preparation, and normalization for both skeleton and hands of the signer.

1) Feature Extraction The feature extraction step has a very important role in distinguishing between the captured signs. The features are extracted from the sequences of depth information. The extracted features from Kinect frames are divided into two parts: 1) skeleton joint features and 2) hand features.

a) Skelton joint features Kinect has the ability to infer the positions of the detected objects, after studying the selected signs carefully. We found that only ten joints of the skeleton are required to represent and describe the sign. These joints are hand (left and right), shoulder (left and right), elbow (left and right), wrist (left and right), spine mid, and head center. All signs are represented and performed with the upper part of the body and the lower part remains static while performing the sign. The captured frames are required to be normalized in order to overcome the variation in signer's position and signer's size.

Position normalization The signer can be in any position while performing the sign as shown in Fig. 3 and this variation can make a conflict to the model, so we performed the position normalization. The captured coordinates (X, Y, Z) for any joint are scaled by subtracting them from the spine-mid coordinates.

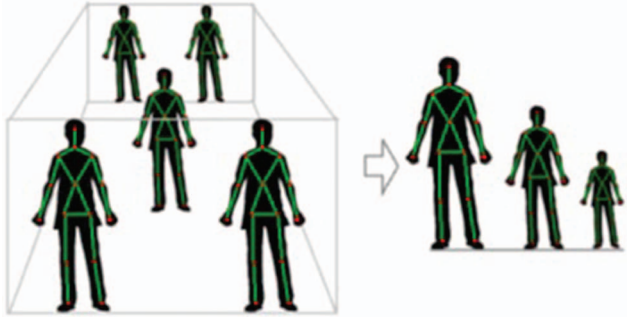


Fig. 3. Position normalization.

The coordinates of the selected joints will be converted from Cartesian coordinates X, Y, and Z into spherical coordinates that are represented by (ϕ, θ, r) as shown in Fig. 4.

The computation of the spherical coordinates is illustrated in the following equations:

$$\sum_{i=1}^{n} r(i) = \sqrt{(J(i)_x - \text{S\_M}_x)^2 + \left(J(i)_y - \text{S\_M}_y\right)^2 + \left(J(i)_z - \text{S\_M}_z\right)^2} \tag{1}$$

$$\sum_{i=1}^{n} \theta(i) = a\tan 2 \times \left(\sqrt{(J(i)_x - \text{S\_M}_x)^2 + \left(J(i)_y - \text{S\_M}_y\right)^2}, J(i)_z - \text{S\_M}_z\right) \tag{2}$$

$$\sum_{i=1}^{n} \phi(i) = a\tan 2\left(\left(J(i)_y - \text{S\_M}_y\right), \left(J(i)_x - \text{S\_M}_x\right)\right) \tag{3}$$

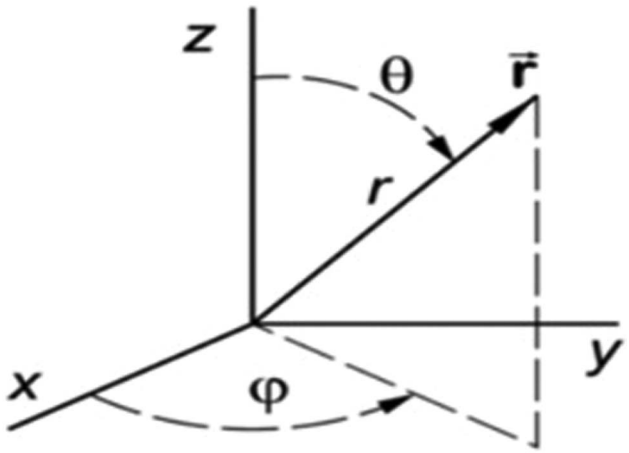


Fig. 4. Spherical coordinates.

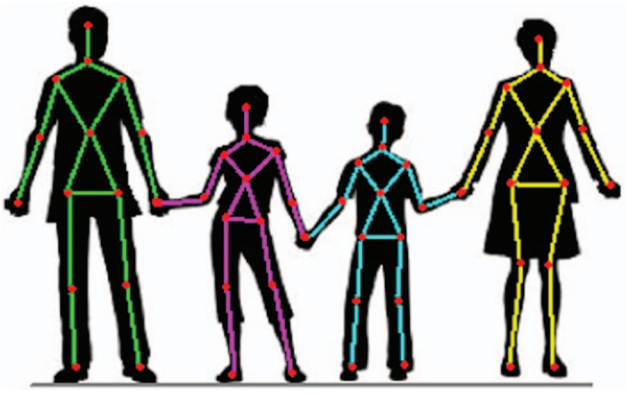


Fig. 5. Size normalization.

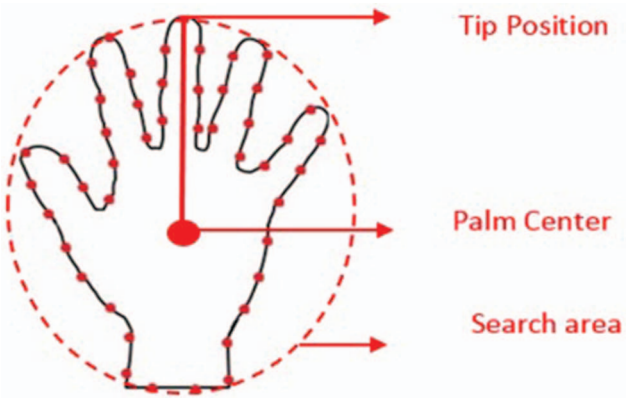


Fig. 7. Hand detection.

where
n is the number of joints from J, r is the radial distance,
S_M_x is the x coordinate of spine-mid joint,
S_M_y is the y coordinate of spine-mid joint, and
S_M_z is the z coordinate of spine-mid joint.

Size normalization To overcome the problem arising from the variation of user's size as shown in Fig. 5, we normalized all the distances that result from the position normalization step by a factor; in our model, we chose this factor as (r_{H,S_M}), which is the distance between the head and spine mid as given in the following equation:

$$\sum_{i=1}^{n} r_{\text{norm}}(i) = \frac{r(i)}{r_{H,S\_M}} \tag{4}$$

where
n is the number of joints from J,
r_{norm} is the normalized radial distance of the joint, and
r_{H,S_M} is the radial distance from head center to spine mid.

Finally, we selected another subset feature added to the spherical coordinates of the selected joints in order to enhance the recognition process as the difference in distance between hand (left and right) and shoulder (left and right). The total number of Kinect features (f_S) is about 32 in spherical coordinates. These features are denoted by ($f_1, f_2, f_3, \ldots, f_{32}$), where the feature vector consists of two sets:

1) {r, $\emptyset$} of right, left {hand, wrist, elbow, shoulder} position.

$\text{Hand}_{\text{Left } r}$, $\text{Hand}_{\text{Right } r}$, $\text{Wrist}_{\text{Left } r}$, $\text{Wrist}_{\text{Right } r}$,

$\text{Elbow}_{\text{Left } r}$, $\text{Elbow}_{\text{Right } r}$, Shoulder_Left_r,

Shoulder_Right_r

$\text{Hand}_{\text{Left } \phi}$, $\text{Hand}_{\text{Right } \phi}$, $\text{Wrist}_{\text{Left } \phi}$, $\text{Wrist}_{\text{Right } \phi}$,

$\text{Elbow}_{\text{Left } \phi}$, $\text{Elbow}_{\text{Right } \phi}$, $\text{Shoulder\_Left}_\phi$,

$\text{Shoulder\_Right}_\phi$

2) {r} of separation between right and left {hand, wrist, elbow, shoulder} as shown in Fig. 6.

b) Hand features Adding the hand features to the skeleton features will give a complete view and accurate description of the performed sign. The extraction of the hand features is based on the algorithm in [36]. The methodology of hand feature extraction starts by detecting hand joints of the tracked human body; the detected coordinates (x, y, z) for the hand represent the palm center. The next step is to specify where the search area of the hand lies; this 3-D area can be limited by the captured hand and tip position as shown in Fig. 7. After specifying the search area, all depth values that do not belong to the hand area can be excluded. The fingers can be detected by applying the algorithm of the convex hull on the search area; the edges of the convex hull above the wrist represent the fingertips as shown in Fig. 8.

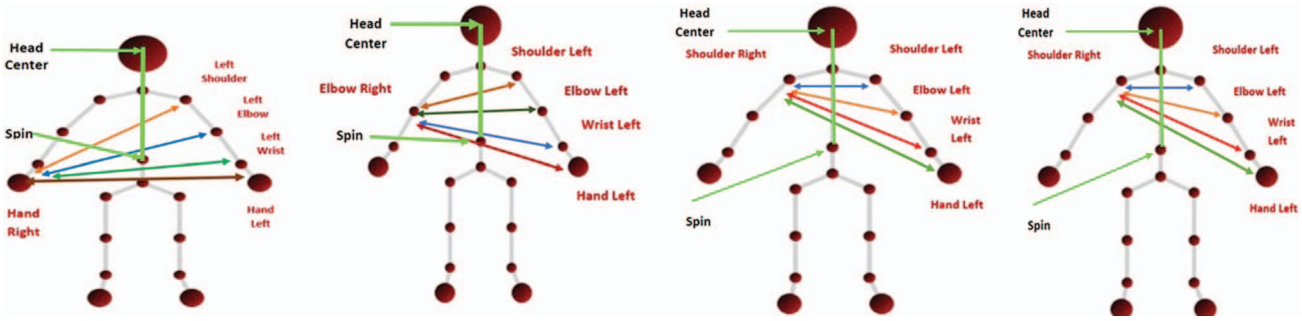


Fig. 6. Features of joint separation. (a) Hand separation. (b) Elbow separation. (c) Elbow separation. (d) Shoulder separation.

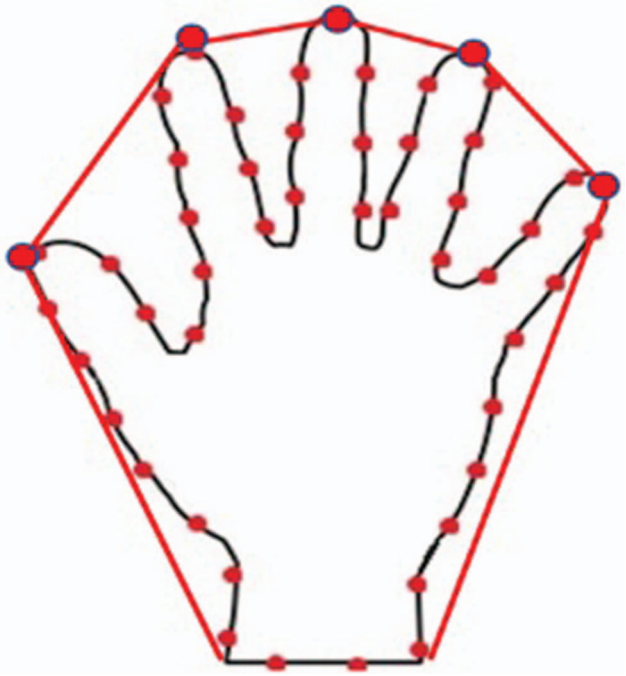


Fig. 8. Convex hull detection.

Finally, the total number of hand features (f_{H}) is about 30 in Cartesian coordinates. These features are denoted by ($f_1, f_2, f_3, \ldots, f_{30}$) where the feature vector consists of fingertip direction that is composed of 3-D data $\langle x_i, y_i, z_i \rangle$. After fingertip positions are detected, the fingers' direction vectors can be easily calculated by subtracting the tip position of each finger from the palm center $P_{\mathrm{C}}(p_x, p_y, p_z)$. The vectors that are pointing from the palm center to fingers can be calculated using the following equation:

$$V_{\mathrm{Direction}} = ((f_x - p_x), (f_y - p_y), (f_z - p_z)). \quad (5)$$

c) Feature integration Feature integration is the process of integrating the feature vectors of both skeleton joint features (f_{S}) and hand features (f_{H}) in order to produce the fused vector $f_{\mathrm{c}} = \{f_{\mathrm{S}}, f_{\mathrm{H}}\}$. The resultant fused feature vector has a dimension of 62. It should be mentioned that the data sequence is synchronized perfectly because they are coming from the same device.

C. Classification

Once the gesture features have been extracted, the descriptor of gestures that the system must classify will be formed. The goal of our system is to recognize the gestures, so after extracting the features, we applied two classifiers: KNN with $K = 1$ and SVM with RBF kernel function (gamma = 0.48 and cost = 0.5). We chose these classifiers after applying different classifiers on the test set. They gave us the best accuracy, are widely used in many pattern recognition applications such as the handwritten digit recognition [37], and are efficient in dealing with multiclass nonlinear classification problems. These two classifiers work as two sources of information. It is better to use two classifiers rather than using one classifier in order to improve the overall accuracy. The combination of information from different sources is critical, especially when developing a system that depends on conflicting, imprecise, and uncertain data. In the proposed model, each classifier takes the sequence of frames that formed the single presegmented gesture and classifies each frame separately to predict the class that frame belongs to. However, there are similarities between some gestures so that, for example, if frames of gestures enter into the classifier, the output may be classified by 70% of frames as Sign_ID = "1", 10% of frames as Sign_ID = 4, and 20% of frames as Sign_ID = 8. These values were considered as BBA that will enter the fusion phase.

To define BBA, let X be the universe that represents all possible states of a system under consideration. In the evidence theory, the BBA assigns belief mass to each element of the power set formed from the underlying universe X. We can consider the function $m : 2^x \rightarrow [0, 1]$ as a BBA, when two conditions occurred:

1) The mass of the empty set is 0 (i.e., $m(\emptyset) = 0$).
2) The masses of the remaining members of the power set add up to a total of 1 ($\sum_{A \subseteq 2^X} m(A) = 1$).

D. Late Fusion

The late fusion occurs by combining the results of the two classifiers (SVM and KNN) and applying the rules of DSmT. The fusion of these classifiers was done on the measurement level, which is more confident. The evidence (results of the classifier) is considered as BBA.

1) Dezert–Smarandache Theory DSmT is an effective fusion method. It can deal with the uncertainty and the data coming from highly conflicting sources. It allows the combination of information that is coming from different independent sources; this information is represented in terms of belief function. Dezert–Smarandache rules combine the conflict evidence accurately, so it is very successful in problems of object recognition [38].

DSmT is a theory of plausible and paradoxical reasoning; it overcame the limitations of DST [39]. We can summarize the comparison between DST and DSmT as follows.

Let $\vartheta = \{\theta_1, \ldots, \theta_n\}$ is a finite set of hypotheses.

- The DST considers a discrete and finite frame of discernment θ based on a set of exhaustive and exclusive elementary elements θ.
- The bodies of evidence are assumed independent and provide their own belief function on the power set θ but with same interpretation for θ [39].

DSmT has two types of models: 1) a free model that combines the evidence without taking the integrity constraint into consideration and 2) a hybrid model that

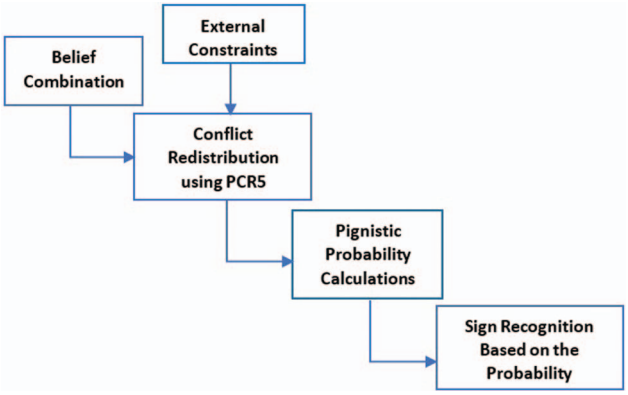


Fig. 9. Fusion framework.

includes all operators such as union and intersection and the constraints that are required to build the class Φ, so it is used in real applications.

Based on that model, the hyper-power set is given by $\vartheta = \{\theta_1, \ldots, \theta_n\}$ as a finite set (called frame) of n exhaustive elements. The free Dedekind's lattice denoted hyper-power set D^θ is defined as

1) $\emptyset, \theta_1, \ldots, \theta_n \in D^\theta$.
2) If $A,\ B \in D^\theta,$ then $A \cap B$ and $A \cup B$ belong to D^θ.
3) No other elements belong to D^θ, except those obtained by using rules 1 or 2 [38].

2) Basic Belief Assignment For any finite discrete frame ϑ, we define a belief assignment as a mapping $m(\cdot)$: $G^\theta \rightarrow [0, 1]$ associated with a given body of evidence, B, that satisfies the following conditions:

$$m(\emptyset) = 0 \quad \text{and} \quad \sum_{A \in G^\theta} m(A) = 1 \tag{6}$$

where G^θ is a hyper-power set of Θ, which is $= \{\emptyset, \theta_1, \theta_2, \theta_1 \cap \theta_2, \theta_1 \cup \theta_2\}$.

In (6), $m(A)$ is the generalization of BBA/mass, where the belief function is defined as

$$\text{Bel}(A) \triangleq \sum_{\substack{B \subseteq A \\ B \in G^\theta}} m(B). \tag{7}$$

In DSmT, there is a two-level process: credal (for combination of evidences) and pignistic (for decision making); i.e., when we need to take a decision, we should depend on a probability function. The classical pignistic probability transformation is defined as [38]

$$\text{BetP}\{A\} = \sum_{X \in 2^\theta} \frac{|X \cap A|}{|X|} m(X) \tag{8}$$

where $|x|$ denotes the cardinality of x (with convention $|\emptyset|/|\emptyset| = 1$, when defining BetP$\{\emptyset\}$). Decisions are achieved by computing the expected utilities of the acts using the subjective/pignistic BetP$\{\cdot\}$ as the probability function needed to compute expectations. It is easy to show that BetP$\{\cdot\}$ is a proper probability function [38].

TABLE I
BBA: Stage 1

	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"	"10"
BBA (S1/KNN)	0.8	0	0	0.2	0	0	0	0	0	0
BBA (S2/SVM)	0.75	0.25	0	0	0	0	0	0	0	0

3) Fusion Framework As introduced earlier, we used the DSmT for the beliefs of each evidence, and then applied the combination rule. We summarize the fusion framework in Fig. 9.

The belief calculation is computed using (6) and (7) and then the conflict is redistributed using PCR5 rule, which is the mathematical form to redistribute the conflicting mass to nonempty sets; the conflict mass should be distributed to the elements that are involved in the partial conflict with respect to their mass, considering the canonical form of the partial conflict as in (9). Finally, the pignistic probability is calculated using (8) in order to decide the performed sign according to the highest probability.

$$m_{\text{PCR5}}(X) = m_{12}(X) + \sum_{\substack{Y \in G^\theta \{X\} \\ X \cap Y = \emptyset}} \left[\frac{m_1(X)^2 m_2(Y)}{m_1(X) + m_2(Y)} + \frac{m_2(X)^2 m_1(Y)}{m_2(X) + m_1(Y)} \right]. \tag{9}$$

The dataset contains 40 signs, so it is divided into four parts in order to simplify the calculation. Sign_ID = "1" is chosen as a common sign between the divided datasets in order to relate them to each other. When the test sign enters into the system, it will pass four stages of fusion with each divided dataset. The goal is to calculate the ranked pignistic probability in order to recognize the performed sign.

The following calculation represents "tested sign with Sign_ID = 1" when entered into the fusion framework. As mentioned earlier, the first step of the model is applying the classification using the two classifiers (SVM and KNN) as two sources of information. Table I presents the results of the two classifiers, which are considered as BBA. The first stage of fusion is done with the first dataset that contains the signs with ID = 1, 2, …, 10.

The second step is applying the classical DSM combination rule. Table II presents the fusion results of the

TABLE II
Fusion Output: Stage 1

	"1"	"2"	"4"	"1 ∩ 2"	"1 ∩ 4"	"2 ∩ 4"
m_{DSm}	0.6	0	0	0.2	0.15	0.05

TABLE III
PCR5 Output: Stage 1

	"1"	"2"	"4"	"1 ∩ 2"	"1 ∩ 4"
m_{PCR5}	0.6	0.025	0.025	0.2	0.15

beliefs after applying the fusion rules.

$$m_{12}(A) \triangleq \sum_{\substack{X_1,X_2,\ldots,X_k \int D^\theta \\ (X_1\cap X_2\cap\cdots\cap X_k)=A}} \prod_{i=1}^{k} m_i(X_i) \tag{10}$$

$$(1) = m_1(1)\cdot m_2(1) = 0.8\times 0.75 = 0.6$$

$$(2) = m_1(2)\cdot m_2(2) = 0\times 0.25 = 0$$

$$(4) = m_1(4)\cdot m_2(4) = 0.2\times 0 = 0$$

$$(1\cap 2) = m_1(1)\cdot m_2(2) + m_2(1)\cdot m_1(2) = 0.2$$

$$(1\cap 4) = m_1(1)\cdot m_2(4) + m_2(1)\cdot m_1(4) = 0.15$$

$$(2\cap 4) = m_1(2)\cdot m_2(4) + m_2(2)\cdot m_1(4) = 0.05.$$

Consequently, redistribute the conflict factor using PCR5 rule.

Redistribute: "$2\cap 4 = \Phi$".

So, we will distribute this conflict proportionally.

$$m_{12}(2) = 0.025$$

$$m_{12}(4) = 0.025.$$

Table III presents the values of the beliefs after applying PCR5 rule.

The pignistic probability can be obtained from the above beliefs using (8). Table IV presents the pignistic probability.

$$\text{CM}(1) = 3,\ \text{CM}(2) = 2,\ \text{CM}(4) = 2,$$

$$\text{CM}(1\cap 2) = 1,\ \text{and CM}(1\cap 4) = 1$$

$$P(1) = \frac{1}{2}\times m_{12}(2) + \frac{1}{3}\times m_{12}(1) + \frac{1}{2}\times m_{12}(4) = 0.225$$

$$P(2) = \frac{1}{2}\times m_{12}(2) + \frac{1}{3}\times m_{12}(1) = 0.2125$$

$$P(4) = \frac{1}{2}\times m_{12}(4) + \frac{1}{3}\times m_{12}(1) = 0.2125$$

$$P(1\cap 2) = \frac{1}{1}\times m_{12}(1\cap 2) = 0.2$$

$$P(1\cap 4) = \frac{1}{1}\times m_{12}(1\cap 2) = 0.15.$$

TABLE IV
Pignistic Probability Output: Stage 1

	"1"	"2"	"4"	"1 ∩ 2"	"1 ∩ 4"
Probability	0.225	0.2125	0.2125	0.2	0.15

TABLE V
BBA: Stage 2

	"1"	"11"	"12"	"13"	"14"	"15"	"16"	"17"	"18"	"19"	"20"
BBA (S1/KNN)	0.64	0	0	0	0.2	0.16	0	0	0	0	0
BBA (S2/SVM)	0.7	0	0	0	0.2	0.1	0	0	0	0	0

Again, reprocess the sign in the second stage with the second dataset, where ID = 11, 12, …, 20. Table V presents the BBA for the second group.

Apply the classical DSM combination rule as in (10). Table VI presents the fusion results of the second group.

$$(1) = m_1(1)\cdot m_2(1) = 0.64\times 0.7 = 0.469$$

$$(14) = m_1(14)\cdot m_2(14) = 0.2\times 0.2 = 0.04$$

$$(15) = m_1(15)\cdot m_2(15) = 0.16\times 0.1 = 0.016$$

$$(1\cap 14) = m_1(1)\cdot m_2(14) + m_2(1)\cdot m_1(14) = 0.268$$

$$(1\cap 15) = m_1(1)\cdot m_2(15) + m_2(1)\cdot m_1(15) = 0.176$$

$$(14\cap 15) = m_1(2)\cdot m_2(4) + m_2(2)\cdot m_1(4) = 0.052.$$

Consequently, redistribute the conflict factor using PCR5 rule. Table VII presents the beliefs of the second group after redistributing the conflict using PCR5 rule.

Redistribute: Φ = "$14\cap 15$".

So, we will distribute this conflict proportionally.

$$m_{12}(14) = 0.04 + (0.72\times 0.052) = 0.07744$$

$$m_{12}(15) = 0.016 + (0.28\times 0.052) = 0.03056.$$

The pignistic probability can be obtained from the above beliefs using (8). Table VIII presents the pignistic probability.

$$\text{CM}(1) = 3,\ \text{CM}(2) = 2,\ \text{CM}(4) = 2,$$

$$\text{CM}(1\cap 2) = 1,\ \text{and CM}(1\cap 4) = 1$$

$$P(1) = \frac{1}{2}\times m_{12}(14) + \frac{1}{3}\times m_{12}(1) + \frac{1}{2}\times m_{12}(15) = 0.203$$

$$P(14) = \frac{1}{2}\times m_{12}(14) + \frac{1}{3}\times m_{12}(1) = 0.18853$$

$$P(15) = \frac{1}{2}\times m_{12}(15) + \frac{1}{3}\times m_{12}(1) = 0.16447$$

$$P(1\cap 14) = \frac{1}{1}\times m_{12}(1\cap 14) = 0.268$$

$$P(1\cap 15) = \frac{1}{1}\times m_{12}(1\cap 15) = 0.176.$$

TABLE VI
Fusion Output: Stage 2

	"1"	"14"	"15"	"1 ∩ 14"	"1 ∩ 15"	"14 ∩ 15"
m_{DSm}	0.448	0.04	0.016	0.268	0.176	0.052

TABLE VII
PCR5 Output: Stage 2

	"1"	"14"	"15"	"1 ∩ 14"	"1 ∩ 15"
m_{PCR5}	0.448	0.07744	0.03056	0.268	0.176

Again, reprocess the sign in the third stage with the third dataset, where ID = 21, 22, …, 30. Table IX presents the BBA of the third group.

Apply the classical DSM combination rule as in (10). Table X presents the fusion results of the third group.

$$(1) = m_1(1) \cdot m_2(1) = 0.8 \times 0.7 = 0.56$$

$$(24) = m_1(24) \cdot m_2(24) = 0.2 \times 0.1 = 0.02$$

$$(26) = m_1\ (26) \cdot m_2(26) = 0 \times 0.2 = 0$$

$$(1 \cap 24) = m_1(1) \cdot m_2(24) + m_2(1) \cdot m_1(24) = 0.22$$

$$(1 \cap 26) = m_1(1) \cdot m_2(26) + m_2\ (1) \cdot m_1(26) = 0.16$$

$$(24 \cap 26) = m_1(2) \cdot m_2(4) + m_2(2) \cdot m_1(4) = 0.04.$$

Consequently, redistribute the conflict factor using PCR5 rule. Table XI presents the beliefs of the third group after redistributing the conflict using PCR5 rule.

Redistribute: = Φ"24 ∩ 26".

So, we will distribute this conflict proportionally.

$$m_{12}(24) = 0.02 + (0.04) = 0.06.$$

The pignistic probability can be obtained from the above beliefs using (8). Table XII presents the pignistic probability.

$$\text{CM}(1) = 3,\ \text{CM}(2) = 2,\ \text{CM}(4) = 2,$$

$$\text{CM}(1 \cap 2) = 1,\ \text{and CM}(1 \cap 4) = 1$$

$$P(1) = \frac{1}{2} \times m_{12}(24) + \frac{1}{3} \times m_{12}(1) + \frac{1}{2} \times m_{12}(26) = 0.2167$$

$$P(24) = \frac{1}{2} \times m_{12}(24) + \frac{1}{3} \times m_{12}(1) = 0.217$$

$$P(26) = \frac{1}{2} \times m_{12}(15) + \frac{1}{3} \times m_{12}(1) = 0.1867$$

$$P(1 \cap 24) = \frac{1}{1} \times m_{12}(1 \cap 14) = 0.22$$

$$P(1 \cap 26) = \frac{1}{1} \times m_{12}(1 \cap 15) = 0.16.$$

TABLE VIII
Pignistic Probability Output: Stage 2

	"1"	"14"	"15"	"1 ∩ 14"	"1 ∩ 15"
DSm	0.203	0.18853	0.16447	0.268	0.176

TABLE IX
BBA: Stage 3

	"1"	"21"	"22"	"32"	"24"	"25"	"26"	"27"	"28"	"29"	"30"
BBA (S1/KNN)	0.8	0	0	0	0.2	0	0	0	0	0	0
BBA (S2/SVM)	0.7	0	0	0	0.1	0	0.2	0	0	0	0

Again, reprocess the sign in the fourth stage with the fourth dataset, where ID = 31, 32, …, 40. Table XIII presents the BBA of the fourth group.

Apply the classical DSM combination rule as in (10). Table XIV presents the fusion results of the fourth group.

$$(1) = m_1(1) \cdot m_2(1) = 0.9 \times 0.95 = 0.855$$

$$(35) = m_1(35) \cdot m_2(35) = 0.05 \times 0.03 = 0.0015$$

$$(39) = m_1(39) \cdot m_2(39) = 0.05 \times 0.2 = 0.001$$

$$(1 \cap 39) = m_1(1) \cdot m_2(39) + m_2(1) \cdot m_1(39) = 0.0655$$

$$(1 \cap 35) = m_1(1) \cdot m_2(35) + m_2(1) \cdot m_1(35) = 0.0745$$

$$(39 \cap 35) = m_1(39) \cdot m_2(35) + m_2(39) \cdot m_1(35) = 0.0025.$$

Consequently, redistribute the conflict factor using PCR5 rule. Table XV presents the beliefs of the fourth group after redistributing the conflict using PCR5 rule.

Redistribute: Φ = "35 ∩ 39".

So, we will distribute this conflict proportionally.

$$m_{12}(39) = 0.001 + (0.001) = 0.002$$

$$m_{12}(35) = 0.0015 + (0.0015) = 0.003.$$

The pignistic probability can be obtained from the above beliefs using (8). Table XVI presents the pignistic probability.

$$\text{CM}(1) = 3,\ \text{CM}(2) = 2,\ \text{CM}(4) = 2,$$

$$\text{CM}(1 \cap 2) = 1,\ \text{and CM}(1 \cap 4) = 1$$

$$P(1) = \frac{1}{2} \times m_{12}(39) + \frac{1}{3} \times m_{12}(1) + \frac{1}{2} \times m_{12}(35) = 0.2875$$

$$P(35) = \frac{1}{2} \times m_{12}(35) + \frac{1}{3} \times m_{12}(1) = 0.2865$$

$$P(39) = \frac{1}{2} \times m_{12}(15) + \frac{1}{3} \times m_{12}(1) = 0.286$$

TABLE X
Fusion Output: Stage 3

	"1"	"24"	"26"	"1 ∩ 24"	"1 ∩ 26"	"24 ∩ 26"
m_{DSm}	0.56	0.02	0	0.22	0.16	0.04

TABLE XI
PCR5 Output: Stage 3

	"1"	"24"	"26"	"1 ∩ 24"	"1 ∩ 26"
m_{PCR5}	0.56	0.06	0	0.22	0.16

$$P(1 \cap 39) = \frac{1}{1} \times m_{12}(1 \cap 14) = 0.0655$$

$$P(1 \cap 35) = \frac{1}{1} \times m_{12}(1 \cap 15) = 0.0745.$$

Table XVII shows the results of pignistic probabilities of the four stages by combining Tables IV, VIII, XII, and XVI in which the Sign_ID is chosen by taking the higher probability value. So, from Table XVII the performed sign is Sign_ID = 1 with the highest probability of 0.2875.

E. Ada-Boosting and Majority Voting

It is one of the fusion techniques that was first introduced by Freund and Schapire in 1996. The ensemble classifier is constructed from multiple weak classifiers. The single classifier can act poorly, but the results of multiple classifiers are expected to be more accurate. We applied the Ada-Boosting method on the two classifiers (SVM and KNN) and because we deal with dynamic gestures (i.e., the captured sign cannot be represented by single frame), so after applying the Ada-Boosting on the received frames, the result is finally selected by majority voting. The main idea of Ada-Boosting is to give higher importance to the more accurate classifiers in the sequence, so it starts by giving equal weights to each observation in dataset. If the prediction of the first classifier is incorrect, then it gives the highest weight to the observation with incorrect prediction. Fig. 10 presents the Ada-Boosting of two classifiers (SVM and KNN).

The algorithm of Ada-Boosting classifier is as follows.

TABLE XII
Pignistic Probability Output: Stage 3

	"1"	"24"	"26"	"1 ∩ 24"	"1 ∩ 26"
DSm	0.2167	0.2167	0.1867	0.22	0.16

TABLE XIII
BBA: Stage 4

	"1"	"31"	"32"	"33"	"34"	"35"	"36"	"37"	"38"	"39"	"40"
BBA (S1/KNN)	0.9	0	0	0	0	0.05	0	0	0	0.05	0
BBA (S2/SVM)	0.95	0	0	0	0	0.03	0	0	0	0.02	0

TABLE XIV
Fusion Output: Stage 4

	"1"	"39"	"35"	"1 ∩ 39"	"1 ∩ 35"	"35 ∩ 39"
m_{DSm}	0.855	0.001	0.0015	0.0655	0.0745	0.0025

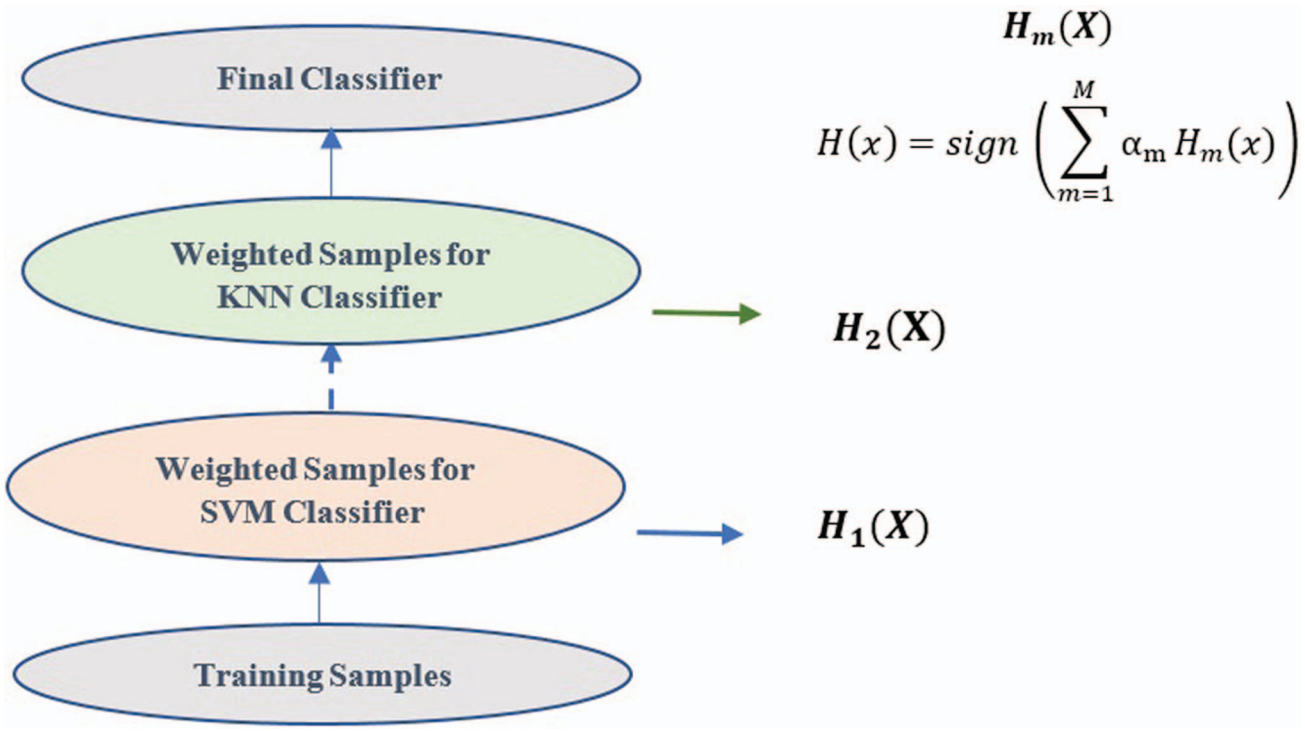


Fig. 10. Ada-Boosting structure.

TABLE XV
PCR5 Output: Stage 4

	"1"	"39"	"35"	"1 ∩ 39"	"1 ∩ 35"
m_{PCR5}	0.855	0.002	0.003	0.0655	0.0745

Table XVI
Pignistic Probability Output: Stage 4

	"1"	"39"	"35"	"1 ∩ 39"	"1 ∩ 35"
DSm	0.2875	0.286	0.2865	0.0655	0.0745

Input: Let $\{(x_1, y_y), \ldots, (x_n, y_n)\}$ be a training set, where n is the number of patterns.

Output: The ensembled classifier $H(x) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m H_m(x)\right)$, where m is the number of classifiers (SVM, KNN).

a) Initialize the weights w_i, $w_i = 1/n$, where $i \in \{1, \ldots, n\}$.

From $m = 1$ to M

b) Train the weak classifier H_m with the new weighted training data.

c) Calculate the error rate err_m of the classifier.

d) Calculate the classifier contribution $\alpha_m = 0.5 \times \log[(1 - \text{err}_m)/\text{err}_m]$.

e) Update the classifiers weights $w_i \leftarrow w_i \exp(-\alpha_m I(y_i \neq H_m(x_i)))$, where w_i is the weight of each input sequence and I is an indicator function:

$$I = \begin{cases} -1, & \text{prediction incorrect, then increase } w_i \\ 1, & \text{prediction correct, then decrease } w_i \end{cases}.$$

f) Output the ensemble classifier

$$H(x) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m H_m(x)\right).$$

F. Dynamic Time Wrapping

DTW is a pattern recognition algorithm that is widely used with dynamic gestures. It applies a direct matching technique because it tries to match the tested gesture with the most similar stored sign in the training set irrespective of the sign's length depending on measuring the distance between the two series. It tries to find the optimal alignment between two time series sequences that are varying in their speed or their time and also have different lengths. For our model, the system receives the sign (test sign) as a set of frames and DTW compares these sequences of frames with stored signs' sequences in the training set; the sequences that are compared must be wrapped in the time dimension to compute the DTW similarity coefficient. The similarity depends on the calculated distance for each sign and then the test sign is matched with the sign that has least distance. Fig. 11 presents the process of alignment for two time-independent sequences. Given two time series $X = (x_1, x_2, x_4, \ldots, x_n)$ with length $n \in N$ and $Y = (y_1, y_2, y_4, \ldots, y_n)$ with length $m \in N$ and let $\mathcal{F}$ be a feature space where X_n,$Y_m \in \mathcal{F}$. DTW will analyze the sequences in order to find the similarities between them and finally find optimal alignment $O(nm)$ [40]. To compare x and y sequences, we need to find the local cost matrix that represents the cost distribution between each two elements in the two sequences as given in the following equation:

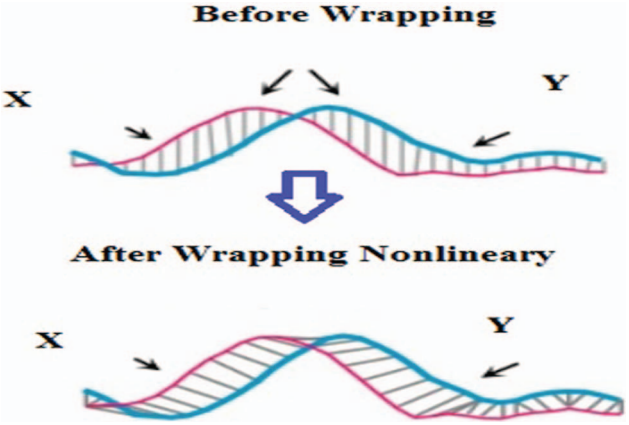


Fig. 11. DTW alignment process.

$$C : \mathcal{F} \times \mathcal{F}. \tag{11}$$

The value of C represents the similarity between the stored sign (x) and the test sign (y); if they are similar, this value must be small, else it must be large, to generate the local cost matrix with a dimension of $(n \times m)$ as shown in Fig. 12. The cost of any position at the local cost matrix $M(i, j)$ can be determined as follows:

$$M(n, m) = d(n, m) + \min\{M(n-1, m-1), M(n-1, m), M(n, m-1)\}. \tag{12}$$

This equation has two parts: the first part is the Euclidean distance $d(i, j)$ between the feature vectors

TABLE XVII
Pignistic Probability Output

Sign index	"1"	"2"	"4"	"1 ∩ 2"	"1 ∩ 4"	"14"	"15"	"1 ∩ 14"	"1 ∩ 15"
BetP{·}	0.2875	0.2125	0.2125	0.2	0.15	0.18853	0.16447	0.268	0.176
Sign index	"24"	"26"	1 ∩ 24""	1 ∩ 26""	"39"	"35"	"1 ∩ 39"	"1 ∩ 35"	–
BetP{·}	0.2167	0.1867	0.22	0.16	0.286	0.2865	0.0655	0.0745	–

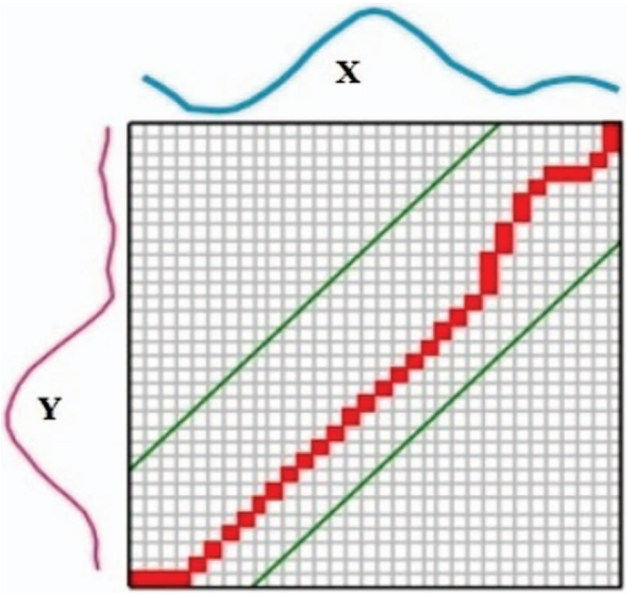


Fig. 12. Local cost matrix.

of the sequences X and Y, and the second part is the minimum cost of the adjacent elements of the cost matrix up to that point [41].

After getting the local cost matrix, we must find the wrapping path through it by applying the following equation to get the wrapping list:

$$\mathrm{wp}_{n,m} = \min\left(c_{n-1,m-1}, c_{n-1,m}, c_{n,m-1}\right) \tag{13}$$

where C is the cost value of each element in the cost matrix.

Finally, apply the following distance equation on the wrapping list in order to calculate the DTW distance:

$$\mathrm{DTW}_d = \frac{1}{p}\sum_{i=1}^{p} w_i \tag{14}$$

where w is the value of each element in the wrapping path.

Fig. 13 is an illustrative example for applying DTW in our model. Let us have a dataset of three stored signs, and a new sign is performed. The DTW algorithm computes the distance between the captured sign and each sign in the training set. Finally, the algorithm matches the sign with the stored sign that corresponds to the least distance.

G. Hidden Markov Model

HMM is a statistical model and time-domain process. It represents the statistical behavior for the observed sequence using a set of hidden states called "hidden network." The model can transition from one state to another with probability assignment [41]. The expression "hidden" comes from the fact that the Markov model constructs a sequence of hidden states from the observed sequence. HMM was successful and achieved a good accuracy with the applications of speech recognition and it is noted that there are similarities between the nature of speech and dynamic gestures [42].

$Q = q_1, q_2, q_3, \ldots, q_n$, a set of n states.

$\pi = \pi_1, \pi_2, \pi_3, \ldots, \pi_n$, the probability distribution over the states.

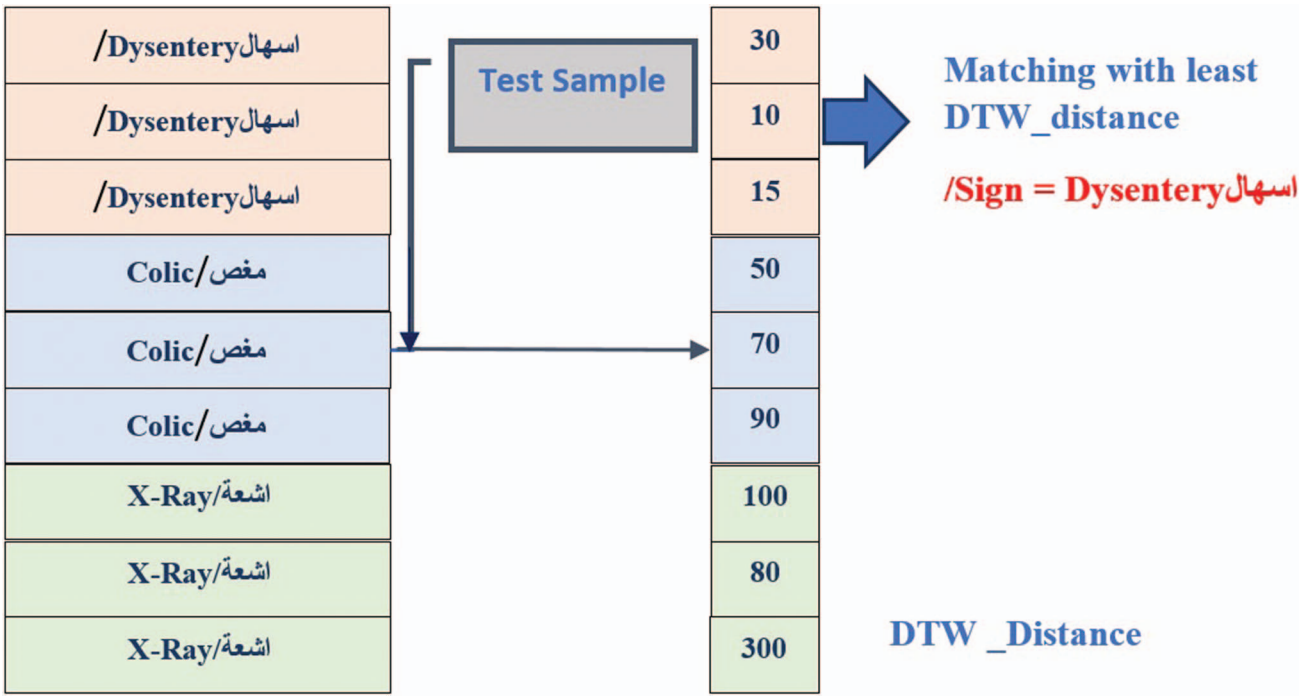


Fig. 13. DTW classification example.

$A = a_{01}, a_{02}, \ldots, a_{n1}, \ldots, a_{nn}$, the matrix A of transition probability that contains the transition probability for the transition from one state to another.

$B = b_j(O_k)$, the observation probability from state j and the observing sequence O_k.

$O = o_1, o_2, \ldots, o_T$, a sequence of T observations.

q_0, q_F, start state and end (final) state.

There are two axioms in the HMM: 1) from the law of probability, the sum of all values on the directed arcs from a given state to other must equal 1 as in (15); and 2) the sum of all π probabilities must equal 1 as in (16).

$$\text{Axiom \#1:} \quad \sum_{i=1}^{n} a_{ij} = 1 \tag{15}$$

$$\text{Axiom \#2:} \quad \sum_{i=1}^{n} \pi_i \,. \tag{16}$$

The Markov model assumed two important assumptions: 1) the probability of each state depends only on the previous state in the state sequence as in (17); and 2) the probability of any observation o_i depends only on the state q_i that produced the observation and not on any other states or any other observations as in (18).

Markov assumption #1:

$$P(q_i|q_1, \ldots, q_{i-1}) = P(q_i|q_{i-1}) \tag{17}$$

Markov assumption #2:

$$P(o_i|q_1, \ldots, q_i, \ldots, q_T, o_1, \ldots, o_i, \ldots, o_T) = P(o_i|q_i). \tag{18}$$

For our model, there are two phases:

1) *Training phase*: In this phase, we fed the model with all gesture sequences and their feature vector to build the model for each sequence and then re-estimate the probability distribution using the Baum–Welch algorithm. Also, K-means clustering is used to clusters all the 3-D sequence's points in the training set into n clusters. This will reduce the data of the stored gestures to a set of discrete states and symbols. Now each point in the training set is converted to a specific symbol that is tightly related to the clustered n states. Fig. 14 presents building of the HMM states for one gesture "Injection/حقن" as an example.
2) *Testing phase*: In this phase, we used the Viterbi decoding algorithm to match the test sign sequence with the stored sign that has highest likelihood L, which is computed using the following equation:

$$L(S_1, \ldots, S_n\,|O_1, \ldots, O_n\,) = \prod_{i=1}^{n} P\,(S_i|O_i) \cdot \prod_{i=1}^{n} P\,(S_i|O_{i-1})\,. \tag{19}$$

Fig. 15 presents the workflow of the system when using the HMM in dynamic gesture recognition.

IV. EXPERIMENTAL RESULTS

The experimental results have two aspects:

1) Recognition accuracy.
2) Latency (execution time).

First, we applied the proposed model using Microsoft Kinect V2, which consists of an IR emitter, an RGB

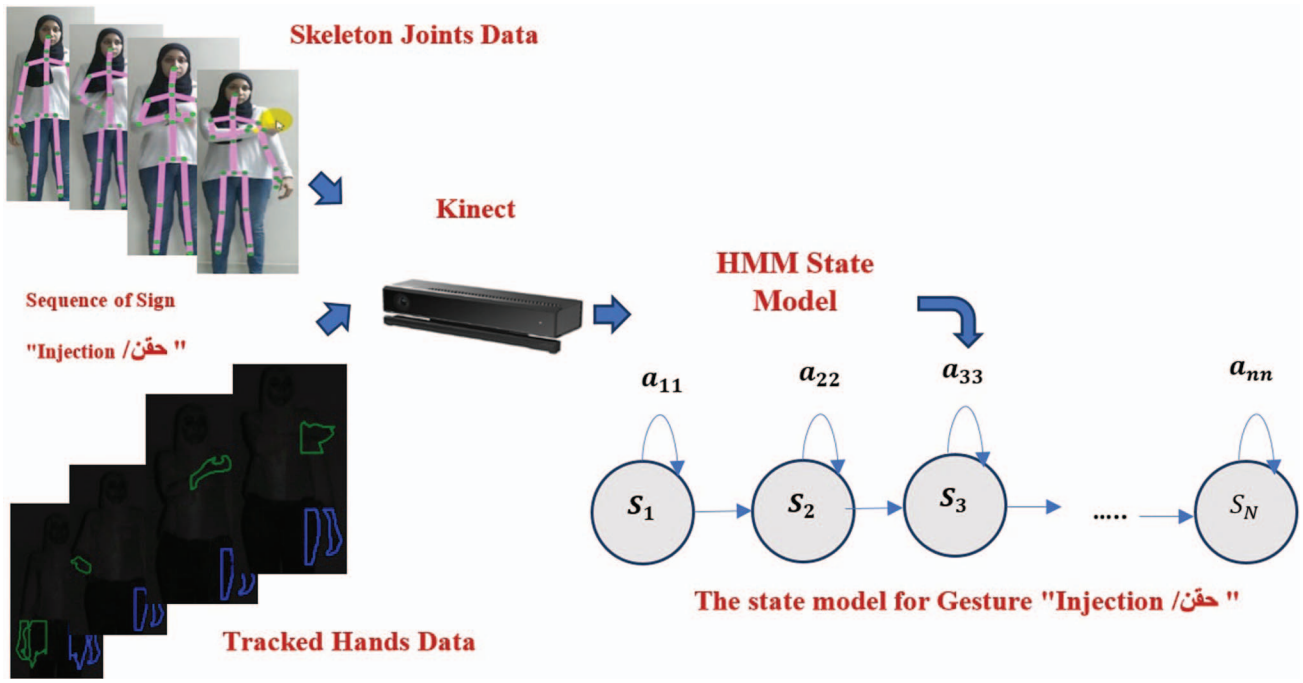


Fig. 14. HMM state model for gesture.

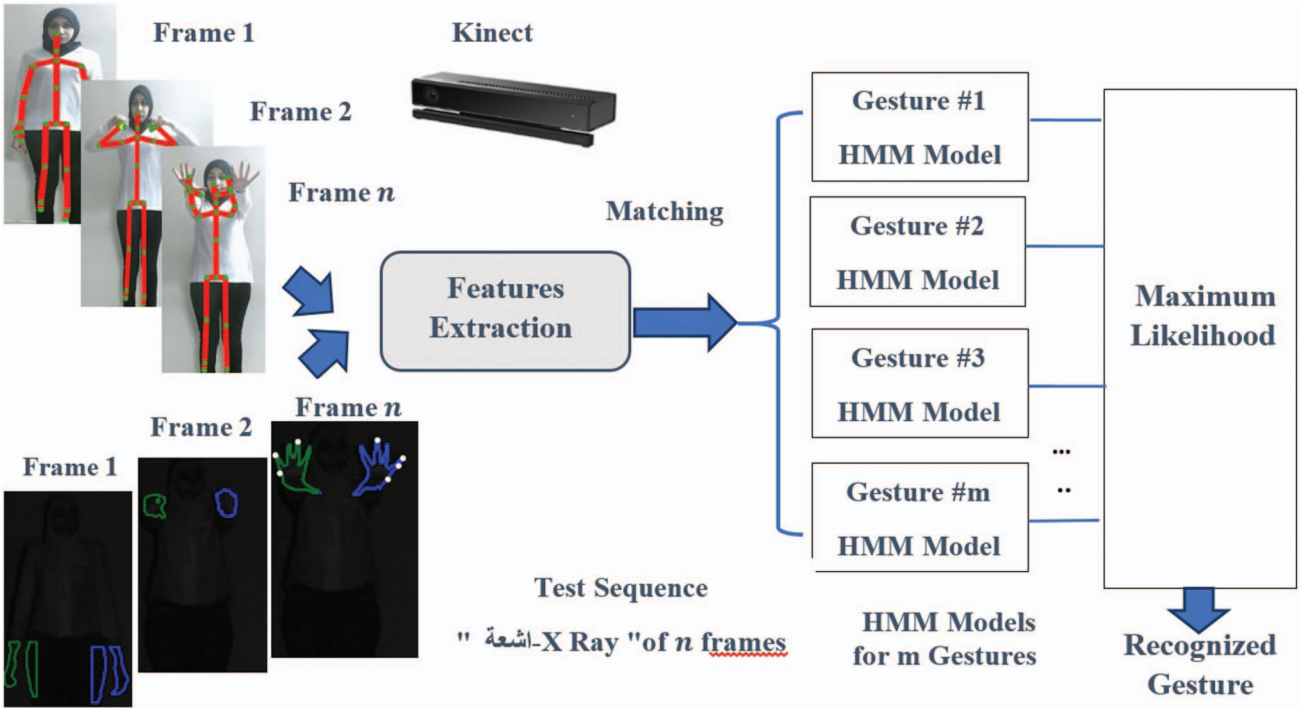


Fig. 15. HMM model for dynamic gesture recognition.

camera, an IR depth sensor, and a microphone array [42]. It is used to acquire signs and obtain the depth streams with a rate of 30 frames per second. We connected Kinect with a laptop that has a 64-bit architecture, Windows 8 operating system, 8 GB of physical memory, and Intel Core i7-5500U and 2.40 GHz with x-64-based processor. The proposed model is developed using Microsoft C# program and Microsoft Kinect SDK library.

A. Arabic Sign Dataset

We chose 40 different gestures in the medical field. They are listed in Table XVIII. We collected the data from three different volunteers in different positions and with different sizes.

B. Proposed Model Accuracy

For each sign, we collected 30 samples from three different signers and divided them into 20 for the training set and 10 for the testing set. The total samples for all signs were 1200 (800 for the training set and 400 for the testing set). The collected signs were dynamic; i.e., the sign was performed by moving body joints such as wrist, elbow, shoulder, and hands. Each sign's stream contained on average 120–200 frames, so the total number of frames was around 40,000 for the training set and 32,500 for the testing set. The feature vector was formed from the skeleton joint features and hand features that were combined to form 62 features. We applied two classifiers (SVM and KNN) in the classification phase. They were applied on the separated frames, and the accuracies were 79% and 66%, respectively. Because the selected signs

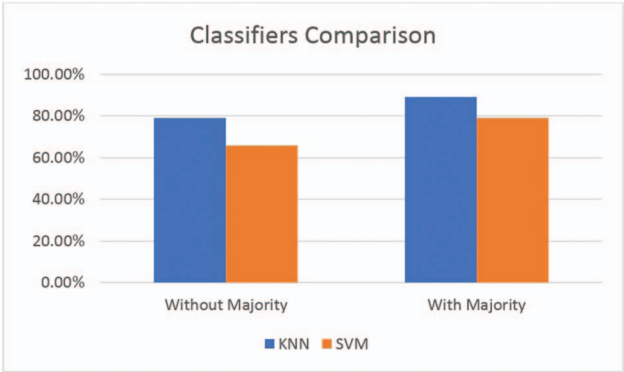


Fig. 16. Classifiers' accuracy comparison.

are dynamic in nature, we can apply the majority voting on the classified frames for each sign in order to get the accuracy of recognizing each sign. The KNN and SVM were improved after majority voting to 89% and 79%, respectively. After applying the DSmT fusion of the two classifier results, the accuracy reached 91.5%. Also, the accuracy of Ada-Boosting reached 90.2%. For the dynamic pattern recognition approach, the accuracies of DTW and HMM reached 82.6% and 79.5%, respectively. Fig. 16 presents the classifier accuracy before and after applying the majority voting for each classifier without applying the fusion. Fig. 17 presents the comparison between the accuracy of each classifier individually and after fusing their results.

It is noticed in Fig. 16 that the DSmT fusion of classifiers improves the model recognition accuracy compared to the individual classifiers and the Ada-Boosting technique. The misclassified signs using SVM reached 21% and using KNN reached 11%, while there were no

TABLE XVIII
Medical Dataset

Index	Arabic Sign	Meaning in English	Index	Arabic Sign	Meaning in English
1	اسهال	Dysentery	21	نزيف	Bleeding
2	اشعة	X-Ray	22	وفاة	Death
3	استقبال	Reception	23	طبيب عظام	Orthopedic doctor
4	رئتان	Two lungs	24	علاج طبيعى	Physical therapy
5	كبد	liver	25	حقن	Injection
6	كلى	kidneys	26	زغللة	Blurred vision
7	معدة	stomach	27	سرطان	Cancer
8	امساك	Constipation	28	جهاز قياس ضغط	Pressure measuring device
9	تحليل	analysis	29	صداع	A headache
10	تطعيم	Vaccination	30	صمم	Deafness
11	شلل	Paralysis	31	طبيب اطفال	Pediatrician
12	طبيب توليد	Obstetrician	32	طبيب انف واذن	Doctor of nose and ear
13	تقيوء	Vomiting	33	طبيب باطنة	Internist
14	تورم	Swelling	34	طبيب عام	General Doctor
15	جرح	Wound	35	كسر عظام	Broken bones
16	حامل	Pregnant	36	فيتامينات	Vitamins
17	حرارة	Fever	37	فشل كلوى	Kidney failure
18	اوردة	Veins	38	قرحة	ulcer
19	حساسية	Allergic	39	قولون	The colon
20	مغص	Colic	40	مختبر	laboratory

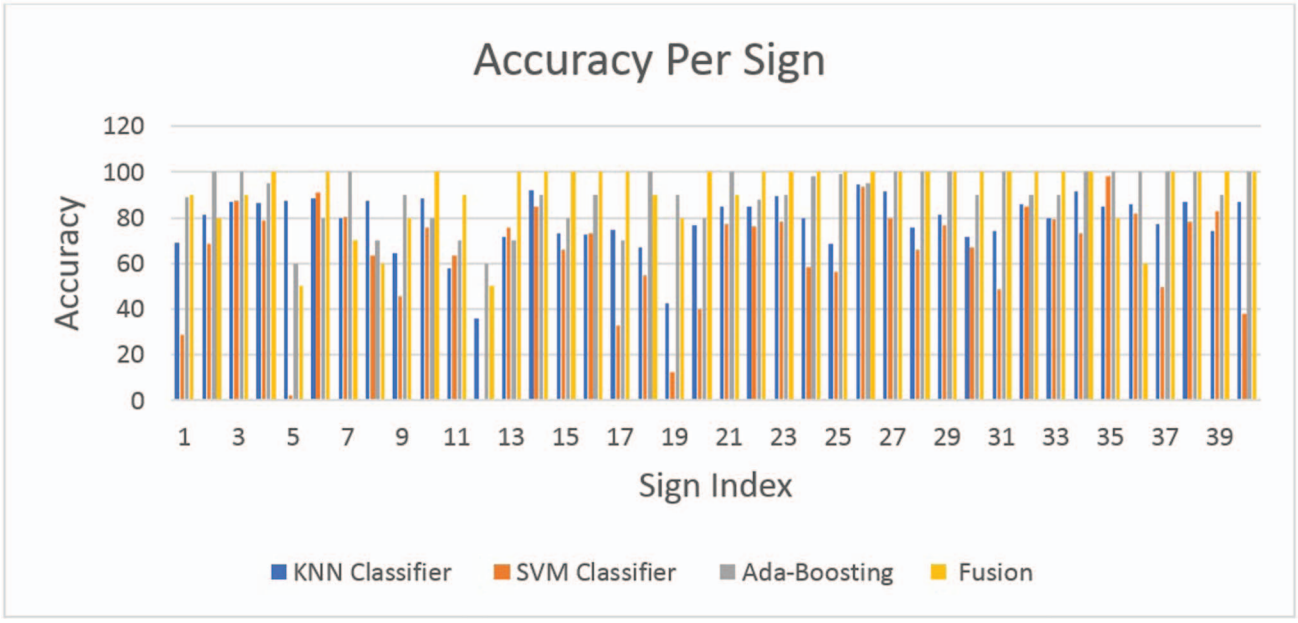


Fig. 17. Accuracy per sign comparison for classifiers versus Ada-Boosting and fusion model.

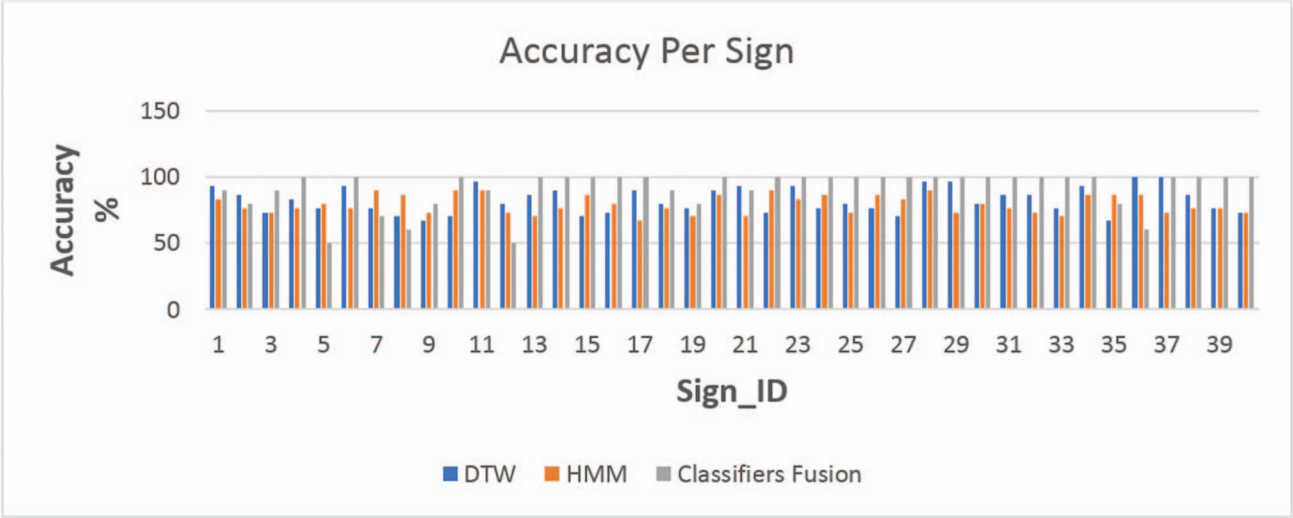


Fig. 18. Accuracy per sign comparison for DTW, HMM, and fusion model.

TABLE XIX
SVM Misclassification

Misclassified signs	Misclassification percentage
1	72%
5	98%
12	100%
17	67%
19	88%
31	51%
40	62%

misclassified signs after using DSmT. Table XIX presents the misclassified signs using only SVM, and Table XX presents the misclassified signs using only KNN.

For DTW and HMM, Fig. 18 presents the comparison between the achieved accuracy per sign using DTW, HMM, and classifier fusion.

From Fig. 17, the DSmT model is more accurate than both DTW and HMM and achieved higher recognition accuracy over the 40 signs.

The system performance is a very important metric, especially when the system works in real time, so the computation latency was computed. We took into consideration that the main processes in the system are performed sequentially and also the frames of Kinect are captured at the rate of 30 frames per second. Table XXI lists the time in seconds as the average time for each process, which was calculated over 30 experiments while performing the selected signs from the dataset. Finally, the total time will be the result of aggregating the times of all processes.

TABLE XX
KNN Misclassification

Misclassified signs	Misclassification percentage
12	64%
19	58%

TABLE XXI
Computation Latency

Process	Time (s)
Sign capturing	6
Preprocessing (data normalization)	2
KNN classifier	8
SVM classifier	5
Late fusion + DSmT fusion	7
Total	28

Also, from the experiments we found that DTW is faster than HMM. Over 30 experiments, DTW takes 5 s on average for recognition and HMM takes 7 s on average. Fig. 19 presents the processing times for both DTW and HMM for X-ray/"اشعة" over ten samples.

V. CONCLUSION AND FUTURE WORK

In this paper, we introduced an automatic system for Arabic sign recognition using Microsoft Kinect V2. The proposed model was applied and tested on 40 Arabic signs that are related to the medical field. Each sign is captured and represented as a depth stream. This stream was analyzed and normalized to overcome the variation of signer's position and size, and then the features of both skeleton (32 features) and hand (30 features) were extracted and integrated in one feature vector with 62 features. The data with these features were used to train the two classifiers (KNN and SVM). Finally, DSmT was used to combine the results of these two classifiers. Three different signers performed the signs in order to build the required dataset. The number of collected samples was 1200 (800 for the training set and 400 for the testing set). The accuracy of the classifiers was 89% for KNN and 79% for SVM. Classifiers' accuracy was compared with the fusion results, which reached 91.5%. Finally, we compared the fusion model with the Ada-Boosting technique, which achieved an accuracy of 90.2%, and other

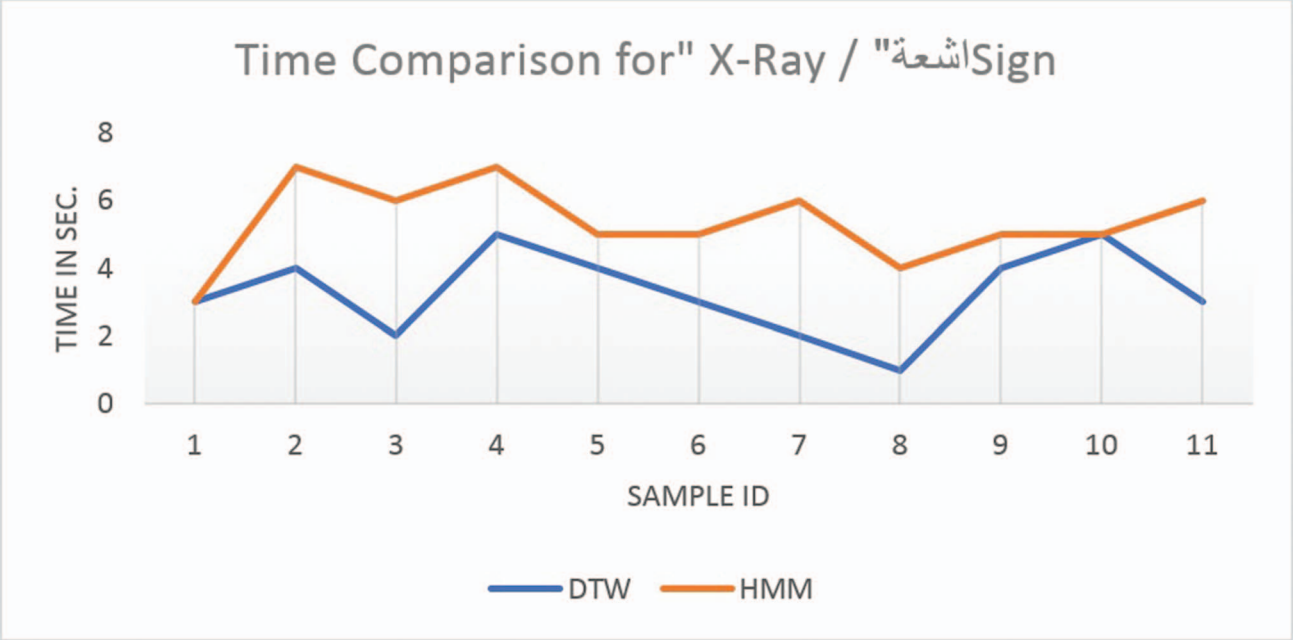


Fig. 19. DTW and HMM processing time comparison.

two algorithms that were widely used in dynamic gesture recognition. These algorithms were DTW and HMM, and they achieved an accuracy of 82.6% and 79.5%, respectively, so our model was more accurate than them. The suggested future work consists of increasing the overall accuracy of the system, improving the model in order to recognize the full sentences, and reducing the computation latency in real time. Also, we suggested the use of deep neural networks to improve the accuracy. Deep learning-based systems can learn efficiently from raw images or video sequences, so it is beneficial to process multimodality data such as the RGB-D data, skeleton, finger points, etc. that can provide rich and wide information of signers' movements. Finally, with the revolution in Internet of Things (IoT), we suggest using IoT devices like wireless wearable devices that will help in monitoring hearing-impaired people to interpret their signs and know their needs without delays.

ACKNOWLEDGMENT

This research was supported and revised by my supervisors Dr. Alaa Hamouda and Prof. Ali Rashed. The authors thank the reviewers for their effort in reviewing and for their insights and comments.

REFERENCES

[1] R. Hoopers et al.
"Analyzing variation in sign languages: Theoretical and methodological issues,"
in *Signed Languages:* Discoveries From International Research, M. Metzger, S. Taub, A. M. Baer, and V. Dively Eds., Washington, DC: Gallaudet University Press, 2001, pp. 135–162.

[2] M. Mohandes, S. Aliyu, and M. Deriche
"Prototype Arabic sign language recognition using multi-sensor data fusion of two Leap Motion controllers,"
in *Proc. 12th Int. Multi-Conf. Syst., Signals Devices*, Piscataway, NJ: IEEE, 2015.

[3] T. Dutta
"Evaluation of the Kinect™ sensor for 3-D kinematic measurement in the workplace,"
Appl. Ergonom., vol. 43, no. 4, pp. 645–649, 2012.

[4] Z. Cai, J. Han, L. Liu, and L. Shao
"RGB-D datasets using Microsoft Kinect or similar sensors: A survey,"
Multimedia Tools Appl., vol. 76, no. 3, pp. 4313–4355, 2017.

[5] C.-H. Chuan, E. Regina, and C. Guardino
"American sign language recognition using Leap Motion sensor,"
in *Proc. 13th Int. Conf. Mach. Learn. Appl.*, Piscataway, NJ: IEEE, 2014.

[6] A. Clark and D. Moodley
"A system for a hand gesture-manipulated virtual reality environment,"
in *Proc. Annu. Conf. South Afr. Inst. Comput. Scientists Inf. Technologists*, New York: ACM, 2016.

[7] D. N. Metaxas, B. Liu, F. Yang, P. Yang, N. Michael, and C. Neidle
"Recognition of nonmanual markers in American sign language (ASL) using non-parametric adaptive 2D–3D face tracking,"
in *Proc. Int. Conf. Lang. Resour. Eval.*, 2012.

[8] M. A. Abdel-Fattah
"Arabic sign language: A perspective,"
J. Deaf Stud. Deaf Educ., vol. 10, no. 2, pp. 212–221, 2005.

[9] I. El Naqa, R. Li, and M. J. Murphy
Machine Learning in Radiation Oncology: Theory and Applications. Cham, Switzerland: Springer, 2015, pp. 57–70.

[10] M. A. Almasre and H. Al-Nuaim
"Comparison of four SVM classifiers used with depth sensors to recognize Arabic sign language words,"
Computers, vol. 6, no. 2, p. 20, 2017.

[11] Y. Bengio, A. Courville, and G. E. Hinton
"Representation learning: A review and new perspectives,"
IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[12] M. Mohandes, M. Deriche, and J. Liu
"Image-based and sensor-based approaches to Arabic sign language recognition,"
IEEE Trans. Human–Mach. Syst., vol. 44, no. 4, pp. 551–557, Aug. 2014.

[13] H. Ahmed, S. O. Gilani, M. Jamil, Y. Ayaz, and S. I. A. Shah
"Monocular vision-based signer-independent Pakistani sign language recognition system using supervised learning,"
Indian J. Sci. Technol., vol. 9, no. 25, pp. 1–16, 2016.

[14] M. Maraqa and R. Abu-Zaiter
"Recognition of Arabic sign language (ArSL) using recurrent neural networks,"
in *Proc. 1st Int. Conf. Appl. Digit. Inf. Web Technol.*, Piscataway, NJ: IEEE, 2008.

[15] K. Assaleh and M. Al-Rousan
"Recognition of Arabic sign language alphabet using polynomial classifiers,"
EURASIP J. Adv. Signal Process., vol. 2005, no. 13, p. 507614, 2005.

[16] O. Al-Jarrah and F. A. Al-Omari
"Improving gesture recognition in the Arabic sign language using texture analysis,"
Appl. Artif. Intell., vol. 21, no. 1, pp. 11–33, 2007.

[17] N. El-Bendary, H. M. Zawbaa, M. S. Daoud, A. E. Hassanien, and K. Nakamatsu
"ArSLAT: Arabic sign language alphabets translator,"
in *Proc. Int. Conf. Comput. Inf. Syst. Ind. Manage. Appl.*, Piscataway, NJ: IEEE, 2010.

[18] K. Assaleh, T. Shanableh, and M. Zourob
"Low complexity classification system for glove-based Arabic sign language recognition,"
in *Proc. 19th Int. Conf. Neural Inf. Process.*, 2012, pp. 262–268.

[19] M. A. Mohandes
"Recognition of two-handed Arabic signs using the CyberGlove,"
Arabian J. Sci. Eng., vol. 38, no. 3, pp. 669–677, 2013.

[20] M. I. Sadek, M. N. Mikhael, and H. A. Mansour
"A new approach for designing a smart glove for Arabic sign language recognition system based on the statistical analysis of the sign language,"
in *Proc. 34th Nat. Radio Sci. Conf.*, Piscataway, NJ: IEEE, 2017.

[21] E. E. Hemayed and A. S. Hassanien
"Edge-based recognizer for Arabic sign language alphabet (ArS2V-Arabic sign to voice),"
in *Proc. Int. Comput. Eng. Conf.*, Piscataway, NJ: IEEE, 2010

[22] M. A. Almasre and H. A. Al-Nuaim
"Using the Hausdorff algorithm to enhance Kinect's recognition of Arabic sign language gestures,"
Int. J. Comput. Sci. Secur., vol. 7, no. 1, p. 2, 2017.

[23] M. ElBadawy, A. S. Elons, H. Sheded, and M. F. Tolba
"A proposed hybrid sensor architecture for Arabic sign language recognition,"
in *Intelligent Systems' 2014*. Cham, Switzerland: Springer, 2015, pp. 721–730.

[24] S. Aliyu, M. Mohandes, M. Deriche, and S. Badran
"Arabic sign language recognition using the Microsoft Kinect,"
in *Proc. 13th Int. Multi-Conf. Syst., Signals Devices*, Piscataway, NJ: IEEE, 2016.

[25] A. B. Jmaa, W. Mahdi, Y. B. Jemaa, and A. B. Hamadou
"Arabic sign language recognition based on HOG descriptor,"
in *Proc. 8th Int. Conf. Graph. Image Process.*, vol. 10225. Bellingham, WA: International Society for Optics and Photonics, 2017.

[26] M. Mohandes, M. A. Deriche, and S. O. Aliyu
"Arabic sign language recognition using multi-sensor data fusion,"
U.S. Patent 9 672 418, Jun. 6, 2017.

[27] O. Rashid, A. Al-Hamadi, and B. Michaelis
"A framework for the integration of gesture and posture recognition using HMM and SVM,"
in *Proc. IEEE Int. Conf. Intell. Comput. Intell. Syst.*, vol. 4. Piscataway, NJ: IEEE, 2009.

[28] Y. Song, Y. Gu, P. Wang, Y. Liu, and A. Li
"A Kinect based gesture recognition algorithm using GMM and HMM,"
in *Proc. 6th Int. Conf. Biomed. Eng. Inform.*, Piscataway, NJ: IEEE, 2013.

[29] P. V. V. Kishore and P. Rajesh Kumar
"A video based Indian sign language recognition system (INSLR) using wavelet transform and fuzzy logic,"
Int. J. Eng. Technol., vol. 4, no. 5, p. 537, 2012.

[30] B. Penelle and O. Debeir
"Multi-sensor data fusion for hand tracking using Kinect and Leap Motion,"
in *Proc. Virtual Reality Int. Conf.*, New York: ACM, 2014.

[31] G. Marin, F. Dominio, and P. Zanuttigh
"Hand gesture recognition with jointly calibrated Leap Motion and depth sensor,"
Multimedia Tools Appl., vol. 75, no. 22, pp. 14991–15015, 2016.

[32] K. Y. Fok, C. T. Cheng, and N. Ganganath
"Live demonstration: A HMM-based real-time sign language recognition system with multiple depth sensors,"
in *Proc. IEEE Int. Symp. Circuits Syst.*, Piscataway, NJ: IEEE, 2015.

[33] X. Yang, X. Chen, X. Cao, S. Wei, and X. Zhang
"Chinese sign language recognition based on an optimized tree-structure framework,"
IEEE J. Biomed. Health Inform., vol. 21, no. 4, pp. 994–1004, 2017.

[34] P. Kumar, H. Gauba, P. P. Roy, and D. P. Dogra
"Coupled HMM-based multi-sensor data fusion for sign language recognition,"
Pattern Recognit. Lett., vol. 86, pp. 1–8, 2017.

[35] Y. Sun et al.
"Gesture recognition based on Kinect and sEMG signal fusion,"
Mobile Netw. Appl., vol. 23, no. 4, pp. 797–805, 2018.

[36] M. Cheriet, N. Kharma, C. L. Liu, and C. Suen
Character Recognition Systems: A Guide for Students and Practitioners. Hoboken, NJ: Wiley, 2007.

[37] A. Lekova, D. Ryan, and R. Davidrajuh
"Fingers and gesture recognition with Kinect v2 sensor,"
Inf. Technol. Control, vol. 14, no. 3, pp. 24–30, 2016.

[38] F. Smarandache and J. Dezert
Advances and Applications of DSmT for Information Fusion, Collected Works, vol. 3. Rehoboth, DE: American Research Press, 2009, 760 pp.

[39] H. Lifang, G. Xin, and H. You
"Efficient combination rule of Dezert–Smarandache theory,"
J. Syst. Eng. Electron., vol. 19, no. 6, pp. 1139–1144, 2008.

[40] M. A. Bautista et al.
"Probability-based dynamic time warping for gesture recognition on RGB-D data,"
in *Advances in Depth Image Analysis and Applications*. Berlin, Germany: Springer, 2013, pp. 126–135.

[41] S. Mitra and T. Acharya
"Gesture recognition: A survey,"
IEEE Trans. Syst., Man, Cybern., Part C: Appl. Rev., vol. 37, no. 3, pp. 311–324, 2007.

[42] E. Gani and A. Kika
"Albanian sign language (AlbSL) number recognition from both hand's gestures acquired by Kinect sensors,"
Int. J. Adv. Comput. Sci. Appl., vol. 7, no. 7, pp. 777–780, 2016.

Basma Hisham received the B.Sc. and M.Sc. degrees from the Faculty of Engineering, Al-Azhar University, Cairo, Egypt, in 2012 and 2015, respectively. He is currently a Teacher Assistant with the Faculty of Engineering, Al-Azhar University, and previously was a Research Assistant with the National Telecommunication Institute.

Alaa Hamouda received the Ph.D. degree in computer engineering from Al-Azhar University, Cairo, Egypt. He is currently an Associate Professor with the Faculty of Engineering, Al-Azhar University. His research interests include data mining, text summarization, opinion mining, multiagent systems, Capability Maturity Model Integration (CMMI), swarm intelligence, and sensory network.

Tensor Decomposition-Based Multitarget Tracking in Cluttered Environments

FELIX GOVAERS
BRUNO DEMISSIE
ALTAMASH KHAN
MARTIN ULMKE
WOLFGANG KOCH

Many real-world applications of target tracking and state estimation are nonlinear filtering problems and can therefore not be solved by closed-form analytical solutions. In the recent past, tensor-based approaches have become increasingly popular due to very effective decomposition algorithms, which allow a compressed representation of discretized, high-dimensional data. It has been shown that by means of a Kronecker format of the Fokker–Planck equation, the Bayesian recursion for prediction and filtering can be solved for probability densities in a canonical polyadic decomposition (CPD). In this paper, the application of this approach on tracking multiple targets in a cluttered environment is presented. It is shown that intensity or probability hypothesis density-based filters can well be implemented using the CPD tensor format.

Manuscript received February 11, 2019; released for publication July 8, 2019.

The authors are with Fraunhofer FKIE, Wachtberg 53343, Germany. E-mail: {felix.govaers, bruno.demissie, altamash.khan, martin.ulmke, wolfgang.koch}@fkie.fraunhofer.de

I. INTRODUCTION

As sensors become more and more ubiquitous, powerful and practical algorithms are required to extract relevant information from noisy, contradicting, ambiguous, or erroneous measurements. Often, the theory of Bayesian state estimation is a good choice to develop algorithms that cope with these challenges. Recent results of different research groups demonstrate that the common recursion of filtering and prediction can also be solved based on a decomposed tensor representation of the underlying multivariate density function. It is well known that by means of a CANDECOMP/PARAFAC decomposition (CPD) of a discretized density, a computationally effective representation can be achieved. Despite the fact that there are still open challenges, we might say that CPD representations are the most promising candidates to beat the curse of dimensionality in many research domains, including nonlinear filtering. In this paper, we apply the CPD-based state estimation to the problem of multitarget tracking in cluttered environments. We derive all update equations for a CPD tensor in the case of a single target and multiple targets when false measurements are present. An evaluation shows the performance of the approach.

The theory of target tracking has exposed a growing family of algorithms to compute the probability density function (pdf) of a system state based on noise-corrupted sensor observations. An estimate of the state is then obtained by taking the mean of the pdf.[1] The corresponding covariance matrix additionally provides a measure of accuracy for this estimate. Bayesian estimation is the framework of recursive filtering methodologies that allow us to process a current measurement by means of a *prior* or *initial* density and a measurement likelihood function that statistically describes the performance of the sensor. Thus, a tracking algorithm is an iterative updating scheme for calculating a conditional pdf $p(\mathbf{x}_k|\mathcal{Z}^k)$ that represents all available knowledge on the object state $\mathbf{x}_k$ at some time t_k, which typically is chosen as the present time. The densities are explicitly conditioned on the sensor data time series $\mathcal{Z}^k$. The iterative scheme consists of two processing steps per update cycle, referred to as *prediction* and *filtering*. The manipulation of the probability densities is given by the following basic equations (see [1], [2] for instance).

Prediction. Assuming the Markov property of the underlying object state, the prediction density $p(\mathbf{x}_k|\mathcal{Z}^{k-1})$ is obtained by combining the evolution model $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ with the previous filtering density $p(\mathbf{x}_{k-1}|\mathcal{Z}^{k-1})$:

$$p(\mathbf{x}_{k-1}|\mathcal{Z}^{k-1}) \xrightarrow[\text{constraints}]{\text{evolution model}} p(\mathbf{x}_k|\mathcal{Z}^{k-1})$$

$$p(\mathbf{x}_k|\mathcal{Z}^{k-1}) = \int \mathrm{d}\mathbf{x}_{k-1} \; \underbrace{p(\mathbf{x}_k|\mathbf{x}_{k-1})}_{\text{evolution model}} \; \underbrace{p(\mathbf{x}_{k-1}|\mathcal{Z}^{k-1})}_{\text{previous filtering}}. \quad (1)$$

[1] Depending on the scenario, for instance, the expectation value, the maximum value, the median, or other statistics of the pdf can be used.

Filtering. The filtering density $p(\mathbf{x}_k|\mathcal{Z}^k)$ is obtained using the Bayes theorem by combining the sensor model $p(\mathbf{z}_k|\mathbf{x}_k)$, also called the "likelihood function," with the prediction density $p(\mathbf{x}_k|\mathcal{Z}^{k-1})$ according to

$$p(\mathbf{x}_k|\mathcal{Z}^{k-1}) \xrightarrow[\text{sensor model}]{\text{current sensor data}} p(\mathbf{x}_k|\mathcal{Z}^k)$$

$$p(\mathbf{x}_k|\mathcal{Z}^k) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)\, p(\mathbf{x}_k|\mathcal{Z}^{k-1})}{\int \mathrm{d}\mathbf{x}_k \underbrace{p(\mathbf{z}_k|\mathbf{x}_k)}_{\text{sensor model}} \underbrace{p(\mathbf{x}_k|\mathcal{Z}^{k-1})}_{\text{prediction}}}. \quad (2)$$

According to this paradigm, an *object track* represents all relevant knowledge on a time-varying object state of interest, including its history and measures that describe the quality of this knowledge. As a technical term, "track" is therefore either a synonym for the collection of densities $p(\mathbf{x}_l|Z^l), l = 1, \ldots, k, \ldots$, or of suitably chosen parameters characterizing them, such as estimates and the corresponding estimation error covariance matrices.

An analytical solution to a recursive computation of these densities is given for instance by the Kalman filter in the case of linear Gaussian models [3]. For nonlinear scenarios, only approximate solutions are feasible. The first-order Taylor approximation is called the *extended Kalman filter* (EKF) that has low computation cost, due to its analytic solution of the prediction and filtering steps (see [1] for instance). The performance of the linearization can be improved by means of deterministic samples chosen at the local neighborhood of the current estimate. This algorithm is known as the *unscented Kalman filter* (UKF) [4]. The term *particle filter* (PF) subsumes all kinds of numerical solutions with nondeterministic samples. Here, knowledge about the state typically is represented by a set of state samples, which implies that the density is approximated by a Dirac mixture. Because the process noise terms are simulated by means of appropriately sampled random vectors, these methods are also known as *sequential Monte Carlo* (SMC) methods.

In the literature, a variety of particle filter algorithms can be found [5]. Still, the basic *sampling importance resampling* (SIR) particle filter [6] is often used due to its robustness. The main drawback of the SIR-PF is that it can suffer from impoverishment of the particle weights. For numerical reasons, resampling has to be used in order to avoid the particles to degenerate. More recently, new algorithms have been proposed based on a log-homotopy transition between the prior and the posterior. For instance, the Daum–Huang filters (see [7] and the references therein) model this transition phase in terms of a physical flow that is determined by a "force" induced by the measurement. This leads to a stochastic differential equation (SDE) that then can be solved numerically by introducing a discretized pseudotime evolving from the prior to the posterior pdf. However, the computation time for solving the SDE is often quite high for standard target tracking scenarios [8]. A different homotopy approach is provided by the *progressive filter* that was published by Hanebeck in [9]. In the progressive filter, an incremental inclusion of the likelihood function is achieved by a partition of the exponent of the likelihood going from zero to one. This prevents particle impoverishment by means of frequent resampling and an appropriately chosen step size. A Kullback–Leibler divergence-based approach to obtain the posterior particles is proposed in [10]. The resulting algorithm is similar to the *ensemble Kalman filter* (EnKF)-based filter proposed in [11], however, the additional noise term in the Kalman-based update is different. The EnKF adds some zero mean Gaussian-distributed noise to the measurement of each sample and applies Kalman filter update equations for each particle. As a consequence, a fast filter results that is consistent and performs well in nonlinear scenarios. The EnKF also has been extended to Gaussian mixtures in [12] and [13].

Instead of nondeterministic samples we can also use a grid of equidistant space points to represent the pdf in the field of view. The prior pdf is then obtained by solving the *Fokker–Planck equation* (FPE), which is equivalent to the integral formulation in (1). Challa and Bar-Shalom for instance use finite differences in [14] to obtain the solution of the FPE and show that a consistent result is obtained even for highly nonlinear problems with large noise variances. This static approach has not become as popular as the particle filters due to the higher computational load. There is a notable change in the way of thinking since it was discovered that separated representations of discretized multidimensional functions have surprisingly good approximation properties [15]. Nowadays it may even be seen as the only known way to overcome the curse of dimensionality [16], at least in cases, where the models can easily be approximated by factorized functions. In other words, approximations by separable functions are of particular interest when the dimensionality of the problem becomes large. To the authors' knowledge, a first attempt to integrate the tensor decomposition into a Bayesian estimation framework was given by Sun and Kumar [17]. Further development of the tensor-based approach and some tracking examples were given in [18] and [19].

In this paper, we demonstrate the performance of the tensor decomposition approach for realistic tracking problems with nonlinear measurement models. Further, we show that it can also be applied to multitarget tracking using the set theory-based approaches where an intensity function is computed instead of a pdf. Multiple numerical evaluations are shown to demonstrate that the tensor decomposition approach can achieve convincing results in terms of estimation accuracy in nonlinear filtering problems.

A. Structure

This paper is structured as follows. In Section II, we review the basic concept of nonlinear filtering using the

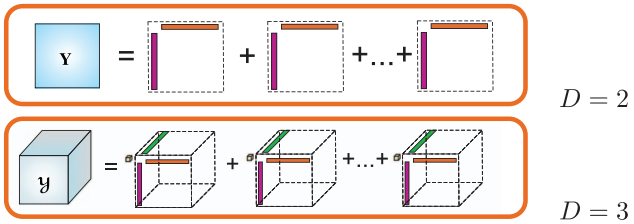


Fig. 1. Scheme of a tensor decomposition along its dimensions using the Kronecker product of vectors for two and three dimensions. Scheme taken from [21].

tensor decomposition approach. This is extended for the handling of nondetections and clutter measurements in Section III. The application on the set-based estimation theory for multitarget tracking is given in Section IV. Then, in Section V numerical examples are given to demonstrate the performance of the approach. The paper ends with conclusions in Section VI.

II. NONLINEAR STATE ESTIMATION BASED ON TENSOR DECOMPOSITION

It was already discovered in 1927 by Hitchcock that a D-way tensor can be represented as a sum of outer products [20]:

$$\mathcal{Y} = \sum_{l=1}^{L} \mathbf{y}_{1,l} \circ \cdots \circ \mathbf{y}_{D,l} \tag{3}$$

where L is the number of components. If for all $d \in \{1, \ldots, D\}$ $\mathbf{y}_{d,l}$ are of a given size $N_d \times 1$, in other words a vector, this representation of the tensor is called *canonical polyadic decomposition* or *CANDECOMP/PARAFAC decomposition*.[2] In this case, the $\mathbf{y}_{d,l}$ are called the *loading vectors*. This decomposition of a tensor is visualized for $D = 2$ and $D = 3$, respectively, in Fig. 1. It should be noted that this decomposition of a two-way tensor (matrix) can easily be obtained by the *singular value decomposition* (SVD). For higher dimensions the problem of finding a decomposed representation becomes NP-hard [22]. However, numerical solutions, such as the *alternating least squares* (ALS) algorithm, exist [23], which yield satisfying results for the problems addressed here in manageable time. For a fixed dimension d, it is assumed that the state space can be discretized into N_d grid points. These can be uniformly spaced with a fixed step size of Δ_{x_d} or chosen specifically for a numerical differentiation such as Chebyshev polynomials [18]. The probability density function restricted to the discretized state space points yields a D-way tensor, which approximates the original function:

$$p(\mathbf{x}_k|\mathcal{Z}^k) \approx [p([\mathbf{x}_d]_i|\mathcal{Z}^k)]_{n_1,\ldots,n_D}. \tag{4}$$

[2] An explanation of the abbreviations involved can be found in [21] and the references therein.

Throughout this paper, this tensor is represented in a decomposed form. Thus, the pdf at time t_k is approximated, again, by a CPD factorization:

$$[p([\mathbf{x}_d]_i|\mathcal{Z}^k)]_{n_1,\ldots,n_D} \approx \sum_{l=1}^{L} \rho_{1,l}^{(t_k)} \circ \cdots \circ \rho_{D,l}^{(t_k)} \tag{5}$$

where L is the number of components, which usually is a fixed user parameter and depends on the computational power of the fusion hardware and processing time constraints, "$\circ$" is the outer product and $\rho_{d,l}^{(t_k)}$ are the so-called *loading vectors* of dimension $N_d \times 1$ for each $d = 1, \ldots, D$ and $l = 1 \ldots, L$.

By means of an appropriate index function (see [19] for instance), an equivalent representation of a tensor can be achieved in its *vectorized* form:

$$p(\mathbf{x}_k|\mathcal{Z}^k) \approx \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}^{(t_k)} \tag{6}$$

where "$\otimes$" is the Kronecker product. For the sake of notational simplicity, the latter form will be used throughout this paper.

It is assumed that the time evolution of the system is described by a continuous-time stochastic system given by

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + \mathbf{G}(\mathbf{x}, t)\mathrm{d}\mathbf{w} \tag{7}$$

where $\mathbf{f}$ is the drift vector, $\mathbf{G}$ is the matrix of all diffusion coefficients, and $\mathrm{d}\mathbf{w}$ are the increments of a multivariate Brownian motion with covariance $\mathbf{Q}t$.

The measurement model is a general possibly nonlinear function $\mathbf{h}$ such that the observation at discrete instants of time t_k is given by

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k, t_k, \mathbf{v}_k) \tag{8}$$

where $\mathbf{v}_k$ is a random variable that represents the measurement noise of the sensor.

It is well known that the posterior pdf conditioned on all sensor data up to time t_k can be computed recursively by means of a prediction-filtering cycle. In recent publications, a tensor decomposition-based state estimation scheme has been proposed [17]–[19], which will be summarized in the remainder of this section.

A. Initialization

For the initialization, it is assumed that the initial pdf is given in a CPD form:

$$p(\mathbf{x}_0|\mathcal{Z}^0) = \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}^{(t_0)}. \tag{9}$$

This can either be achieved by an analytical decomposition into sums of products of a given pdf evaluated at

the discretization points or numerically by the ALS for instance.

B. Prediction

It is well known that the time evolution of the pdf is described by the FPE, for which the drift and diffusion parameters are given by the stochastic differential equation in (7):

$$\frac{\partial p}{\partial t} = -\sum_{i=1}^{D} \frac{\partial([\mathbf{f}]_i p)}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^{D} \frac{\partial^2([\mathbf{B}]_{i,j} p)}{\partial x_i \partial x_j} \tag{10}$$

where $\mathbf{B} = \mathbf{G}\mathbf{Q}\mathbf{G}^\top$ is the combined diffusion coefficient matrix. In the above equation, $[\mathbf{f}]_i$ denotes the ith entry of the drift vector and $[\mathbf{B}]_{i,j}$ is the entry in the ith row and jth column of the diffusion matrix. It is assumed that all components can be represented in a separable form such that

$$[\mathbf{f}]_i(\mathbf{x}) = \sum_{k=1}^{K_i} \prod_{d=1}^{D} f_{d,k}^{i}(x_d) \tag{11}$$

$$[\mathbf{B}]_{i,j}(\mathbf{x}) = \sum_{k=1}^{K_{i,j}} \prod_{d=1}^{D} B_{d,k}^{i,j}(x_d) \tag{12}$$

where K_i and $K_{i,j}$ are the number of components of the functions $[\mathbf{f}]_i$ and $[\mathbf{B}]_{i,j}$, respectively. By means of differentiation matrices and the FPE parameters in a separable form on the discretized grid we obtain an FPE operator $\mathbb{L}$ such that

$$\frac{\partial p}{\partial t} = \mathbb{L} p \tag{13}$$

where p is the pdf in a tensorized form. In the recent literature, two different approaches have been proposed to compute a numerical solution of the FPE for CPD tensors. The first solution is based on the ALS, where the FPE operator is augmented with a differentiation matrix $\mathbf{D}_t$ for the time dimension, which is accumulated with the pdf [17]:

$$(\mathbf{D}_t - \mathbb{L}) p(\mathbf{x}, t) = 0. \tag{14}$$

Since the trivial solution $p = 0$ of (14) has to be avoided, constraints are added to the least square optimization such that the pdf is normalized and matches the previous pdf for the start time.

The second solution uses the tensor exponential of the FPE operator since it holds that

$$p(\mathbf{x}, t_k) = \exp\{\Delta_t \cdot \mathbb{L}\}\, p(\mathbf{x}, t_{k-1}) \tag{15}$$

where Δ_t is the time difference $t_k - t_{k-1}$. Here, the tensor exponential is approximated by means of a Taylor series [19].

C. Filtering

For the recursive prediction-filtering cycle, it is required to compute the posterior pdf, which incorporates the information of a given observation $\mathbf{z}_k$ at time t_k. This is achieved by means of the Bayes theorem

$$p(\mathbf{x}_k|\mathcal{Z}^k) = \frac{p(\mathbf{z}_k|\mathbf{x}_k) \cdot p(\mathbf{x}_k|\mathcal{Z}^{k-1})}{\int \mathrm{d}\mathbf{x}_k\, p(\mathbf{z}_k|\mathbf{x}_k) \cdot p(\mathbf{x}_k|\mathcal{Z}^{k-1})} \tag{16}$$

where $p(\mathbf{z}_k|\mathbf{x}_k)$ is the likelihood function. Since the likelihood evaluated on the discretized state space is a tensor, it is assumed that it is also given in a CPD form:

$$p(\mathbf{z}_k|\mathbf{x}_k) = \sum_{l'=1}^{L'} \bigotimes_{d=1}^{D} \lambda_{d,l'}. \tag{17}$$

As for the initial estimation pdf, this can be achieved by numerical methods if the likelihood function cannot be decomposed analytically. As a consequence, the posterior is given by

$$p(\mathbf{x}_k|\mathcal{Z}^k) = \frac{1}{c}\sum_{l=1}^{L}\sum_{l'=1}^{L'} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k-1)} \odot \lambda_{d,l'} \tag{18}$$

$$=: \sum_{l=1}^{\tilde{L}} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k)} \tag{19}$$

where "$\odot$" is the Hadamard (pointwise) product, $\tilde{L} = L \cdot L'$, and c is the normalization constant [17]. In order to keep the number of components L fixed, a mixture reduction technique such as tensor deflation has to be applied [21].

Note that the normalization constant of a CPD tensor p can easily be computed by the following summation[3]:

$$\int \mathrm{d}\mathbf{x}\, p(\mathbf{x}) = \sum_{l=1}^{L} \prod_{d=1}^{D} \Delta_{x_d} \sum_{i_d=1}^{N_d} [\rho_{d,l}]_{i_d} \tag{20}$$

where Δ_{x_d} is the discretization size of the state space in the dth dimension.

III. TRACKING IN CLUTTER

If the sensor produces multiple measurements at time t_k of a single target, the interpretation of the measurement origins is ambiguous. Measurements that come from unwanted objects or false detections are often referred to as *clutter*. If it is assumed that the target is detected with probability p_D, we have one additional interpretation assuming that the target is not detected at all. Therefore, let Z_k denote the set of m_k measurements $\mathbf{z}_k^1, \ldots, \mathbf{z}_k^{m_k}$ produced at time t_k.

[3] More details on implementation issues of the CPD tensor approach are given in the Appendix.

Let $j_k = 0$ denote the data interpretation hypothesis that the object has not been detected at all by the sensor at time t_k and so all measurements have to be considered as clutter. A hypothesis denoted by $j_k \in \{1, \ldots, m_k\}$ refers to the interpretation that the object has been detected, $\mathbf{z}_k^{j_k} \in Z_k$ being the corresponding measurement, and the remaining sensor data being clutter. Evidently, the set $\{0, \ldots, m_k\}$ describes mutually exclusive and exhaustive data interpretations. Due to the total probability theorem we have the following equation [24]:

$$p(Z_k, m_k|\mathbf{x}_k) = p_D\, p(Z_k, m_k|\mathbf{x}_k, O) + (1 - p_D)\, p(Z_k, m_k|\neg O) \quad (21)$$

where p_D is the probability of detection and O is the enumeration of a detection event. Often, it is assumed that the clutter measurements can be modeled statistically by a Poisson distribution in the number of measurements, which are uniformly distributed in the state space. Therefore, we have

$$\begin{aligned} p(Z_k, m_k|\neg O) &= p(Z_k|m_k, \neg O)p(m_k|\neg O) \\ &= p_F(m_k) \cdot |\mathrm{FoV}|^{-m_k} \end{aligned} \quad (22)$$

$$p_F(m_k) = \frac{\bar{m}^{m_k}}{m_k!} e^{-\bar{m}} \quad (23)$$

where $\bar{m}$ is the mean number of false measurements, p_F its Poisson distribution, and $|\mathrm{FoV}|$ is the size of the field of view. For the density conditioned on a detection, we can enumerate the data interpretation that the jth measurement is from the target:

$$p(Z_k|\mathbf{x}_k, O) = \sum_{j=1}^{m_k} p(Z_k|\mathbf{x}_k, O, j)p(j|O, \mathbf{x}_k). \quad (24)$$

Since hypothesis j implies that there are $m_k - 1$ false measurements, we have the following equation:

$$\begin{aligned} p(Z_k, m_k|\mathbf{x}_k, O, j) &= p(Z_k|m_k, \mathbf{x}_k, O, j)p(m_k|O) \\ &= p_F(m_k - 1)|\mathrm{FoV}|^{-(m_k-1)} \cdot p(\mathbf{z}_k^j|\mathbf{x}_k) \end{aligned} \quad (25)$$

where $p(\mathbf{z}_k^j|\mathbf{x}_k)$ is the density describing the sensor statistics for measuring a target with state $\mathbf{x}_k$. Furthermore, it is assumed that all hypotheses have the same prior probability, which yields

$$p(j|O, \mathbf{x}_k) = \frac{1}{m_k}. \quad (26)$$

Together we obtain

$$\begin{aligned} p(Z_k|\mathbf{x}_k) = p_D\, &\frac{1}{m_k} p_F(m_k - 1)|\mathrm{FoV}|^{-(m_k-1)} \sum_{j=1}^{m_k} p(\mathbf{z}_k^j|\mathbf{x}_k) \\ &+ (1 - p_D)p_F(m_k) \cdot |\mathrm{FoV}|^{-m_k} \end{aligned} \quad (27)$$

$$= p_F(m_k)|\mathrm{FoV}|^{-m_k} \cdot \left(p_D \frac{|\mathrm{FoV}|}{\bar{m}} \sum_{j=1}^{m_k} p(\mathbf{z}_k^j|\mathbf{x}_k) + (1 - p_D) \right) \quad (28)$$

$$= \frac{p_F(m_k)|\mathrm{FoV}|^{-m_k}}{\rho_F} \cdot \left(p_D \sum_{j=1}^{m_k} p(\mathbf{z}_k^j|\mathbf{x}_k) + (1 - p_D)\rho_F \right) \quad (29)$$

where $\rho_F = \frac{\bar{m}}{|\mathrm{FoV}|}$ denotes the clutter density.

Since it is sufficient to model the likelihood function up to proportionality, we can neglect the constant factor and obtains

$$p(Z_k|\mathbf{x}_k) \propto (1 - p_D)\rho_F + p_D \sum_{j=1}^{m_k} p(\mathbf{z}_k^j|\mathbf{x}_k). \quad (30)$$

Again, it is assumed that a decomposed form of the sensor statistics $p(\mathbf{z}_k^j|\mathbf{x}_k)$ is available:

$$p(\mathbf{z}_k^j|\mathbf{x}_k) = \sum_{l'=1}^{L'} \bigotimes_{d=1}^{D} \lambda_{d,l}^j. \quad (31)$$

This leads to a posterior pdf given by

$$\begin{aligned} p(\mathbf{x}_k|\mathcal{Z}^k) = \frac{1}{c} \Bigg(&(1 - p_D)\rho_F \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k-1)} \\ &+ p_D \sum_{j=1}^{m_k} \sum_{l=1}^{L} \sum_{l'=1}^{L'} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k-1)} \odot \lambda_{d,l'}^j \Bigg). \end{aligned} \quad (32)$$

Again, it is obvious that the posterior is already given in a decomposed form, however, the number of components has increased by a factor of $1 + L' \times m_k$ and therefore a deflation algorithm has to be applied for recursions with a constant number of loading vectors.

IV. MULTITARGET TRACKING

For a multitarget scenario, Bayesian filters can be derived by means of the theory of point set statistics. In the case that all targets are significantly separated, it is well possible to apply the single target filters from above on individual clusters. In all other cases, *probability hypothesis density* (PHD) [25] or intensity-based filters [26] have been proven particularly useful since association-free implementations are available that are easy to implement and of low computational complexity. Since there is no free lunch, this comes with a loss of target identities. However, methods have been proposed to overcome this problem.

The basic idea of the PHD or intensity filter is to model the conditional pdf of the set $\mathbf{x}_k^1, \ldots, \mathbf{x}_k^n$ of n target states as an inhomogeneous Poisson point process, that is, the number of targets is assumed to be Poisson

distributed:

$$p(\mathbf{x}_k^1, \ldots, \mathbf{x}_k^n, n) = p(\mathbf{x}_k^1, \ldots, \mathbf{x}_k^n | n)\, p(n) \tag{33}$$

where

$$p(n) = \exp(-\mu)\frac{\mu^n}{n!} \tag{34}$$

$$\mu = \int \mathrm{d}\mathbf{x}\, f(x) \tag{35}$$

$$p(\mathbf{x}_k^1, \ldots, \mathbf{x}_k^n | n) = n! \cdot \prod_{j=1}^{n} p(\mathbf{x}_k^j). \tag{36}$$

The factor of $n!$ in (36) comes from the summation over all possible permutations of target identities, since the set is order free. The function f is the so-called intensity or PHD. Its integral μ is the mean number of targets and the spatial distribution of a target can be obtained by normalization of f:

$$p(\mathbf{x}_k^j) = \frac{f(\mathbf{x}_k^j)}{\int \mathrm{d}\mathbf{x}\, f(\mathbf{x})}. \tag{37}$$

Combining the above equations directly yields

$$p(\mathbf{x}_k^1, \ldots, \mathbf{x}_k^n, n) = \exp\left\{-\int \mathrm{d}\mathbf{x}\, f(x)\right\} \prod_{j=1}^{n} f(\mathbf{x}_k^j). \tag{38}$$

Therefore, f fully characterizes the point process.

In Bayesian set filters, it is sufficient to compute the prior and posterior intensity function f, if the Poisson assumption from above holds:

$$f_{k-1|k-1} \xrightarrow{\text{prediction}} f_{k|k-1} \tag{39}$$

$$f_{k|k-1} \xrightarrow{\text{filtering}} f_{k|k}. \tag{40}$$

In this section, it is shown that a recursive computation of the multitarget intensity function can be achieved by means of a decomposed tensor representation. For the initialization or previous filtering step, it is assumed that the intensity is given in a decomposed form:

$$f_{k-1|k-1} = \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k-1|k-1)}. \tag{41}$$

Due to the relationship (37), the time evolution of the intensity function is described by the FPE:

$$\frac{\partial f}{\partial t} = -\sum_{i=1}^{D} \frac{\partial([\mathbf{f}]_i f)}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^{D} \frac{\partial^2([\mathbf{B}]_{i,j} f)}{\partial x_i \partial x_j}. \tag{42}$$

Another way to see this is the equivalence of the FPE to the Chapman–Kolmogorov equation [27], which is a more common way to compute the prior of a set theory-based filter [26]. As a consequence, it is possible to use one of the tensor propagators described in Section II-B.

Often, birth and death processes are modeled in addition to the system evolution to handle appearing or disappearing targets. Let $\boldsymbol{\nu}(\mathbf{x})$ be an intensity of a target birth point process given in a CPD representation

$$\boldsymbol{\nu}(\mathbf{x}) = \sum_{l_b=1}^{L_b} \bigotimes_{d=1}^{D} \nu_{d,l_b} \tag{43}$$

and $p_{\mathrm{TD}}(\mathbf{x})$ be the probability of target death such that the probability of survival is given by

$$p_{\mathrm{S}}(\mathbf{x}) = 1 - p_{\mathrm{TD}}(\mathbf{x}) = \sum_{l_d=1}^{L_d} \bigotimes_{d=1}^{D} p_{d,l_d}^{\mathrm{S}}. \tag{44}$$

If

$$f'_{k|k-1} = \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k-1)\prime} \tag{45}$$

denotes the CPD tensor approximation of the intensity after the application of the time propagator, then the prior intensity is obtained by the superposition of the birth and death processes [26]. The result again is a CPD tensor, where deflation algorithms have to be applied to keep the number of loading vectors constant:

$$f_{k|k-1} = \sum_{l_b=1}^{L_b} \bigotimes_{d=1}^{D} \nu_{d,l_b} + \sum_{l_d=1}^{L_d}\sum_{l=1}^{L} \bigotimes_{d=1}^{D} p_{d,l_d}^{\mathrm{S}} \odot \rho_{d,l}^{(k|k-1)\prime}. \tag{46}$$

In many practical applications, it is sufficient to model the birth process as a constant birth rate $\boldsymbol{\nu}(\mathbf{x}) = \mathbf{b}$ and analogously set a constant probability of target death $p_{\mathrm{TD}}(\mathbf{x}) = \mathbf{d}$. In this case, the prior reduces to

$$f_{k|k-1} = \frac{\mathbf{b}}{\prod_{d=1}^{D} \Delta_{x_d} N_d} \bigotimes_{d=1}^{D} 1_d + (1 - \mathbf{d}) \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k-1)\prime} \tag{47}$$

where 1_d is the vector consisting of N_d ones.

A. Multitarget Filtering

Let $Z_k = \mathbf{z}_k^1, \ldots, \mathbf{z}_k^{m_k}$ be the set of observations produced at time t_k. As shown by Mahler [25], it is possible to approximate the posterior pdf of a multitarget Poisson point process by the following intensity function:

$$f_{k|k} = \left((1 - p_D) + \sum_{j=1}^{m_k} \frac{p(\mathbf{z}_k^j|\mathbf{x})\, p_D}{\lambda(\mathbf{z}_k^j)}\right) f_{k|k-1} \tag{48}$$

$$\lambda(\mathbf{z}_k^j) = \lambda_c(\mathbf{z}_k^j) + \int \mathrm{d}\mathbf{x}\, p(\mathbf{z}_k^j|\mathbf{x})\, p_D\, f_{k|k-1}(\mathbf{x}) \tag{49}$$

where $\lambda_c(\mathbf{z})$ is the Poisson intensity of the clutter point process. It is assumed that the prior intensity is given in a CPD form where $\rho_{d,l}^{(k|k-1)}$ are its loading vectors for $l = 1, \ldots, L$ and $d = 1, \ldots, D$. Using the sensor model

from (31), we obtain the following update equation in a tensorized form:

$$f_{k|k} = \Bigg((1 - p_D) \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k-1)} + \sum_{j=1}^{m_k} \frac{p_D}{\lambda(\mathbf{z}_k^j)} \sum_{l=1}^{L} \sum_{l'=1}^{L'} \bigotimes_{d=1}^{D} \rho_{d,l}^{(k|k-1)} \odot \lambda_{d,l'}^{j} \Bigg) \tag{50}$$

where the expected measurement likelihood $\lambda(\mathbf{z}_k^j)$ is given by

$$\lambda(\mathbf{z}_k^j) = \lambda_c(\mathbf{z}_k^j) + \sum_{l=1}^{L} \sum_{l'=1}^{L'} p_D \prod_{d=1}^{D} \Delta_{x_d} \sum_{i=1}^{N_d} [\rho_{d,l}^{(k|k-1)}]_i \cdot [\lambda_{d,l'}^{j}]_i. \tag{51}$$

Here, the PHD filter update equations were used. The extension to higher order statistics filter for the number of targets (CPHD) [28], the iFilter [29], or (generalized) multi-Bernoulli filters [30] is straightforward.

V. NUMERICAL EVALUATION

In this section, we demonstrate the performance of the described algorithms by means of numerical simulations. It is divided into three parts describing the simulation setup and results of a nonlinear scenario (a), a scenario with ambiguous measurements (b), and a multitarget scenario (c), respectively. In all scenarios, a four-dimensional state space was used such that

$$\mathbf{x} = (x, y, \dot{x}, \dot{y})^\top. \tag{52}$$

The simulated target(s) move according to a discretized almost constant velocity model where the transition and process noise model is given by

$$\mathbf{x}_{k+1} = \mathbf{F}_{k|k-1}\mathbf{x}_k + \mathbf{w}_{k|k-1} \tag{53}$$

$$\mathbf{F}_{k|k-1} = \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix} \otimes \mathbf{I}_2 \tag{54}$$

$$\mathbf{w}_{k|k-1} \sim \mathcal{N}(\mathbf{O},\ \mathbf{Q}_{k|k-1}) \tag{55}$$

$$\mathbf{Q}_{k|k-1} = q \cdot \begin{pmatrix} \mathbf{O} & \mathbf{O} \\ \mathbf{O} & T\mathbf{I}_2 \end{pmatrix} \tag{56}$$

where the power spectral density was set to $q = 0.1$.

A. Nonlinear Example

In the nonlinear scenario, a single target is observed by means of a bistatic radar with one transmitter antenna (Tx) and two receiving antennas (Rx_1 and Rx_2). Once a second, the bistatic ranges $|Tx-\mathbf{x}|+|Rx_i-\mathbf{x}|$ for $i = 1, 2$ are measured with additive Gaussian noise with a standard deviation of $\sigma_{br} = 1.0m$. The receivers were located at $(0, 8)^\top$, and $(0, 12)^\top$, respectively, and the transmitter was set in between at $(0, 10)^\top$.

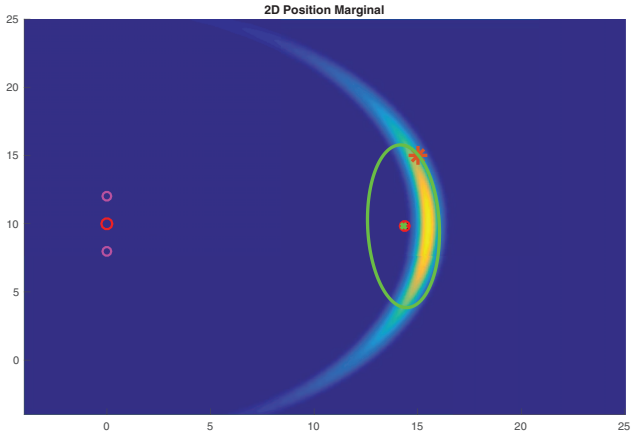


Fig. 2. Exemplary initial pdf based on two bistatic range measurements of a target (brown star) where the transmitter is located at the center (red circle) of two receivers (purple circles). It can be seen that the Gaussian approximation (green) does not reflect the measurement uncertainty in an appropriate way, in contrast to that the 2D marginal of the tensor representation (blue/yellow color range) gives a precise approximation to the true Bayes posterior.

The same measurements also were processed by an EKF and a bootstrap PF using importance resampling. The initial state and covariance for these filters was, respectively, estimated by the first and second moment of the likelihood function of the first measurement. The initial velocity was set to zero with a standard deviation of $2\frac{m}{s}$. The initial pdf of the position is shown exemplary in Fig. 2.

The results of the numerical evaluation are shown in Fig. 3. We have plotted the root mean squared error (RMSE) of 50 Monte Carlo simulations. Clearly, the tensor approach reaches the performance of the PF, which is close to the Cramer–Rao lower bound.

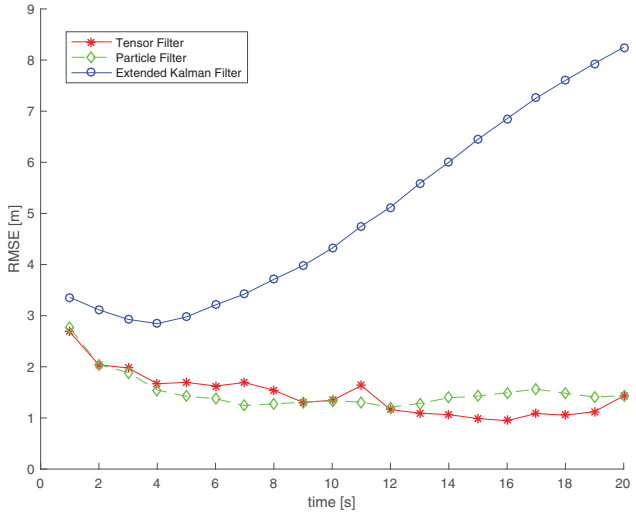


Fig. 3. Root mean squared error (RMSE) of 50 Monte Carlo simulations. It can be seen that the tensor decomposition-based approach has equal or better estimation performance than the extended Kalman filter or the particle filter.

B. Multitarget Example

The multitarget scenario demonstrates the ability of the tensor decomposition-based PHD filter as described in Section IV to estimate the number of targets and their respective states. Since the PHD intensity is a label-free statistic, an evaluation of the estimation errors would require some state extraction algorithm, which is out of the scope of this paper. As a consequence, the x–y marginals of the intensity function of an exemplary simulation are presented.

In the multitarget scenario, two targets are initialized with states $\mathbf{x}_0^1 = (0, 0, 1, 1)^\top$ and $(10, 10, -1, -1)^\top$, respectively.

Fig. 4 shows the intensity function in x–y coordinates after the initialization and after 50 time steps of 0.1 s.

C. Filtering in High Dimensions

The third example scenario can be considered a toy problem. It is specified to demonstrate the power of the tensor decomposition approach in high-dimensional filtering problems. These high dimensions can easily appear in practical scenarios such as SLAM-based [31] navigation data fusion.

The scenario demonstrated here is a one-step filtering of a given prior density using a linear measurement. For various dimensions D, the prior is given by a mean

$$\mathbf{x}_{0|0} = (50, \ldots, 50)^\top \tag{57}$$

and a covariance matrix $\mathbf{P}_{0|0} = [\mathbf{P}_{0|0}]_{i,j}$ for $i, j = 1, \ldots, D$, where

$$[\mathbf{P}_{0|0}]_{i,j} = 100, \text{ if } i == j \tag{58}$$

$$[\mathbf{P}_{0|0}]_{i,j} = 50, \quad \text{else.} \tag{59}$$

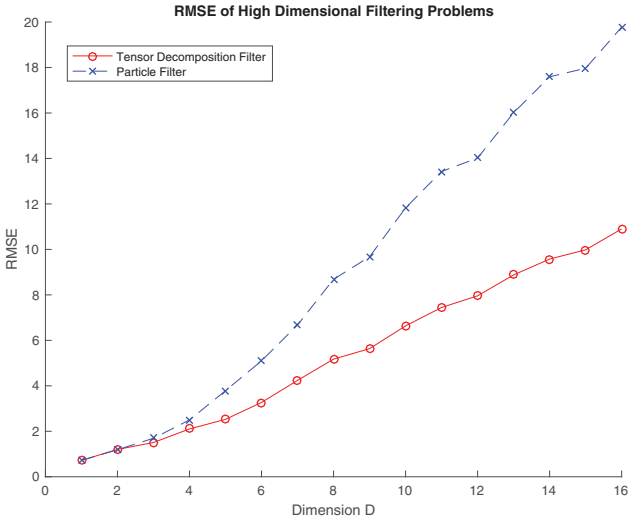


Fig. 5. Root mean squared error (RMSE) for 100 Monte Carlo simulations on a single measurement update for various dimensions.

The true state of the system is drawn randomly according to the prior distribution. The measurement vector is the true state corrupted by additive Gaussian noise, where the covariance matrix is given by the unity matrix $\mathbf{I}_D$. For the tensor approach, $L = 2{,}000$ components were used and for the particle filter $N_{\text{pf}} = 10{,}000$ particles were used.[4] For each dimension 100 Monte Carlo simulations were used.

It can be seen from Fig. 5 that the particle filter degrades due to the curse of dimensionality. The tensor decomposition approach clearly outperforms the particle filter in higher dimensions where it is obvious that the difference increases when the D grows. In Fig. 6, the mean processing times for both filters are summarized.

[4]The number of particles and components, respectively, was chosen such that the processing time is of equal magnitude.

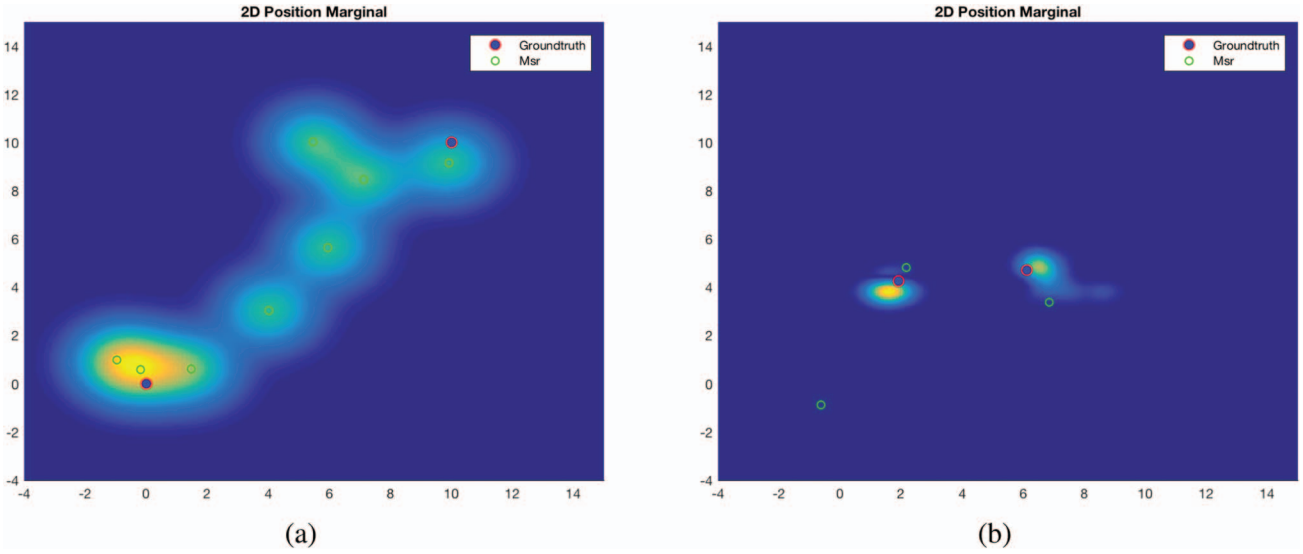


Fig. 4. Multitarget intensity function after the initialization (a) and after 50 time steps (b).

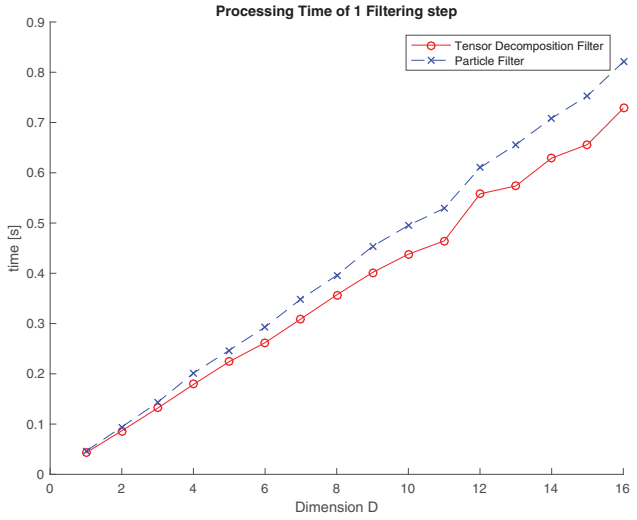


Fig. 6. Mean processing times for the tensor decomposition filter and the particle filter.

VII CONCLUSION

In this paper, we have applied the CPD-based approach of nonlinear Bayesian filtering on practical multitarget tracking problems with false measurements. By means of the *probabilistic data association* (PDA) likelihood, the update formula for single targets in cluttered environments for CPD tensors was derived. Then, it was shown that the CPD representation can also be used to apply set theory-based filters such as the PHD filter. In a numerical evaluation the performance was compared to the existing methods for nonlinear filtering. In a toy example it has been shown that the CPD is a powerful representation of a density function in particular if the problem is high dimensional.

APPENDIX

In the Appendix, we would like to provide some details and assistance for the interested reader who wants to implement some of the algorithms above. Since most engineers in data fusion and tracking work with Gaussian mixtures, particles, and optimization algorithms, the tensor decomposition-based data fusion can be considered quite new and unknown. This section should help to start from the scratch to implement the CPD tensors and the corresponding filters.

Implementation

A CPD tensor, which is given by

$$p(\mathbf{x}) = \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}$$

is fully described by L *loading vectors* for each dimension $d = 1, \ldots, D$. It is useful to store these vectors as matrices, called *loading matrices*, $\mathbf{U}_1, \ldots, \mathbf{U}_D$. For a given dimension d, each loading vector $\rho_{d,l}$ has by definition N_{x_d} entries, therefore, the loading matrix $\mathbf{U}_d$ is of the size $N_{x_d} \times L$. In the following explanations, we will use the following short notation for the above CPD tensor:

$$p(\mathbf{x}) = [\mathbf{U}_1, \ldots, \mathbf{U}_D] = [\mathbf{U}_d]_{d=1}^{D}.$$

The multiplication with an exemplary decomposed likelihood function

$$\ell(\mathbf{z}, \mathbf{x}) = \bigotimes_{d=1}^{D} \lambda_d$$

then reduces to simple matrix multiplication:

$$p(\mathbf{x}) \cdot \ell(\mathbf{z}, \mathbf{x}) = [\operatorname{diag}[\lambda_d]\, \mathbf{U}_d]_{d=1}^{D}.$$

If the likelihood function has multiple components as in (17), the same operation is computed for each of the components and the resulting matrices are appended horizontally. Also it should be noted that likelihood functions of lower dimensions can easily be incorporated by setting $\lambda_d = \mathbb{1}_d$ for $d > d'$, where d' is the dimension of the likelihood.

Integration

The integral of a CPD tensor

$$\int \mathrm{d}\mathbf{x} \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}$$

can easily be computed by means of cheap computational operations. This can be seen by the fact that

$$\begin{aligned}\int \mathrm{d}\mathbf{x} \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l} &= \sum_{l=1}^{L} \int \mathrm{d}\mathbf{x} \bigotimes_{d=1}^{D} \rho_{d,l} \\ &= \sum_{l=1}^{L} \sum_{i_1,\ldots,i_D} \prod_{d=1}^{D} [\rho_{d,l}]_{i_d} \cdot \Delta_{x_d} \\ &= \sum_{l=1}^{L} \prod_{d=1}^{D} \Delta_{x_d} \sum_{i_d=1}^{N_d} [\rho_{d,l}]_{i_d}.\end{aligned}$$

Computing the Mean Vector

Again, it is assumed that the pdf is given in a CPD-tensorized form:

$$p(\mathbf{x}) = \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}.$$

The mean $\mathrm{E}[\mathbf{x}] = \hat{\mathbf{x}} = [\hat{\mathbf{x}}_d]_d$ is given by

$$\hat{\mathbf{x}}_d = \int \mathrm{d}\mathbf{x}\, \mathbf{x}_d \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}.$$

Using the integration rule from above, we obtain

$$\hat{\mathbf{x}}_d = \sum_{l=1}^{L} \prod_{j \neq d} \left\{ \Delta_{x_j} \sum_{i_j=1}^{N_j} [\rho_{j,l}]_{i_j} \right\} \Delta_{x_d} \sum_{i_d=1}^{N_d} [\operatorname{diag}[[\mathbf{x}_d]]\, \rho_{d,l}]_{i_d}.$$

Computing the Covariance Matrix

Analogously, the covariance matrix $\operatorname{cov}[\mathbf{x}] = \mathbf{P} = [\mathbf{P}_{ij}]_{i,j}$ can be computed by $\frac{1}{2}D(D+1)$ integration operations. For given i and j, the covariance is given by

$$\mathbf{P}_{ij} = \int \mathrm{d}\mathbf{x}\, (\mathbf{x}_i - \hat{\mathbf{x}}_i)(\mathbf{x}_j - \hat{\mathbf{x}}_j) \sum_{l=1}^{L} \bigotimes_{d=1}^{D} \rho_{d,l}.$$

For $i \neq j$, we have

$$\begin{aligned}\mathbf{P}_{ij} = \sum_{l=1}^{L} \prod_{k\neq i \wedge k \neq j} \left\{ \Delta_{x_k} \sum_{i_k=1}^{N_k} [\rho_{k,l}]_{i_k} \right\} \\ \cdot \Delta_{x_i} \sum_{i_i=1}^{N_i} [\operatorname{diag}[[\mathbf{T}_{\hat{\mathbf{x}}_i}\mathbf{x}_i]]\, \rho_{i,l}]_{i_i} \\ \cdot \Delta_{x_j} \sum_{i_j=1}^{N_j} [\operatorname{diag}\left[[\mathbf{T}_{\hat{\mathbf{x}}_j}\mathbf{x}_j]\right] \rho_{j,l}]_{i_j}\end{aligned}$$

where $\mathbf{T}_{\hat{\mathbf{x}}_j}$ is the affine translation such that $[(\mathbf{x}_j - \hat{\mathbf{x}}_j)]_i = \mathbf{T}_{\hat{\mathbf{x}}_j}\mathbf{x}_j$. The diagonal elements similarly are given by

$$\begin{aligned}\mathbf{P}_{ii} = \sum_{l=1}^{L} \prod_{k\neq i} \left\{ \Delta_{x_k} \sum_{i_k=1}^{N_k} [\rho_{k,l}]_{i_k} \right\} \\ \cdot \Delta_{x_i} \sum_{i_i=1}^{N_i} [\operatorname{diag}\left[[([\mathbf{x}_i]_l - \hat{\mathbf{x}}_i)^2]_l\right] \rho_{i,l}]_{i_i}.\end{aligned}$$

REFERENCES

[1] Y. Bar-Shalom, X. Li, and T. Kirubarajan
Estimation with Applications to Tracking and Navigation. Wiley-Interscience, New York, NY, 2001.

[2] W. Koch
Tracking and Sensor Data Fusion: Methodological Framework and Selected Applications. Springer, Heidelberg, Germany, 2014.

[3] R. E. Kalman
"A New Approach to Linear Filtering and Prediction Problems,"
Transactions of the ASME—Journal of Basic Engineering, vol. 82, no. Series D, pp. 35–45, 1960.

[4] S. J. Julier and J. K. Uhlmann
"New Extension of the Kalman Filter to Nonlinear Systems,"
in *Signal Processing, Sensor Fusion, and Target Recognition VI*, I. Kadar
Ed., vol. 3068, Jul. 1997, pp. 182–193.

[5] B. Ristic, S. Arulampalam, and N. Gordon
Beyond the Kalman Filter: Particle Filters for Tracking Applications. Artech House, Boston, MA, 2004.

[6] N. Gordon, D. Salmond, and A. Smith
"Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation,"
Radar and Signal Processing, IEE Proceedings F, vol. 140, no. 2, pp. 107–113, Apr. 1993.

[7] F. Daum and J. Huang
"Particle Flow for Nonlinear Filters, Bayesian Decisions and Transport,"
in *2013 16th International Conference on Information Fusion (Fusion)*, Jul. 2013, pp. 1072–1079.

[8] M. Khan and M. Ulmke
"Non-Linear and Non-Gaussian State Estimation using log-Homotopy Based Particle Flow Filters,"
in *Sensor Data Fusion: Trends, Solutions, Applications (SDF), 2014*, Oct. 2014, pp. 1–6.

[9] U. Hanebeck
"Pgf 42: Progressive Gaussian Filtering with a Twist,"
in *2013 16th International Conference on Information Fusion (Fusion)* Jul. 2013, pp. 1103–1110.

[10] T. Yang, P. Mehta, and S. Meyn
"Feedback Particle Filter,"
IEEE Transactions on Automatic Control, vol. 58, no. 10, pp. 2465–2480, Oct. 2013.

[11] G. Evensen
"Sequential Data Assimilation with a Nonlinear Quasi-Geostrophic Model using Monte Carlo Methods to Forecast Error Statistics,"
Journal of Geophysical Research, vol. 99, pp. 10143–10162, 1994.

[12] S. Reich
"A Gaussian-Mixture Ensemble Transform Filter,"
Quarterly Journal of the Royal Meteorological Society, vol. 138, pp. 222–233, Jan. 2012.

[13] M. Glodek, M. Schels, and F. Schwenker
"Ensemble Gaussian Mixture Models for Probability Density Estimation (report),"
Computational Statistics, vol. 28, no. 1, p. 127(12), Feb. 2013.

[14] S. Challa and Y. Bar-Shalom
"Nonlinear Filter Design using Fokker Planck Kolmogorov Probability Density Evolutions,"
IEEE Transactions on AeroSpace and Electronic Systems, vol. 36, no. 1, pp. 309–315, Jan. 2000.

[15] G. Beylkin and M. J. Mohlenkamp
"Algorithms for Numerical Analysis in High Dimensions,"
SIAM Journal on Scientific Computing, vol. 26, pp. 2133–2159, 2005.

[16] B. N. Khoromskij
"Tensors-Structured Numerical Methods in Scientific Computing: Survey on Recent Advances,"
Chemometrics and Intelligent Laboratory Systems, vol. 110, no. 1, pp. 1–19, 2012.

[17] Y. Sun and M. Kumar
"Nonlinear Bayesian Filtering Based on Fokker Planck Equation and Tensor Decomposition,"
in *2015 18th International Conference on Information Fusion (Fusion)*, July 2015, pp. 1483–1488.

[18] M. A. Khan, M. Ulmke, B. Demissie, F. Govaers, and W. Koch
"Combining Log-Homotopy Flow with Tensor Decomposition Based Solution for Fokker Planck Equation,"
in *Proceedings of the 19th International Conference on Information Fusion*, Istanbul, Turkey, July 2016.

[19] B. Demissie, F. Govaers, and M. A. Khan
"Nonlinear Filter Design using Fokker-Planck Propagator in Kronecker Tensor Format,"
in *Proceedings of the 19th International Conference on Information Fusion*, 2016.

[20] F. L. Hitchcock
"The Expression of a Tensor or a Polyadic as a Sum of Products,"
Journal of Mathematics and Physics, vol. 6, no. 1–4, pp. 164–189, 1927 [Online]. Available: http://dx.doi.org/10.1002/sapm192761164.

[21] A.-H. Phan, P. Tichavsky, and A. Cichocki
"Tensor Deflation for CANDECOMP/PARAFAC. Part 1: Alternating SubSpace Update Algorithm,"
IEEE Transactions on Signal Processing, vol. 63, no. 22, pp. 5924–5938, Nov. 2015.

[22] C. J. Hillar and L.-H. Lim
"Most Tensor Problems are NP-Hard,"
Journal of the ACM, vol. 60, no. 6, pp. 45:1–45:39, Nov. 2013 [Online]. Available: http://doi.acm.org/10.1145/2512329.

[23] R. A. Harshman
"Foundations of the PARAFAC Procedure: Models and Conditions for an 'Explanatory' Multimodal Factor Analysis," UCLA Working Papers in Phonetics, 1970.

[24] W. Koch
in *Signal Processing Handbook*,
S. Stergiopoulos
Ed. CRC Press, Boca Raton, FL, 2000.

[25] R. Mahler
"Multitarget Bayes Filtering via First-Order Multitarget Moments,"
IEEE Transactions on Aeropace and Electronic Systems, vol. 39, no. 4, pp. 1152–1178, Nov. 2003.

[26] R. Streit
Poisson Point Processes: Imaging, Tracking, and Sensing. Springer, 2010.

[27] C. Gardiner
Stochastic Methods. Springer, 2009.

[28] R. P. S. Mahler
"Multitarget Bayes Filtering via First-Order Multitarget Moments,"
IEEE Transactions on Aeropace and Electronic Systems, vol. 39, no. 4, pp. 1152–1178, Oct. 2003.

[29] R. L. Streit and L. D. Stone
"Bayes Derivation of Multitarget Intensity Filters,"
in *2008 11th International Conference on Information Fusion*, June 2008, pp. 1–8.

[30] M. Beard, S. Reuter, K. Granström, B. T. Vo, B. N. Vo, and A. Scheel
"A Generalised Labelled Multi-Bernoulli Filter for Extended Multi-Target Tracking,"
in *2015 18th International Conference on Information Fusion (Fusion)*, July 2015, pp. 991–998.

[31] D. Törnqvist, T. B. Schön, R. Karlsson, and F. Gustafsson
"Particle Filter SLAM with High Dimensional Vehicle Model,"
Journal of Intelligent and Robotic Systems, vol. 55, no. 4, pp. 249–266, 2009 [Online]. Available: http://dx.doi.org/10.1007/s10846-008-9301-y.

Felix Govaers received the diploma in mathematics in 2007 and the Ph.D. degree in computer science in 2012, both from the University of Bonn, Bonn, Germany. Since 2009, he has been with the Department of Sensor Data Fusion and Information Fusion, Fraunhofer FKIE, where he led the research group distributed systems from 2014 to 2017. He is currently the Deputy Head with the Department of Sensor Data Fusion and Information Fusion, Fraunhofer FKIE. His research interests include data fusion for state estimation in nonlinear scenarios and in sensor networks, which includes track extraction, processing of delayed measurements, as well as the distributed Kalman filter and track-to-track fusion, and advances in state estimation, such as particle flow and homotopy filters and the random finite set theory approaches. His current research projects are tensor decomposition-based approaches to multitarget tracking. Since 2008, he has been an active member of the International Society of Information Fusion (ISIF) community. He organized the ISIF-cosponsored Sensor Data Fusion Workshop in Germany for many years as the Technical Program Chair. Since 2017, he has been an active member of the ISIF Board of Directors. At FUSION 2016, he was the Publication Chair of the organization team. Since 2014, he was an Associate Editor for the *IEEE Transactions on Aerospace and Electronic Systems*.

Bruno Demissie received the Diploma Degree in theoretical physics from the Ruhr-University-Bochum, Bochum, Germany and the Dr. rer. nat. degree from Carl-von-Ossietzky University, Oldenburg, Germany, in 1994 and 1997, respectively. He was a Postdoctoral Researcher with the Phillips-University, Marburg, Germany. From February 1998 until March 2000, he was a MINERVA fellow with the Weizmann Institute of Science, Rehovot, Israel. Since 2000, he has been with the Fraunhofer Institute for Communications, Information Processing and Ergonomics. He has been a member of the editorial board of the *International Journal of Electronics and Communications* (AEUE). Since 2015, he has been an Associate Editor and subsequently a Senior Area Editor for *IEEE Signal Processing Letters*.

Altamash Khan received the B.E. in electrical engineering from the National University of Sciences and Technology (NUST), Islamabad, Pakistan, in 2006 and the M.Sc. degree in wireless systems from KTH Royal Institute of Technology, Stockholm, Sweden in 2011. He also received the Ph.D. degree in computer science from the University of Bonn, Bonn, Germany in 2018. His doctoral research focused on the development of log-homotopy-based particle filters for providing efficient statistical estimation for complex systems. From April 2014 to March 2017, he was a Marie Curie Fellow at FKIE Fraunhofer, Wachtberg, Germany. Since July 2017, he has been with Zenuity, Gothenburg, Sweden, as an Algorithm Development Engineer for advanced driving assistance systems in the sensor fusion group. His research interests include nonlinear and non-Gaussian state estimation methods, multitarget tracking, signal processing for automotive applications, and wireless sensors networks.

Martin Ulmke received the Dipl.-Phy. (M.S.) and Dr. rer. nat. (Ph.D.) degrees from the Aachen Technical University (RWTH), Aachen, Germany, in 1991 and 1995, respectively. From 1995 to 1998, he was a Research Associate in condensed matter theory at the University of California in Davis, CA, USA and at the University of Augsburg, Augsburg, Germany. From 1998 to 2001, he was a System Engineer with MTU Aero Engines, Munich, Germany. Since 2001, he has been a part of the scientific staff with the Department of Sensor Data and Information Fusion, Fraunhofer FKIE, where he was the Head of Research Group "Distributed Sensor Systems".

Wolfgang Koch received the Dr. rer. nat. degree in theoretical physics from the Aachen Technical University, Aachen, Germany. For many years, he was the Head of the Department of Sensor Data and Information Fusion, Fraunhofer FKIE, an institute of the Fraunhofer Society, the largest institution for applied research in Europe. He has authored or coauthored a well-referenced textbook on target tracking and sensor data fusion, 16 handbook chapters, and above 200 journal and conference articles on various topics within the area of sensor data and information fusion. He was one of the coeditors of the two-volume handbook *Novel Radar Techniques and Applications*. He was a Fellow of the IEEE, where he was with the *IEEE Transactions on Aerospace and Electronic Systems* and was a member of the board of governors of the Aerospace and Electronics Systems Society. In 2015, he was the IEEE Distinguished Lecturer and Chairman of the German IEEE AESS chapter. In 2013, he was the President of the ISIF. His research interests include the NATO Science and Technology Organization. He received the Habilitation Degree in applied computer science from Bonn University, Bonn, Germany, where he gave lecture series on sensor data and information fusion. In 2016, he was the General Co-chair of the IEEE ISF International Conference on Information Fusion, Heidelberg, Germany.

CRLB for Estimation of 3D Sensor Biases in Spherical Coordinates and Its Attainability

MICHAEL KOWALSKI
DJEDJIGA BELFADEL
YAAKOV BAR-SHALOM
PETER WILLETT

In order to carry out data fusion, it is crucial to account for the imprecision of sensor measurements due to systematic errors. This requires estimation of the sensor measurement biases. In this paper, we consider a three-dimensional multisensor–multitarget maximum likelihood bias estimation approach for both additive and multiplicative biases in the measurements. Multiplicative biases can more accurately represent real biases in many sensors; however, they increase the complexity of the estimation problem. By converting biased measurements into pseudo-measurements of the biases, it is possible to estimate biases separately from target state estimation. The conversion of the spherical measurements to Cartesian measurements, which has to be done using the unbiased conversion, is the key that allows estimation of the sensor biases without having to estimate the states of the targets of opportunity. The measurements provided by these sensors are assumed time-coincident (synchronous) and perfectly associated. We evaluate the Cramér–Rao lower bound on the covariance of the bias estimates, which serves as a quantification of the available information about the biases. Through the use of the iterated least squares, it is proved that it is possible to achieve statistically efficient estimates.

Manuscript received January 18, 2019; revised April 9, 2019; released for publication July 8, 2019. Approved for Public Release 17-MDA-9446 (Dec. 15, 2017). There are no conflict of interest or financial disclosure statements relevant to this publication.

M. Kowalski, Y. Bar-Shalom, and P. Willett are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT, USA
D. Belfadel is with the Department of Electrical Engineering, Fairfield University, Fairfield, CT, USA
E-mail: michael.p.kowalski@uconn.edu, dbelfadel@fairfield.edu, yaakov.bar-shalom@uconn.edu, peter.willett@uconn.edu

1557-6418/19/$17.00

I. INTRODUCTION

Bias estimation and compensation are essential steps in distributed tracking systems. The objective of sensor registration is to estimate the biases in sensor measurements, such as scale (multiplicative) and offset (additive) biases in range, azimuth, and elevation measurements, clock bias, and/or uncertainties in sensor positions [4]. Owing to this, much effort has been devoted in the last few years to bias estimation procedures for multisensor–multitarget tracking systems. Biases in sensors have been approximated in several ways, including error in sensor pointing and additive biases in the measurements. However, real sensor biases can be more complex than such approximations. One reason is a combination of both multiplicative and additive biases in measurements. That is, a bias may cause increased error in a target that is further away from the sensor or on the periphery of the sensor's field of view.

In [19] and [20], a joint track-to-track bias estimation and fusion approach based on equivalent measurements of the local tracks was proposed. In [14], an approach is used to carry out track-to-track association by assuming additive biases in 2D Cartesian coordinates. In [11], another approach based on pseudo-measurements along with expectation–maximization (EM) to perform joint fusion and registration was proposed. A different method that uses a multistart local search to handle the joint track-to-track association and bias estimation problem was introduced in [21]. The concept of pseudo-measurement was used in [15] for exact bias estimation with further extensions in [16] and [17]. In addition, these methods require perfect knowledge about each local filter and its dynamic model. Also, as the number of sensors increases, the bias estimation problem suffers from the curse of dimensionality because of the commonly used stacked bias vector implementation [10]. In [5], [7], and [9], pseudo-measurements are used with maximum likelihood (ML) to estimate a combination of rotational biases, position biases, and additive measurement biases in addition to presenting the hybrid Cramér–Rao lower bound (HCRLB) as a metric for evaluating estimator efficiency. This is expanded upon in [6] and [8] with EM methods used instead of ML.

In this paper, the novelty is to use the method and Cramér–Rao lower bound (CRLB) derived in [13] combined with an ML method using iterated least squares (ILS) to solve the problem of estimating both multiplicative and additive biases for three-dimensional (3D) spherical sensors. The primary novelty of this work is to extend the work in [15]–[17] to 3D sensors including an unbiased conversion of the measurement covariance found in [18]. This work also builds on previous bias estimation research in [5], [7], and [9] by using a nonlinear weighted ML method to avoid the problems with biased estimates and lack of statistical efficiency. Additionally, multiplicative biases are used instead of the combination of rotational biases, position biases, and

additive measurement biases. The difficulty with this is that it increases the complexity of the bias estimation problem: now there are two sources of error in the measurement other than noise and the error from the multiplicative bias varies depending on the location of the target. More targets are required and it becomes necessary for the targets to be spaced such that the multiplicative bias can be differentiated from the additive bias. Therefore, an analysis of the CRLB is made to determine whether this method can achieve accuracy in bias estimates that is comparable to the error from noise. The CRLB gives a lower bound on the accuracy of bias estimates when using the pseudo-measurements, allowing analysis of performance when the measurement conversion is used rather than the raw measurements. Through the use of the pseudo-measurement model described in [15]–[17], it is possible to avoid the need to estimate the states of the targets and estimate only the sensor biases. Once the sensor biases are estimated, they can be removed from the measurements and the (nearly) bias-free measurements can then be used in tracking systems.

An important metric when using pseudo-measurements is the HCRLB that is discussed in [5], [7], and [9] and evaluated using ML methods. The HCRLB is the CRLB but calculated using all measurements and a parameter vector including the bias variables and all nuisance variables. In this case, the nuisance variables that are included are the target states. The removal of the target state in the calculation of the CRLB may result in a higher metric than the true lower bound that takes into account all the nuisance parameters available to the estimator. This means that it is necessary to include this metric in simulation results to understand how much accuracy is lost using the pseudo-measurement model. It is important to note that the HCRLB is a lower bound and may not be achieved by an estimator; however, EM approaches can be used to improve results such that they are closer to the HCRLB [6], [8]. Furthermore, estimating every nuisance variable may be computationally intensive, which would make the pseudo-measurement method attractive despite the loss of accuracy. Finally, calculating the CRLB does not require information about the target state, unlike the HCRLB that requires an estimate of the target states.

The paper is structured as follows. The bias model and the assumptions for bias estimation are discussed in Section II. In Section III, a review of the exact bias estimation method is given. The key to create the bias pseudo-measurements in Cartesian coordinates, which allows avoiding the need to estimate the states of the targets of opportunity, is to use the unbiased transformation from spherical to Cartesian [18]. The pseudo-measurement model is presented in Section III-A and the ILS estimator is described in Section III-B. Section III-C presents the calculation of the CRLB for the proposed method. Section IV demonstrates the performance of the method for synchronous sensors and compares the root mean squared error (RMSE) of the estimator with the CRLB. Conclusions are discussed in Section V.

II. PROBLEM FORMULATION

A. Coordinate Frames and Measurement Space

In a typical 3D sensor, the measured values of position are in spherical coordinates—range, azimuth, and elevation. Assume there are N_S synchronized sensors, with known positions, reporting range, azimuth, and elevation measurements in spherical coordinates of $t = 1, \ldots, N_T$ targets in the common surveillance region with K total time steps. The true range, azimuth, and elevation are represented by $r_{s,t}(k)$, $\theta_{s,t}(k)$, and $\alpha_{s,t}(k)$, respectively. The noise- and bias-free measurements originating from target t for sensor s at time k are

$$\begin{aligned} r_{s,t}(k) &= \sqrt{x_{s,t}(k)^2 + y_{s,t}(k)^2 + z_{s,t}(k)^2} \\ \theta_{s,t}(k) &= \tan^{-1}\left(\frac{y_{s,t}(k)}{x_{s,t}(k)}\right) \\ \alpha_{s,t}(k) &= \tan^{-1}\left(\frac{z_{s,t}(k)}{\sqrt{x_{s,t}(k)^2 + y_{s,t}(k)^2}}\right). \end{aligned} \tag{1}$$

Each sensor views the target using its own sensor reference frame; therefore,

$$\begin{aligned} \mathbf{x}_{s,t}(k) &= \begin{bmatrix} x_{s,t}(k) \\ y_{s,t}(k) \\ z_{s,t}(k) \end{bmatrix} = \begin{bmatrix} x_t(k) - x_s(k) \\ y_t(k) - y_s(k) \\ z_t(k) - z_s(k) \end{bmatrix} \\ &= \mathbf{x}_t(k) - \mathbf{x}_s(k) \end{aligned} \tag{2}$$

where $\mathbf{x}_t(k) = [x_t(k), y_t(k), z_t(k)]$ is the true position in Cartesian coordinates of target t at time step k and $\mathbf{x}_s(k) = [x_s(k), y_s(k), z_s(k)]$ is the true position in Cartesian coordinates of sensor s at time step k. Transforming (1) to a Cartesian coordinate frame yields

$$\begin{aligned} x^c_{s,t}(k) &= r_{s,t}(k)\cos\left(\theta_{s,t}(k)\right)\cos\left(\alpha_{s,t}(k)\right) \\ y^c_{s,t}(k) &= r_{s,t}(k)\sin\left(\theta_{s,t}(k)\right)\cos\left(\alpha_{s,t}(k)\right) \\ z^c_{s,t}(k) &= r_{s,t}(k)\sin\left(\alpha_{s,t}(k)\right). \end{aligned} \tag{3}$$

For a given sensor, each measurement is modeled as a function of the actual (true) target state, systematic errors (biases), and random errors (noise). The model for the measurements originating from a target with additive and multiplicative biases at time k in spherical coordinates for sensor s is

$$\begin{aligned} z_{s,t}(k) &= \begin{bmatrix} r^m_{s,t}(k) \\ \theta^m_{s,t}(k) \\ \alpha^m_{s,t}(k) \end{bmatrix} \\ &= \begin{bmatrix} \left[1 + \epsilon^r_s(k)\right] r_{s,t}(k) + b^r_s + w^r_s(k) \\ \left[1 + \epsilon^\theta_s(k)\right] \theta_{s,t}(k) + b^\theta_s + w^\theta_s(k) \\ \left[1 + \epsilon^\alpha_s(k)\right] \alpha_{s,t}(k) + b^\alpha_s + w^\alpha_s(k) \end{bmatrix} \\ & \qquad s = 1, \ldots, N_S, \quad t = 1, \ldots, N_T \end{aligned} \tag{4}$$

where $r^m_{s,t}(k)$, $\theta^m_{s,t}(k)$, and $\alpha^m_{s,t}(k)$ are the measured range, azimuth, and elevation, respectively, b^r_s, b^θ_s, and b^α_s are the offset biases in the range, azimuth, and elevation, respectively, and $\epsilon^r_s(k)$, $\epsilon^\theta_s(k)$, and $\epsilon^\alpha_s(k)$ are the scale biases in the range, azimuth, and elevation, respectively. The measurement noises $w^r_s(k)$, $w^\theta_s(k)$, and $w^\alpha_s(k)$ in range, azimuth, and elevation are zero mean with corresponding variances σ^2_r, σ^2_θ, and σ^2_α, respectively, and are assumed mutually independent. The bias vector for sensor s is

$$\beta_s = [\, b^r_s \quad b^\theta_s \quad b^\alpha_s \quad \epsilon^r_s \quad \epsilon^\theta_s \quad \epsilon^\alpha_s \,]^{\mathrm{T}} \tag{5}$$

and is modeled as an unknown constant over a certain window of scans (nonrandom variable). Consequently, the ML estimator [2] or the least-squares estimator [1] can be used for bias estimation. On the other hand, a Gauss–Markov random model [22] can also be used, in which case a Kalman filter can be adopted for bias estimation. We model the measurement equation (4) as

$$z_{s,t}(k) = \begin{bmatrix} r_{s,t}(k) \\ \theta_{s,t}(k) \\ \alpha_{s,t}(k) \end{bmatrix} + C_{s,t}(k)\beta_s + \begin{bmatrix} w^r_s(k) \\ w^\theta_s(k) \\ w^\alpha_s(k) \end{bmatrix} \tag{6}$$

where

$$C_{s,t}(k) \triangleq \begin{bmatrix} 1 & 0 & 0 & r_{s,t}(k) & 0 & 0 \\ 0 & 1 & 0 & 0 & \theta_{s,t}(k) & 0 \\ 0 & 0 & 1 & 0 & 0 & \alpha_{s,t}(k) \end{bmatrix}. \tag{7}$$

Here, the measured azimuth $\theta^m_{s,t}(k)$, elevation $\alpha^m_{s,t}(k)$, and range $r^m_{s,t}(k)$ can be utilized in (7) without any significant loss of performance [15]–[17].

The problem is to estimate the bias vectors β_s for all sensors. After bias estimation, all the biases can be compensated for to obtain the state estimates. Since the motion equations of targets are naturally expressed in Cartesian coordinates, if the spherical measurements can be converted to Cartesian (via nonlinear transformation) without introducing coordinate conversion bias and obtaining the correct covariance for the converted measurements, one can then perform the state estimation within a completely linear framework. Then, sensor s has the measurement equation in Cartesian coordinates (with the same $H_s(k) = H(k)$ for all sensors)

$$z^c_{s,t}(k) = H(k)\mathbf{x}_t(k) + B_{s,t}(k)C_{s,t}(k)\beta_s + \mathbf{x}_s(k) + w_s(k) \tag{8}$$

where the state vector is

$$\mathbf{x}_t(k) = \left[x_t(k)\ y_t(k)\ z_t(k)\right]^{\mathrm{T}} \tag{9}$$

and $H(k)$ is the measurement matrix given by

$$H(k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \triangleq H. \tag{10}$$

Using the measured azimuth $\theta^m_{s,t}(k)$, elevation $\alpha^m_{s,t}(k)$, and range $r^m_{s,t}(k)$ from sensor s, the Jacobian of the Cartesian measurements with respect to the biases in each coordinate, $B_{s,t}(k)$, can be written (omitting subscripts s and t, superscript m, and time step k for simplicity) as

$$B_{s,t}(k) = \begin{bmatrix} \cos\theta\cos\alpha & -r\sin\theta\cos\alpha & -r\cos\theta\sin\alpha \\ \sin\theta\cos\alpha & r\cos\theta\cos\alpha & -r\sin\theta\sin\alpha \\ \sin\alpha & 0 & r\cos\alpha \end{bmatrix}. \tag{11}$$

The transformation of the measurements from spherical to Cartesian coordinates that has to be used is the unbiased one [18]. This was found necessary to ensure the accuracy of the bias estimates and is discussed in more detail at the end of the section.

The unbiased conversion converts the original measurements with the following equations:

$$\begin{aligned} x^{c,m}_{s,t}(k) &= \lambda^{-1}_\theta \lambda^{-1}_\alpha r_{s,t}(k)\cos\theta_{s,t}(k)\cos\alpha_{s,t}(k) + x_s(k) \\ y^{c,m}_{s,t}(k) &= \lambda^{-1}_\theta \lambda^{-1}_\alpha r_{s,t}(k)\sin\theta_{s,t}(k)\cos\alpha_{s,t}(k) + y_s(k) \\ z^{c,m}_{s,t}(k) &= \lambda^{-1}_\alpha r_{s,t}(k)\sin\alpha_{s,t}(k) + z_s(k) \end{aligned} \tag{12}$$

$$z^{c,m}_{s,t}(k) = \begin{bmatrix} x^{c,m}_{s,t}(k) \\ y^{c,m}_{s,t}(k) \\ z^{c,m}_{s,t}(k) \end{bmatrix}. \tag{13}$$

The new (unbiased) covariance matrix of the measurements in Cartesian coordinates (omitting indexes m and k in the measurements for simplicity) is given by

$$R_{s,t}(k) = \begin{pmatrix} R^{s,t}_{xx} & R^{s,t}_{xy} & R^{s,t}_{xz} \\ R^{s,t}_{xy} & R^{s,t}_{yy} & R^{s,t}_{yz} \\ R^{s,t}_{xz} & R^{s,t}_{yz} & R^{s,t}_{zz} \end{pmatrix} \tag{14}$$

$$\begin{aligned} R^{s,t}_{xx} &= (\lambda^{-2}_\theta \lambda^{-2}_\alpha - 2) r^2_{s,t}\cos^2\theta_{s,t}\cos^2\alpha_{s,t} \\ &\quad + \frac{1}{4}(r^2_{s,t} + \sigma^2_r)(1 + \lambda'_\theta \cos 2\theta_{s,t})(1 + \lambda'_\alpha \cos 2\alpha_{s,t}) \\ R^{s,t}_{yy} &= (\lambda^{-2}_\theta \lambda^{-2}_\alpha - 2) r^2_{s,t}\sin^2\theta_{s,t}\cos^2\alpha_{s,t} \\ &\quad + \frac{1}{4}(r^2_{s,t} + \sigma^2_r)(1 - \lambda'_\theta \cos 2\theta_{s,t})(1 + \lambda'_\alpha \cos 2\alpha_{s,t}) \\ R^{s,t}_{zz} &= (\lambda^{-2}_\alpha - 2) r^2_{s,t}\sin^2\alpha_{s,t} \\ &\quad + \frac{1}{2}(r^2_{s,t} + \sigma^2_r)(1 - \lambda'_\alpha \cos 2\alpha_{s,t}) \\ R^{s,t}_{xy} &= (\lambda^{-2}_\theta \lambda^{-2}_\alpha - 2) r^2_{s,t}\sin\theta_{s,t}\cos\theta_{s,t}\cos^2\alpha_{s,t} \\ &\quad + \frac{1}{4}(r^2_{s,t} + \sigma^2_r)\lambda'_\theta \sin 2\theta_{s,t}(1 + \lambda'_\alpha \cos 2\alpha_{s,t}) \\ R^{s,t}_{xz} &= (\lambda^{-1}_\theta \lambda^{-2}_\alpha - \lambda^{-1}_\theta - \lambda_\theta) r^2_{s,t}\cos\theta_{s,t}\sin\alpha_{s,t}\cos\alpha_{s,t} v \\ &\quad + \frac{1}{2}(r^2_{s,t} + \sigma^2_r)\lambda_\theta \lambda'_\alpha \cos\theta_{s,t}\sin 2\alpha_{s,t} \\ R^{s,t}_{yz} &= (\lambda^{-1}_\theta \lambda^{-2}_\alpha - \lambda^{-1}_\theta - \lambda_\theta) r^2_{s,t}\sin\theta_{s,t}\sin\alpha_{s,t}\cos\alpha_{s,t} \\ &\quad + \frac{1}{2}(r^2_{s,t} + \sigma^2_r)\lambda_\theta \lambda'_\alpha \sin\theta_{s,t}\sin 2\alpha_{s,t} \end{aligned} \tag{15}$$

where

$$\lambda_\theta = e^{-\sigma_\theta^2/2}$$
$$\lambda'_\theta = e^{-2\sigma_\theta^2} = \lambda_\theta^4$$
$$\lambda_\alpha = e^{-\sigma_\alpha^2/2}$$
$$\lambda'_\alpha = e^{-2\sigma_\alpha^2} = \lambda_\alpha^4. \tag{16}$$

The debiasing coefficients (16) are used in the calculation of the converted covariance matrix and this conversion bias interferes with the estimation of the consistent measurement biases. If the debiasing coefficients are zero, then the converted covariance matrix results in negative values, which causes negative values in the CRLB. Furthermore, the conversion bias adds to the error resulting from the measurement biases. The estimator has difficulty in differentiating this error from the error from the biases. If there is no noise or extremely little noise, it is possible to use the standard conversion to estimate the biases, but without CRLB efficiency. This is unreliable though and results may vary depending on the number of targets, their positions, and the magnitude of the biases, and in any case, the unbiased conversion adds little numerical complication.

Additionally, the calculation of the covariance matrix is necessary for use in ML methods in order to avoid biased estimates and to generate the CRLB. If a least-squares method is used but with identity noise rather than an accurate measurement noise matrix, it is likely to result in statistically inefficient estimates and potentially biased estimates [5], [7], [9].

III. SYNCHRONOUS SENSOR REGISTRATION FOR THE 3D CASE

In this section, the bias estimation method introduced in [15]–[17] for synchronous sensors with known sensor locations is reviewed and extended to the 3D case, with various simulations and the calculation of the lower bounds for bias estimation in multisensor–multitarget scenarios.

The estimator uses a batch of measurements from a number of time steps to estimate the biases. The parameter vector to be estimated consists of the biases, and pseudo-measurements are used to measure the effect of the biases. The pseudo-measurements remove the true target states in order to only measure the effect of the biases. The target states are not estimated with this estimator.

The dynamic equation for the target state is

$$\mathbf{x}(k) = \left[\mathbf{x}_t(k)^{\mathrm{T}}, \dot{\mathbf{x}}_t(k)^{\mathrm{T}}, \ddot{\mathbf{x}}_t(k)^{\mathrm{T}}\right]^{\mathrm{T}} \tag{17}$$

$$\mathbf{x}(k+1) = F(k)\mathbf{x}(k) + \mathbf{v}(k) \tag{18}$$

where $F(k)$ is the transition matrix and $\mathbf{v}(k)$ is a zero-mean additive white Gaussian noise with covariance $Q(k)$.

Because the local trackers are not able to estimate the biases on their own, they yield inaccurate estimates of tracks by assuming no bias in their measurements. Hence, the state space model considered by local trackers for a specific target t and sensor s is

$$\mathbf{x}_t(k+1) = \mathbf{x}_t(k) + F_t(k)\left[\dot{\mathbf{x}}_t(k)^{\mathrm{T}}\ddot{\mathbf{x}}_t(k)^{\mathrm{T}}\right]^{\mathrm{T}} + v(k) \tag{19}$$

$$z_{s,t}(k) = H(k)\mathbf{x}_t(k) + w_s(k) \tag{20}$$

where $F_t(k)$ is a submatrix of $F(k)$. In this method, the transition matrix can be unknown as the target state is not estimated. The difference between (4) and (20) is that the latter has no bias term and, as a result, the local tracks are bias-ignorant [15]–[17]. Note that this mismatch should be compensated for.

A. The Pseudo-Measurement of the Bias Vector

In this subsection, a discussion on how to find an informative pseudo-measurement by using the local tracks for the case $N_S = 2$ synchronized 3D sensors is presented, generalizing the method given in [15]–[17].

The pseudo-measurement of the bias vector is defined as

$$z_t^p(k) \triangleq z_{1,t}^c(k) - z_{2,t}^c(k) \tag{21}$$

In the above equation, the true position of the target is eliminated because of cancellation since each such position is multiplied by the same matrix (24). This results in the following equation:

$$z_t^p(k) = B_{1,t}(k)C_{1,t}(k)\beta_1 - B_{2,t}(k)C_{2,t}(k)\beta_2 + w_1(k) - w_2(k). \tag{22}$$

The pseudo-measurement of the bias vector can be written as

$$z_t^p(k) = \mathcal{H}_t(k)\mathbf{b} + \tilde{w}(k) \tag{23}$$

where the pseudo-measurement matrix $\mathcal{H}$, the bias parameter vector $\mathbf{b}$, and the pseudo-measurement noise $\tilde{w}(k)$ are defined as

$$\mathcal{H}_t(k) \triangleq \begin{bmatrix} (B_{1,t}(k)C_{1,t}(k))^{\mathrm{T}} \\ (-B_{2,t}(k)C_{2,t}(k))^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{24}$$

$$\mathbf{b} \triangleq \left[\beta_1^{\mathrm{T}}, \beta_2^{\mathrm{T}}\right]^{\mathrm{T}} \tag{25}$$

and

$$\tilde{w}(k) \triangleq w_1(k) - w_2(k). \tag{26}$$

The bias pseudo-measurement noises $\tilde{w}$ are additive white Gaussian with zero mean, and their covariance is

$$\mathcal{R}_t(k) = R_{1,t}(k) + R_{2,t}(k). \tag{27}$$

The key property of (26) is its whiteness, which results in an exact bias estimate. In this approach, there is no

approximation in deriving (23)–(27) unlike the methods previously proposed in [12], [23], and [24]. This was one of the main contributions of [15].

B. The ILS Method

If the biases are constant for each measurement over the batch of scans, then an ILS method can be used. This estimator finds the ML estimate [3] of the bias vector **b**. This estimator uses the Jacobian calculated previously in (24) for the pseudo-measurements of the bias vector as well as the noise covariance matrix (27). The measurements and matrices must be stacked in a batch for the estimator. The measurement batch is

$$\mathbf{z}^p = \left[z_1^p(1)^{\mathrm{T}}, \ldots, z_1^p(K)^{\mathrm{T}}, z_2^p(1)^{\mathrm{T}}, \ldots, z_{N_T}^p(K)^{\mathrm{T}}\right]^{\mathrm{T}}. \quad (28)$$

The Jacobian matrix batch is defined for each estimator iteration j as

$$\mathbf{H}^j = [\mathcal{H}_1^j(1)^{\mathrm{T}}, \ldots, \mathcal{H}_1^j(K)^{\mathrm{T}}, \mathcal{H}_2^j(1)^{\mathrm{T}}, \ldots, \mathcal{H}_{N_T}^j(K)^{\mathrm{T}}]^{\mathrm{T}}. \quad (29)$$

The noise covariance for the batch is a diagonal matrix composed of the individual covariance matrices

$$\mathbf{R} = \begin{bmatrix} \mathcal{R}_1(1) & 0 & 0 & 0 & 0 \\ 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & \mathcal{R}_1(K) & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & 0 & \mathcal{R}_{N_T}(K) \end{bmatrix}. \quad (30)$$

The equation for each iteration j of the estimator is

$$\mathbf{b}_e^{j+1} = \mathbf{b}_e^j + [\mathbf{H}^{j\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}^j]^{-1}\mathbf{H}^{j\mathrm{T}}\mathbf{R}^{-1}[\mathbf{z}^p - \mathbf{h}(\mathbf{b}_e^j)]. \quad (31)$$

At each iteration, the current state estimate is used to generate a predicted measurement vector to compare to the actual measurements

$$\mathbf{h}(\mathbf{b}_e^j) = \mathbf{H}^j\mathbf{b}_e^j. \quad (32)$$

When the state no longer changes significantly, then the estimator stops and takes the final iteration as its estimated parameter. To initialize the estimator, the biases are assumed to be zero

$$\mathbf{b}_e^0 = [0, 0, \ldots, 0]^{\mathrm{T}}. \quad (33)$$

In order for the estimator to be observable, a bare minimum of measurements is needed to satisfy the requirement that there will be at least one pseudo-measurement per parameter vector element. This results in the following inequality:

$$3KN_T\,(N_S - 1) \geq 6N_S. \quad (34)$$

This inequality can be simplified to

$$KN_T - \frac{KN_T}{N_S} \geq 2. \quad (35)$$

In practice, more measurements than this are required together with measurement diversity to obtain satisfactory accuracy. In order to have sufficient measurement diversity, there must be targets spaced such that for one target the error from the multiplicative bias is larger than the error from the additive bias and for another target the error from the additive bias is larger than the error from the multiplicative bias.

C. CRLB for the Biases

To investigate the performance of the estimator, it is necessary to calculate the CRLB. The CRLB is defined [3] as the inverse of the Fisher information matrix.

$$\mathrm{CRLB} = \mathbf{J}^{-1} = [\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}]^{-1}. \quad (36)$$

The CRLB is based on the batch of Jacobians that is calculated in (29) and (24) as well as the batch of noise covariance matrices calculated in (30), (27), and (14). The calculation of the CRLB does not require any knowledge of the target state, although the spherical measurements are used in calculating (11) and (7). It will be shown in the next section that the covariance of the bias estimates attains the CRLB; i.e., the ML estimator is efficient for this problem.

We additionally calculate the HCRLB that is a more accurate lower bound as some of the information in the 3D spherical measurements has been eliminated to produce the pseudo-measurements. The HCRLB takes into account the nuisance variables not originally estimated, in this case the target positions. The parameter vector for the HCRLB is

$$\psi = \left[\mathbf{b}^{\mathrm{T}}, \mathbf{x}_{t1}(1), \ldots, \mathbf{x}_{t1}(K), \ldots, \mathbf{x}_{N_t}(K)\right]^{\mathrm{T}}. \quad (37)$$

The HCRLB is defined as

$$\mathrm{HCRLB} = [\mathbf{H}_\psi^{\mathrm{T}}\mathbf{R}_\psi^{-1}\mathbf{H}_\psi]^{-1} \quad (38)$$

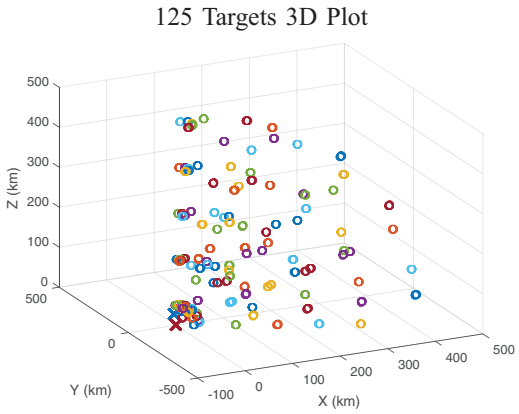
where the Jacobian and covariance associated with the HCRLB are defined as

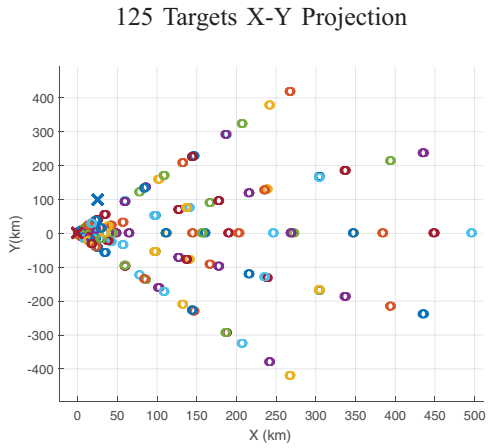
$$\begin{aligned} \mathbf{H}_\psi &= \nabla_\psi \mathbf{z} \\ &= \nabla_\psi \left[z_{s1,t1}(1)^{\mathrm{T}}, \ldots, z_{s1,t1}(K)^{\mathrm{T}}, \ldots, z_{N_s,N_t}(K)^{\mathrm{T}}\right]^{\mathrm{T}} \end{aligned} \quad (39)$$

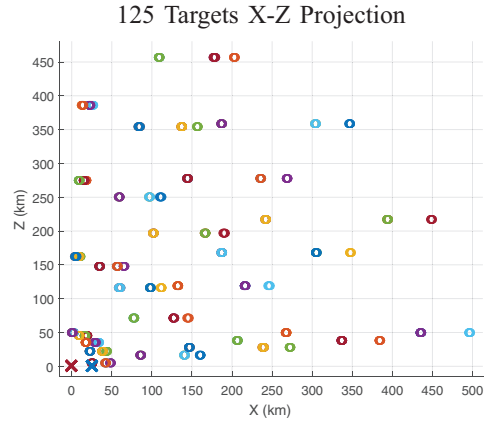
$$\mathbf{R}_\psi = \begin{bmatrix} \sigma_r^2 & 0 & 0 & \ldots \\ 0 & \sigma_\alpha^2 & 0 & \ldots \\ 0 & 0 & \sigma_\epsilon^2 & \ldots \\ \ldots & \ldots & \ldots & \ldots \end{bmatrix}. \quad (40)$$

For brevity, the individual derivatives are not included. The HCRLB is calculated using the true values of the biases and target states.

In situations where the target motion is unknown, the HCRLB tends not to deviate far from the CRLB as the information about the target states is not very accurate compared to the large amount of data contributing to the biases.







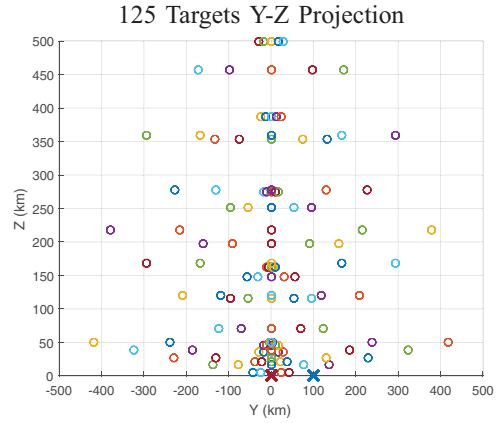
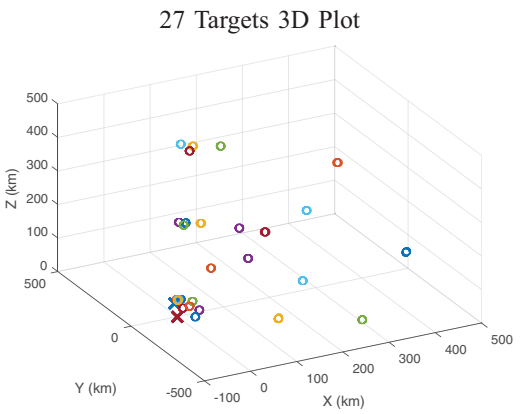
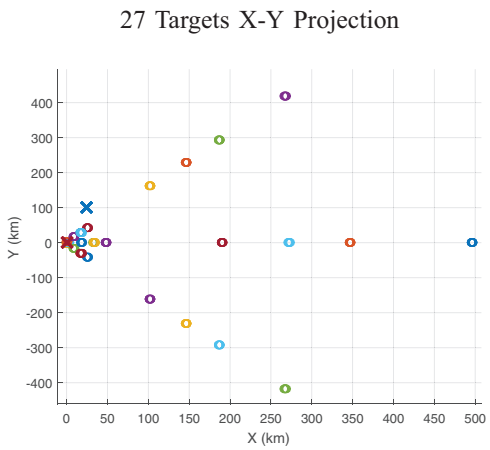
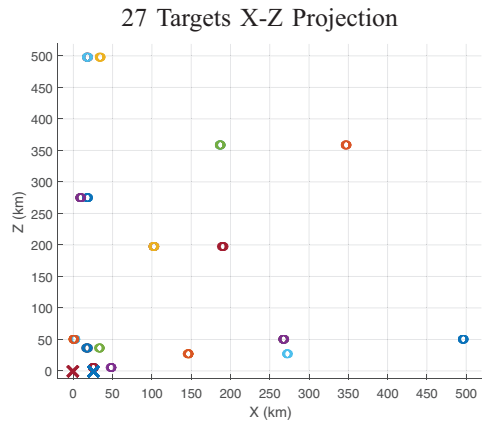


Figure 1. 125-target layout with projections. The × symbols represent sensors and the ○ symbols represent targets. (a) 3D plot. (b) X–Y projection. (c) X–Z projection. (d) Y–Z projection.







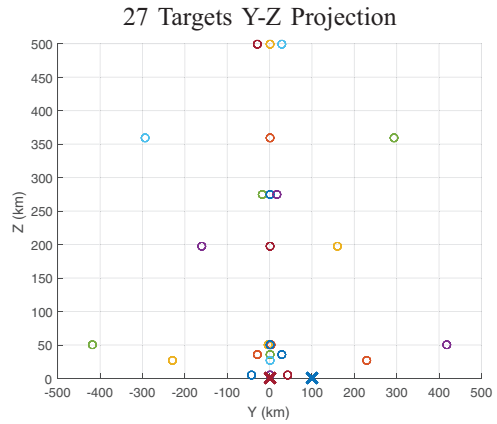


Figure 2. 27-target layout with projections. The × symbols represent sensors and the ○ symbols represent targets. (a) 3D plot. (b) X–Y projection. (c) X–Z projection. (d) Y–Z projection.

IV SIMULATION RESULTS

Simulations are made to test the performance of the approach proposed. Estimation of the biases can be difficult as a number of distinct targets must be used in order to differentiate the effects of the multiplicative and additive biases. In our simulation, sensor 1 is fixed at (0,0,0) km, sensor 2 is fixed at (25,100,0) km, and target velocity is $(-300, 0, 0)$ m/s. The target positions are set up in a cone extended from sensor 1. The ranges are in $[50, 500]$ km, the azimuths are in $[-1, 1]$ rad, and the elevations are in [0.1, 1.5] rad. The additive and multiplicative biases, as well as the noise variances, are in range $[20\ \text{m}, 10^{-4}, 100\ \text{m}^2]$, and in azimuth and elevation $[3\ \text{mrad}, 3 \times 10^{-3}, 1\ \text{mrad}^2]$.

The target positions include a swath of range, azimuth, and elevation that allows each bias to make its effect apparent versus the other. In order to ensure this, the targets are radially placed in a cone from one sensor. In cases of high range, azimuth, and elevation values, the multiplicative biases dominate, whereas in cases of low range, azimuth, and elevation values, the additive biases dominate. In our simulations, the targets move at 300 m/s across time steps with ten measurements at one measurement per second. The sensor configuration is shown in Fig. 1 for 125 targets and in Fig. 2 for 27 targets.

The results of the simulations include the CRLB, RMSE from n_{MC} Monte Carlo runs, and a probability interval around the CRLB for each bias. The probability interval is calculated for the 95% region using the bias error samples from the Monte Carlo runs. The 95% probability interval is calculated by the following equations where σ_{SE} is the standard deviation of the squared error from the n_{MC} Monte Carlo runs:

$$0.95 = P(a < \text{RMSE} < b) \tag{41}$$

$$a = \sqrt{\text{CRLB} - 1.96 \cdot \frac{\sigma_{\text{SE}}}{\sqrt{n_{\text{MC}}}}} \tag{42}$$

$$b = \sqrt{\text{CRLB} + 1.96 \cdot \frac{\sigma_{\text{SE}}}{\sqrt{n_{\text{MC}}}}}. \tag{43}$$

A. Baseline Simulations

The first simulations are a baseline test to determine the performance and efficiency of the estimator. To begin, a simulation was performed with $N_T = 125$ targets and $K = 10$ time steps, the results of which are shown in Table III. In this simulation, it is shown that it is possible to achieve RMSE values that are compatible to the CRLB. The CRLB and RMSE are based on the error in the final bias estimates. The RMSE lies within the 95% probability interval around the CRLB in all cases; thus, the estimator is proved to be efficient [3]. Furthermore, the CRLB values are compared to the true bias and the noise standard deviation. Table III also contains the results that show that the residual bias RMSE is consistently lower than the noise standard deviation. Furthermore, the error from RMSE is lower than the noise standard deviation for

TABLE I
$n_{\text{MC}} = 100$ Runs, $K = 10$ Time Steps, and $N_T = 125$ Targets

Component	CRLB square root	HCRLB square root	RMSE	95% Probability interval around CRLB	Noise standard deviation	Uncorrected bias
Sensor 1 range additive	4.96 m	4.96 m	4.7 m	[4.3, 5.5] m	10 m	20 m
Sensor 1 range multiplicative	4.21×10^{-5}	4.21×10^{-5}	3.88×10^{-5}	$[3.66 \times 10^{-5}, 4.63 \times 10^{-5}]$	10 m	10^{-4}
Sensor 1 azimuth additive	4.85×10^{-2} mrad	4.85×10^{-2} mrad	5.55×10^{-2} mrad	$[3.98 \times 10^{-2}, 5.45 \times 10^{-2}]$ mrad	1 mrad	3 mrad
Sensor 1 azimuth multiplicative	7.89×10^{-5}	7.89×10^{-5}	7.56×10^{-5}	$[6.81 \times 10^{-5}, 8.80 \times 10^{-5}]$	1 mrad	3×10^{-3}
Sensor 1 elevation additive	1.15×10^{-1} mrad	1.14×10^{-1} mrad	1.27×10^{-1} mrad	$[9.24 \times 10^{-2}, 1.33 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 1 elevation multiplicative	9.42×10^{-5}	9.41×10^{-5}	9.82×10^{-5}	$[7.74 \times 10^{-5}, 1.08 \times 10^{-4}]$	1 mrad	3×10^{-3}
Sensor 2 range additive	5.67 m	5.67 m	5.4 m	[5.0, 6.2] m	10 m	20 m
Sensor 2 range multiplicative	4.43×10^{-5}	4.43×10^{-5}	4.33×10^{-5}	$[3.80 \times 10^{-5}, 4.92 \times 10^{-5}]$	10 m	10^{-4}
Sensor 2 azimuth additive	8.44×10^{-2} mrad	8.44×10^{-2} mrad	9.32×10^{-2} mrad	$[6.65 \times 10^{-2}, 9.83 \times 10^{-2}]$ mrad	1 mrad	3 mrad
Sensor 2 azimuth multiplicative	6.82×10^{-5}	6.82×10^{-5}	8.00×10^{-5}	$[5.33 \times 10^{-5}, 7.97 \times 10^{-5}]$	1 mrad	3×10^{-3}
Sensor 2 elevation additive	9.66×10^{-2} mrad	9.66×10^{-2} mrad	1.00×10^{-1} mrad	$[8.18 \times 10^{-2}, 1.10 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 2 elevation multiplicative	9.58×10^{-5}	9.56×10^{-5}	9.65×10^{-5}	$[8.15 \times 10^{-5}\ 1.07 \times 10^{-4}]$	1 mrad	3×10^{-3}

TABLE II
$n_{\mathrm{MC}} = 100$ Runs, $K = 10$ Time Steps, and $N_T = 27$ Targets

Component	CRLB square root	HCRLB square root	RMSE	95% Probability interval around CRLB	Noise standard deviation	Uncorrected bias
Sensor 1 range additive	7.7 m	7.7 m	8.3 m	[6.5, 8.7] m	10 m	20 m
Sensor 1 range multiplicative	8.84×10^{-5}	8.84×10^{-5}	9.15×10^{-5}	$[7.59 \times 10^{-5}, 9.84 \times 10^{-5}]$	10 m	10^{-4}
Sensor 1 azimuth additive	1.12×10^{-1} mrad	1.12×10^{-1} mrad	1.05×10^{-1} mrad	$[9.65 \times 10^{-2}, 1.23 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 1 azimuth multiplicative	1.81×10^{-4}	1.81×10^{-4}	1.73×10^{-4}	$[1.56 \times 10^{-4}, 2.02 \times 10^{-4}]$	1 mrad	3×10^{-3}
Sensor 1 elevation additive	2.28×10^{-1} mrad	2.28×10^{-1} mrad	2.19×10^{-1} mrad	$[1.93 \times 10^{-1}, 2.59 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 1 elevation multiplicative	1.82×10^{-4}	1.81×10^{-4}	1.72×10^{-4}	$[1.49 \times 10^{-4}, 2.08 \times 10^{-4}]$	1 mrad	3×10^{-3}
Sensor 2 range additive	11.1 m	11.1 m	11.4 m	[9.5, 12.4] m	10 m	20 m
Sensor 2 range multiplicative	9.64×10^{-5}	9.63×10^{-5}	1.01×10^{-4}	$[8.21 \times 10^{-5}, 1.08 \times 10^{-4}]$	10 m	10^{-4}
Sensor 2 azimuth additive	2.00×10^{-1} mrad	2.00×10^{-1} mrad	2.03×10^{-1} mrad	$[1.70 \times 10^{-1}, 2.25 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 2 azimuth multiplicative	1.53×10^{-4}	1.53×10^{-4}	1.47×10^{-4}	$[1.32 \times 10^{-4}, 1.71 \times 10^{-4}]$	1 mrad	3×10^{-3}
Sensor 2 elevation additive	1.76×10^{-1} mrad	1.76×10^{-1} mrad	1.77×10^{-1} mrad	$[1.52 \times 10^{-1}, 1.98 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 2 elevation multiplicative	1.82×10^{-4}	1.82×10^{-4}	1.79×10^{-4}	$[1.56 \times 10^{-4}, 2.03 \times 10^{-4}]$	1 mrad	3×10^{-3}

TABLE III
$n_{\mathrm{MC}} = 100$ Runs, $K = 1$ Time Steps, and $N_T = 125$ Targets

Component	CRLB square root	HCRLB square root	RMSE	95% Probability interval around CRLB	Noise standard deviation	Uncorrected bias
Sensor 1 range additive	16.4 m	16.4 m	17.5 m	[13.3, 18.8] m	10 m	20 m
Sensor 1 range multiplicative	1.36×10^{-4}	1.36×10^{-4}	1.24×10^{-4}	$[1.16 \times 10^{-4}, 1.52 \times 10^{-4}]$	10 m	10^{-4}
Sensor 1 azimuth additive	1.53×10^{-1} mrad	1.53×10^{-1} mrad	1.59×10^{-1} mrad	$[1.25 \times 10^{-1}, 1.72 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 1 azimuth multiplicative	2.53×10^{-4}	2.53×10^{-4}	2.32×10^{-4}	$[2.21 \times 10^{-4}, 2.80 \times 10^{-4}]$	1 mrad	3×10^{-3}
Sensor 1 elevation additive	3.66×10^{-1} mrad	3.65×10^{-1} mrad	3.61×10^{-1} mrad	$[3.13 \times 10^{-1}, 4.12 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 1 elevation multiplicative	3.01×10^{-4}	3.01×10^{-4}	2.83×10^{-4}	$[2.65 \times 10^{-4}, 3.31 \times 10^{-4}]$	1 mrad	3×10^{-3}
Sensor 2 range additive	18.7 m	18.7 m	17.8 m	[16.1, 20.8] m	10 m	20 m
Sensor 2 range multiplicative	1.43×10^{-4}	1.43×10^{-4}	1.27×10^{-4}	$[1.24 \times 10^{-4}, 1.59 \times 10^{-4}]$	10 m	10^{-4}
Sensor 2 azimuth additive	2.68×10^{-1} mrad	2.68×10^{-1} mrad	2.83×10^{-1} mrad	$[2.19 \times 10^{-1}, 3.08 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 2 azimuth multiplicative	2.19×10^{-4}	2.19×10^{-4}	2.28×10^{-4}	$[1.81 \times 10^{-4}, 2.50 \times 10^{-4}]$	1 mrad	3×10^{-3}
Sensor 2 elevation additive	3.09×10^{-1} mrad	3.09×10^{-1} mrad	3.02×10^{-1} mrad	$[2.68 \times 10^{-1}, 3.47 \times 10^{-1}]$ mrad	1 mrad	3 mrad
Sensor 2 elevation multiplicative	3.05×10^{-4}	3.05×10^{-4}	3.05×10^{-4}	$[2.59 \times 10^{-4}, 3.43 \times 10^{-4}]$	1 mrad	3×10^{-3}

all cases except range multiplicative bias at the larger ranges.

This initial simulation contains many targets; therefore, another simulation is made with 27 targets instead. The results are displayed in same manner as before in Table I. The results show that the performance is not reduced much more than the 125-target case. The estimator is still efficient and has error in the angle biases that is lower than the noise standard deviation and the full bias. The range biases are significantly worse, and

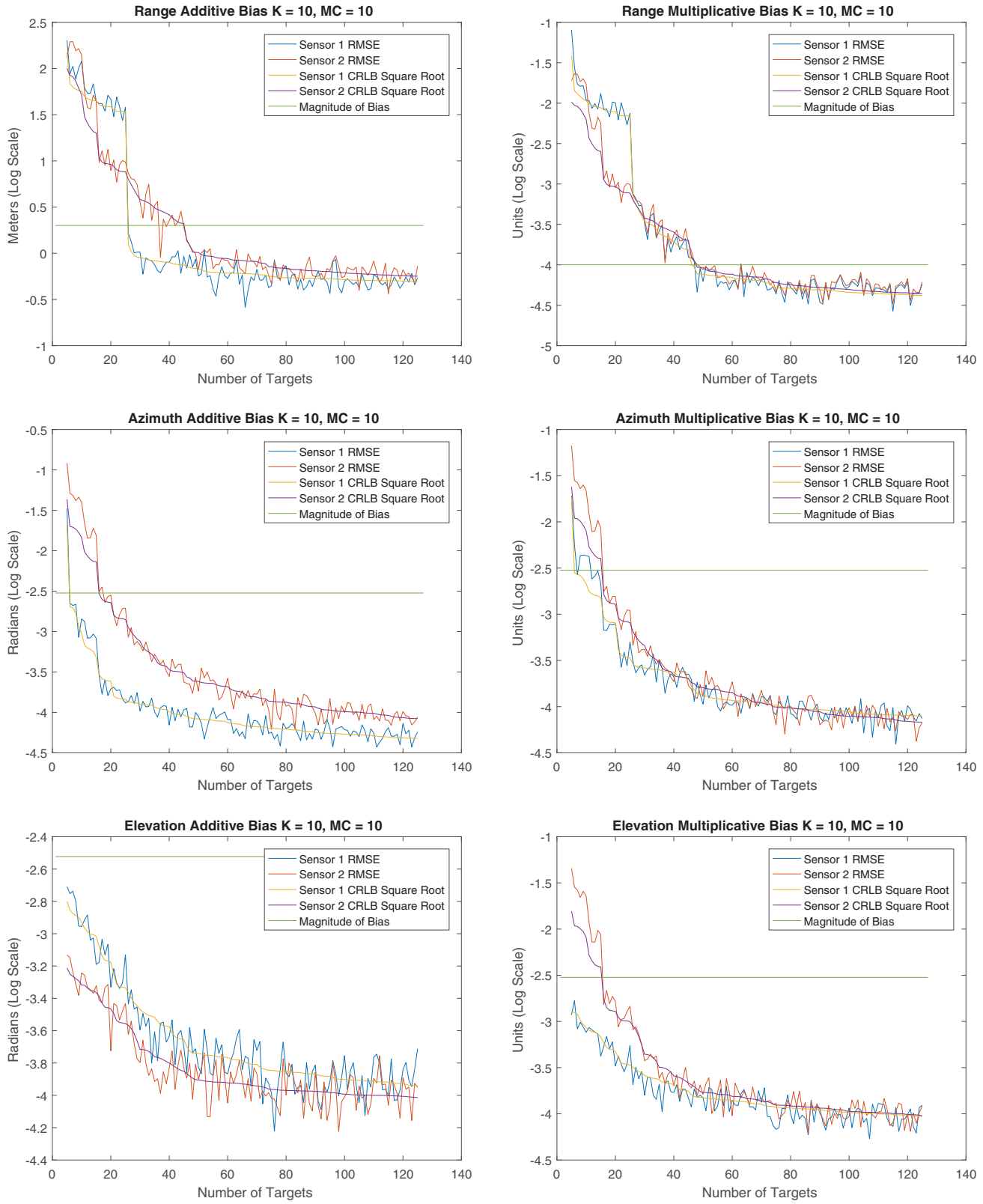


Figure 3. Comparing RMSE and CRLB with number of targets, $K = 10$ time steps.

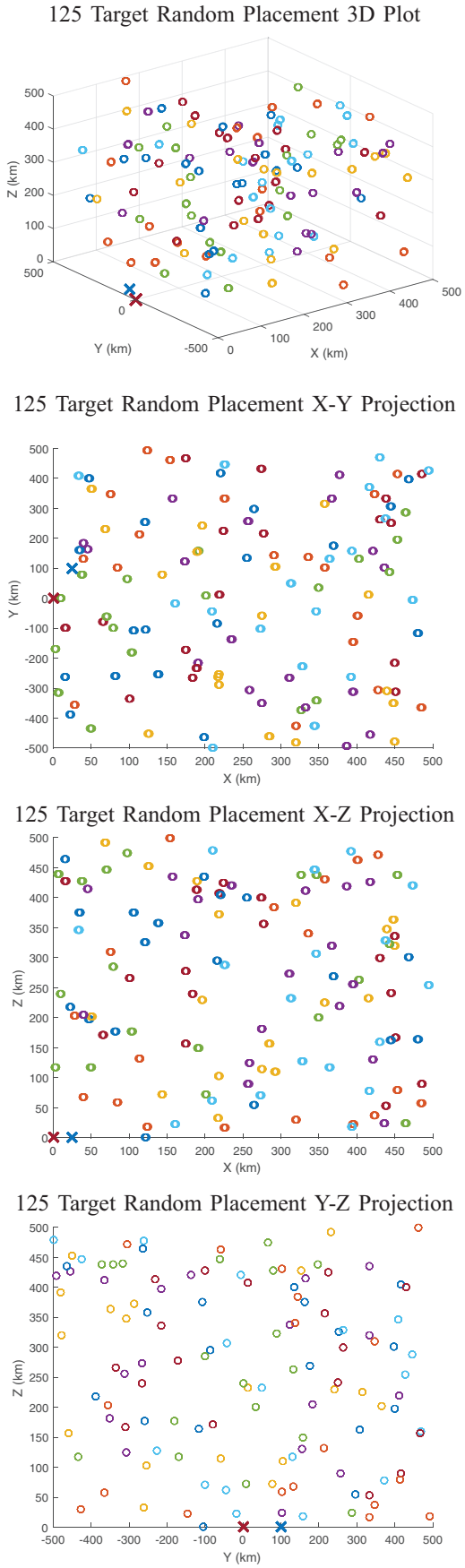


Figure 4. 125 random target layout with projections. The × symbols represent sensors and the ○ symbols represent targets. (a) 3D plot. (b) X–Y projection. (c) X–Z projection. (d) Y–Z projection.

the error in the range additive bias is almost equal to the noise standard deviation. The range multiplicative bias has RMSE that is nearly equal to the full bias, meaning that the estimation of this bias is comparable to not estimating it at all. Finally, a simulation is performed in which only a single measurement is available from each time step and with 125 targets. These results are provided in Table II. In this simulation, we see the results are very similar to the previous simulation with 10 time steps and 27 targets. The range bias estimates have RMSE that is poor and comparable to the full bias. The angle bias RMSE values are still lower than the noise standard deviation. The estimator is efficient although the CRLB itself is very poor for the range biases.

The HCRLB values for these simulations are nearly identical to the CRLB values, meaning that very little accuracy has been lost by using pseudo-measurements instead of the original measurements. This shows that when little information is known about the nuisance parameters, then the pseudo-measurement method is effective for avoiding the need to estimate the target state.

Efficient estimates are possible with this estimator and it is possible to reduce the CRLB to reasonable levels of variance by using measurements from many targets. In the case of only a single time step, the error is larger than the magnitude of the bias, meaning that it is necessary to include more measurements in order to achieve reasonable results. Furthermore, it is possible through bias estimation to reduce the error from the biases to levels that are less than the standard deviation of the noise, as shown in Tables I–III.

B. Comparing Performance Versus Number of Targets

Additionally, it is important to evaluate the bias estimation performance versus the number of targets. To simulate this, the number of targets is varied from 5 to 125 targets, starting with low-range targets and slowly expanding outward according to the target cone shown in Fig. 1. This means that range measurements have poor diversity and the estimates for the range biases are less accurate for small numbers of targets. The results of this simulation are given in Fig. 3. The results show that at around 45 targets the CRLB and RMSE are near the lowest point, and that further addition of targets continues to improve the results at a slow rate. Furthermore, we see that in the case of angle biases once there are 30 targets the bias RMSE values are about one-tenth of the full bias value. It is likely though that in a different target layout the results may differ, as this layout includes different combinations of range, azimuth, and elevation to ensure that the multiplicative biases can be estimated and not confused with the additive biases. To observe this difference, another simulation is made with random target placement.

In each Monte Carlo run, the targets are placed uniformly in a cube around the cone previously used. An example of this placement is given in Fig. 4. The same

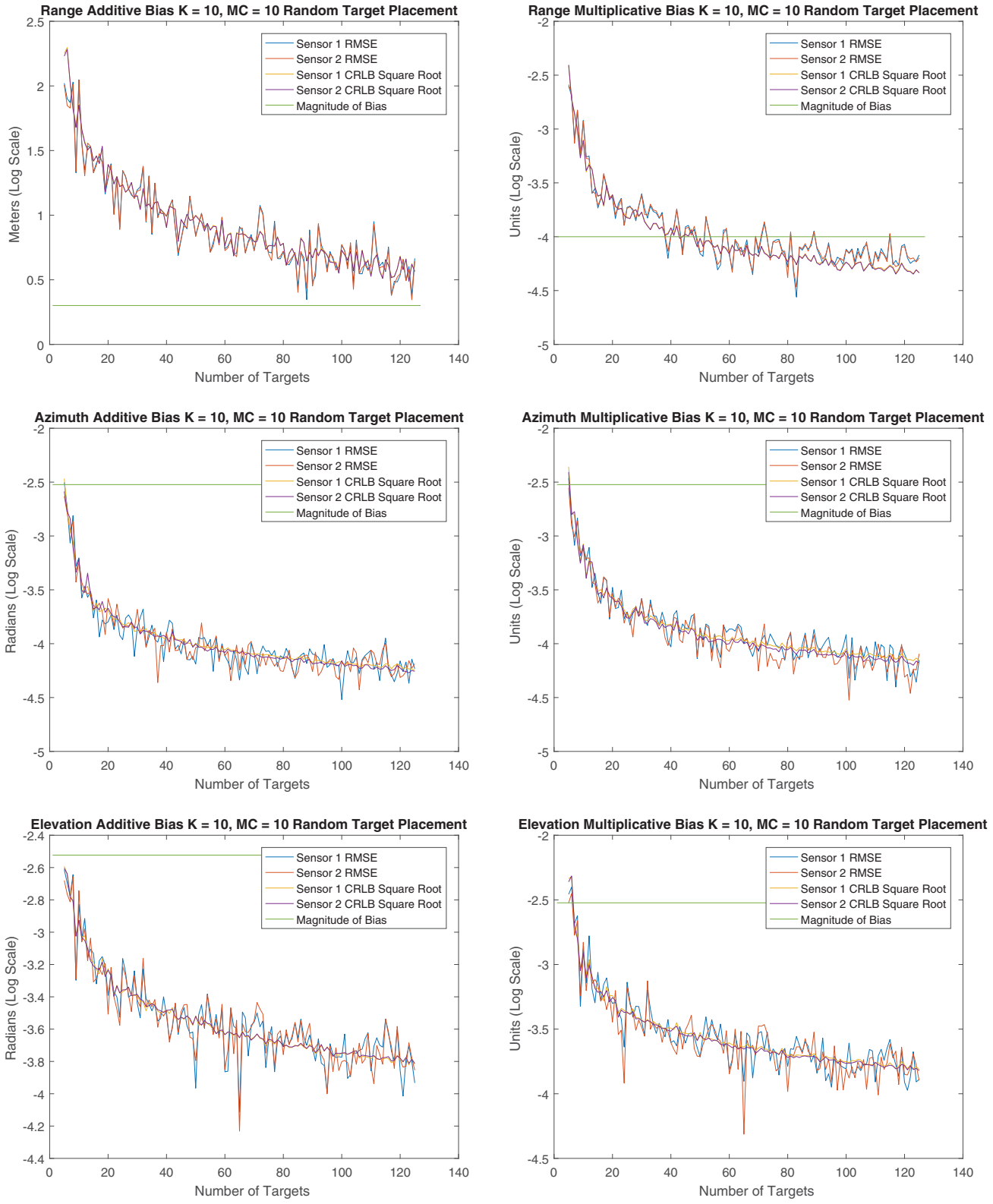


Figure 5. Comparing RMSE and CRLB with number of targets, $K = 10$ time steps.

simulation is made as earlier and the results are shown in Fig. 5. The results are nearly the same as before, except for the range additive bias. This is a result of the poor range diversity, especially of low-range targets. Overall, these results show that it is important to have measurement diversity to reduce the CRLB to reasonable levels and it is useful to have a large number of targets for this reason.

V. Conclusion

In this paper, an ML method is used to accurately estimate both multiplicative and additive biases in a

two-sensor scenario. Measurement data are converted into pseudo-measurements to isolate the effects of the biases in order to estimate them while ignoring estimation of the target state. The results show that despite the 12 separate estimated biases it is possible to match the RMSE and CRLB of the bias estimates by using a sufficient number of targets positioned in a manner to differentiate the biases. This proves the method is statistically efficient, although CRLB values are subject to the sensor and target geometry. In good conditions, the estimator can reduce the error from RMSE in bias estimates to a fraction of the noise standard deviation.

REFERENCES

[1] T. Asparouhov and B. Muthén
"Weighted least squares estimation with missing data,"
Mplus Technical Appendix, Jan. 2010.

[2] A. Balleri, A. Nehorai, and J. Wang
"Maximum likelihood estimation for compound-Gaussian clutter with inverse gamma texture,"
IEEE Trans. Aerosp. Electron. Syst., vol. 43, no. 2, PP. 775–779, Apr. 2007.

[3] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan
Estimation With Applications to Tracking and Navigation: Theory, Algorithms and Software, New York: Wiley–Interscience, 2001.

[4] Y. Bar-Shalom, X. Tian, and P. Willett
Tracking and Data Fusion: A Handbook of Algorithms.
Storrs, CT: YBS Publishing, 2011.

[5] S. Fortunati, A. Farina, F. Gini, A. Graziano, M. S. Greco, and S. Giompapa
"Least squares estimation and Cramér–Rao type lower bounds for relative sensor registration process,"
IEEE Trans. Signal Process., vol. 59, no. 3, pp. 1075–1087, Mar. 2011.

[6] S. Fortunati, F. Gini, A. Farina, A. Graziano, M. S. Greco, and S. Giompapa
"On the application of the expectation–maximisation algorithm to the relative sensor registration problem,"
IET Radar, Sonar Navigation, vol. 7, no. 2, pp. 191–203, Feb. 2013.

[7] S. Fortunati, F. Gini, M. Greco, A. Farina, A. Graziano, and S. Giompapa
"An identifiability criterion in the presence of random nuisance parameters,"
in *Proc. 20th Eur. Signal Process. Conf.*, Aug. 2012, pp. 1194–1198.

[8] S. Fortunati, F. Gini, M. S. Greco, A. Farina, A. Graziano, and S. Giompapa
"An EM-based approach to the relative sensor registration in multi-target scenarios,"
in *Proc. IEEE Radar Conf.*, May 2012, pp. 602–607.

[9] S. Fortunati, F. Gini, M. S. Greco, A. Farina, A. Graziano, and S. Giompapa
"Least squares estimation and hybrid Cramér–Rao lower bound for absolute sensor registration,"
in *Proc. Tyrrhenian Workshop Adv. Radar Remote Sens.*, Sep. 2012, pp. 30–35.

[10] B. Friedland
"Treatment of bias in recursive filtering,"
IEEE Trans. Autom. Control, vol. 14, no. 4, pp. 359–367, Aug. 1969.

[11] D. Huang, H. Leung, and E. Bosse
"A pseudo-measurement approach to simultaneous registration and track fusion,"
IEEE Trans. Aerosp. Electron. Syst., vol. 48, no. 3, pp. 2315–2331, Jul. 2012.

[12] K. Kastella, B. Yeary, T. Zadra, R. Brouillard, and E. Frangione
"Bias modeling and estimation for GMTI applications,"
in *Proc. Int. Conf. Inf. Fusion*, Aug. 2000.

[13] M. Kowalski, D. Belfadel, Y. Bar-Shalom, and P. Willett
"CRLB for estimation of 3D sensor biases in spherical coordinates,"
Proc. SPIE Defense Secur.: Signal Process., Sensor/Inf. Fusion, and Target Recognit. XXVII, vol. 10646, no. 63, Apr. 2018.

[14] M. Levedahl
"Explicit pattern matching assignment algorithm,"
Proc. SPIE, vol. 4728, pp. 461–469, 2002. [Online]. Available: https://doi.org/10.1117/12.478526.

[15] X. Lin, Y. Bar-Shalom, and T. Kirubarajan
"Exact multisensor dynamic bias estimation with local tracks,"
IEEE Trans. Aerosp. Electron. Syst., vol. 40, no. 2, pp. 576–590, Apr. 2004.

[16] X. Lin, Y. Bar-Shalom, and T. Kirubarajan
"Multisensor–multitarget bias estimation of asynchronous sensors,"
Proc. SPIE, vol. 5429, pp. 105–116, 2004. [Online]. Available: https://doi.org/10.1117/12.542272.

[17] X. Lin, Y. Bar-Shalom, and T. Kirubarajan
"Multisensor multitarget bias estimation for general asynchronous sensors,"
IEEE Trans. Aerosp. Electron. Syst., vol. 41, no. 3, pp. 899–921, Jul. 2005.

[18] M. Longbin, S. Xiaoquan, Z. Yiyu, S. Z. Kang, and Y. Bar-Shalom
"Unbiased converted measurements for tracking,"
IEEE Trans. Aerosp. Electron. Syst., vol. 34, no. 3, pp. 1023–1027, Jul. 1998.

[19] N. N. Okello and S. Challa
"Joint sensor registration and track-to-track fusion for distributed trackers,"
IEEE Trans. Aerosp. Electron. Syst., vol. 40, no. 3, pp. 808–823, Jul. 2004.

[20] N. Okello and B. Ristic
"Maximum likelihood registration for multiple dissimilar sensors,"
IEEE Trans. Aerosp. Electron. Syst., vol. 39, no. 3, pp. 1074–1083, Aug. 2003.

[21] D. J. Papageorgiou and J. D. Sergi
"Simultaneous track-to-track association and bias removal using multistart local search,"
in *Proc. IEEE Aerosp. Conf.*, Mar. 2008.

[22] H. Rue and L. Held
Gaussian Markov Random Fields: Theory and Applications.
London, UK: Chapman & Hall/CRC, 2005.

[23] P. J. Shea, T. Zadra, D. Klamer, E. Frangione, R. Brouillard, and K. Kastella
"Precision tracking of ground targets,"
in *Proc. IEEE Aerosp. Conf. (Cat. No. 00TH8484)*, vol. 3, 2000, pp. 473–482.

[24] B. A. van Doorn and H. Blom
"Systematic error estimation in multisensor fusion systems,"
Proc. SPIE, vol. 1954, pp. 450–461, Oct. 1993.

Michael Kowalski received the B.S. degree in electrical engineering from the University of Connecticut, Storrs, CT, USA, in 2015. He is currently working toward the Ph.D. degree in electrical engineering at the Department of Electrical and Computer Engineering, University of Connecticut. His research interests include bias estimation as well as sensor fusion and target tracking.

Djedjiga Belfadel received the B.S. degree from the University of Mouloud Mammeri, Tizi Ouzou, Algeria, in 2003, the M.S. degree from the University of New Haven, West Haven, CT, USA, in 2008, and the Ph.D. degree from University of Connecticut, Storrs, CT, USA, in 2015, all in electrical engineering. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Fairfield University, Fairfield, CT, USA. From 2009 to 2011, she was an Electrical Engineer with Evax Systems, Inc., Branford, CT, USA. Her research interests include target tracking, data association, sensor fusion, machine vision, and other aspects of estimation.

Yaakov Bar-Shalom (F'84) received the B.S. and M.S. degrees from the Technion, Haifa, Israel, in 1963 and 1967, respectively, and the Ph.D. degree from Princeton University, Princeton, NJ, USA, in 1970, all in electrical engineering. He is currently a Board of Trustees Distinguished Professor with the Department of Electrical and Computer Engineering and Marianne E. Klewin Professor with the University of Connecticut, Storrs, CT, USA. His current research interests include estimation theory, target tracking, and data fusion. He has authored or coauthored 600 papers and book chapters, and coauthored/edited 8 books, including *Tracking and Data Fusion* (YBS Publishing, 2011). He was elected Fellow of IEEE for contributions to the theory of stochastic systems and of multitarget tracking. He was an Associate Editor of the *IEEE Transactions on Automatic Control* and *Automatica*. He was General Chairman of the 1985 ACC, General Chairman of FUSION 2000, President of ISIF in 2000 and 2002, and Vice President for Publications during 2004–2013. Since 1995, he is a Distinguished Lecturer of the IEEE Aerospace and Electronic Systems Society. He was the corecipient of the M. Barry Carlton Award for the best paper in the *IEEE Transactions on Aerospace and Electronic Systems* in 1995 and 2000, and the recipient of the J. Mignona Data Fusion Award from the DoD JDL Data Fusion Group in 2002. He is a member of the Connecticut Academy of Science and Engineering. In 2008, he was the recipient of the IEEE Dennis J. Picard Medal for Radar Technologies and Applications, and in 2012 the Connecticut Medal of Technology. He was listed by academic.research.microsoft (top authors in engineering) as #1 among the researchers in aerospace engineering based on the citations of his work. He was the recipient of the 2015 ISIF Award for a Lifetime of Excellence in Information Fusion. This award was renamed in 2016 as the Yaakov Bar-Shalom Award for a Lifetime of Excellence in Information Fusion. He has the following Wikipedia page: https://en.wikipedia.org/wiki/Yaakov Bar-Shalom

Peter Willett received the B.A.S. degree in engineering science from the University of Toronto, Toronto, Ontario, Canada, in 1982, and the M.E., M.Sc., and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, USA, in 1983, 1984, and 1986, respectively. He is a faculty member with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT, USA, since 1986. Since 1998, he has been a Professor, and since 2003 an IEEE Fellow. His primary areas of research include statistical signal processing, detection, machine learning, communications, data fusion, and tracking. He is Chief Editor of *IEEE Aerospace and Electronic Systems Magazine* (2018–2020). He was Editor-in-Chief of *IEEE Signal Processing Letters* (2014–2016) and *IEEE Transactions on Aerospace and Electronic Systems* (2006–2011). He was also AESS Vice President for Publications (2012–2014). He is a member of the IEEE Fellows Committee, Ethics Committee, and Periodicals Committee, as well as the IEEE Signal Processing Society's Technical Activities and Conference Boards. He is also a member of the IEEE AESS Board of Governors and was Chair of IEEE Signal Processing Society's Sensor-Array and Multichannel Technical Committee.

Performance Improvement of Measurement Association Using a System with two 2D Sensors and one 3D Sensor

CÉSAR CONTRERAS
JOHN LANGFORD
LARRY AMMANN
JOHN ZWECK
BRIAN MARKS

A measurement-to-measurement data association problem is formulated for a target tracking system consisting of one or two 2D sensors and a 3D sensor. Operating conditions are identified under which performance is improved by using two 2D sensors and a 3D sensor instead of one 2D sensor and a 3D sensor. To facilitate this study, two algorithms are introduced to compute near-optimal solutions of the corresponding three-way assignment problem: a single-step algorithm based on two independent two-way assignment problems, and a related iterative algorithm that explicitly enforces a compatibility condition between measurements made by the 2D sensors. Simulation studies show that the position estimates obtained with the three-sensor system are much more accurate than those obtained with a two-sensor system whenever there is large uncertainty in the 3D sensor in the dimension orthogonal to the plane of the 2D sensor in the two-sensor system. Moreover, whenever there is large uncertainty in the measurements from the 3D sensor in the common dimension of the 2D sensors, the percentage of correct matches with the the iterative assignment algorithm for the three-sensor system is significantly better than that with a two-sensor system. The degree to which the methods and results can be extended to more realistic 3D radar and 2D camera models is discussed and inferences for aerospace and missile defense applications are drawn.

Manuscript received January 18, 2019; revised April 9, 2019; released for publication July 8, 2019.

C. Contreras, J. Langford, L. Ammann, and J. Zweck are with the Department of Mathematical Sciences, The University of Texas at Dallas, Richardson, TX 75080, USA.
B. Marks is with The Johns Hopkins University Applied Physics Laboratory, Laurel, MD 20723, USA.
This project was supported under the NSF Enriched Doctoral Training Program entitled "Team Training Mathematical Scientists Through Industrial Collaborations," DMS Award #1514808.

I. INTRODUCTION

Multitarget, multisensor tracking systems for aerospace and missile defense applications are typically based on a network of radars [1], [2], [6], [11]. In computer vision and robotics, tracking systems are often based on a network consisting of cameras or radars or both. These systems are used to track the movement of people and robots, for perception systems in autonomous vehicles, and for surveillance applications [15], [18]–[20]. Therefore, an important problem is to associate and fuse tracks generated by networks of heterogeneous 2D and 3D sensors. The simplest such data association problem is the measurement-to-measurement association (M2MA) problem, which is that of associating measurements from different sensors to form composite measurements that can then be used by a centralized tracking system to generate a single set of tracks using data from several sensors [3], [4]. In this paper, we adopt the approach of [4] and [6] in which target states and measurements are represented as random vectors. However, recently a Bayesian inference approach to target tracking based on message passing and the sum–product algorithm has also been shown to be particularly effective [12].

We classify data association problems, such as the M2MA problem, by the number of sensors in the system and the dimensions of the Euclidean spaces of the data recorded by the sensors. For example, we will classify a system with one 3D radar and two 2D cameras as a 322-sensor system, and a system with one 3D radar and one 2D camera as a 32-sensor system.

Since target tracking with three sensors is significantly more complex than with two sensors, it is important to identify situations in which the addition of a third sensor results in more accurate estimates of the target positions. Deb *et al.* [7] studied the M2MA problem for 322-, 222-, and 32-sensor systems using target scenarios that simulated a squadron of fighter jets flying in formation. They found that ghosting and resolution problems due to specific geometric configurations of the targets were a major source of position errors, but that these were smaller for a 322-sensor than a 222-sensor system.

The main contribution of this paper is to identify sensor operating conditions under which the performance of a target tracking system can be improved by using a 322-sensor system rather than a 32-sensor system. Specifically, in the context of M2MA, we study how the percentage of correct assignments and the average target position error depend on the angle between the planes of the two 2D sensors and on the orientation of the covariance ellipsoid of the 3D sensor with respect to the camera planes. By contrast, the results of Deb *et al.* [7] were obtained with fixed sensor orientations. In Section VI, we show that the percentage of correct assignments with a 322-sensor system is significantly better than that with a 32-sensor system whenever the

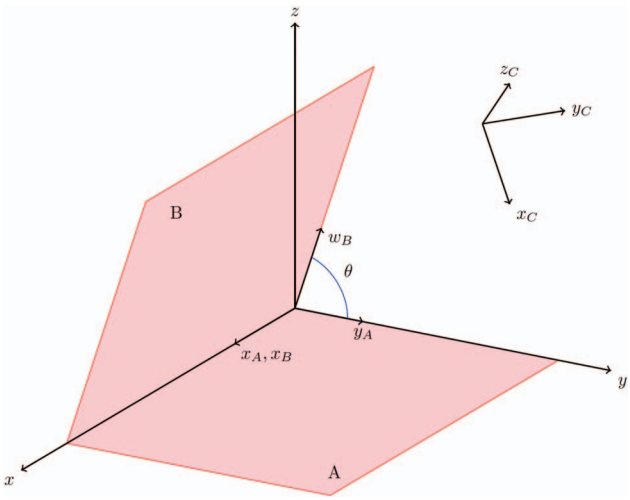


Fig. 1. The geometric configuration and coordinate systems for the three-sensor system.

uncertainty in the measurements made by the 3D sensor is large compared to that of the two 2D sensors along the line of intersection of the planes of the two 2D sensors (i.e., along the x-axis in Fig. 1). We also show that the root-mean-square error (RMSE) in the estimated target positions is much smaller for the 322-sensor system when the uncertainty in the 3D sensor measurements is large in the dimension orthogonal to the plane of the 2D sensor. In Section VII, we discuss the implications of these results for aerospace and missile defense applications. In particular, our results suggest that there can be a significant performance advantage to a target tracking system with two cameras and a 3D radar as opposed to one camera and a 3D radar.

The data association problems that arise in target tracking and sensor fusion can be formulated as multidimensional assignment problems [16]. The problem of computing the most likely assignment of measured sensor data to a collection of unknown targets involves optimizing a log-likelihood cost function subject to a collection of constraint equations that specify the set of feasible assignments. The cost function defined by Deb *et al.* [7] allows for realistic sensor models and includes spurious measurements and missed detections. They formulated the resulting three-way assignment problem as an integer linear program in which the coefficients of the cost function are defined in terms of unknown target positions. Since the number of coefficients is equal to the number of ways to select three measurements, one from each sensor, the computation of the coefficients involves solving a large number of nonlinear least-squares problems for the target positions. Indeed, Deb *et al.* found that up to 80% of the computational time of the assignment algorithm was taken up with solving these least-squares problems.

Although multidimensional assignment problems are NP-hard for $N > 3$ sensors, there are efficient iterative algorithms to compute suboptimal solutions based on greedy algorithms, simulated annealing, and Lagrangian relaxation-based methods [14], [17]. For example, in their work Deb *et al.* [7] employed a nearly optimal three-way assignment algorithm grounded in linear programming theory that trades off computational time for some loss of association accuracy. A key step in their method is to incorporate one of the constraint equations into the cost function via Lagrange multipliers and to solve the resulting two-way assignment problem using Bertsekas' auction algorithm [5] or Munkres' algorithm [13]. One advantage of this approach is that it provides upper and lower bounds for the cost of the optimal solution. In a simulation study, they found that the average gap between these bounds can be reduced to less than 2%. Such Lagrangian relaxation methods have been extended to N-way assignments [8], [17].

Since we did not have access to an implementation of a state-of-the-art assignment algorithm for the M2MA problem [8], [17], to compare the performance of 322- and 32-sensor systems we considered an idealized situation with simplified target-to-sensor transformations and with no spurious measurements or missed detections. With these simplifying assumptions, we were able to define a cost function whose coefficients do not depend on unknown target positions, thus avoiding the high computational cost of solving the nonlinear least-squares problems in [7], [8], and [17].

Furthermore, to obtain near-optimal three-way assignments we used the simplified geometric configuration of the sensors to devise two algorithms, each of which reduces the 322-sensor assignment problem to a sequence of 32-sensor problems that we solve using Munkres' algorithm. We will refer to these algorithms, which are described in Sections III and IV, as the single-step four-dimensional (4D) algorithm and the iterative five-dimensional (5D) algorithm. Although we do not prove that the iterative 5D algorithm converges to the optimal solution, we performed simulation studies (not described here) verifying that the method yields optimal assignments for small problems ($\approx$10 targets) by comparing to results obtained by an exhaustive search.

In Section VII, we will show that the simplified target-to-sensor transformations we used are valid approximations of realistic radar and camera models for applications in which the targets are located at a significant distance from the sensors and are confined to a sufficiently small region of the field of view. We will also argue that the main conclusions obtained in Section VI on the performance advantages of a 322-sensor system over a 32-sensor system should remain valid if a state-of-the-art method was used that incorporates spurious measurements and missed detections.

In Section II, we introduce the model we used for the 322-sensor association problem, and in Sections III and

IV we derive formulas for the cost function in the case of orthogonal and nonorthogonal 2D sensor planes, respectively. In Section V, we derive a formula for the estimated target positions. In Section VI, we describe the numerical simulations we performed to quantify the performance of the methods we developed. In Section VII, we discuss the extent to which our methods and results can be generalized to realistic 322-sensor systems. Finally, in Section VIII, we present our conclusions.

II. SYSTEM MODEL

In this section, we describe the system models we used to compare the performance of 322- and 32-sensor systems.

We consider a target tracking scenario with three sensors: two 2D sensors (A and B) and one 3D sensor (C). We assume that all three sensors make N measurements of the same N targets, and we assume that there is no systematic bias in any of the sensors. However, mismatches can occur because of the inherent uncertainty in the measurements made by each sensor. We modeled the 2D sensors using parallel projections rather than the more realistic perspective projections often used to model cameras. In addition, we assume that the 3D sensor uses the identity transformation to map each target to a measurement in a 3D rectangular coordinate system, rather than employing a more realistic radar model based on the transformation from XYZ to RUV space, as is described, for example, in [9]. In Section VII, we will explain why these simplified models are reasonable.

In Fig. 1, we show a schematic diagram of the geometric configuration of the three sensors. The planes of the two 2D sensors intersect in a line, L, and we let θ denote the angle between these two planes. We define a coordinate system, (x, y, z), adapted to Sensors A and B as follows. We choose the origin of the coordinate system to be a point on L, the x-axis to be along the line L, the y-axis to be perpendicular to the x-axis and in the plane of Sensor A, and the z-axis to be orthogonal to the xy-plane.

We generate the N targets in the (x, y, z) coordinate system, and suppose that each sensor records the measured positions of each target in a coordinate system adapted to that sensor. Without any loss of generality, we choose the coordinate systems (x_A, y_A) adapted to Sensor A, (x_B, w_B) adapted to Sensor B, and (x_C, y_C, z_C) adapted to Sensor C such that

$$\begin{aligned}\begin{bmatrix} x_A & y_A \end{bmatrix} &= \begin{bmatrix} x & y \end{bmatrix} \\ &= \begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} x & y & z \end{bmatrix} M_A \end{aligned} \tag{1}$$

$$\begin{aligned}\begin{bmatrix} x_B & w_B \end{bmatrix} &= \begin{bmatrix} x & \cos(\theta)y + \sin(\theta)z \end{bmatrix} \\ &= \begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \cos\theta \\ 0 & \sin\theta \end{bmatrix} \\ &= \begin{bmatrix} x & y & z \end{bmatrix} M_B, \end{aligned} \tag{2}$$

and

$$\begin{bmatrix} x_C & y_C & z_C \end{bmatrix} = \begin{bmatrix} x & y & z \end{bmatrix} R, \tag{3}$$

for some 3×3 orientation-preserving orthogonal matrix R. The projection matrices, M_A and M_B, in (1) and (2) map the 3D vector $\begin{bmatrix} x & y & z \end{bmatrix}$ to the planes of Sensors A and B.

We assume that the errors in the positions measured by Sensors A, B, and C are independent Gaussian random vectors with covariance matrices Σ_A, Σ_B, and $\widetilde{\Sigma}_C$, where Σ_A and Σ_B are 2×2 matrices and $\widetilde{\Sigma}_C$ is a 3×3 matrix. We assume that these variances are known from sensor calibration measurements, and are given in the coordinate system adapted to each sensor. For simplicity, henceforth we suppose that the measurements made by Sensor C have been transformed from (x_C, y_C, z_C) to (x, y, z) coordinates via (3) and that the covariance matrix $\widetilde{\Sigma}_C$ has been transformed to $\Sigma_C = R\widetilde{\Sigma}_C R^T$.

The notational conventions we use are summarized in Table I. In addition, we use the following notation for submatrices of an $N \times M$ matrix A. Let $\mathbf{i} = (i_1, \ldots, i_K)$ and $\mathbf{j} = (j_1, \ldots, j_L)$ be vectors of row and column indices of A, respectively. Then, let $A[\mathbf{i}, \mathbf{j}]$ denote the $K \times L$ submatrix of A consisting of the rows and columns indexed by $\mathbf{i}$ and $\mathbf{j}$, respectively. Also, let $A[*, \mathbf{j}] = A[(1, \ldots, N), \mathbf{j}]$ and $A[\mathbf{i}, *] = A[\mathbf{i}, (1, \ldots, M)]$.

Let $\begin{bmatrix} X & Y & Z \end{bmatrix}$ denote the $N \times 3$ matrix of the true positions of the N targets in the coordinate system (x, y, z) adapted to Sensors A and B. Let A_{err} and B_{err} denote the $N \times 2$ matrices of position errors for Sensors A and B, respectively, and let C_{err} denote the $N \times 3$ matrix of position errors for Sensor C. The rows of A_{err} are independent and identically distributed (i.i.d.) multivariate normal distributions with mean zero and covariance matrix Σ_A, $A_{\text{err}}[n, *] \sim MN(0, \Sigma_A)$, and similarly for Sensors B and C. Then, the positions of the N targets as measured by Sensors A, B, and C are given by

$$\begin{bmatrix} \widehat{X}_A & \widehat{Y}_A \end{bmatrix} = \begin{bmatrix} X_A & Y_A \end{bmatrix} + A_{\text{err}} \in \mathbf{R}^{N \times 2} \tag{4}$$

$$\begin{bmatrix} \widehat{X}_B & \widehat{W}_B \end{bmatrix} = \begin{bmatrix} X_B & W_B \end{bmatrix} + B_{\text{err}} \in \mathbf{R}^{N \times 2} \tag{5}$$

$$\begin{bmatrix} \widehat{X}_C & \widehat{Y}_C & \widehat{Z}_C \end{bmatrix} = \begin{bmatrix} X_C & Y_C & Z_C \end{bmatrix} + C_{\text{err}} \in \mathbf{R}^{N \times 3} \tag{6}$$

respectively. The matrix $\begin{bmatrix} X_A & Y_A \end{bmatrix}$ in (4) is obtained via an unknown permutation, π_A, of the rows of the matrix $\begin{bmatrix} X & Y & Z \end{bmatrix}$ using the projection defined in (1). Similarly, the matrix $\begin{bmatrix} X_B & W_B \end{bmatrix}$ in (5) is obtained via an unknown permutation, π_B, of the rows of the matrix $\begin{bmatrix} X & Y & Z \end{bmatrix}$ using the projection defined in (2). These permutations

TABLE I
Notational Conventions Used in This Paper

Variable	Meaning
$(x_A, y_A), (x_B, w_B), (x_C, y_C, z_C)$	Coordinate systems adapted to Sensors A, B, and C
$[X, Y, Z]$	$N \times 3$ matrix of true target positions
$[X_A, Y_A], [X_B, W_B], [X_C, Y_C, Z_C]$	$N \times K$ matrices of permuted true target positions for Sensors A, B ($K = 2$), and C ($K = 3$)
$\Sigma_A, \Sigma_B, \Sigma_C$	$K \times K$ covariance matrices for Sensors A, B, and C
$A_{\text{err}}, B_{\text{err}}, C_{\text{err}}$	$N \times K$ matrices of position errors for Sensors A, B, and C
$[\widehat{X}_A, \widehat{Y}_A], [\widehat{X}_B, \widehat{W}_B], [\widehat{X}_C, \widehat{Y}_C, \widehat{Z}_C]$	$N \times K$ matrices of permuted measured target positions for Sensors A, B, and C

model the fact that the three sensors do not record the data from the N targets in the same order.

Let S_N denote the set of all permutations of N symbols. With each permutation, $\pi \in S_N$, we can associate an $N \times N$ matrix, also denoted by π, with the property that left multiplication of a matrix by π permutes the rows of that matrix. Specifically, the permutation matrix corresponding to the permutation, $\pi(1, 2, \ldots, N) = (\pi(1), \pi(2), \ldots, \pi(N))$, is given by

$$\pi = \begin{bmatrix} \mathbf{e}_{\pi(1)} \\ \mathbf{e}_{\pi(2)} \\ \vdots \\ \mathbf{e}_{\pi(n)} \end{bmatrix} \tag{7}$$

where the row vector $\mathbf{e}_j$ is the jth standard basis vector. Therefore, by (4) and (5),

$$\begin{bmatrix} \widehat{X}_A & \widehat{Y}_A \end{bmatrix} = \pi_A \begin{bmatrix} X & Y \end{bmatrix} + A_{\text{err}} \tag{8}$$

$$\begin{bmatrix} \widehat{X}_B & \widehat{W}_B \end{bmatrix} = \pi_B \begin{bmatrix} X & W \end{bmatrix} + B_{\text{err}} \tag{9}$$

where $W = \pi_B^{-1} W_B$ is $N \times 1$.

Our goal is to determine the permutations, π_A and π_B, that associate each of the N targets recorded by Sensors A and B, respectively, with those recorded by Sensor C. We formulate this association problem as a combinatorial optimization problem that involves the minimization of a cost function

$$\ell : S_N \times S_N \to [0, \infty) \tag{10}$$

where $\ell(\widehat{\pi}_A, \widehat{\pi}_B)$ represents the cost of using permutations $\widehat{\pi}_A$ and $\widehat{\pi}_B$ to associate data from Sensors A and B with those from Sensor C, respectively.

In the following sections, we will define the cost function, ℓ, in terms of a Mahalanobis distance using the requirement that, in the absence of measurement error, the orthogonal projections of the target positions measured by Sensor C onto the planes of Sensors A and B should match the target positions measured by Sensors A and B, respectively. Specifically, the permutations π_A and π_B should be chosen so that

$$\pi_A^{-1} \begin{bmatrix} X_A & Y_A \end{bmatrix} = \begin{bmatrix} X_C & Y_C & Z_C \end{bmatrix} M_A \tag{11}$$

$$\pi_B^{-1} \begin{bmatrix} X_B & W_B \end{bmatrix} = \begin{bmatrix} X_C & Y_C & Z_C \end{bmatrix} M_B. \tag{12}$$

III. COST FUNCTION FOR ORTHOGONAL SENSOR PLANES

To simplify the discussion, we first consider the special case that the planes of Sensors A and B are orthogonal ($\theta = \pi/2$). In the presence of measurement error, by (4) and (6), the difference in the positions measured by Sensors A and C is

$$\begin{aligned} &\pi_A^{-1} \begin{bmatrix} \widehat{X}_A & \widehat{Y}_A \end{bmatrix} - \begin{bmatrix} \widehat{X}_C & \widehat{Y}_C \end{bmatrix} \\ &= \left(\pi_A^{-1} \pi_A \begin{bmatrix} X & Y \end{bmatrix} + \pi_A^{-1} A_{\text{err}} \right) \\ &\quad - \left(\begin{bmatrix} X & Y \end{bmatrix} + C_{\text{err}}[*, (1, 2)] \right) \\ &= A_{\text{err}} - C_{\text{err}}[*, (1, 2)] \end{aligned} \tag{13}$$

since the rows of A_{err} are i.i.d. random variables

$$\pi_A^{-1} A_{\text{err}} \overset{(d)}{=} \pi_A A_{\text{err}} \overset{(d)}{=} A_{\text{err}}$$

where $\overset{(d)}{=}$ denotes equality in distribution. Similarly,

$$\pi_B^{-1} \begin{bmatrix} \widehat{X}_B & \widehat{W}_B \end{bmatrix} - \begin{bmatrix} \widehat{X}_C & \widehat{Z}_C \end{bmatrix} = B_{\text{err}} - C_{\text{err}}[*, (1, 3)]. \tag{14}$$

Let $D_{N \times 4}(\pi_A, \pi_B)$ be the $N \times 4$ data matrix

$$D_{N \times 4}(\pi_A, \pi_B) = \begin{bmatrix} D_1 & D_2 & D_3 & D_4 \end{bmatrix}$$

where

$$\begin{aligned} D_1 &= \pi_A^{-1} \widehat{X}_A - \widehat{X}_C \\ D_2 &= \pi_A^{-1} \widehat{Y}_A - \widehat{Y}_C \\ D_3 &= \pi_B^{-1} \widehat{X}_B - \widehat{X}_C \\ D_4 &= \pi_B^{-1} \widehat{W}_B - \widehat{Z}_C. \end{aligned}$$

By (13) and (14), the vector $\vec{x}(\pi_A, \pi_B) \in \mathbf{R}^{4N}$ obtained by concatenating the columns of the matrix $D_{N \times 4}(\pi_A, \pi_B)$ is multivariate normally distributed with mean zero and a $4N \times 4N$ covariance matrix, $\widetilde{\Sigma}_{4\text{D}}$, that is determined by Σ_A, Σ_B, and Σ_C. Our goal then is to determine the permutations $(\widehat{\pi}_A, \widehat{\pi}_B)$ that maximize the likelihood function

$$L((\pi_A, \pi_B)|\vec{x}) = \exp\left[-\vec{x}(\pi_A, \pi_B)^T \widetilde{\Sigma}_{4\text{D}}^{-1} \vec{x}(\pi_A, \pi_B) \right] \tag{15}$$

which is equivalent to minimizing the Mahalanobis distance

$$\ell(\pi_A, \pi_B) = \vec{x}(\pi_A, \pi_B)^T \widetilde{\Sigma}_{4\text{D}}^{-1} \vec{x}(\pi_A, \pi_B). \tag{16}$$

The advantage of defining $D_{N\times 4}$ using differences of measurements is that the unknown positions of the targets are removed, simplifying the computation of the cost function ℓ.

Since $S_N \times S_N$ has $(N!)^2$ elements, an exhaustive search for the global minimum of ℓ is intractable for $N \gtrapprox 10$. On the other hand, the simpler problem of associating measurements made by two sensors can be formulated as an optimization problem on S_N for which $\mathcal{O}(N^3)$-algorithms, such as Munkres' algorithm, exist [13]. To obtain an efficient, approximate solution to the 322-sensor assignment problem, we devised two algorithms that use Munkres' algorithm to solve two-way assignment subproblems. We refer to these two algorithms as the 4D and 5D algorithms.

Before describing these algorithms, we derive an alternate formula for the Mahalanobis distance, ℓ, in (16). Recall that the definition of ℓ was motivated by the requirements, (8) and (9), that the projections of the measured data from Sensor C onto the planes of Sensors A and B should be aligned with the data obtained from Sensors A and B. We observe from (8) and (9) that, in the absence of measurement error, the compatibility condition

$$\pi_A^{-1} X_A = \pi_B^{-1} X_B \tag{17}$$

must also hold for the optimal choice of permutations. This condition will be used in the 5D algorithm to further constrain the search for the optimal pair of permutations.

By (4) and (5), we observe that in the presence of measurement error the difference in the x-coordinate of the measured positions of Sensors A and B is

$$\pi_A^{-1}\widehat{X}_A - \pi_B^{-1}\widehat{X}_B = A_{\text{err}}[*,1] - B_{\text{err}}[*,1]. \tag{18}$$

Then, we define an $N \times 5$ data matrix by

$$D_{N\times 5}(\pi_A, \pi_B) = \begin{bmatrix} D_{N\times 4}(\pi_A, \pi_B) & D_5 \end{bmatrix} \tag{19}$$

where

$$D_5 = \pi_A^{-1}\widehat{X}_A - \pi_B^{-1}\widehat{X}_B.$$

Let $\widetilde{\Sigma}_{5\text{D}}$ be the $5N \times 5N$ covariance matrix associated with the data in $D_{N\times 5}$. Since $D_5 = D_1 - D_3$ for any set of measurements, the covariance matrix $\widetilde{\Sigma}_{5\text{D}}$ is not positive definite but rather has rank $4N$. A calculation shows that $\widetilde{\Sigma}_{5\text{D}}$ is the 5×5 block matrix whose (k,l)-block is the $N \times N$ matrix

$$(\widetilde{\Sigma}_{5\text{D}})_{kl} \;=\; (\Sigma_{5\text{D}})_{kl}\, I_{N\times N} \tag{20}$$

where $I_{N\times N}$ is the $N \times N$ identity matrix and $\Sigma_{5\text{D}}$ is the 5×5 matrix given by

$$\Sigma_{5\text{D}} = \begin{bmatrix} \Sigma_{AC} & C_{ACBC} & C_{ACAB} \\ C_{ACBC}^T & \Sigma_{BC} & C_{BCAB} \\ C_{ACAB}^T & C_{BCAB}^T & \Sigma_{AB} \end{bmatrix} \tag{21}$$

whose entries are the matrices

$$\Sigma_{AC} = \Sigma_A + \Sigma_C[(1,2),(1,2)]$$

$$\Sigma_{BC} = \Sigma_B + \Sigma_C[(1,3),(1,3)]$$

$$\Sigma_{AB} = \Sigma_A[1,1] + \Sigma_B[1,1]$$

$$C_{ACBC} = \begin{bmatrix} \Sigma_C[1,1] & \Sigma_C[1,3] \\ \Sigma_C[1,2] & \Sigma_C[2,3] \end{bmatrix}$$

$$C_{ACAB} = \Sigma_A[*,1]$$

$$C_{BCAB} = -\Sigma_B[*,1].$$

Since $\Sigma_{5\text{D}}$ is singular, we can use the Moore–Penrose pseudoinverse of $\Sigma_{5\text{D}}$ to define the distribution of each row of D. Let

$$\Sigma_{5\text{D}} = VEV^T$$

denote the eigendecomposition of $\Sigma_{5\text{D}}$, arranged so that the last eigenvalue in E is 0 and the first four are positive. Let V_4 denote the first four columns of V and let E_4 denote the submatrix consisting of the first four rows and columns of E. Then, the Moore–Penrose pseudoinverse of $\Sigma_{5\text{D}}$ is

$$\Sigma_{5\text{D}}^{-} = V_4 E_4^{-1} V_4^T.$$

Therefore, the log-likelihood function in (16) for parameters $\widehat{\pi}_A$ and $\widehat{\pi}_B$ can also be expressed as

$$l(\widehat{\pi}_A, \widehat{\pi}_B) = \sum_{i=1}^{n} D_i(\widehat{\pi}_A, \widehat{\pi}_B) V_4 E_4^{-1} V_4^T D_i^T(\widehat{\pi}_A, \widehat{\pi}_B) \tag{22}$$

where $D_i(\widehat{\pi}_A, \widehat{\pi}_B)$ denotes the ith row of $D_{N\times 5}(\widehat{\pi}_A, \widehat{\pi}_B)$.

The problem of simultaneously finding permutations $\widehat{\pi}_A$ and $\widehat{\pi}_B$ that minimize the association cost is not a standard association problem since $\ell : S^N \times S^N \to \mathbf{R}$ rather than $\ell : S^N \to \mathbf{R}$. Instead, we can treat the problem as two separate association problems by using columns 1 and 2 of $D_{N\times 5}$ to associate the targets measured by A with those from C and using columns 3 and 4 of $D_{N\times 5}$ to associate the measurements from B with those from C. Specifically, the first association subproblem is obtained by extracting columns 1 and 2 from $D_{N\times 5}$ to obtain the $N \times 3$ data matrix

$$D^{(AC)}(\widehat{\pi}_A) = \begin{bmatrix} \widehat{\pi}_A\widehat{X}_A - \widehat{X}_C, & \widehat{\pi}_A\widehat{Y}_A - \widehat{Y}_C \end{bmatrix} \tag{23}$$

which only depends on $\widehat{\pi}_A$. The cost function for this subproblem is defined by

$$\ell_{AC}(\widehat{\pi}_A) = \sum_{i=1}^{n} D_i^{(AC)} \Sigma_{AC}^{-1} D_i^{(AC)T}. \tag{24}$$

Similarly, the second subproblem, which is to associate the data from B with those from C, is determined by the data matrix

$$D^{(BC)}(\widehat{\pi}_B) = \begin{bmatrix} \widehat{\pi}_B^{-1}\widehat{X}_B - \widehat{X}_C, & \widehat{\pi}_B^{-1}\widehat{W}_B - \widehat{Z}_C \end{bmatrix} \tag{25}$$

and the corresponding cost function

$$\ell_{BC}(\widehat{\pi}_B) = \sum_{i=1}^{n} D_i^{(BC)} \Sigma_{BC}^{-1} D_i^{(BC)T}. \tag{26}$$

This approach is the essence of the 4D algorithm for the 322-sensor association problem, which involves solving two independent matching problems: an A-to-C and a B-to-C match, from which an A-to-B match can be inferred. The 4D algorithm can be summarized as follows:

#initialize
$\widehat{\pi}_B = \text{argmin}(l_{BC}(\widehat{\pi}_B))$
$\widehat{\pi}_A = \text{argmin}(l_{AC}(\widehat{\pi}_A))$
end

In the 5D algorithm, we explicitly impose the additional consistency condition encoded in column 5 of $D_{N\times 5}$ that represents the differences in the observed positions between A and B in their shared dimension. With this iterative algorithm, we first use a current estimate for the A-to-C match and optimize the B-to-C match. The Mahalanobis distance objective function for this B-to-C match involves a $3N \times 3N$ covariance matrix that incorporates the additional A–B consistency condition. In the second stage of each iteration, we switch the roles of A and B and optimize the A-to-C match. Given a reasonable initial guess for the A-to-C match, only a few iterations are typically required for the algorithm to converge.

In detail, the 5D algorithm is given as follows. Suppose we have available an initial permutation, $\widehat{\pi}_B^{(0)}$, for Sensor B and wish to associate the measurements made by Sensor A with those made by Sensor C given $\widehat{\pi}_B^{(0)}$. This data association problem can be based on columns 1, 2, and 5 of (21) with $\widehat{\pi}_B^{(0)}$ fixed

$$D^{(ABC)}\big(\widehat{\pi}_A|\widehat{\pi}_B^{(0)}\big) = [D_1,\ D_2,\ D_5]. \tag{27}$$

The covariance matrix for $D^{(ABC)}$ is the submatrix consisting of rows and columns 1, 2, and 5 of (21)

$$\Sigma_{ABC} = \Sigma_{5D}[(1,2,5),(1,2,5)]. \tag{28}$$

The matrix Σ_{ABC} is nonsingular and the objective function for $\widehat{\pi}_A$ given $\widehat{\pi}_B^{(0)}$ is given by

$$\ell_{ABC}\big(\widehat{\pi}_A|\widehat{\pi}_B^{(0)}\big) = \sum_{i=1}^{n} D_i^{(ABC)}\Sigma_{ABC}^{-1}\big(D_i^{(ABC)}\big)^T. \tag{29}$$

Let $\widehat{\pi}_A^{(0)}$ denote the permutation that minimizes this cost function. Since $\ell_{ABC} : S_N \to \mathbf{R}$, an approximation to $\widehat{\pi}_A^{(0)}$ can be obtained using a two-way assignment algorithm [10], [13]. Once we have an estimate for $\widehat{\pi}_A^{(0)}$, we can then refine the initial estimate of π_B using columns 3, 4, and 5 of (21)

$$D^{(BAC)}\big(\widehat{\pi}_B|\widehat{\pi}_A^{(0)}\big) = [D_3,\ D_4,\ D_5]. \tag{30}$$

The covariance matrix for $D^{(BAC)}$ is

$$\Sigma_{BAC} = \Sigma_{5D}[(3,4,5),(3,4,5)]. \tag{31}$$

The matrix Σ_{BAC} is nonsingular and, as above, the cost function for the new estimate $\widehat{\pi}_B$ is

$$\ell_{BAC}\big(\widehat{\pi}_B|\widehat{\pi}_A^{(0)}\big) = \sum_{i=1}^{n} D_i^{(BAC)}\Sigma_{BAC}^{-1}\big(D_i^{(BAC)}\big)^T. \tag{32}$$

Now let $\widehat{\pi}_B^{(1)}$ denote the permutation that minimizes this cost function. The process of solving these alternating two-way assignment problems can be repeated using the updated estimates from the previous step, continuing until the estimates remain unchanged.

It remains to define an initial estimate for π_B. This can be obtained using columns 3 and 4 of $D_{N\times 5}$

$$D^{(BC)}(\widehat{\pi}_B) = \big[\widehat{\pi}_B\widehat{X}_B - \widehat{X}_C,\ \widehat{\pi}_B\widehat{W}_B - \widehat{Z}_C\big] \tag{33}$$

to assign the data from B to those from C without A. The cost function for this problem is

$$\ell_{BC}(\widehat{\pi}_B) = \sum_{i=1}^{n} D_i^{(BC)}\Sigma_{BC}^{-1}\big(D_i^{(BC)}\big)^T. \tag{34}$$

The resulting 5D algorithm can be expressed as follows:

#initialize
$\widehat{\pi}_B^{(0)} = \text{argmin}(l_{BC}(\widehat{\pi}_B))$
$\widehat{\pi}_A^{(0)} = \text{argmin}(l_{ABC}(\widehat{\pi}_A|\widehat{\pi}_B^{(0)}))$

#iterate
for i in 1 : maxiter
{
$\widehat{\pi}_B^{(i)} = \text{argmin}(l_{BAC}(\widehat{\pi}_B|\widehat{\pi}_A^{(i-1)}))$
$\widehat{\pi}_A^{(i)} = \text{argmin}(l_{ABC}(\widehat{\pi}_A|\widehat{\pi}_B^{(i-1)}))$
if $\widehat{\pi}_A^{(i)} == \widehat{\pi}_A^{(i-1)}$ and $\widehat{\pi}_B^{(i)} == \widehat{\pi}_B^{(i-1)}$
break
}

IV. COST FUNCTION FOR NONORTHOGONAL SENSOR PLANES

We now consider the more realistic situation in which the sensor planes A and B are not orthogonal, $(\theta \neq \pi/2)$. In this case, the $N \times 5$ data matrix is given by

$$D = \begin{bmatrix} D_1 & D_2 & D_3 & D_4 & D_5 \end{bmatrix} \tag{35}$$

where

$$\begin{aligned} D_1 &= \widehat{\pi}_A^{-1}\widehat{X}_A - \widehat{X}_C \\ D_2 &= \widehat{\pi}_A^{-1}\widehat{Y}_A - \widehat{Y}_C \\ D_3 &= \widehat{\pi}_B^{-1}\widehat{X}_B - \widehat{X}_C \\ D_4 &= \widehat{\pi}_B^{-1}\widehat{W}_B - \cos(\theta)\widehat{Y}_C - \sin(\theta)\widehat{Z}_C \\ D_5 &= \widehat{\pi}_A^{-1}\widehat{X}_A - \widehat{\pi}_B^{-1}\widehat{X}_B. \end{aligned}$$

Notice that the only column that is different from the case of orthogonal sensor planes is D_4, which corresponds to the orthogonal projection of the measurements made by Sensor C onto the tilted plane of Sensor

B. The associated $N \times 5$ error matrix is given by

$$E = \begin{bmatrix} E_1 & E_2 & E_3 & E_4 & E_5 \end{bmatrix} \tag{36}$$

where

$$\begin{aligned} E_1 &= A_{\text{err}}[*, 1] - C_{\text{err}}[*, 1] \\ E_2 &= A_{\text{err}}[*, 2] - C_{\text{err}}[*, 2] \\ E_3 &= B_{\text{err}}[*, 1] - C_{\text{err}}[*, 1] \\ E_4 &= B_{\text{err}}[*, 2] - \cos(\theta) C_{\text{err}}[*, 2] - \sin(\theta) C_{\text{err}}[*, 3] \\ E_5 &= A_{\text{err}}[*, 1] - B_{\text{err}}[*, 1]. \end{aligned}$$

Since columns 1, 2, and 5 of (35) are the same as in the nonorthogonal case, the covariance matrix Σ_{ABC} is the same as before. On the other hand, Σ_{BAC} is now given by

$$\Sigma_{BAC} = \begin{bmatrix} \Sigma_{BC}(1) & \Sigma_{BC}(1,2) & -\Sigma_B[1,1] \\ \Sigma_{BC}(1,2) & \Sigma_{BC}(2) & -\Sigma_B[2,1] \\ -\Sigma_B[1,1] & -\Sigma_B[2,1] & \Sigma_{AB}(1) \end{bmatrix} \tag{37}$$

where

$$\begin{aligned} \Sigma_{BC}(1) &= \Sigma_B[1,1] + \Sigma_C[1,1] \\ \Sigma_{BC}(1,2) &= \Sigma_B[1,2] + C_\theta \Sigma_C[1,2] + S_\theta \Sigma_C[1,3] \\ \Sigma_{BC}(2) &= \Sigma_B[2,2] + C_\theta^2 \Sigma_C[2,2] + \Gamma_\theta \Sigma_C[2,3] \\ &\quad + S_\theta^2 \Sigma_C[3,3] \end{aligned}$$

with $C_\theta = \cos\theta, S_\theta = \sin\theta$, and $\Gamma_\theta = 2 S_\theta C_\theta$.

V. ESTIMATION OF TARGET POSITIONS

Once the target assignments have been made, the target positions can be estimated using a weighted combination of the positions measured by each sensor in each coordinate, where the weights are inversely proportional to the standard deviations of the sensor position errors. For simplicity, in this section we assume that the sensor planes are orthogonal and the covariance matrices Σ_A, Σ_B, and Σ_C are diagonal. In principle, it is possible to relax these simplifying assumptions. Let

$$\widehat{P} = (\widehat{X}_A, \widehat{Y}_A, \widehat{X}_B, \widehat{W}_B, \widehat{X}_C, \widehat{Y}_C, \widehat{Z}_C)^T \tag{38}$$

denote the positions of an arbitrary target as measured by each sensor after assignments have been made for A and B and let

$$\sigma_{AX}, \sigma_{AY}, \sigma_{BX}, \sigma_{BW}, \sigma_{CX}, \sigma_{CY}, \sigma_{CZ} \tag{39}$$

denote the standard deviations of the position errors in the respective coordinates. Then, the estimated target positions are given by

$$\begin{aligned} \widehat{X} &= w_{XA}\widehat{X}_A + w_{XB}\widehat{X}_B + w_{XC}\widehat{X}_C \\ \widehat{Y} &= w_{YA}\widehat{Y}_A + w_{YC}\widehat{Y}_C \\ \widehat{Z} &= w_{ZB}\widehat{W}_B + w_{ZC}\widehat{Z}_C \end{aligned} \tag{40}$$

where $w_{XA} = \sigma_{AX}^{-1}/\delta_X$, the other weights are defined similarly, and

$$\begin{aligned} \delta_X &= \left(\sigma_{AX}^{-1} + \sigma_{BX}^{-1} + \sigma_{CX}^{-1}\right) \\ \delta_Y &= \left(\sigma_{AY}^{-1} + \sigma_{CY}^{-1}\right) \\ \delta_Z &= \left(\sigma_{BW}^{-1} + \sigma_{CZ}^{-1}\right). \end{aligned}$$

Note that a sensor with higher relative variability will have lower weight associated with its observed positions.

VI. RESULTS

In this section, we present the results of numerical simulations designed to compare the performance of a 32-sensor system, in which target matching is performed between pairs of measurements using Munkres' algorithm, to that of a 322-sensor system, in which target matching is performed between triples of measurements using the 5D algorithm. We also compare the performance of the 4D and 5D algorithms for a 322-sensor system. First, we study the percentage of correct assignments with the different methods for six choices of the covariance matrix of Sensor C in the special case that the planes of the two 2D sensors are orthogonal. Second, we investigate how the percentage of correct assignments for the 322-sensor system with the 5D algorithm depends on the angle, θ, between the planes of the two 2D sensors, and also on the orientation of the 3D sensor relative to that of the two 2D sensors. Third, we compare the RMSE in the estimated target positions obtained using a 32-sensor system, a 322-sensor system with the 4D algorithm, and a 322-sensor system with the 5D algorithm. Finally, we present results that illustrate how the execution time of the 5D algorithm depends on both the number of targets and the average separation between the targets.

A. Target and Sensor Simulation Scenarios

To evaluate how the performance of the 32- and 322-sensor system M2MA algorithms depends on the average separation between the targets, we generated data by simulating a collection of N targets with randomly distributed constant velocities that originate from the same location, and move apart like a cluster of exploding fireworks. The velocity, $\mathbf{v}_n$, of the nth target is given by

$$\mathbf{v}_n = \mathbf{v}_{\text{g}} + v_n \mathbf{u}_n \tag{41}$$

where $\mathbf{v}_{\text{g}}$ is a common group velocity and $v_n \mathbf{u}_n$ is the drift velocity of the nth target relative to the group. Here, v_n is the drift speed and $\mathbf{u}_n$ is a unit drift-direction vector. The drift speeds, v_n, are chosen from a gamma distribution with mean, μ_{d}, and standard deviation, σ_{d}, and the drift directions are sampled from a uniform distribution on the unit sphere. We ran the simulation of the target trajectories for a total time T with time steps of size Δt. We generated data in two target simulation scenarios, I

TABLE II
Parameters Used to Generate Targets in Simulation Scenarios I and II

Parameter	Meaning	Scenario I	Scenario II
N	Number of targets	50	50
$\mathbf{v}_g$	Group velocity of targets	$(1, 1, \sqrt{2})$ m/s	100 $(1, 1, \sqrt{2})$ m/s
μ_d	Mean speed for target drift	1 m/s	100 m/s
σ_d	Standard deviation of speed for target drift	0.25 m/s	20 m/s
T	Total target simulation time	1.4 s	1.4 s
Δt	Time step for target simulation	0.02 s	0.02 s

Table III
Sensor Parameters Used in the Simulations

Parameter	Meaning	Values
Σ_A, Σ_B	Sensor covariance matrices	$I_{2\times 2}$
Σ_C	Sensor covariance matrices (cigars)	diag[16, 1, 1], diag[1, 16, 1], diag[1, 1, 16]
	Sensor covariance matrices (pancakes)	diag[1, 16, 16], diag[16, 1, 16], diag[16, 16, 1]
θ	Angle between sensor planes A and B	$0^\circ, 10^\circ, 60^\circ, 90^\circ$
ϕ	Rotation angle of Sensor C	$0^\circ, 30^\circ, 60^\circ, 90^\circ$

and II. The parameter values for these two scenarios are shown in Table II.

After each time step, the target positions were orthogonally projected onto the planes of the two 2D sensors. The measurement uncertainty of Sensors A and B was modeled by adding Gaussian white noise to these target positions with mean zero and covariance matrices Σ_A and Σ_B, respectively. To simulate measurements from Sensor C, we added Gaussian white noise with mean zero and covariance matrix Σ_C to the computed target positions. We replicated each target trajectory simulation $M = 1000$ times, each with a different choice of the random parameters, v_n, $\mathbf{u}_n$, and different noise realizations in each sensor. The performance results we present below were obtained by averaging over these M replications.

The parameter values we used to specify the geometric configurations and measurement uncertainties of the sensors are summarized in Table III. For the results in Sections VI-B, VI-C, and VI-E, the planes of the two 2D sensors were orthogonal ($\theta = 90^\circ$). However, in Section VI-D we also present results for nonorthogonal sensor planes using the values of θ given in Table III. For the two 2D sensors, the covariance ellipsoid was chosen to be the unit sphere. We considered both cigar-shaped and pancake-shaped covariance ellipsoids for the 3D sensor. With cigar-shaped ellipsoids, the length of the major axis was 4 and the other two axes were of length 1. With pancake-shaped ellipsoids, the length of the minor axis was 1 and the other two axes were of length 4. Unless otherwise noted, the covariance matrix, Σ_C, of the 3D sensor was chosen to be diagonal. However, at the end of Section VI-D, we present results for a nondiagonal covariance matrix obtained by rotating the matrix diag[16, 1, 1] about the y-axis by the angles, ϕ, shown in the last row of Table III. To illustrate how the choice of covariance matrix is related to the measurement uncertainty of a realistic 3D radar, we observe that a radar that has a limited field of view, points in the z-direction, and has high range resolution and low angular resolution corresponds to a 3D sensor with a pancake-shaped covariance ellipsoid with $\Sigma_C = \text{diag}[\alpha, \alpha, \beta]$, where $\alpha \gg \beta$.

In Fig. 2a, we show the projection onto the xz-plane of the actual trajectories of the 50 targets for one random realization of the target trajectories in Scenario I. The target tracks diverge more from each other in Scenario II than in Scenario I. In Fig. 2b, we show the projection onto the xz-plane of these trajectories as measured by Sensor C in the case that $\Sigma_C = \text{diag}[16, 1, 1]$, and in Fig. 2c we show the targets in the xz-plane at the final time as measured by Sensors B and C. For this result, the planes of the 2D sensors were orthogonal.

At each time step, we computed the average distance between pairs of targets. Since $\Sigma_A = \Sigma_B = \text{Id}$, we plot the performance of each algorithm as a function of the normalized average separation between the targets, where the normalization factor is the square root of the trace of Σ_C.

B. Data Association With 32- and 322-Sensor Systems

In Fig. 3, we compare the performance of a 32-sensor system, consisting of sensors X and C ($X = A$ or B), to that of the 322-sensor system by plotting the percentage of correct two-way assignments for the X-to-C match as a function of the normalized average separation between the targets. These results were obtained in Scenario II in the case that the planes of Sensors A and B were orthogonal, and for the six choices of co-

variance matrix, Σ_C, shown in Table III. The results for the three cigar-shaped covariance ellipsoids are shown in the top row of Fig. 3, and those for the pancake-shaped ellipsoids are shown in the bottom row. In each subfigure, we compare the results obtained using the two 32-sensor systems to the 322-sensor system. For the 32-sensor systems, we used Munkres' algorithm to solve the assignment problem, which we refer to as the 2D algorithm in Fig. 3. Although the 5D algorithm produces three-way A-to-B-to-C matches, to make fair comparisons to the 32-sensor systems we only consider the percentages of correct two-way A-to-C and B-to-C assignments.

We begin with two preliminary observations. First, due to the inherent symmetry, for the 2D algorithm the percentage of correct B-to-C matches in Fig. 3b is statistically identical to the percentage of correct A-to-C matches in Fig. 3c. However, for the 5D algorithm, the A-to-C performance curve in Fig. 3b is slightly different from the B-to-C performance curve in Fig. 3c since this iterative algorithm is not invariant to the y-to-z coordinate switch. (The first iteration always involves a B-to-C match.) Second, in Fig. 3b, the performance of both algorithms is significantly better for the B-to-C match than for the A-to-C match since when $\Sigma_C = \text{diag}[1, 16, 1]$, the projection of Σ_C onto the plane of Sensor A is diag[1, 16], whereas the projection onto the plane of Sensor B is diag[1, 1].

Next, we compare the percentage of correct two-way matches for the 2D and 5D algorithms. In Fig. 3, we observe that the performance of the two algorithms is comparable whenever the (1,1) entry of Σ_C is comparable to that of Σ_A and Σ_B (see Fig. 3b–d). The reason is that when the uncertainty in the A–C and B–C data in their common x-dimension is comparable to the uncertainty in the A–B data, there is no advantage to be gained by applying the A-to-B consistency check in the 5D algorithm. On the other hand, when the (1,1) entry of Σ_C is significantly larger than that of Σ_A and Σ_B, the performance is generally much better with the 5D algorithm (see Fig. 3a, e, and f).

C. 322-Sensor System: 4D Versus 5D Algorithm

In this subsection, we compare the performance of the 4D and 5D algorithms for the 322-sensor system. In Fig. 4, we plot the percentage of correct three-way A-to-B-to-C matches as a function of the normalized average separation between the targets. By a correct assignment for the A-to-B-to-C match, we mean that the A-to-C and B-to-C (and hence the A-to-B) matches are all correct. As in Fig. 3, these results were obtained in Scenario II in the case that the planes of Sensors A and B were orthogonal, $\theta = 90^\circ$.

First, we discuss how the percentage of correct three-way matches for the 4D algorithm depends on the choice of Σ_C. In Fig. 4a and b, we see that when the normalized average separation is 10, the percentage of correct assignments is 75% with $\Sigma_C = \text{diag}[1, 16, 1]$ but only 65% with $\Sigma_C = \text{diag}[16, 1, 1]$. The reason for the poorer performance when the uncertainty in the Sensor C measurements is larger on the x-axis is that this axis is common to both 2D sensors. Consequently, the large uncertainty in the Sensor C data remains when the data are projected onto the xy-plane of Sensor A and onto the xz-plane of Sensor B. On the other hand, when $\Sigma_C = \text{diag}[1, 16, 1]$ the large uncertainty in the Sensor C data only remains when the data are projected onto one of these two planes, leading to better performance of the 4D algorithm in this case.

We now compare the percentage of correct three-way matches with the 4D and 5D algorithms. For the same reasons as in the discussion of Fig. 3, the 5D algorithm significantly outperforms the 4D algorithm when the (1,1) entry of Σ_C is significantly larger than that of Σ_A and Σ_B, while the performance of the two algorithms is comparable in the other three cases. The reason the 5D algorithm outperforms the 4D algorithm in some cases is

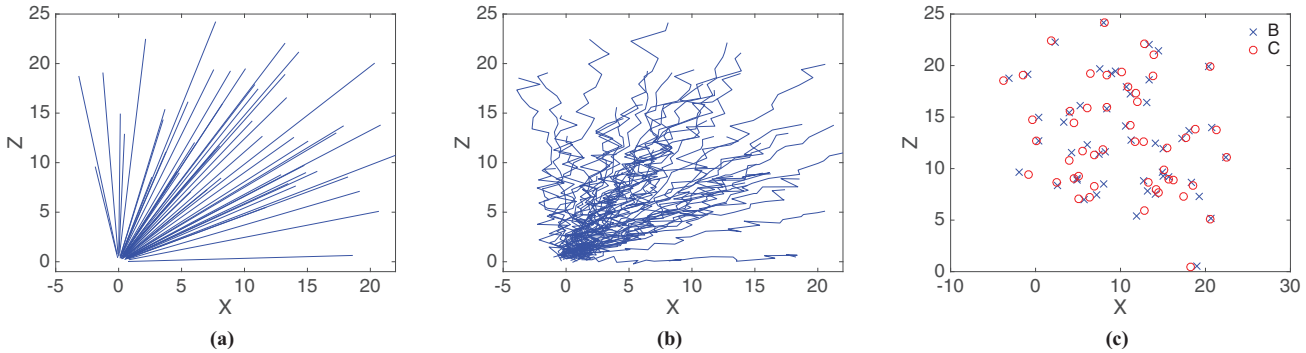
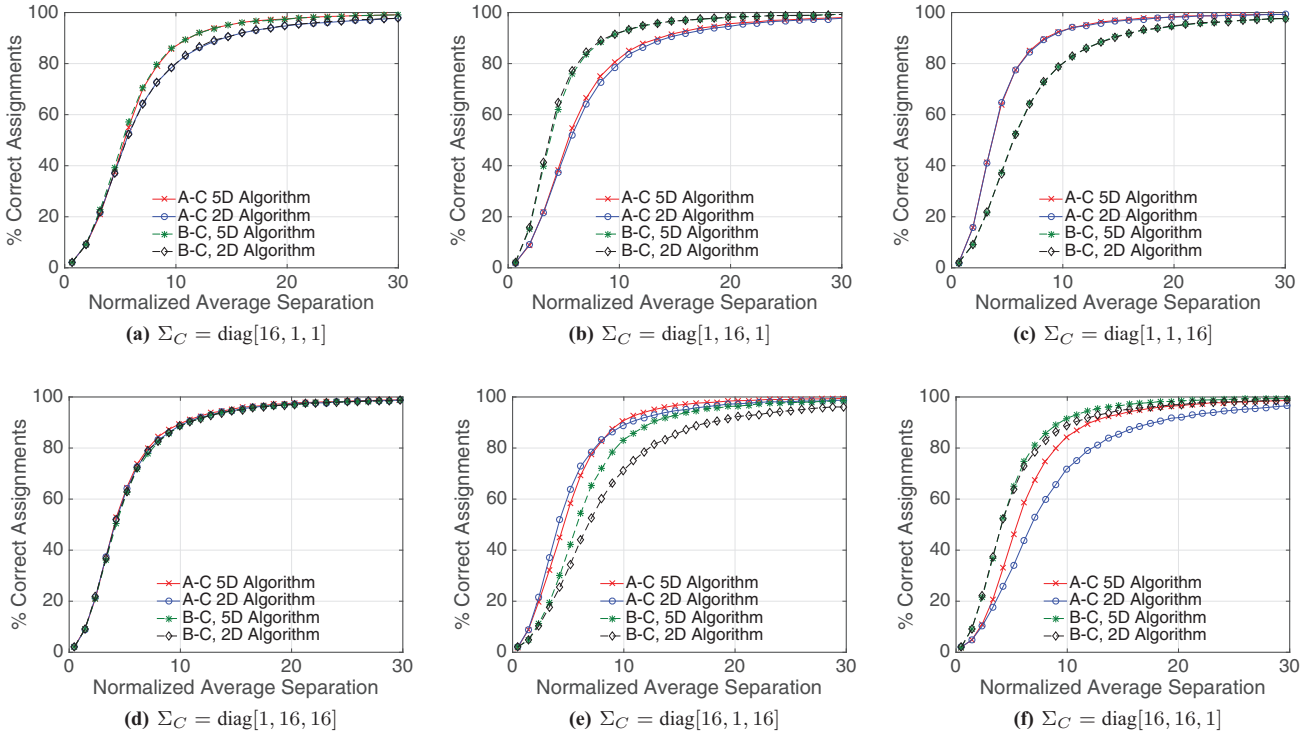


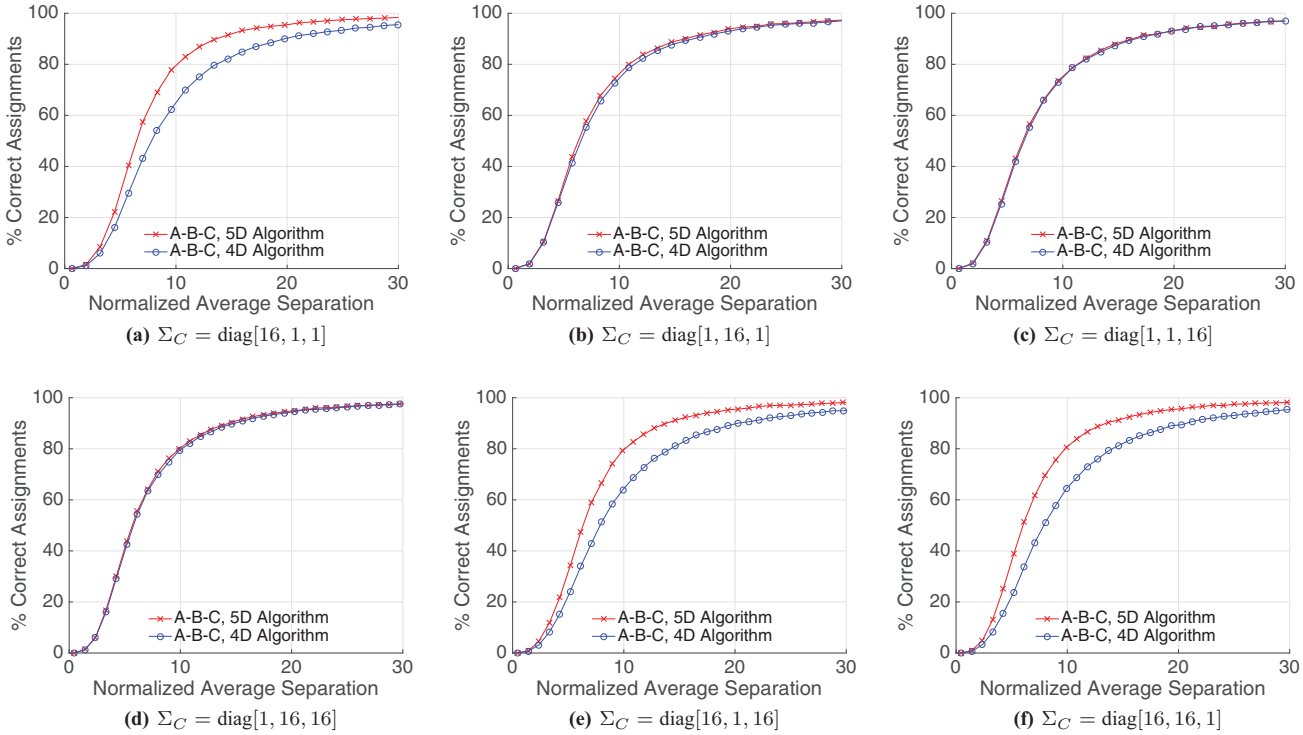
Fig. 2. (a) Projection into the x–z plane of the actual trajectories of 50 moving targets for one random realization of the target trajectory simulation in Scenario I. (b) Projection into the x–z plane of the trajectories of the same 50 moving targets as observed by Sensor C after the addition of Gaussian white noise with covariance matrix $\Sigma_C = \text{diag}[16, 1, 1]$. (c) Measured targets in the x–z plane as observed by Sensor B together with the projection into the x–z plane of the measured targets as observed by Sensor C. The planes of the two 2D sensors were orthogonal ($\theta = 90^\circ$) and the measurements were taken at the final time.



(a) $\Sigma_C = \text{diag}[16, 1, 1]$ **(b)** $\Sigma_C = \text{diag}[1, 16, 1]$ **(c)** $\Sigma_C = \text{diag}[1, 1, 16]$

(d) $\Sigma_C = \text{diag}[1, 16, 16]$ **(e)** $\Sigma_C = \text{diag}[16, 1, 16]$ **(f)** $\Sigma_C = \text{diag}[16, 16, 1]$

Fig. 3. Percentage of correct assignments for two-way matches as a function of the normalized average separation between the targets in Scenario II with orthogonal 2D sensor planes ($\theta = 90^\circ$) and $\phi = 0^\circ$. The results shown are for the A-to-C match using the 5D algorithm (red lines with crosses), the A-to-C match using Munkres' algorithm (blue lines with circles), the B-to-C match using the 5D algorithm (green dashed lines with stars), and the B-to-C match using Munkres' algorithm (black dashed lines with diamonds). The results with cigar-shaped covariance ellipsoids for Sensor C are shown in the top row, and those with pancake-shaped ellipsoids are shown in the bottom row.



(a) $\Sigma_C = \text{diag}[16, 1, 1]$ **(b)** $\Sigma_C = \text{diag}[1, 16, 1]$ **(c)** $\Sigma_C = \text{diag}[1, 1, 16]$

(d) $\Sigma_C = \text{diag}[1, 16, 16]$ **(e)** $\Sigma_C = \text{diag}[16, 1, 16]$ **(f)** $\Sigma_C = \text{diag}[16, 16, 1]$

Fig. 4. Percentage of correct assignments for the three-way A-to-B-to-C match as a function of the normalized average separation between the targets in Scenario II with orthogonal 2D sensor planes ($\theta = 90^\circ$) and $\phi = 0^\circ$. The results shown were obtained using the 5D algorithm (red line with crosses) and the 4D algorithm (blue line with circles). The results with cigar-shaped covariance ellipsoids for Sensor C are shown in the top row, and those with pancake-shaped ellipsoids are shown in the bottom row.

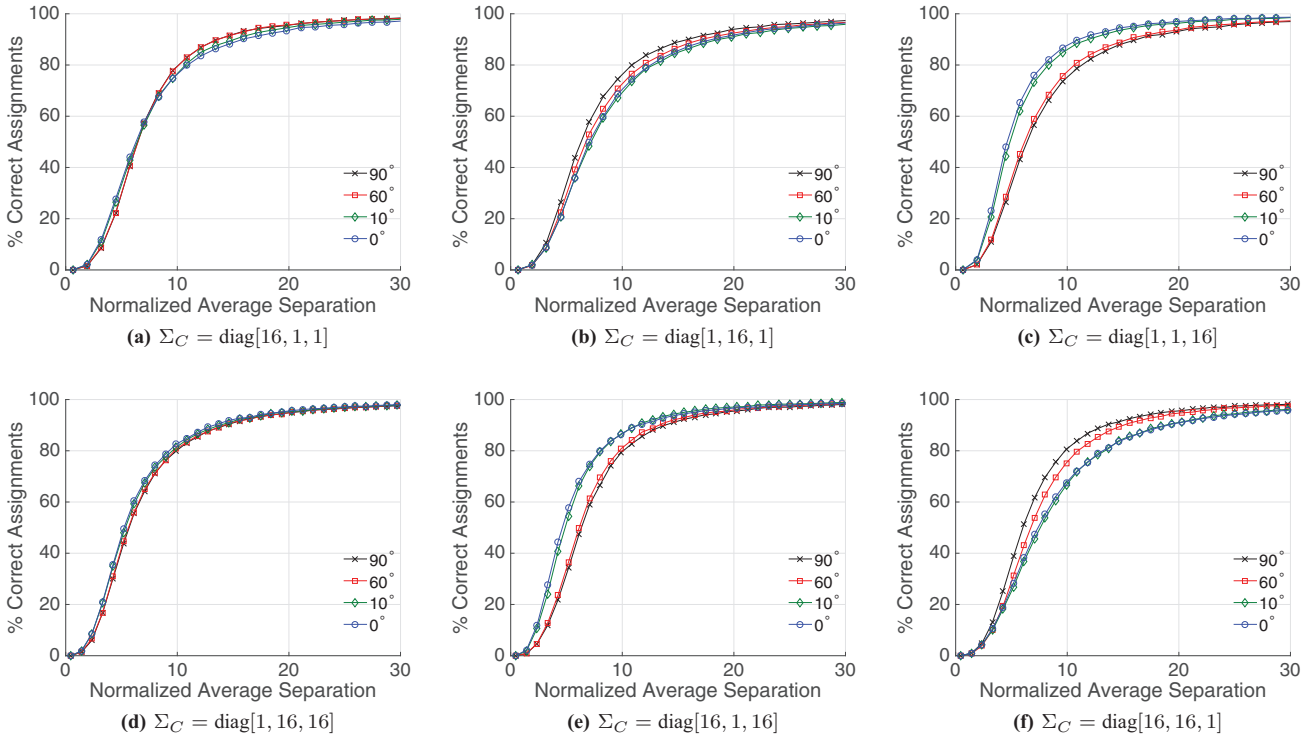
that the application of the A–B consistency check in column 5 of $D_{N\times 5}$ rules out some incorrect A-to-B matches made by the 4D algorithm. As a consequence, during the iteration process the 5D algorithm also corrects for some of the matching errors made by the poorer performing of the A-to-C and B-to-C matches.

D. Relative Sensor Orientation Effects for 322-Sensor

Having identified situations in which the performance with the 5D algorithm is significantly better than that with the other two methods, we now focus on how the percentage of correct three-way matches with the 5D algorithm depends on the relative orientation of the three sensors. We begin by studying how the 5D algorithm performs as the angle, θ, between the planes of the 2D sensors decreases from $\theta = 90^\circ$ to $\theta = 0^\circ$. These results were obtained with target simulation Scenario II. As $\theta \to 0^\circ$, the plane of Sensor B converges to that of Sensor A and neither of the 2D sensors has the ability to distinguish between two targets with the same (x, y)-value but different z-values. Although this observation suggests that the percentage of correct assignments should decrease as $\theta \to 0^\circ$, we will show that in many cases this decrease is slight. Moreover, in certain situations the performance is actually significantly better when the 2D sensor planes are parallel than when they are perpendicular, due to the suppression of noise from Sensor C that is orthogonal to the common plane of the 2D sensors.

In Fig. 5a and d, the percentage of correct three-way matches with the 5D algorithm is insensitive to the angle, θ, between the planes of Sensors A and B. The reason for this performance insensitivity is that the covariance ellipsoid for Sensor C has a circular cross section in the yz-plane, and so the projection of Σ_C onto the plane of Sensor B is independent of θ. Consequently, the performance of the B-to-C match (not shown) is completely independent of θ. Nevertheless, we do not expect the percentage of correct three-way matches with the 5D algorithm to be totally independent of θ, since the relative alignment of the two 2D sensors changes as θ changes.

The scenario in Fig. 5f, for which $\Sigma_C = \text{diag}[16, 16, 1]$, closely corresponds to a 322-sensor system in which all three sensors are mounted on a single platform with the 2D sensors pointing in similar directions (so that the A-to-B match solves the stereopsis problem), with the 3D sensor being a radar with good range accuracy, and with the 2D sensors having better angular measurement accuracy than the radar. In Fig. 5b and f, the percentage of correct three-way matches decreases more as $\theta \to 0^\circ$ than in Fig. 5a and d since in these cases the area of the covariance ellipse of the projection of Σ_C onto the plane of Sensor B increases as $\theta \to 0^\circ$. As a consequence, the performance of the B-to-C match (not shown) degrades as $\theta \to 0^\circ$. When $\theta = 0^\circ$, the performance is worse when $\Sigma_C = \text{diag}[16, 16, 1]$ than with the other five covariance matrices, since the covariance ellipse of the projection of Σ_C onto the xy-plane is largest in this case.



(a) $\Sigma_C = \text{diag}[16, 1, 1]$ **(b)** $\Sigma_C = \text{diag}[1, 16, 1]$ **(c)** $\Sigma_C = \text{diag}[1, 1, 16]$

(d) $\Sigma_C = \text{diag}[1, 16, 16]$ **(e)** $\Sigma_C = \text{diag}[16, 1, 16]$ **(f)** $\Sigma_C = \text{diag}[16, 16, 1]$

Fig. 5. Percentage of correct assignments for the A-to-B-to-C match using the 5D algorithm as a function of the normalized average distance between the targets in Scenario II with $\phi = 0^\circ$. The different curves show the results for different values of the angle, θ, between the planes of the two 2D sensor. The results with cigar-shaped covariance ellipsoids for Sensor C are shown in the top row, and those with pancake-shaped ellipsoids are shown in the bottom row.

In Fig. 5c and e, we see that rather than decreasing, the percentage of correct three-way matches increases significantly as $\theta \rightarrow 0^\circ$. The reason for this behavior is that when θ is small the large uncertainty of the Sensor C measurements in the z-dimension is (almost) completely suppressed when the Sensor C data are (almost) orthogonally projected onto the planes of both Sensors A and B. Since the uncertainty in the z-dimension is relatively large and affects all the measurements made by Sensor C, its suppression plays a much larger role in increasing the performance than the very minor reduction in performance that occurs whenever two targets with similar (x, y)-coordinates but different z-coordinates cannot be distinguished by nearly horizontal 2D sensors.

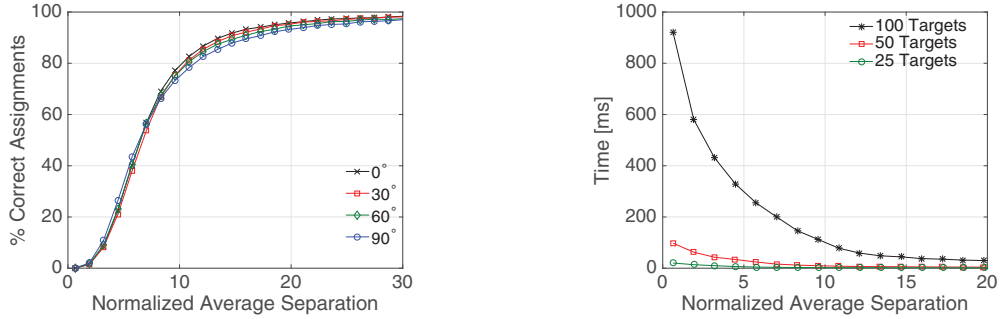
In Fig. 6a, we show the results we obtained when Σ_C is not diagonal. These results were obtained with target simulation Scenario II and $\theta = 90^\circ$. Specifically, we chose Σ_C to be a rotation about the y-axis by an angle, ϕ, of the matrix diag[16, 1, 1]. For these simulations, we chose $\phi = 0^\circ, 30^\circ, 60^\circ$, and 90°, as in Table III, so that when $\phi = 0^\circ$, $\Sigma_C = \text{diag}[16, 1, 1]$, and when $\phi = 90^\circ$, $\Sigma_C = \text{diag}[1, 1, 16]$. These results reproduce the expected decrease in the percentage of correct three-way matches that we see when comparing the $\theta = 90^\circ$ plots in Fig. 5a and b, and therefore serve to verify the correctness of the theory we developed for nondiagonal covariance matrices.

E. Target Position Estimates

For each replication, we computed the estimated target positions using (40) for three different assignment methods: Munkres' algorithm using data from one 2D sensor (A) and the 3D sensor, and the 4D and 5D algorithms. In Fig. 7, we plot the normalized RMSE in the estimated target positions as a function of the normalized average separation for the three assignment methods. The RMSE is the square root of the trace of the covariance matrix of the estimated target positions. In Fig. 7, both the RMSE and the average separation between the targets are normalized by the square root of the trace of the covariance matrix of Sensor C. Since this normalization factor equals the RMSE of the target positions as measured by Sensor C alone, when the normalized RMSE is less than 1 the RMSE performance of the algorithm is better than the RMSE performance with Sensor C alone. These results were obtained in target simulation Scenario I with $\theta = 90^\circ$ and $\phi = 0^\circ$. (The targets are less well separated in Scenario I than in Scenario II.)

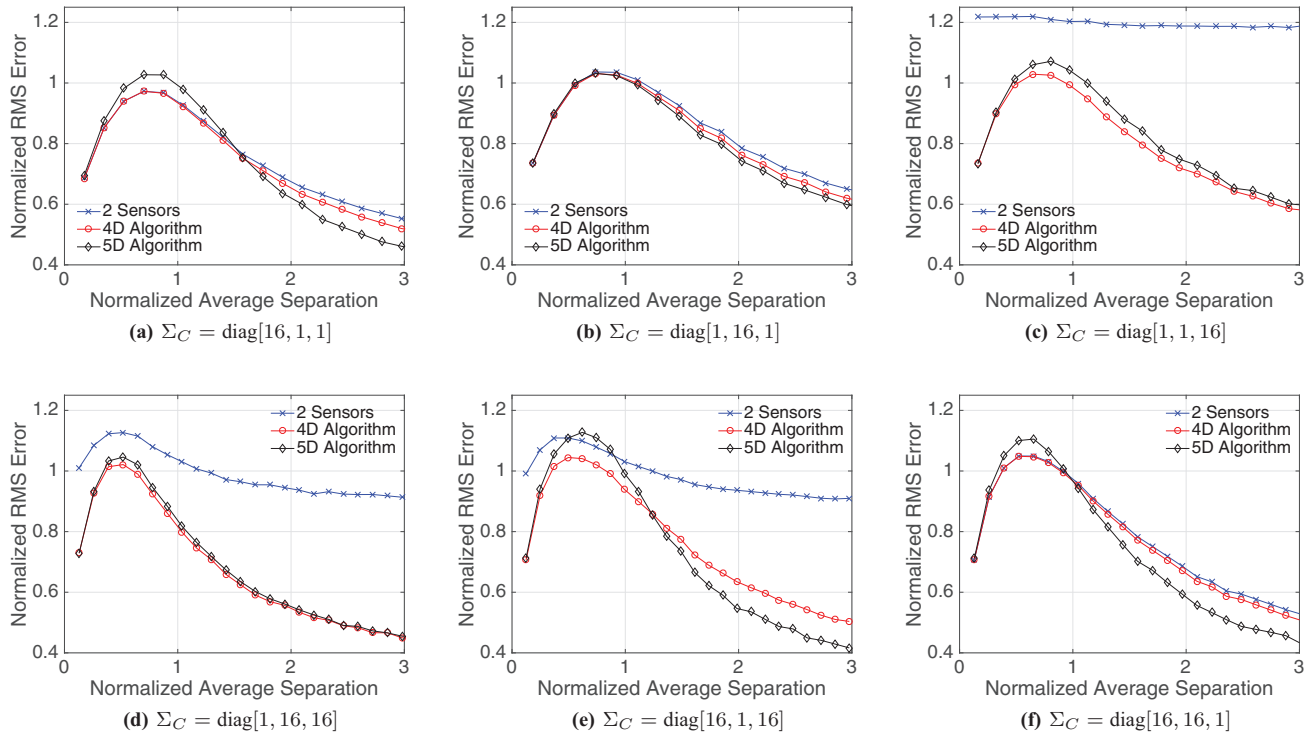
In Fig. 7a, we show the results with $\Sigma_C = \text{diag}[16, 1, 1]$. The initial rapid rise of the RMSE occurs simply because the distance between the targets increases, thereby increasing the RMSE for *any* choice of assignment. However, once the target separation exceeds the uncertainty in the Sensor C data, i.e., once the normalized average separation is greater than 1, the performance of all three algorithms is better than the performance of Sensor C alone, and all three of the RMSE curves decrease as the target separation increases. Therefore, whenever the target separation exceeds the uncertainty in the Sensor C measurements, it is advantageous to use a 32- or 322-sensor system instead of a 3D sensor alone, provided that the uncertainty of the 2D sensor measurements is small compared to that of the 3D sensor. When the average separation distance is greater than 2, the performance is worst with the two-sensor assignment method and best with the 5D algorithm. On the other hand, when the normalized average separation is less than 1, the 5D algorithm does not perform as well as the other two methods. This result holds for essentially the same reasons we gave for the poorer performance of the 5D algorithm for the A-to-C match in Fig. 3e.

Finally, in Fig. 7c, we see that with $\Sigma_C = \text{diag}[1, 1, 16]$, the RMSE obtained using the two-sensor method remains large irrespective of the average separation between the targets, since the z-coordinate of the estimated target position in (40) is based only on the estimate, $\widehat{Z}_C$, which has large uncertainty, $\sigma_{CZ} = 4$. This result demonstrates that there can be a significant advantage to using a second 2D sensor, especially in situations where there is a large uncertainty in the measurements made by Sensor C in the dimension that is orthogonal to the plane of the first 2D sensor.



(a) (b)

Fig. 6. (a) Percentage of correct assignments for the 5D algorithm as a function of the normalized average separation between the targets in Scenario II with orthogonal 2D sensor planes ($\theta = 90^\circ$). The different curves show the results for covariance matrices, Σ_C, obtained by rotating the matrix diag[16, 1, 1] about the y-axis through different angles, ϕ. (b) Average execution time in milliseconds for $N = 25, 50$, and 100 targets.



(a) $\Sigma_C = \text{diag}[16, 1, 1]$ **(b)** $\Sigma_C = \text{diag}[1, 16, 1]$ **(c)** $\Sigma_C = \text{diag}[1, 1, 16]$

(d) $\Sigma_C = \text{diag}[1, 16, 16]$ **(e)** $\Sigma_C = \text{diag}[16, 1, 16]$ **(f)** $\Sigma_C = \text{diag}[16, 16, 1]$

Fig. 7. Normalized RMSE in the estimated target positions as a function of normalized average distance between targets in Scenario I with $\theta = 90^\circ$ and $\phi = 0^\circ$. The results shown are for position estimates obtained after matching data from two sensors, A and C, using Munkres' algorithm (blue line with crosses), from all three sensors using the 4D algorithm (red lines with circles), and from all three sensors using the 5D algorithm (black lines with diamonds). The results with cigar-shaped covariance ellipsoids for Sensor C are shown in the top row, and those with pancake-shaped ellipsoids are shown in the bottom row.

F. Execution Time of Algorithms

In Fig. 6b, we show how the execution time of the 5D algorithm scales as the number of targets and the normalized average separation between the targets are varied. For this study, we implemented the algorithm in MATLAB on a MacBook Pro laptop with a 2.5 GHz Intel Core I5 processor. The three curves in the plot were obtained using $N = 25$, 50, and 100 targets. Each curve shows the average execution time (in milliseconds) of the 5D algorithm as a function of the normalized average separation between the targets. The increase in the computational time as the normalized average separation decreases is due to the fact that Munkres' algorithm requires more iterations to obtain a two-way match when the targets are more closely spaced. Indeed, we found that the number of iterations of the 5D algorithm is largely independent of the normalized average separation. The mean of the ratio of the average execution times of the 5D and 4D algorithms, taken over all values of the average separation, was 2.1 for 25 targets, 2.3 for 50 targets, and 3.2 for 100 targets. Since the 4D algorithm involves two applications of Munkres' algorithm, for 50 targets it is therefore about 4.6 times as expensive to obtain an assignment for the 322-sensor system using the 5D algorithm as it is for the 32-sensor system using Munkres' algorithm.

VII. DISCUSSION

In the previous sections, we used idealized sensor models to investigate how the performance of 322- and 32-sensor systems depends on the relative alignment of the 2D sensors and on the orientation of the covariance matrix of the 3D sensor. In this section, we first discuss the extent to which the theory we developed can be generalized to more realistic 3D radar and 2D camera models. Then, we summarize the main results from Section VI and interpret them in the context of more realistic systems.

Let $\mathbf{x} = (x_1, x_2, x_3)$ denote the position of a target in $\mathbb{R}^3$. In place of Sensor A, we consider a camera located at the origin of $\mathbb{R}^3$ pointing in the x_3-direction. We model camera measurements using a perspective projection as

$$(\widehat{w}_1, \widehat{w}_2) = \widehat{\mathbf{w}} = \psi_{2D}(\mathbf{x}) + \mathbf{u} = \alpha \left(\frac{x_1}{x_3}, \frac{x_2}{x_3} \right) + (u_1, u_2) \tag{42}$$

where α is the focal length and $\mathbf{u} \sim MN(\mathbf{0}, \Sigma_{2\times 2})$. In place of Sensor C, we consider a 3D radar located at $\mathbf{y} \in \mathbb{R}^3$ that makes measurements [16]

$$\widehat{\mathbf{z}} = \psi_{3D}(\mathbf{x} - \mathbf{y}) + \mathbf{v} \tag{43}$$

where ψ_{3D} is the rectangular to spherical coordinate transformation and $\mathbf{v} \sim MN(\mathbf{0}, \Sigma_{3\times 3})$. As in (11), we

transform the 3D radar measurement to the measurement plane of the camera via

$$\widehat{\boldsymbol{\xi}} \;=\; \Psi(\widehat{\mathbf{z}}) \tag{44}$$

where $\Psi = \psi_{2\mathrm{D}} \circ \psi_{3\mathrm{D}}^{-1}$ is the transformation from the space of radar measurements to the camera plane.

Since the main advantage of a third sensor is to resolve cluttered scenes [7], it is reasonable to assume that the targets are confined to a sufficiently small region, $\mathcal{R}$, of $\mathbb{R}^3$. Then

$$\widehat{\boldsymbol{\xi}} \;\approx\; \psi_{2\mathrm{D}}(\mathbf{x}-\mathbf{y}) + T_{\mathbf{x}}\mathbf{v} \tag{45}$$

where $T_{\mathbf{x}}$ is the 2×3 matrix of the linearization of Ψ at $\psi_{3\mathrm{D}}(\mathbf{x}-\mathbf{y})$. If $\bar{\mathbf{x}}$ denotes a nominal center for $\mathcal{R}$, which can be determined from radar measurements, then $T_{\mathbf{x}} \approx T_{\bar{\mathbf{x}}}$, which is independent of the particular target. As in (13), we take the difference of the measurements (42) and (45) in the camera plane to obtain

$$\widehat{\boldsymbol{\xi}} - \widehat{\mathbf{w}} \approx \psi_{2\mathrm{D}}(\mathbf{x}-\mathbf{y}) - \psi_{2\mathrm{D}}(\mathbf{x}) + T_{\bar{\mathbf{x}}}\mathbf{v} - \mathbf{u}. \tag{46}$$

For aerospace and defense applications, we may assume that $\mathcal{R}$ is at a large distance from the camera, so that $x_3 = \bar{x}_3 + \Delta x_3$, with $\Delta x_3 \ll \bar{x}_3$. Therefore,

$$(w_1, w_2) = \psi_{2\mathrm{D}}(x_1, x_2, x_3) \approx \frac{\alpha}{\bar{x}_3}(x_1, x_2) \tag{47}$$

and so as in (13), we can eliminate the unknown target location, $\mathbf{x}$, to obtain

$$\widehat{\boldsymbol{\xi}} - \widehat{\mathbf{w}} \approx -(y_1, y_2) + T_{\bar{\mathbf{x}}}\mathbf{v} - \mathbf{u}. \tag{48}$$

Similarly, in place of the compatibility condition (18) for Sensors A and B, we have that

$$\frac{\bar{x}_3}{\alpha_A}\widehat{w}_1^A \;-\; \frac{\bar{x}_2}{\alpha_B}\widehat{w}_1^B \;=\; \frac{\bar{x}_3}{\alpha_A}u_1^A \;-\; \frac{\bar{x}_2}{\alpha_B}u_1^B. \tag{49}$$

Using (48) and (49) and a formula for $T_{\bar{\mathbf{x}}}$, we can then compute replacements for the covariance matrices, $\Sigma_{4\mathrm{D}}$ in (15) and $\Sigma_{5\mathrm{D}}$ in (21). In this manner, the 4D and 5D algorithms can be extended to the scenario described above in which the targets are confined to a sufficiently small region far from the sensors. Moreover, the results in Section VI can also be applied to this scenario. These results can be summarized as follows.

First, in Section VI-B we showed that whenever the measurement uncertainty of the 3D sensor is large compared to that of the 2D sensors along the common axis of the planes of the 2D sensors, the percentage of correct two-way matches for the 322-sensor system with the 5D algorithm is significantly better than that of a 32-sensor system with the 2D algorithm. Under the same operating conditions, the percentage of correct three-way matches for the 322-sensor system with the 5D algorithm is significantly better than that with the 4D algorithm. With other relative sensor orientations, the percentage of correct matches was mostly independent of the choice of algorithm.

Second, with the aid of the bottom row of Fig. 5 we can make some conclusions about the performance of a 322-sensor system that includes a 3D radar with higher range resolution than angular resolution. Here, we also suppose that the 2D cameras have better angular resolution than the radar, and that the targets are confined to a small region, $\mathcal{R}$, as above. In this context, we refer to the vector $\bar{\mathbf{x}} - \mathbf{y}$ as the pointing direction of the radar. By the pointing direction of a camera, we mean the normal vector to the camera plane. We first consider a scenario in which all three sensors are mounted on the same platform with one of the cameras pointing in the same direction as the radar. In this scenario, the percentage of correct three-way matches decreases as the pointing direction of the second camera is lined up with that of the first (see Fig. 5f). On the other hand, for the scenario in Fig. 5d, the percentage of correct three-way matches is largely independent of the angle, θ, between the pointing directions of the cameras, and is moreover somewhat greater than for the scenarios in Fig. 5e and f. The covariance matrix that yields the largest percentage of correct three-way matches, i.e., that in Fig. 5d, is given by $\Sigma_C = \mathrm{diag}[1, 16, 16]$. This orientation of Sensor C corresponds to a scenario in which the radar is pointing along the $+x$-axis, perpendicular to the pointing directions of both cameras. Since all three sensors are observing the same small region, $\mathcal{R}$, this configuration can only be achieved if the sensors are viewing the targets from different locations.

Finally, for the above scenario in which the pointing direction of the radar is perpendicular to the pointing directions of the cameras, the results in Fig. 7d and e show that the RMSE in the estimated target positions is unacceptably large for a 32-sensor system. This is so because the large uncertainty in the radar measurements in the pointing direction of the camera cannot be resolved by the camera measurement. However, with the addition of the second camera, the RMSE performance is greatly improved.

Because we compare sensor measurements to sensor measurements rather than sensor measurements to unknown target locations, it is unlikely that the algorithms we developed can be extended to deal with missed detections as in [7] and [16]. Nevertheless, if we were to use the approach of [7], the above conclusions should still hold with an unknown number of targets, missed detections, and spurious measurements. Indeed, we would expect the performance gains for the 322-sensor to be even greater in this situation and/or when employing the Lagrangian relaxation assignment algorithm [17] in place of the 5D algorithm.

VIII. CONCLUSION

We used idealized sensor models and a special-purpose assignment algorithm for the M2MA problem to investigate the potential advantages of target tracking based on a 322-sensor system rather than a 32-sensor system. Depending on the relative alignment of the 2D sensors and the orientation of the covariance ellipsoid

of the 3D sensor, we found that position estimates obtained with the 322-sensor system can be much more accurate than those obtained with a 32-sensor system. Furthermore, we identified cases in which the percentage of correct assignments was significantly larger for the 322-sensor with the 5D algorithm than for the 32-sensor with Munkres' algorithm. Finally, we discussed applications of our results to specific scenarios based on realistic 3D radar and 2D camera models.

The overarching conclusion from our simulation study is that the performance of a 32- or 322-sensor system in which the covariance ellipsoid of the 3D sensor sensor is sufficiently aspherical can exhibit a strong dependence on the relative directions from which the sensors view the scene. Moreover, provided that the sensors are appropriately aligned, there can be a significant performance advantage to a target tracking system with two cameras and a radar as opposed to one camera and a radar.

ACKNOWLEDGMENTS

The authors thank Cody Harmon and Joseph Burnett for illuminating conversations and the reviewers for their suggestions that significantly improved this paper.

REFERENCES

[1] Y. Bar-Shalom and W. D. Blair
Multitarget–Multisensor Tracking: Applications and Advances, vol. III. Norwood, MA: Artech House, 2000.

[2] Y. Bar-Shalom and X.-R. Li
Multitarget–Multisensor Tracking: Principles and Techniques. Storrs, CT: YBS Publishing, 1995.

[3] Y. Bar-Shalom and R. W. Osborne
"Data association,"
in *Encyclopedia of Systems and Control*, J. Baillieul and T. Samad
Eds. Berlin, Germany: Springer, 2015, pp. 251–259.

[4] Y. Bar-Shalom, P. Willet, and X. Tian
Tracking and Data Fusion. Storrs, CT: YBS Publishing, 2011.

[5] D. P. Bertsekas
"Auction algorithms for network flow problems: A tutorial introduction,"
Comput. Optim. Appl., vol. 1, no. 1, pp. 7–66, 1992.

[6] S. Blackman and R. Popoli
Design and Analysis of Modern Tracking Systems. Norwood, MA: Artech House, 1999.

[7] S. Deb, K. Pattipati, and Y. Bar-Shalom
"A multisensor–multitarget data association algorithm for heterogeneous sensors,"
IEEE Trans. Aerosp. Electron. Syst., vol. 29, no. 2, pp. 560–568, Apr. 1993.

[8] S. Deb, M. Yeddanapudi, K. Pattipati, and Y. Bar-Shalom
"A generalized *S*-D assignment algorithm for multisensor–multitarget state estimation,"
IEEE Trans. Aerosp. Electron. Syst., vol. 33, no. 2, pp. 523–538, Apr. 1997.

[9] S. Herman and A. Poore
"Nonlinear least-squares estimation for sensor and navigation biases,"
in *Signal and Data Processing of Small Targets 2006*, vol. 6236. Bellingham, WA: International Society for Optics and Photonics, 2006, p. 623617.

[10] R. Jonker and A. Volgenant
"A shortest augmenting path algorithm for dense and sparse linear assignment problems,"
Computing, vol. 38, no. 4, pp. 325–340, 1987.

[11] W. Koch
Tracking and Sensor Data Fusion: Methodological Framework and Selected Applications. Berlin, Germany: Springer, 2016.

[12] F. Meyer et al.
"Message passing algorithms for scalable multitarget tracking,"
Proc. IEEE, vol. 106, no. 2, pp. 221–259, Feb. 2018.

[13] J. Munkres
"Algorithms for the assignment and transportation problems,"
J. Soc. Ind. Appl. Math., vol. 5, no. 1, pp. 32–38, 1957.

[14] C. H. Papadimitriou and K. Steiglitz
Combinatorial Optimization: Algorithms and Complexity. Chelmsford, MA: Courier Corporation, 1982.

[15] A. Petrovskaya and S. Thrun
"Model based vehicle detection and tracking for autonomous urban driving,"
Auton. Robots, vol. 26, no. 2–3, pp. 123–139, 2009.

[16] A. B. Poore
"Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking,"
Comput. Optim. Appl., vol. 3, no. 1, pp. 27–57, 1994.

[17] A. B. Poore and A. J. Robertson III
"A new Lagrangian relaxation based algorithm for a class of multidimensional assignment problems,"
Comput. Optim. Appl., vol. 8, no. 2, pp. 129–150, 1997.

[18] D. Smith and S. Singh
"Approaches to multisensor data fusion in target tracking: A survey,"
IEEE Trans. Knowl. Data Eng., vol. 18, no. 12, pp. 1696–1710, Dec. 2006.

[19] G. Thomaidis, L. Spinoulas, P. Lytrivis, M. Ahrholdt, G. Grubb, and A. Amditis
"Multiple hypothesis tracking for automated vehicle perception,"
in *Proc. Intell. Veh. Symp*. Piscataway, NJ: IEEE, 2010, pp. 1122–1127.

[20] Z. Wu, N. I. Hristov, T. H. Kunz, and M. Betke
"Tracking-reconstruction or reconstruction-tracking? Comparison of two multiple hypothesis tracking approaches to interpret 3D object motion from several camera views,"
in *Proc. Workshop Motion Video Comput*. Piscataway, NJ: IEEE, 2009, pp. 1–8.

César Contreras received the Bachelor's degree and the Master's degree in electronics engineering from Nuevo Laredo Institute of Technology, Nuevo Laredo, Tamaulipas, Mexico, in 2000 and 2004, respectively, and the Master's degree in mathematics from Texas A&M International University, Laredo, TX, USA, in 2014. He held several positions in the industry and is currently working toward the Ph.D. degree in applied mathematics from The University of Texas at Dallas. His research interests include control theory, nonlinear dynamical systems, electromagnetic theory, and semiconductor physics.

John Langford received the B.S. degree in physics from Texas A&M University, College Station, TX, USA, in 2011, and the M.S. degree in physics from The University of Texas at Dallas, Richardson, TX, USA, in 2014. He is currently working toward the Ph.D. degree in statistics at The University of Texas at Dallas. His current research interests include applications of deep learning in genomics.

Larry Ammann received the B.A. degree in physics and the Ph.D. degree in statistics from Florida State University, Tallahassee, FL, USA, in 1969 and 1976, respectively. He is currently a Professor of Statistics with the Department of Mathematical Sciences, The University of Texas at Dallas, Richardson, TX, USA. His research interests include robust multivariate statistical methods, signal processing, statistical computing, applied probability, and remote sensing.

John Zweck received the B.Sc. degree with honors from the University of Adelaide, Adelaide, Australia, in 1988, and the Ph.D. degree in mathematics from Rice University, Houston, TX, USA, in 1993. He has performed research in differential geometry, computer vision, mathematical modeling of engineering systems, and nonlinear waves. He is currently a Professor with the Department of Mathematical Sciences, The University of Texas at Dallas, Richardson, TX, USA. From 2006 to 2012, he was an Associate Professor of Mathematics and Statistics and an Affiliate Associate Professor of Computer Science and Electrical Engineering with the University of Maryland Baltimore County (UMBC). From 2003 to 2006, he was an Assistant Professor with the Department of Mathematics and Statistics, UMBC, and from 2000 to 2003, he was a Research Associate with the Department of Computer Science and Electrical Engineering, UMBC. He is a member of the Society for Industrial and Applied Mathematics.

Brian Marks received the B.S. degree in physics and mathematics from North Carolina State University, Raleigh, NC, USA, in 1995, and the Ph.D. degree in engineering sciences and applied mathematics from Northwestern University, Evanston, IL, USA, in 2000. He is currently a senior member of the technical professional staff at the Johns Hopkins University Applied Physics Laboratory, Laurel, MD, USA, where he has been working since 2013. He previously held academic positions at University of Maryland, Baltimore County and at Indiana University Bloomington. His research interests include applications of mathematics and statistics in the domains of engineering and physics, with recent focus on remote sensing, signal processing, tracking, and multisensor integration.

INTERNATIONAL SOCIETY OF INFORMATION FUSION

ISIF Website: http://www.isif.org

2019 BOARD OF DIRECTORS*

2017–2019	2018–2020	2019–2021
Wolfgang Koch	Fredrik Gustafsson	Kathryn Laskey
Simon Godsill	X. Rong Li	Felix Govaers
Murat Efe	Zhansheng Duan	Simon Maskell

*Board of Directors are elected by the members of ISIF for a three year term.

PAST PRESIDENTS

Lyudmila Mihaylova, 2018	Joachim Biermann, 2011	Chee Chong, 2004
Lyudmila Mihaylova, 2017	Stefano Coraluppi, 2010	Xiao-Rong Li, 2003
Jean Dezert, 2016	Elisa Shahbazian, 2009	Yaakov Bar-Shalom, 2002
Darin Dunham, 2015	Darko Musicki, 2008	Pramod Varshney, 2001
Darin Dunham, 2014	Erik Blasch, 2007	Yaakov Bar-Shalom, 2000
Wolfgang Koch, 2013	Pierre Valin, 2006	Jim Llinas, 1999
Roy Streit, 2012	W. Dale Blair, 2005	Jim Llinas, 1998

SOCIETY VISION

The International Society of Information Fusion (ISIF) is the premier professional society and global information resource for multidisciplinary approaches for theoretical and applied information fusion technologies.

SOCIETY MISSION

Advocate

To advance the profession of fusion technologies, propose approaches for solving real-world problems, recognize emerging technologies, and foster the transfer of information.

Serve

To serve its members and engineering, business, and scientific communities by providing high-quality information, educational products, and services.

Communicate

To create international communication forums and hold international conferences in countries that provide for interaction of members of fusion communities with each other, with those in other disciplines, and with those in industry and academia.

Educate

To promote undergraduate and graduate education related to information fusion technologies at universities around the world. Sponsor educational courses and tutorials at conferences.

Integrate

Integrate ideas from various approaches for information fusion, and look for common threads and themes–look for overall principles, rather than a multitude of point solutions. Serve as the central focus for coordinating the activities of world-wide information fusion related societies or organizations. Serve as a professional liaison to industry, academia, and government.

Disseminate

To propagate the ideas for integrated approaches to information fusion so that others can build on them in both industry and academia.

Call for Papers

The Journal of Advances in Information Fusion (JAIF) seeks original contributions in the technical areas of research related to information fusion. Authors are encouraged to submit their manuscripts for peer review http://isif.org/journal.

Call for Reviewers

The success of JAIF and its value to the research community is strongly dependent on the quality of its peer review process. Researchers in the technical areas related to information fusion are encouraged to register as a reviewer for JAIF at http://jaif.msubmit.net. Potential reviewers should notify via email the appropriate editors of their offer to serve as a reviewer.

Journal of Advances in Information Fusion Volume 14 Number 1 June 2019 ISSN 1557-6418