June 2022 Volume 17 Number 1 ISSN 1557-6418

JAIF22

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.

Survey on Clutter Spatial Intensity Estimation Methods for Target Tracking

FAHAD A. M.
SARITHA D. S.

This paper is a comprehensive and up-to-date survey on clutter spatial intensity estimation methods proposed over a span of two decades for radar target tracking. Here, the methods are grouped into three categories. The first category of methods is applicable to measurements in a track validation gate, while the second category of methods compute the clutter spatial intensity of any measurement in measurement space. Finally, the third category uses the concept of clutter generators, which act as the source of clutter measurements. Then, probability hypothesis density filter equations are used to derive the clutter spatial intensity of measurements. The above classification of different methods is based on the techniques used and on the assumptions made while computing the spatial intensity of clutter. This paper emphasizes the underlying ideas and assumptions of each of the methods so that the reader could understand not only how each method works but also their pros and cons. Also, an effort is made to bring out the interrelationship between different methods, wherever possible.

Manuscript received February 13, 2021; revised September 23, 2021; released for publication July 7, 2022.

The authors are with the Central Research Laboratory, Bharat Electronics Ltd., Jalahalli, Bengaluru, India (e-mail: fahad@bel.co.in; sarithads@bel.co.in).

1557-6418/22/$17.00

I. INTRODUCTION

The surveillance systems like radar generate detections (measurements) from the received signals based on a defined detection threshold. These detections may be from targets of interest or from random objects in surveillance space. If a target is present among the detections, then it is detected with a detection probability ($P_D < 1$). The detections from random objects are termed clutter or false alarms. These detections are fed to multitarget trackers (MTTs) without any prior information about the source of measurements. The primary task of any MTT algorithm is to solve the measurement-origin uncertainty, i.e., to identify target-originated measurements and clutter measurements. The above task is called "measurement-to-track association" in the target tracking world.

Usually, in multitarget tracking, the clutter is modeled as a nonhomogeneous Poisson process (NHPP) in measurement space [1]. The distribution of an NHPP process is fully described by its spatial intensity function [6], [7]. Hence, the clutter process can be explained by some clutter spatial intensity function, otherwise called a jargon clutter density in target tracking. Many practical MTTs require the clutter spatial intensity to compute the measurement likelihood ratio (i.e., ratio of the probability that a measurement is generated by a target to the probability that it is clutter) for the purpose of measurement-to-track association and new track initialization [8]–[10], [12]. An overoptimistic estimate for clutter spatial intensity leads to lower data association probabilities of measurement-to-tracks, thereby degrading the performance of MTT. Whereas, a pessimistic estimate of clutter spatial intensity may lead to faster confirmation of false tracks. Hence, it is important to have an optimal estimate of clutter spatial intensity at the tracker.

This paper is a consolidated survey on clutter spatial intensity estimation methods proposed over a span of two decades for radar target tracking. The research work on clutter spatial intensity estimators is scattered over many literatures with no consolidation of the work available till date. This paper is the first comprehensive and up-to-date survey on clutter spatial intensity estimators [3]–[5] to the best of the authors' knowledge. The contributions of the authors are as follows. We broadly group the different spatial intensity estimators into three categories. The first category of methods [1], [12], [14] computes the clutter spatial intensity of those measurements that are only in the track validation gate. The second category of methods [11], [18], [20]–[24], [38], [46], [49], [50] evaluates spatial intensity at any detection point in the surveillance region using some nonparametric density estimation methods. Finally, the third category of methods [27]–[29], [31]–[33], [36], [39], [40], [48] uses a notion of clutter generators independent of the actual targets, and they act as the source of clutter measurements. Thereafter, a probability hypothesis density

(PHD) filter-based recursions are derived for evaluating the intensity of clutter generators. This intensity is used to estimate the required clutter spatial intensity at any detection point. Here, this survey emphasizes the underlying ideas and assumptions of each of the methods so that the reader could understand not only how each method works but also their pros and cons. Moreover, this paper also brings out the interrelationship between different methods, wherever possible. However, the issues related to the implementation of methods are not discussed in this paper.

The rest of the paper is organized as follows: A mathematical definition for clutter spatial intensity is presented in Section II. The clutter density estimation methods based on a track validation gate are explained in Section III. Thereafter, Section IV describes the clutter spatial intensity estimators for any detection in measurement space. The clutter density estimators based on a clutter generator assumption are explained in Section V. Finally, an interrelation between the methods is discussed in Section VI.

II. DEFINITION OF CLUTTER SPATIAL INTENSITY

Assume that the clutter measurements follow an NHPP with spatial intensity function $c(\mathbf{z})$ varying with the d-dimensional location in space $\mathbf{z}$, and if $c(\mathbf{z})$ is continuous and locally integrable in small infinitesimal volume dV, then [41]

$$\lambda = \int_{V}^{V+dV} c(\mathbf{z})\mathbf{dz}, \tag{1}$$

where λ is the mean number of clutter points in the volume. Thus, clutter spatial intensity can be defined as the average number of clutter measurements in dV. From (1), it can be seen that the number of clutter measurements in surveillance space is not uniformly distributed but dependent on spatial coordinate $\mathbf{z}$. The clutter spatial intensity estimators presented in the subsequent sections either try to estimate or find some approximations for this clutter spatial intensity.

III. CLUTTER SPATIAL INTENSITY ESTIMATORS BASED ON TRACK VALIDATION GATE

It is common in the target tracking world to define a validation gate in order to select only the probable measurements corresponding to tracks. A simple ellipsoidal gate [2] is used as the validation gate in the case of single target trackers (STTs). Whereas, a union of validation gates (effective validation gate) of those tracks sharing measurements is used in the case of MTT. The type of estimators discussed in this section only evaluate the clutter spatial intensity of measurements that are in the validation gate/effective validation gate of track(s). Such estimators assume the spatial variation of clutter to be uniformly distributed in the volume of the validation gate/effective validation gate. The clutter spatial intensity estimate in such estimators is calculated as

$$\hat{\lambda}(\mathbf{z}) = \frac{\hat{m}_k}{V}, \tag{2}$$

where $\hat{m}_k$ is the expected number of clutter measurements in the validation gate, V is the volume of the validation gate, and $\hat{\lambda}(\mathbf{z})$ is the estimate for clutter spatial intensity. The effective track validation gate volume discussed in [13] can be used for clutter intensity estimation when the measurements are shared by different tracks. In [1], MTT uses sample spatial intensity (2) obtained from the set of measurements in the effective validation gate as the clutter spatial intensity. In this case, $\hat{m}_k = m_k$, where m_k is the total number of measurements in the validation gate. However, the method in [1] gives a biased estimate as it does not distinguish between target originated measurements and clutter measurements in the validation gate.

However, the method in [14] uses "track perceivability"[15] and that in [12] uses "track existence probability"[16] in order to give an unbiased estimate for the clutter spatial intensity of measurements in the effective track validation gate. The conditional mean estimator of spatial intensity for false alarms proposed in [14] is defined, for a single target, on the assumption that at any time there is one or no target measurement present in the validation gate. Therein, the mean number of false measurements is computed by excluding the target-originated measurements stochastically. The mean number of false alarms $\hat{m}_k$ can be evaluated as [14]

$$\hat{m}_k = m_k - \frac{r_k \frac{m_k}{V}}{\hat{\lambda}_k + r_k \frac{m_k}{V}}. \tag{3}$$

In the above, r_k, a measure of target-originated measurements, is defined as

$$r_k = \frac{P_D P_G P\left(\chi_k \mid Z^{k-1}\right)}{1 - P_D P_G P\left(\chi_k \mid Z^{k-1}\right)}. \tag{4}$$

Here, $P\left(\chi_k \mid Z^{k-1}\right)$ denotes the predicted track quality parameter, i.e., the track exists at time k given the set of previous measurements, and P_D and P_G denoting the track-detection probability and the probability that the target measurement is in its validation gate, respectively. Once $\hat{m}_k$ is computed, the spatial intensity of clutter can be approximated as given in (2). The conditional estimate of (3) is dependent on an initial estimate $\hat{\lambda}_k$, which can be computed using a maximum-likelihood (ML) estimator [14] as

$$\hat{\lambda}_k = \frac{m_k}{2V}\left(1 - r_k + \sqrt{(1-r_k)^2 + \frac{4m_k - 1}{m_k} r_k}\right). \tag{5}$$

In [12], the predicted track existence probability of tracks is used to stochastically identify target-originated measurements under the assumption of at most one detection per target. In a single-target case, the mean

number of false alarms therein can be shown to be

$$\hat{m}_k = m_k - P_D P_G P\left(\chi_k \mid Z^{k-1}\right). \tag{6}$$

Thereafter, (2) can be used to evaluate clutter spatial intensity. However, for a multitarget case, the extension is slightly exhaustive [12].

Validation gates in MTT are created using innovation matrices of tracks [2]. Hence, a measurement shared by different tracks will have different false alarm spatial intensities when used in updating different tracks. So, the effective track validation gate used in the above methods cannot be a suitable choice to estimate the spatial clutter intensity. Furthermore, the estimators based on a track validation gate only evaluate the clutter spatial intensity of those measurements inside the track validation gate. However, the false-alarm intensity of measurements outside the gates of tracks is also required as shown in [17] to compute the cost of newly initialized tracks.

IV. CLUTTER SPATIAL INTENSITY ESTIMATORS IN MEASUREMENT SPACE

Spatial intensity estimators of this category try to evaluate the false-alarm intensity of any detection point in measurement space or in a surveillance region. They utilize statistical methods like nonparametric density estimation [19] or expectation maximization (EM) toward this purpose. The classic clutter map estimator in [18] and [11] divides the whole measurement space into cells, and the false-alarm spatial intensity of measurements belonging to a cell is assumed constant. The false-alarm intensity in cell i is given by

$$\hat{\lambda}(\mathbf{z}) = \frac{\text{number of false alarms in cell } i}{\text{volume of cell} \times \text{number of scans}}. \tag{7}$$

Note that (7) returns a smoothed clutter intensity over a number of scans. Here, the spatial intensity estimate depends on the size of the cell, which also determines the bias of the estimate. The clutter map estimator can be compared to a multivariate histogram method [19]. The histogram method is discontinuous and unsuitable for data of two or more dimensions [19] and also gives a block nature to the estimated spatial intensity function. Even though the notion of dividing the measurement space into cells captures nonhomogeneity in some sense, the spatial intensity estimate evaluated by a clutter map method is overoptimistic, as shown in [18].

Two other methods, namely spatial clutter map estimator [18], [20] and temporal clutter map estimator [18], also require the surveillance region to be divided into cells manually and assume a stationary Poisson clutter process in each cell. Both methods define a mathematical distance to compute the inverse of clutter spatial intensity, called sparsity, of the measurements falling in a cell. The former relies on the nearest neighbor measurement distance, which is equal to the mathematical distance from the center of the cell to the nearest neighbor measurement, and the latter uses interarrival time between two consecutive measurements falling in the cell as the mathematical distance. The estimates obtained are averaged over time in order to smoothen the spatial intensity. However, the exact definition of mathematical distance between the center of the cell and the nearest measurement is not given therein. In order to use the nearest neighbor distance or the interarrival time between measurements in evaluating the clutter spatial intensity, it is assumed that the spatial intensity is homogeneous and isotropic in and around the cell. But, the inverse of the defined mathematical distance does not hold in the case of NHPP [41]. It was shown in [18] that a spatial clutter map estimator is effective when compared to a temporal clutter map estimator in a dynamic clutter.

The need to divide the measurement space into cells is eliminated in a spatial clutter measurement density estimator (SCMDE) [21] unlike the previous methods. It proposes to calculate the sparsity at any detection point by evaluating the volume of hypersphere with center at the detection point and radius equal to the l_2-norm distance to nth nearest detection point. The sparsity therein is the inverse of clutter spatial intensity at that point. The sparsity $\hat{\gamma}(\mathbf{z})$ in SCMDE is defined as

$$\hat{\gamma}(\mathbf{z}) = \frac{V\left(r^n(\mathbf{z})\right)}{n}, \tag{8}$$

where $V\left(r^n(\mathbf{z})\right)$ is the volume of the d-dimensional hypersphere with center $\mathbf{z}$ and radius $r^n(\mathbf{z})$ equal to the distance to the nth nearest measurement. Here, the volume of hypersphere is defined as

$$V\left(r^n(\mathbf{z})\right) = C_d \left(r^n(\mathbf{z})\right)^d, \tag{9}$$

where the constant C_d is defined as

$$C_d = \frac{2}{d}\frac{\pi^{d/2}}{\Gamma\left(d/2\right)}. \tag{10}$$

Here, $\Gamma(.)$ is the Gamma function. SCMDE has similar assumptions of homogeneity around the detection point like the former methods of this category. Moreover, the sparsity is assumed independent at points far from $\mathbf{z}$, thereby accounting for NHPP but with local homogeneity. It has been shown in [21] that the higher the order n of sparsity, the higher the bias in estimate in the case of a nonhomogeneous clutter process. This is due to the fact that as n increases, we tend to encounter more overlapping nonhomogenous areas, and an assumption of homogeneity in those areas could yield a biased estimate. SCMDE is similar to a k-nearest neighbor density estimation technique, and the spatial intensity estimate obtained is discontinuous in nature [19]. It has been shown in [21] and [22] that SCMDE is effective in STT because an unbiased false-alarm spatial intensity estimate is obtained at the target detections while penalizing the clutter measurements. However, it is not suitable for MTT [22] as it does not distinguish target measurements from clutter measurements. In [22] and [23], an improvement for SCMDE is proposed by

calculating the clutter probability of each measurement in the hypersphere. The clutter probability $P\left(\chi^0_{k,j} \mid Z^{k-1}\right)$ of each jth measurement $\mathbf{z}_{k,j}$, given T_k potential tracks at time k, is computed as [22]

$$P\left(\chi^0_{k,j} \mid Z^{k-1}\right) = \frac{1}{1+\sum_{\tau \epsilon T_k} \frac{P^{\tau}_{k,j}}{1-P^{\tau}_{k,j}}}. \tag{11}$$

Here, $\chi^0_{k,j}$ is the event that the measurement $\mathbf{z}_{k,j}$ has not originated from any given tracks and $P^{\tau}_{k,j}$ is the prior probability that the measurement originated from the track τ described in [12]. The MTTSCMDE proposed therein computes the sparsity at any detection point as

$$\hat{\gamma}(\mathbf{z}_{k,j}) = \frac{V\left(r^n(\mathbf{z}_{k,j})\right)}{\sum_{l=1}^{m} C^l_{k,j}}. \tag{12}$$

Here, $C^l_{k,j}$ is the clutter probability of the lth nearest detection point computed using (11).

Even though similarities can be drawn between SCMDE and MTTSCDME based on their assumptions, there lies a significant dissimilarity between them in the method adopted to evaluate clutter spatial intensity at a point. SCMDE utilizes the number of measurements in a hypersphere in computing the sparsity estimate. These measurements can be target detections as well as clutter measurements, which may lead to a biased estimate. Whereas, MTTSCMDE uses the mean number of clutter measurements with the aid of clutter measurement probability to reduce the bias in clutter spatial intensity estimate. The mathematical distance used in SCMDE as well as MTTSCMDE is not well-defined for range (r)/bearing (θ)/range-rate ($\dot{r}$) measurement space in cases like Doppler radar. One solution could be to write the clutter spatial intensity estimate $c(r, \theta, \dot{r})$ as the product of position clutter measurement intensity $c(r, \theta)$ and clutter likelihood of range-rate measurement $p(f_d)$, as shown in [42]. However, this amounts to an assumption that the clutter distribution in range rate space is homogeneous with respect to range and bearing. This assumption may not be true for all real scenarios due to anisotropy in the field of view of range rate, range, and bearing. Another possible solution could be to use a weighted normalized distance that reflects the actual distance of measurements in nonhomogeneous measurement space, as shown in [38].

In [24], the clutter spatial intensity estimation problem is fitted into a kernel density estimation (KDE) [19] framework. A multivariate Gaussian kernel density estimator that can handle the measurement origin uncertainty as well as the continuous arrival of measurements is proposed therein to evaluate the spatial intensity of false alarms. The measurement-origin uncertainty is solved using the joint association events and their corresponding probabilities given by MTTs [8], [10]. This is possible because the measurement origin is decided in an association event. If an association event χ_i with probability $P(\chi_i)$ gives a set of clutter measurements $Z^C_k = \left\{\mathbf{z}_{k,1}, \mathbf{z}_{k,2}, \ldots, \mathbf{z}_{k,n}\right\}$, then the normalized spatial intensity estimate conditional on the association event can be evaluated as [24]

$$\lambda^i(\mathbf{z}) = \sum_{j=1}^{n} w_j K_H\left(\mathbf{z} - \mathbf{z}_{k,j}\right). \tag{13}$$

Here, $K_H(.)$ is the Gaussian kernel with a positive definite bandwidth matrix $\mathbf{H}$ and corresponding weight w_j. Furthermore, using the following relation between conditional expectation and unconditional expected value

$$\mathbb{E}[Z] = \mathbb{E}[\mathbb{E}[Z \mid X]], \tag{14}$$

the clutter spatial intensity at a point $\mathbf{z}$ in measurement space can be computed as

$$\hat{\lambda}(\mathbf{z}) = \sum_{i=1}^{m} P(\chi_i)\, \lambda^i(\mathbf{z}). \tag{15}$$

In (14), $\mathbb{E}$ denotes the expectation operator. The performance of KDE depends on the choice of bandwidth rather than the shape of the kernel used [19]. Hence, a method to compute an optimal bandwidth is given in [24], wherein the gradient of the cost function is derived using a cross-validation technique [25]. A fixed bandwidth in a KDE method affects data with a long tail distribution [19]. Hence, an adaptive kernel bandwidth matrix based on adaptive KDE [19] is adopted in [24], which ensures the kernels at lower intensity areas have a different bandwidth from that of kernels at areas with high intensity. The probability density estimate obtained by KDE inherits all the continuity and differentiability properties of the kernel used [19]. Hence, the clutter spatial intensity estimate of (15) is continuous as the Gaussian kernel has continuity and differentiability of all orders. The method in [24] does not rely on the calculation of mathematical distance unlike SCMDE and MTTSCMDE. Hence, it is applicable to nonlinear sensors like Doppler radar. It has been shown in [24] that it is effective in target tracking in a background clutter with slowly varying spatial intensity. However, a background clutter with fast varying clutter spatial intensity could mean the need to compute optimum bandwidth frequently. This could add to computational complexity in radars with higher sampling rates.

The method in [46], [49], and [50] assumes a time-invariant clutter intensity function, which is estimated using cumulative measurements collected over some window of time. Furthermore, it assumes that the number of target measurements is sparse compared to clutter measurements. Given $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m\}$, the cumulative set of measurements in the period $[1, L]$, and a set C of components, the clutter intensity therein is defined as the product of the average number of clutter measurements and the spatial density function, which can be expressed as

$$\lambda(\mathbf{z}_j) = \hat{\lambda} f(\mathbf{z}_j \mid C). \tag{16}$$

Here, $\hat{\lambda}$ denotes the estimate of the average number of clutter measurements, $f(\mathbf{z}_j \mid C)$ denotes the spatial probability density function, and $\lambda(\mathbf{z}_j)$ denotes the clutter intensity at a measurement $\mathbf{z}_j$ of $\mathbb{Z}$. The spatial probability density function is represented by an N-component finite mixture model (FMM) [47] as

$$f(\mathbf{z}_j \mid C) = \sum_{i=1}^{N} w_j f(\mathbf{z}_j \mid c_i) \qquad \ni \sum_{i=1}^{N} w_j = 1, \quad (17)$$

where w_j denotes the weight of each component and $f(\mathbf{z}_j \mid c_i)$ denotes the likelihood of the measurement $\mathbf{z}_j$ with each component c_i. Gaussian mixture models were adopted in the said paper; hence, the likelihood $f(\mathbf{z}_j \mid c_i)$ is Gaussian with parameters $\theta_i = \{\mu_i, \Sigma_i\}$, wherein μ_i and Σ_i represent the mean and covariance of component c_i, respectively. Note that an implicit one-to-one mapping between measurements and components can be seen therein. Finally, $\hat{\lambda}$ and the parameters of components in (16) are estimated as follows. If $M_1, M_2, \ldots, M_L$ represent the number of measurements in each sampling time in $[1, L]$ and assuming each M_l follow a Poisson distribution, then $\hat{\lambda}$ can be given by an ML estimate expressed as

$$\hat{\lambda} = \frac{1}{L} \sum_{l=1}^{L} M_l. \quad (18)$$

Moving on, if $f(\mathbb{Z} \mid \Theta)$ given in (19) denotes the likelihood function of a parameter set, Θ, of components and by assuming the measurements in set $\mathbb{Z}$ are independent of each other, then

$$f(\mathbb{Z} \mid \Theta) = \prod_{j=1}^{m} \sum_{i=1}^{N} w_i f(\mathbf{z}_j \mid c_i) \quad (19)$$

the ML estimate for Θ can be derived by

$$\hat{\Theta} = \underset{\Theta}{argmax} \left\{ log f(\mathbb{Z} \mid \Theta) \right\}. \quad (20)$$

An EM algorithm is proposed in the same paper to solve (20) iteratively. It has been shown in [46] that the number of target measurements should be negligible when compared to clutter measurements to obtain an unbiased estimate for clutter intensity. Hence, the method has no inherent mechanism to discriminate among clutter and target measurements.

A method for updating clutter intensity online as well is proposed in [51] as an extension to [46]. It has a mechanism to distinguish clutter measurements from target measurements, which is explained below. At first, a PHD [26] filter is run to update the target states. The measurements nearest to the target states are regarded as target measurements. Hence, a set of clutter measurements for the current sampling time can be determined. The update for $\hat{\lambda}$ for the current sampling time is computed using its value in the previous sampling time and the number of clutter measurements at the current sampling time [51]. The component weights, means, and covariances are also updated based on the obtained set of clutter measurements. The set of clutter measurements is divided into subsets of clutter measurements belonging to each of the components using a χ^2 test based on the previous estimates of components [51]. Then, the new component parameters and their weights are updated using the said subset of clutter measurements. It is shown in [46] that the EM algorithm proposed therein is susceptible to giving an underestimate or overestimate of the number of components and their corresponding parameters. These issues are also addressed in [51] by adding new components, wherein the component parameters are misestimated. However, the addition of components may lead to an explosion in the number of components beyond control. Hence, some pruning and merging methods need to be adopted to reduce the nonrelevant components.

V. CLUTTER SPATIAL INTENSITY ESTIMATORS BASED ON CLUTTER GENERATORS

The methods in this section assume some unknown background targets represented using auxiliary variables called clutter generators that generate the clutter measurements. Following the random finite set (RFS) theory [26], the clutter spatial intensity estimation problem then becomes an intensity estimation problem of clutter generators. In [27] and [28], a set of clutter generators $C = \{c_1, c_2, \ldots, c_m\}$ defined in space $\mathbb{C}$, which is disjoint from the target state space $\mathbb{X}$ and measurement space $\mathbb{M}$, is assumed. Furthermore, assuming the clutter process as a Poisson mixture process, an approximate Bayesian estimator based on a PHD filter is derived for computing the intensity of clutter generators. However, the PHD filter proposed in [27] and[28] is intractable [29] as the PHD update equation requires a summation over all the partitions of measurement set at the current time, which is combinatorial in nature and no practical implementation of PHD equations is provided therein.

In [32], two assumptions about the clutter generators are made toward simplifying the above-mentioned problem. The first assumption is that each clutter generator c_i generates only one clutter measurement z_i^C and the second is that the predicted clutter generator distribution can be assumed as a Poisson point process. Then, a tractable PHD filter for the estimation of the intensity of clutter generators is proposed. Similar assumptions of one-to-one mapping between clutter generators and measurements can be seen in [29]–[31]. In [32], the clutter process is assumed to be NHPP whose intensity function is approximated by a Gaussian mixture with unknown mean and covariance. If $Z_k = \{\mathbf{z}_{k,1}, \mathbf{z}_{k,2}, \ldots, \mathbf{z}_{k,n}\}$ is the set of d-dimensional measurements and $c = \{c_1, c_2, \ldots, c_m\}$ the set of clutter generators at current time k, then the intensity function at any

detection point $\mathbf{z}_{k,j}(j = 1, 2, \ldots, n)$ is given by [32]

$$c(\mathbf{z}_j) = \sum_{i=1}^{m} \frac{m_i C_1}{\sqrt{|\Sigma_i|}} \exp\left\{-\frac{1}{2}(\mathbf{z}_j - \mu_i)^T \Sigma_i^{-1}(\mathbf{z}_j - \mu_i)\right\}$$
$$= \sum_{i=1}^{m} m_i C_2 \exp\left\{-\frac{1}{2}(\mathbf{z}_j - \mu_i)^T \rho_i(\mathbf{z}_j - \mu_i)\right\}, \quad (21)$$

where $C_1 = \frac{1}{(2\pi)^{d/2}}$ and $C_2 = C_1\sqrt{|\rho_i|}$.

Here, the mean and covariance of the ith Gaussian component are $\mu_i \in R^d$ and $\Sigma_i \in R^{d\times d}$, respectively. Also, ρ_i denotes the precision matrix, i.e., the inverse of Σ_i, $\mathbf{z}_j$ denotes the detection point $\mathbf{z}_{k,j}$, and $m_i \in R^+$ represents the expected number of clutter measurements corresponding to the ith Gaussian component. Hereafter, the symbol $\mathbf{z}_j$ will be used in place of $\mathbf{z}_{k,j}$ for simplicity. Also, $c_i = \{m_i, \mu_i, \Sigma_i\}\,(i = 1, 2, \cdot, m)$ is the ith clutter generator. All the parameters of c_i are random. Hence, the clutter generator c_i is a random point process. Again, under the assumption of a one-to-one relation between clutter generators and measurements, the clutter generator parameters reduce to $c_i = \{\mu_i, \Sigma_i\}$, as shown in [32]. Also, a Gaussian likelihood function defined below is assumed for $\mathbf{z}_j$ given c_i [32]:

$$f(\mathbf{z}_j \mid c_i) = \frac{1}{(2\pi)^{d/2}\sqrt{|\Sigma_i|}} \exp\left\{\frac{1}{2}(\mathbf{z}_j - \mu_i)^T \Sigma_i^{-1}(\mathbf{z}_j - \mu_i\right\}. \quad (22)$$

Here, μ_i and Σ_i denote the position and extension of the clutter generator c_i, respectively. Using (21) and (22), the clutter spatial intensity estimation problem becomes an estimation of Gaussian random variable with unknown mean and covariance. Furthermore, using the assumption that the state evolution of targets is statistically independent of the state evolution of clutter generator, the prediction of target PHD is decoupled from the prediction of clutter generator PHD, and the prediction equations of [26] are applied to targets and clutter generators separately [32]. The predicted PHD $D^C_{k|k-1}(c)$ for clutter generators is given as [32]

$$D^C_{k|k-1}(c) = b^C_k(c) + \int f^C_{k|k-1}(c) D^C_{k|k}(c) dc. \quad (23)$$

Here, $b^C_k(c)$ denotes the PHD of newborn clutter generators, and $f^C_{k|k-1}(c)$ denotes the state transition function for clutter generators defined in (33) and (34) of [32]. On the other hand, given the measurement set Z^k, the update equations for posterior PHD $D_{k|k}(\mathbf{x})$ of targets and posterior PHD $D^C_{k|k}(c)$ of clutter generators derived in [32] are coupled as

$$D_{k|k}(\mathbf{x}) = p_M D_{k|k-1}(\mathbf{x}) + \sum_{\mathbf{z}_j \in Z^k} \frac{P_D f(\mathbf{z}_j \mid \mathbf{x})}{L(\mathbf{z})} D_{k|k-1}(\mathbf{x})$$

$$D^C_{k|k}(c) = \sum_{\mathbf{z}_j \in Z^k} \frac{f(\mathbf{z}_j \mid c)}{L(\mathbf{z})} D^C_{k|k-1}(c). \quad (24)$$

Here, $p_M = 1 - P_D$ and $L(\mathbf{z})$ are the predicted pseudo-likelihoods for the measurement set Z^k defined as

$$L(\mathbf{z}) = P_D \int D_{k|k-1}(\mathbf{x}) f(\mathbf{z} \mid \mathbf{x}) \mathbf{dx} + \int D^C_{k|k-1}(c) f(\mathbf{z} \mid c) dc. \quad (25)$$

The pseudo-likelihood function $L(\mathbf{z})$ consists of two parts. The first part $P_D \int D_{k|k-1}(\mathbf{x}) f(\mathbf{z} \mid \mathbf{x}) \mathbf{dx}$ represents the target-originated measurements and the second part $\int D^C_{k|k-1}(c) f(\mathbf{z} \mid c) dc$ represents the clutter measurements. It is evident from (25) that [32] does not need a separate filter to identify target-originated measurements. Using the idea of translated point process [43], $L(\mathbf{z})$ can be viewed as a sum of two independent translated Poisson processes: the first one denoted by $P_D \int D_{k|k-1}(\mathbf{x}) f(\mathbf{z} \mid \mathbf{x}) \mathbf{dx}$ in (25) is a translated Poisson process (using basic assumption of [26]) of predicted targets with point transition density $P_D f(\mathbf{z} \mid \mathbf{x})$ and the second one represented by $\int D^C_{k|k-1}(c) f(\mathbf{z} \mid c) dc$ in (25) with point transition density $f(\mathbf{z} \mid c)$. Thus, the predicted clutter spatial intensity at a detection point $\mathbf{z}_j$ can be given by [29]

$$\hat{\lambda}(\mathbf{z}_j) = \int f(\mathbf{z} \mid c) D^C_{k|k-1}(c) dc. \quad (26)$$

Finally, using the fact that a normal-Wishart distribution is the conjugate prior of Gaussian distribution with unknown mean and covariance [34], a practical implementation for the recursion of $D^C_{k|k-1}(c)$ and $D^C_{k|k}(c)$ is shown in [32] using a normal-Wishart mixture. Similar derivations of PHD recursions for the intensity of clutter generator can be seen in [29]. It has been shown in [32] that the number of normal-Wishart mixture components grows out of bound; hence, pruning and merging techniques similar to a Gaussian mixture probability hypothesis density (GMPHD) filter [37] has to be adopted for practical implementation. In the same paper, a zero-velocity model for the dynamic evolution of clutter generators is adopted. However, in sea-based radars, the radial velocity of clutter is usually high due to the presence of moving waves on the sea surface. The radial velocity of these waves can be significant depending on the sea state that is influenced by various weather and wind conditions [45]. Hence, an accurate modeling for the clutter random process used in track-before-detect (TBD) filters [44] could be adopted to improve the accuracy of clutter spatial intensity estimate.

In [33], an extension to [32] is presented in order to use the output of standard MTTs like the joint integrated probabilistic data association (JIPDA) [10] in deriving the clutter generator PHD. The update equation (24) for clutter generator PHD $D^C_{k|k}(c)$ in [32] requires a predicted PHD $D_{k|k-1}(\mathbf{x})$ of targets. However, the standard MTTs like JIPDA readily provide the state estimates of targets as well as the association events that identify target-originated measurements. Hence, the output

of JIPDA is used in [33] to derive a recursion for PHD of clutter generators. In that paper, the posterior target state outputs from standard trackers are approximated by multiBernoulli RFS (MBe RFS) [35], [36]. In addition, the possible interaction between the clutter generator and the target during prediction is neglected. Then, by using the association events and their corresponding probabilities generated by JIPDA, an approximate Bayesian clutter intensity estimator is obtained. The posterior target state pdf at time $k-1$ obtained from legacy trackers is an MBe RFS of the form [33]

$$\Pi_{k-1} = \left\{ r^i_{k-1}, p_{k-1}(\mathbf{x}_{k-1}) \right\}_{i=1}^{M_{k-1}}. \tag{27}$$

Here, r^i_{k-1} denotes the posterior target existence probability, $p_{k-1}(\mathbf{x}_{k-1})$ denotes the posterior pdf of the ith target state, and M_{k-1} denotes the cardinality of targets. Then, the predicted target state pdf at time k is also an MBe RFS, which can be used to generate association events. If each association event χ_i with probability $P(\chi_i)$ generates a set $Z^k(W_{\chi_i})$ of clutter measurements, then, using the similar assumptions of [32], the posterior PHD $D^C_{k|k}(c)$ of clutter generator can be represented by [33]

$$D^C_{k|k}(c) = \sum_{\chi_i} P(\chi_i) \left\{ \sum_{\mathbf{z}_j \in Z^k(W_{\chi_i})} \frac{f(\mathbf{z}_j \mid c) D^C_{k|k-1}(c)}{\int f(\mathbf{z} \mid c) D^C_{k|k-1}(c) dc} \right\}. \tag{28}$$

Here, $f(\mathbf{z}_j \mid c)$ is the likelihood function in (22). The PHD prediction for clutter generators as well as the predicted clutter spatial intensity remains the same as (23) and (26), respectively.

The methods in [32] and [33] are different from the kernel estimation-based method [24] in two aspects. The KDE-based method is a nonparametric estimation method, and as the number of measurements increases, the bandwidth of kernels decreases. In addition, no two kernels are merged at any point of time. On the other hand, the approximate Bayesian methods of [32] and [33] are parametric methods wherein the clutter area is a summation of ellipses (normal-Wishart mixture geometry). Also, the clutter generator components are merged for practical implementation. If the shape of the clutter area is close to the summation of ellipses, then [32] and [33] can give better results than [24] with fewer observations [33]. However, if the shape of the clutter area is significantly different from the summation of ellipses, then the methods in [32] and [33] can be biased.

In [39], a GMPHD-based [37] clutter spatial intensity estimator called interactive clutter measurement density estimator (ICMDE) is proposed, assuming a Poisson point process for clutter generators. Here, the assumption of one-to-one mapping of clutter generator to measurements is retained, but the likelihood function for clutter generator is assumed to be Gaussian with known covariance. Also, a nearly constant velocity model is assumed for the dynamic state evolution of clutter generators. Then, using the above assumptions, the spatial intensity function becomes a Gaussian mixture with known covariance for each component. The intensity estimation problem then reduces to the estimation of Gaussian random variable with a known covariance and unknown mean. Again, using the fact that the conjugate prior of Gaussian distribution with known covariance is also Gaussian [34], GMPHD-based recursions for the PHD of clutter generator are presented therein. The covariance required for GMPHD recursion is calculated based on the sparsity [22] of each measurement, as shown in [39]. Furthermore, ICMDE utilizes the calculated clutter probability of measurements (11) using the tracker output to distinguish target-originated measurements from clutter measurements while evaluating the sparsity of each of the measurements. Hence, ICMDE can be easily integrated to existing MTTs [10], [12]. In [40], a multiscan version of ICMDE is explained. The assumption of known covariance for the likelihood function could make it an attractive choice for practical implementation when compared to methods in [32] and [33]. However, the problem of finding the nearest neighbor detections in nonhomogenous measurement space for the computation of sparsity exists in [39]. The solutions mentioned in [42] and [38] could be adopted to circumvent it.

The nonhomogenous clutter intensity is estimated in [36] and [48] based on clutter generators but by dropping the standard Poisson assumption for clutter process. Therein, the clutter generators are modeled analogous to actual targets with separate models for births, death processes, and transition density. Furthermore, assuming that the targets and clutter generators are independent, MBe RFS recursions of [35] are applied to targets and clutter generators separately. Since clutter generators are modeled with separate models for evolution over time, both the stationary and the nonstationary clutters can be addressed by the said method.

VI. DISCUSSION ON INTERRELATION BETWEEN METHODS AND CONCLUSION

The clutter spatial intensity estimators discussed in the above sections have been categorized into three groups. The categories are decided based on the techniques used and on the assumptions made while computing the spatial intensity of clutter. The first type of methods only compute the spatial intensity of measurements in the validation gate of tracks, assuming uniform spatial intensity inside it. The second group of methods compute the spatial intensity of clutter at any detection point in measurement space. In those methods, some variation of nonparametric density estimation techniques is used to evaluate the spatial intensity of false alarms. The third group of methods assume auxiliary variables called clutter generators, disjoint from target space and measurement space. Then, using RFS theory, PHD recursion equations are used to derive the intensity for clutter generators, thereby evaluating the clutter spatial intensity

Table I
Clutter Spatial Intensity Estimators Under Different Categories

Category	Technique used	Assumptions	References
Methods based on track validation gate	-	Uniform clutter spatial intensity in validation gate	[1], [12], [14]
Methods for any detection point in measurement space	Nonparametric density estimation	1) Constant clutter spatial intensity in each cell. 2) Homogeneous Poisson process in the near proximity of measurement. 3) Nonhomogeneous clutter spatial intensity. 4) FMM approximation.	1) [11], [18], [20] 2) [18], [21]–[23], [38] 3) [24] 4) [46], [49]–[51]
Methods based on clutter generator	PHD filter/MBe RFS	1) Multiple measurements per clutter generator. 2) One measurement per clutter generator.	1) [27], [28]. 2) [29]–[33], [36], [39], [40], [48]

Table II
Pros and Cons of Clutter Spatial Intensity Estimation Methods

Method	Pros	Cons
Nonparametric clutter intensity estimation	Uniform clutter intensity for all measurements in track(s) validation gate.	1) Different clutter intensities for the same measurement in different track validation gates. 2) Unable to distinguish between target and clutter measurements.
Nonparametric clutter intensity estimation with track perceivability	1) Uniform clutter intensity for all measurements in track(s) validation gate. 2) Target measurements are distinguished from clutter measurements using track perceivability.	Different clutter intensities for the same measurement in different track validation gates.
Cluttermap method, spatial clutter map method	Nonhomogenous clutter intensity estimate obtained by dividing measurement space into cells.	1) Naive method with uniform intensity for all measurements in a cell. 2) Intensity estimate obtained has block nature and biased. 3) Unable to distinguish between target and clutter measurements.
SCMDE	1) Clutter intensity estimate for any measurement in measurement space is obtained without dividing the measurement space into cells. 2) Nonhomogenous clutter intensity estimate with local homogeneity.	1) Clutter intensity estimate obtained has block nature. 2) Unable to explicitly distinguish between target and clutter measurements. 3) Not suitable for MTT. 4) Not directly suitable for sensors with nonhomogenous measurement space.
MTTSCMDE	1) Nonhomogenous clutter intensity estimate, which in unbiased. 2) Clutter probability of measurements aids in identifying target measurements. 3) Suitable for MTT.	1) Not directly suitable for sensors with nonhomogenous measurement space. 2) Intensity estimate obtained may have block nature.
Kernel density clutter intensity estimation	1) Efficient method with unbiased and smooth clutter intensity estimate for any measurement. 2) Suitable for MTT and all type of sensors.	Efficiency comes at the cost of computational complexity of KDE.
Online FMM method	1) Recursively updates clutter intensity. 2) Discriminates between target and clutter measurements. 3) Addresses the shortcoming of estimates given by EM algorithm.	1) Nearest neighbor method to identify target measurements is not guaranteed to give correct measurements corresponding to a target. 2) Pruning and merging methods required to stop explosion of number of components. 3) Clutter intensity is assumed time invariant; hence, no transition densities are defined for components.
Integrated clutter intensity estimation	1) Recursively updating clutter intensity estimate. 2) Ease of integration with legacy trackers like JIPDA.	Use of static transition density for clutter generators.
ICMDE	1) Recursively updating clutter intensity estimate. 2) Ease of integration with legacy trackers like JIPDA and LMIPDA.	Not directly suitable for sensors with nonhomogenous measurement space.

of measurements. Table I shows the classification of the spatial intensity estimators under different categories.

The methods in [1], [14], and [12] are similar in the way they compute the clutter spatial intensity of track(s) validated measurements. However, the former has no means to discriminate between clutter and target measurements, whereas the latter two use (4) to distinguish target measurements from clutter measurements. The clutter map method and spatial clutter map estimator are based on a multivariate histogram, whereas SCMDE and MTTSCMDE are based on a k-nearest density estimation technique. The former three methods have no provision to identify target measurements, while the latter uses the clutter probability of each measurement for this purpose, hence a better choice above others in multitarget tracking. The methods similar to k-nearest neighbor density estimation use the mathematical distance to the nearest measurement in evaluating the clutter spatial intensity. They are forced to make a homogeneous assumption for clutter spatial intensity in and around the desired measurement as nonhomogeneous assumptions do not hold for the inverse of the defined mathematical distance. Moreover, they are not easily adaptable to nonlinear sensors like Doppler radar. The method based on KDE is a more general approach applicable to any type of sensors. However, the effectiveness of the approach comes at the cost of computational load of KDE.

One-to-one mapping between clutter generators and measurements can be seen throughout the methods of category three, but the clutter intensity filter recursion implementations vary among each other. The methods in [32] and [33] use a normal-Wishart mixture implementation to account for unknown mean and covariance of Gaussian mixture, whereas a Gaussian mixture implementation is used in [40] to accommodate for the known covariance. However, the former two methods cannot be directly integrated to linear multitarget (LM) tracker [12] variants, whereas the latter can be seamlessly integrated to both LM variants and JIPDA trackers. The method in [32] requires the predicted PHD of targets as shown in (24), whereas the method in [33] uses the output of JIPDA tracker to represent target PHD and then the clutter PHD is updated using the association events. Finally, the pros and cons of each of the methods are tabulated in Table II.

ACKNOWLEDGMENT

The authors would like to thank the management of Bharat Electronics Ltd. and Central Research Laboratory for their support in the work. They would also like to extend their gratitude to Mr. Pardhasaradhi Bethi for his useful contribution in editing.

REFERENCES

[1] Y. Bar-Shalom and X. R. Li
Multitarget-Multisensor Tracking: Principles and Techniques. Storrs, CT, USA: YBS Publications, 1985.

[2] S. Blackman and R. Popoli
Design and Analysis of Modern Tracking Systems. Norwood, MA, USA: Artech House, 1999.

[3] S. Gould, R. Hartley, K. Mele, S. H. Rezatofighi, B. N. Vo, and B. T. Vo
"Multi-target tracking with time-varying clutter rate and detection profile: application to time-lapse cell microscopy sequences,"
IEEE Trans. Med. Imag., vol. 34, no. 6, pp. 1336–1348, Jun. 2015.

[4] V. Bharti, D. E. Clark, E. Delande, and I. Schlangen
"Joint multi-object and clutter rate estimation with the single-cluster PHD filter,"
in *Proc. IEEE 14th Int. Symp. Biomed. Imag.*, Apr. 2017, pp. 1087–1091.

[5] W. An, Y. Long, and R. Zhu
"Multitarget tracking based on PHD smoother with unknown clutter spatial density,"
Prog. Electromagnetics Res., vol. 82, pp. 123–133, 2018.

[6] D. J. Daley and D. V. Jones
An Introduction to the Theory of Point Processes: Elementary Theory of Point Processes, New York, NY, USA: Springer, 2003.

[7] J. Kingman
Poisson Processe, (Oxford Studies in Probability). Oxford, U.K.: Oxford Univ. Press, 1993.

[8] D. Read
"An algorithm for tracking multiple targets,"
IEEE Trans. Autom. Control, vol. 24, no. 6, pp. 843–854, Dec. 1979.

[9] Y. Bar-Shalom, T. Fortman, 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.

[10] R. Evans and D. Musicki
"Joint integrated probabilistic data association: JIPDA,"
IEEE Trans. Aerosp. Electron. Syst., vol. 40, no. 3, pp. 1093–1099, Jul. 2004.

[11] R. Evans and D. Musicki
"Clutter map information for data association and track initialization,"
IEEE Trans. Aerosp. Electron. Syst., vol. 40. no. 2, pp. 387–398, Jul. 2004.

[12] D. Musicki and B. L. Scala
"Multi-target tracking in clutter without measurement assignment,"
IEEE Trans. Aerosp. Electron. Syst., vol. 44, no. 3, pp. 877–896, Jul. 2008.

[13] M. R. Morelande and D. Musicki
"Gate volume estimation for target tracking," in *Proc. 7th Int. Conf. Inf. Fusion*, 2004.

[14] N. Li and X. R. Li
"Integrated real-time estimation of clutter density for tracking,"
IEEE Trans. Signal Process., vol. 48, no. 10, pp. 2797, Oct. 2000.

[15] N. Li and X. R. Li
"Target perceivability and its applications,"
IEEE Trans. Signal Process., vol. 46, no. 11, pp. 2588–2604, Nov. 2001.

[16] R. Evans, D. Musicki, and S. Stankovic
"Integrated probabilistic data association (IPDA),"
in *Proc. 31st IEEE Conf. Decis. Control*, 1992, pp. 3796–3798.

[17] Y. Bar-Shalom, S. S. Blackman, and R. J. Fitzgerald
"Dimensionless score function for multiple hypothesis tracking,"
IEEE Trans. Aerosp. Electron. Syst., vol. 43, no. 1, pp. 392–400, May 2007.

[18] B. Moran, M. Morelande, D. Musicki, and S. Suvorova

"Clutter map and target tracking,"
in *Proc. 7th Int. Conf. Inf. Fusion*, pp. 8, 2005.

[19] B. W. Silverman
Density Estimation for Statistics and Data Analysis, London, U.K.: CRC Press, 1986.

[20] T. Hanselmann, D. Musicki, and M. Palaniswami
"Adaptive target tracking in slowly changing clutter,"
in *Proc. 9th Int. Conf. Inf. Fusion*, 2006, pp. 1–8.

[21] D. Musicki and T. L. Song
"Adaptive clutter measurement density estimation for improved target tracking,"
IEEE Trans. Aerosp. Electron. Syst., vol. 47, no. 2, pp. 1457–1466, Apr. 2011.

[22] S. Y. Chong, H. J. Kim, S. H. Park, and T. L. Song
"Adaptive estimation of spatial clutter measurement density using clutter measurement probability for enhanced multi-target tracking," *Sensors*, vol. 20, no. 1, pp. 114, 2020.

[23] D. H. Hant, S. H. Park, T. L. Song, and Y. Xie
"Spatial clutter measurement density estimation with the clutter probability for improving multi-target tracking performance in cluttered environments,"
in *Proc. Int. Conf. Control, Autom. Inf. Sci.*, 2018 pp. 90–95.

[24] X. Chen, T. Kirubarajan, M. Mcdonald, and R. Tharmarasa
"Online clutter estimation using a Gaussian kernel density estimator for multitarget tracking,"
Institution Eng. Technol. Radar, Sonar Navigation, vol. 9, no. 1, pp. 1–9, Jan. 2015.

[25] L. Wasserman
All of Nonparametric Statistics, New York, NY, USA: Springer, 2006.

[26] R. Mahler
"Multitarget Bayes filtering via first-order multitarget moments,"
IEEE Trans. Aerosp. Electron. Syst., vol. 39, no. 4, pp. 1152–1178, Oct. 2003.

[27] R. Mahler
"CPHD and PHD filters for unknown backgrounds I: Dynamic data clustering," *Proc. SPIE*, vol. 7330, May 2009, Art. no. 73300K.

[28] R. Mahler
"CPHD and PHD filters for unknown backgrounds II: Multitarget filtering in dynamic clutter," *Proc. SPIE*, vol. 7330, May 2009, Art. no. 73300L.

[29] A. El-Fallah and R. Mahler
"CPHD and PHD filters for unknown backgrounds, part III: tractable multitarget filtering in dynamic clutter," *Proc. SPIE*, vol. 7698, Apr. 2010, Art. no. 76980F.

[30] A. El-Fallah and R. Mahler
"CPHD filtering with unknown probability of detection," *Proc. SPIE*, vol. 7697, Apr. 2010, Art. no. 76970F.

[31] R. Mahler, B. N. Vo, and B. T. Vo
"CPHD filtering with unknown clutter rate and detection profile,"
IEEE Trans. Signal Process., vol. 59, no. 8, pp. 3497–3513, Aug. 2011.

[32] X. Chen, T. Kirubarajan, M. Pelletier, and R. Tharmarasa
"Integrated clutter estimation and target tracking using Poisson point processes,"
IEEE Trans. Aerosp. Electron. Syst., vol. 48, no. 2, pp. 1210–1235, Apr. 2012.

[33] X. Chen, T. Kirubarajan, M. Pelletier, and R. Tharmarasa
"Integrated bayesian clutter estimation with JIPDA/MHT trackers,"
IEEE Trans. Aerosp. Electron. Syst., vol. 49, no. 1, pp. 395–414, Jan. 2013.

[34] M. H. Degroot
Optimal Statistical Decisions, Hoboken, NJ, USA: Wiley, 2005.

[35] A. Cantoni, B. N. Vo, and B. T. Vo
"The cardinality balanced multi-target multi-Bernoulli filter and its implementations,"
IEEE Trans. Signal Process., vol. 57, no. 2, pp. 409, Oct. 2008.

[36] R. Hoseinnezhad, R. Mahler, B. N. Vo, and B. T. Vo
"Multi-Bernoulli filtering with unknown clutter intensity and sensor field-of-view,"
in *Proc. 45th Annu. Conf. Inf. Sci. Syst.*, May 2011, pp. 1–6.

[37] W. K. Ma and B. N. Vo
"The Gaussian mixture probability hypothesis density filter,"
IEEE Trans. Signal Process., vol. 54, no. 11, pp. 4091–4104, Oct. 2006.

[38] W. C. Kim and T. L. Song
"Spatial clutter measurement density estimation in nonhomogeneous measurement spaces," *Proc. 18th Int. Conf. Inf. Fusion*, Sep. 2015, pp. 1772–1777.

[39] W. C. Kim and T. L. Song
"Interactive clutter measurement density estimator for multi-target data association,"
Institution Eng. Technol. Radar, Sonar Navigation, vol. 11, no. 1, pp. 125–132, Jan. 2017.

[40] J. S. Bae, W. C. Kim, D. Musicki, and T. L. Song
"A multi scan clutter density estimator,"
in *Proc. 16th Int. Conf. Inf. Fusion*, 2013, pp. 707–713.

[41] J. Moller and R. P. Waagepetersen
Statistical Inference and Simulation for Spatial Point Processes, Boca Raton, FL, USA: Chapman & Hall, 2003.

[42] D. Musicki
"Doppler-aided target tracking in heavy clutter,"
in *Proc. 13th Int. Conf. Inf. Fusion*, 2010, pp. 1–7.

[43] M. Miller and D. L. Snyder
Random Point Processes in Time and Space, New York, NY, USA: Springer, 1991.

[44] S. P. Ebenezer
Multiple Radar Target Tracking in Environments With High Noise and Clutter, Tempe, AZ, USA: Arizona State University, 2015.

[45] R. J. Tough, D. K. Ward, and S. Watts
Sea Clutter: Scattering, the K Distribution and Radar Performance, London, U.K.: The Institution of Engineering and Technology, 2006.

[46] C. Han, F. Han, and W. Liu
"Estimating unknown clutter intensity for PHD filter,"
IEEE Trans. Aerosp. Electron. Syst., vol. 46, no. 4, (Oct. 2010), pp. 2066–2078.

[47] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake
"Finite mixture models,"
Annu. Rev. Statist. Appl., vol. 6, no. 1, pp. 355–378, Mar. 2019.

[48] S. He, H. S. Shin, and A. Tsourdos
"Joint probabilistic data association filter with unknown detection probability and clutter rate," *Sensors*, vol. 18, no. 1, Jan. 2018.

[49] C. Han, F. Han, and N. Lv
"Unknown clutter estimation by FMM approach in multitarget tracking algorithm," *Math. Problems Eng.*, vol. 2014 , pp. 1–11, Mar. 2014.

[50] R. Zhu, Y. Long, and W. An
"Multitarget tracking based on PHD smoother with unknown clutter spatial density,"
Prog.Electromagnetics Res., vol. 82, pp. 123–133, Jan. 2018.

[51] C. Cui and Y. Gong
"Offline estimation and online update algorithm of clutter intensity in multi-target tracking,"
in *Proc. 5th Int. Conf. Comput. Commun. Syst.*, pp. 848–853, 2020.

Fahad A. M. received the Master's degree in communication systems engineering from the National Institute of Technology, Surat, India, in 2014.

In 2015, he joined the Central Research Laboratory, Bharat Electronics Ltd., Bengaluru, India, and he is currently a Member (Research Staff). His research interests include multitarget tracking, multisensor fusion, and radar bias estimation.

Saritha D. S. received the Master's degree in computer network engineering.

In 2008, she joined the Central Research Laboratory, Bharat Electronics Ltd., Bengaluru, India, and she is currently a Senior Member (Research Staff). Her research interests include device driver development for embedded systems, radar display software development, multitarget tracking using radar, and radar measurement simulator development.

In 2009, she was bestowed with the Research and Development Excellence award by Bharat Electronics.

Full State Information Transfer Across Adjacent Cameras in a Network Using Gauss Helmert Filters

RONG YANG
YAAKOV BAR-SHALOM

This paper develops three-dimensional (3D) Cartesian tracking algorithms for a high-resolution wide field of view (FOV) camera surveillance system. This system consists of a network linking multiple narrow FOV cameras side-by-side looking at adjacent areas. In such a multi-camera system, a target usually appears in the FOV of one camera first, and then shifts to an adjacent one. The tracking algorithms estimate target 3D positions and velocities dynamically using the angular information (azimuth and elevation) provided by multiple cameras. The target state (consisting of position and velocity) is not fully observable when it is detected by the first camera only. Once it moves into the FOV of the next camera, the state can then be fully estimated. The main challenge is how to transfer the state information from the first camera to the next one when the target moves across cameras. In this paper, we develop an approach, designated as Cartesian state estimation with full maximum likelihood information transfer (fMLIT), to cope with this challenge. Since the fMLIT consists of an implicit state relationship, the conventional Kalman-like filters (for explicit constraints) are not suitable. We then develop three Gauss–Helmert filters, and test them with simulation data.

Manuscript received August 20, 2021; revised November 11, 2021; released for publication July 7, 2022

Refereeing of this contribution was handled by David Crouse.
Authors' addresses: R. Yang, DSO National Laboratories, 12 Science Park Drive, Singapore 118225 (E-mail: yrong@dso.org.sg); Y. Bar-Shalom, Department of ECE, University of Connecticut, Storrs, CT 06269, USA (E-mail: yaakov.bar-shalom@uconn.edu).

I. INTRODUCTION

As more and more cameras are used in surveillance systems, intensive research and development works have been conducted on target detection and tracking using cameras. Most of them deal with extended targets (such as people or vehicles) in near range. After illegal drone intrusions were frequently reported, the camera surveillance has been extended to the more challenging applications which track small air targets in a relatively far range. This paper will focus on developing appropriate algorithms to track small air targets in three-dimensional (3D) Cartesian space using a network of cameras.

In the earlier stage, the computer vision research focused on detection of targets from images, and association of the detections from the same targets over frames based on various features (color, shape, edge, etc.) [9], [16], [18]. Later, it was extended target tracking over video frames to help the track continuity when the detection was imperfect [3], [19]. Further extensions focused on estimation of the target location and trajectory in Cartesian 3D space instead of the two-dimensional (2D) image space. A direct conversion (from 2D image to 3D Cartesian space) can be applied when the camera projection matrix and the target range are available [8], [17]. Another approach is to estimate target range from the ratio of target size and image size (assuming both are known). This is actually an old technique called Stadiametric range finding. Recently, artificial intelligence techniques were introduced to this approach for better accuracy [5].

However, a small air target in a relatively far range (the focus of this paper) has only a few pixels in a video frame. The target image size and range information (with sky background) cannot be obtained accurately. It is hard to obtain the target locations or trajectories in 3D Cartesian space using the above-mentioned techniques. Triangulation from multiple cameras is then a suitable method. In [11], a drone location is estimated by the measured azimuths and elevations from multiple cameras, assuming the cameras are widely spaced and detect a target simultaneously.

In this paper, we consider a realistic camera deployment as shown in Fig. 1. It is a camera network with multiple high-resolution narrow field of view (NFOV) stationary cameras located side by side. Each of them covers a small area and overlaps with the neighbors a little. In this system, a target is detected by one camera only most of time, and the triangulation cannot be performed always. This paper will focus on developing appropriate approaches to track targets in 3D Cartesian space even for cameras with non-overlapping regions. In such a multi-camera system, a target usually appears in the field of view (FOV) of one camera first, and then shifts to an adjacent one. The proposed tracking algorithms will estimate target 3D positions and velocities dynamically using the angular information (azimuth and elevation) provided by multiple cameras. The target state

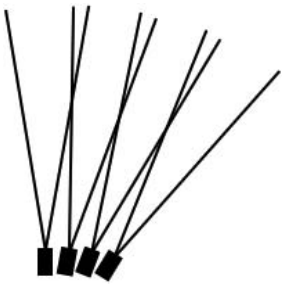


Fig. 1. A wide field of view surveillance system consisting of four NFOV cameras.

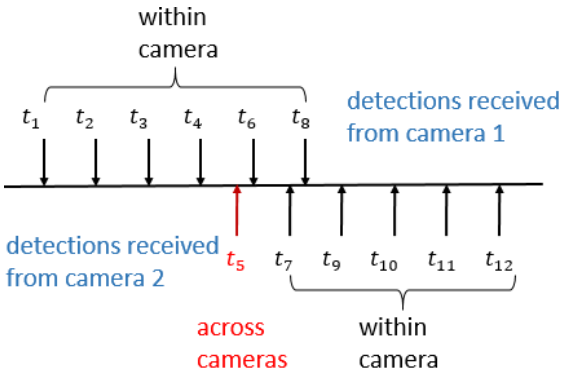


Fig. 2. A target flies from the FOV of camera 1 to FOV of camera 2. The across-camera event (crossover) occurs at time t_5 when camera 2 detects the target for the first time. The others are considered as within-camera events.

(consisting of position and velocity) is not fully observable when the target is detected by the first camera only. Once the target moves into the FOV of the next camera, the state can then be estimated completely using suitable algorithms. The main challenge is how to transfer the useful state information from the first camera to the next one when the target moves across cameras. We will formulate the problem with an unique Gauss–Helmert model (GHM), which provides an implicit constraint that is used to transfer full state information (including 3D position and velocity) across cameras. The approach is designated as Cartesian state estimation with full maximum likelihood information transfer (fMLIT).

A conventional dynamic estimation problem is formulated by two basic models, the state transition model and the measurement model. Usually these two models are in explicit forms as

$$\mathbf{x}(t_k) = \mathbf{f}[\mathbf{x}(t_{k-1})] + \mathbf{v}(t_k, t_{k-1}), \tag{1}$$

$$\mathbf{z}(t_k) = \mathbf{h}[\mathbf{x}(t_k)] + \mathbf{w}(t_k), \tag{2}$$

where $\mathbf{x}(t_k)$ is the state vector to be estimated at time t_k, $\mathbf{z}(t_k)$ is the measurement vector observed by a sensor, $\mathbf{f}[\cdot]$ and $\mathbf{h}[\cdot]$ are the state transition function and measurement function, respectively, and $\mathbf{v}(\cdot)$ and $\mathbf{w}(\cdot)$ are the process noise and measurement noise, respectively. Based on these models, the Kalman-like filters has the following two generic steps to perform estimation:

- Predict the state from time t_{k-1} to t_k using the state transition model (1). The predicted state is $\hat{\mathbf{x}}(t_k|t_{k-1})$.
- Update the predicted state $\hat{\mathbf{x}}(t_k|t_{k-1})$ by the measurement $\mathbf{z}(t_k)$ using the measurement model (2). The updated state is $\hat{\mathbf{x}}(t_k|t_k)$.

However, to perform fMLIT across cameras, an implicit constraint needs to be taken into consideration in addition to the measurement model given in (2) (the details will be given later in the Section II-B). The implicit relationship is between the predicted state $\hat{\mathbf{x}}(t_k|t_{k-1})$ from the previous camera and the current state $\mathbf{x}(t_k)$. Thus, the following GHM is used to replace the measurement model in (2) at the crossover time (details to be given later—see (20)).

$$\mathbf{g}[\mathbf{x}(t_k), \hat{\mathbf{x}}(t_k|t_{k-1}), \mathbf{z}(t_k)] = \mathbf{w}_{\mathrm{g}}, \tag{3}$$

where $\mathbf{w}_{\mathrm{g}}$ is a small zero-mean Gaussian model error. Thus the Kalman-like filters cannot be applied, and need a further development.

The GHM is commonly used for similarity estimation in geodetic science [10], [12], and is also applied to computer vision and curve fitting [4], [7], [13], [14], [15]. It was introduced in dynamic estimation for solving unknown propagation delay problem in the state transition model in [20], [21]. The Unscented GHF (UGHF) was developed to solve the problem. A number of works were conducted for various applications with implicit state transition models [22], [23], [24].

The above mentioned Gauss–Helmert filter (GHF) and its applications use the GHM in their state transition models. The problem in this paper requires an implicit measurement model. Consequently, we will discuss three GHFs and three other algorithms which totally or partially ignore the implicit constraints in the GHM. Their performance will be demonstrated using simulation data.

The rest of paper is structured as following. Section II formulates the problem. Section III develops the six estimation algorithms. Section IV presents the simulation results, and Section V draws the conclusions.

II. PROBLEM FORMULATION

To track a target using the camera suite shown in Fig. 1, we need to formulate the dynamic estimation problem with the state transition model and measurement model within and across cameras, respectively. Fig. 2 illustrates the within and across camera events. When a target flies from the FOV of camera 1 to the FOV of camera 2, it is detected by camera 1 first at times t_1, t_2, t_3, t_4, t_6, and t_8. Camera 2 detects the target at times $t_5, t_7, t_9, t_{10}, t_{11}$, and t_{12}. The across camera event occurs at time t_5 when the target is firstly detected by camera 2. All other occurrences are considered as within-camera events, even they fall in overlapping FOVs. The problem formulate is then based on these two events.

A. Dynamic Estimation Models for the Within-Camera Event

The state vector to be estimated at time t_k is defined as

$$\mathbf{x}(t_k) = [x(t_k)\ y(t_k)\ z(t_k)\ \dot{x}(t_k)\ \dot{y}(t_k)\ \dot{z}(t_k)]'. \quad (4)$$

It consists of the position and velocity components of a target in 3D Cartesian coordinates. The measurement vector from the ith camera (with $i = 1, \ldots, n$, and n is the total number of cameras in the system) is

$$\mathbf{z}_{s_i}(t_k) = [a_{s_i}(t_k)\ e_{s_i}(t_k)]', \quad (5)$$

where $a_{s_i}(t_k)$ is the measured (noisy) azimuth from true North clockwise, and $e_{s_i}(k)$ is the measured elevation up from the horizontal, with the reference to the ith camera position. The state transition model uses the nearly constant velocity (NCV) model [2] as

$$\mathbf{x}(t_k) = \mathbf{F}\mathbf{x}(t_{k-1}) + \mathbf{v}(t_k, t_{k-1}), \quad (6)$$

where

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & T(t_k) & 0 & 0 \\ 0 & 1 & 0 & 0 & T(t_k) & 0 \\ 0 & 0 & 1 & 0 & 0 & T(t_k) \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (7)$$

with time interval

$$T(t_k) = t_k - t_{k-1}, \quad (8)$$

and $\mathbf{v}(t_k, t_{k-1})$ is the white Gaussian process noise with covariance

$$\mathbf{Q}(t_k, t_{k-1}) = \begin{bmatrix} \frac{T(t_k)^3}{3} & 0 & 0 & \frac{T(t_k)^2}{2} & 0 & 0 \\ 0 & \frac{T(t_k)^3}{3} & 0 & 0 & \frac{T(t_k)^2}{2} & 0 \\ 0 & 0 & \frac{T(t_k)^3}{3} & 0 & 0 & \frac{T(t_k)^2}{2} \\ \frac{T(t_k)^2}{2} & 0 & 0 & T(t_k) & 0 & 0 \\ 0 & \frac{T(t_k)^2}{2} & 0 & 0 & T(t_k) & 0 \\ 0 & 0 & \frac{T(t_k)^2}{2} & 0 & 0 & T(t_k) \end{bmatrix} q, \quad (9)$$

and q is the Cartesian acceleration power spectral density (PSD). The measurement model is

$$\mathbf{z}_{s_i}(t_k) = \mathbf{h}[\mathbf{x}(t_k), \mathbf{x}_{s_i}] + \mathbf{w}(t_k) \quad (10)$$

with

$$\mathbf{h}[\mathbf{x}(t_k), \mathbf{x}_{s_i}] = \begin{bmatrix} \operatorname{atan}\left[\frac{x(t_k) - x_{s_i}(t_k)}{y(t_k) - y_{s_i}(t_k)}\right] \\ \operatorname{atan}\left[\frac{z(t_k) - z_{s_i}(t_k)}{\sqrt{[x(t_k) - x_{s_i}(t_k)]^2 + [y(t_k) - y_{s_i}(t_k)]^2}}\right] \end{bmatrix}, \quad (11)$$

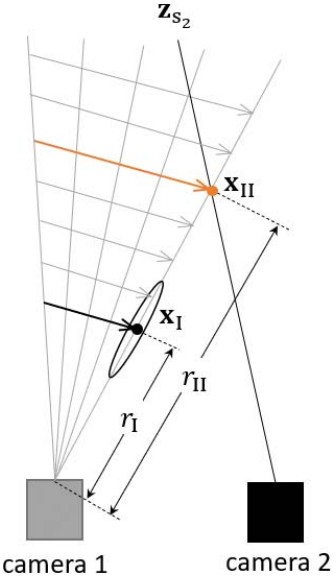


Fig. 3. The ground truth is at $\mathbf{x}_{II}$. The estimated state from camera 1 is $\mathbf{x}_I$. The large estimate bias of $\mathbf{x}_I$ is because the state is not observable from the measurements from one camera.

where $\mathbf{x}_{s_i} = [x_{s_i}\ y_{s_i}\ z_{s_i}]'$ is the position of the ith camera, which provides the measurement, and $\mathbf{w}(t_k)$ is the white Gaussian measurement noise with covariance

$$\mathbf{R} = \operatorname{diag}(\sigma_a^2\ \sigma_e^2), \quad (12)$$

with σ_a and σ_e, which are the measurement error standard deviations of azimuth and elevation, respectively.

To apply these models to our problem directly one has the following issue. The state is not fully observable when a target is detected by one camera only, as the target range cannot be obtained. Fig. 3 shows that the possible target trajectories are a set of parallel lines (assuming the target is in constant velocity motion, and does not head to or move away from the camera directly) [6]. All these parallel trajectories share the same heading, and the speeds on parallel lines are proportional to their ranges (the distances to the camera). The proof of the target heading observability and range unobservablity from a single stationary camera, shown in Fig. 3, is given in Appendix A. The state estimate, using this incompletely observable models, can fall on any one of possible trajectories depending on the initial condition. The initial state estimate is usually set at a particular point along the line of sight (LOS) of the first detection in 3D space and zero velocities with predefined large range and velocity errors. When the state estimate is updated by the angular measurements, it is adjusted to suitable range and speed to match the angular change. This process can lead to a biased estimation for this unobservable problem, namely, the ground truth can fall outside the state uncertainty region. Fig. 4 and its enlarged version Fig. 5 show an estimation example. There are two cameras located at $(0, 0, 50)$ m and $(1, 0, 50)$ m, respectively. The target is in the FOV of the first camera at beginning where the target trajectory is not observable, and moves to the

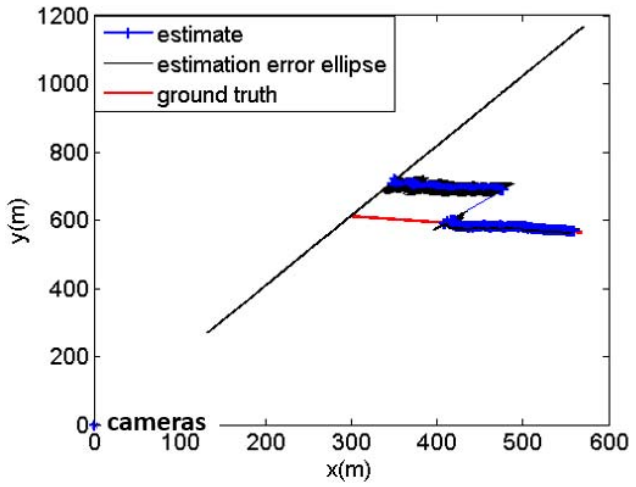


Fig. 4. Estimation from angle-only measurements using two adjacent cameras (here the error ellipses are very elongated).

FOV of the second camera. The track has an initial state with large error in range to cover the ground truth. After the second time cycle, the filter-calculated estimation error is much smaller, and the state converges to a trajectory (blue) parallel to the ground truth (red). Obviously, the estimation is seen to be biased in range. Using a very large uncertainty for the initial state does not solve the problem.

While we cannot make the state observable in this situation, we can transfer all useful information obtained from the first camera to the subsequent estimation. A Gauss–Helmert (GH)-based information transfer can make the state fully observable from the across-camera event (given in the next subsection).

After the across-camera event, the estimation is carried out continuously in the FOV of camera 2. Although the state is still not fully observable in theory (namely, the measurements are from camera 2 only), the bias is, in spite of the marginal observability, reduced significantly after the full information transfer (see Figs. 4 and 5).

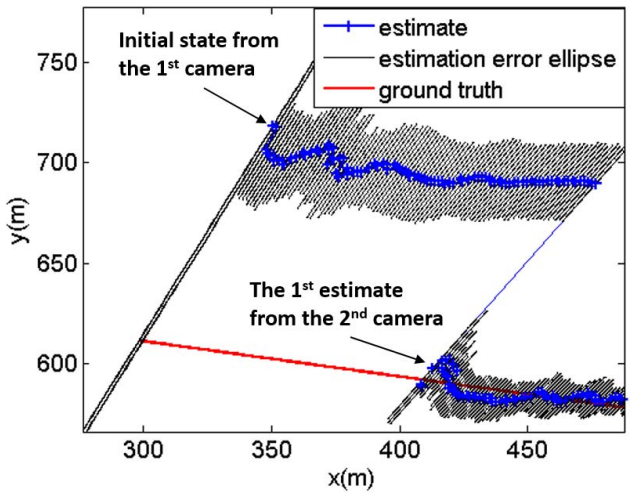


Fig. 5. An enlarged version of Fig. 4.

B. Dynamic Estimation Models for the Across-Camera Event

At the across-camera event, the state vector and the state transition model are the same as those of the within-camera event given in (4) and (6). Assuming the across-camera event happens at time t_k, we predict the state estimate $\hat{\mathbf{x}}(t_{k-1}|t_{k-1})$ and its error covariance $\mathbf{P}(t_{k-1}|t_{k-1})$ to time t_k by

$$\hat{\mathbf{x}}(t_k|t_{k-1}) = \mathbf{F}\hat{\mathbf{x}}(t_{k-1}|t_{k-1}), \tag{13}$$

$$\mathbf{P}(t_k|t_{k-1}) = \mathbf{F}\mathbf{P}(t_{k-1}|t_{k-1})\mathbf{F}' + \mathbf{Q}(t_k, t_{k-1}). \tag{14}$$

To simplify the expression, we denote the predicted state and its error covariance as

$$\mathbf{x}_{\mathrm{I}} = \hat{\mathbf{x}}(t_k|t_{k-1}), \tag{15}$$

$$\mathbf{P}_{\mathrm{I}} = \mathbf{P}(t_k|t_{k-1}), \tag{16}$$

and the state at t_k (see Fig. 3) as

$$\mathbf{x}_{\mathrm{II}} = \mathbf{x}(t_k). \tag{17}$$

Intuitively, the predicted state $\mathbf{x}_{\mathrm{I}}$ can be updated to $\mathbf{x}_{\mathrm{II}}$ by the second camera measurement $\mathbf{z}_{s_2}(t_k)$ based on the measurement model giving in (10). However, it faces the following issues. First, the predicted state $\mathbf{x}_{\mathrm{I}}$ can be significantly biased compared to the ground truth due to the unobservability in the range. Fig. 3 shows an example that the estimated range from the first camera is r_{I}, but the actual range is r_{II}. The error covariance $\mathbf{P}_{\mathrm{I}}$ does not include camera 2's measurement $\mathbf{z}_{s_2}(t_k)$ in its uncertainty region. The Kalman-like filters cannot bring the state $\mathbf{x}_{\mathrm{I}}$ close to $\mathbf{x}_{\mathrm{II}}$ enough with $\mathbf{z}_{s_2}(t_k)$, as the filters are designed to reduce the random error, and are not effective for a large bias error. Second, the state information estimated by the first camera is not fully utilized. The headings in $\mathbf{x}_{\mathrm{I}}$ and $\mathbf{x}_{\mathrm{II}}$ should be consistent, and the speeds should be proportional to their ranges. Thus, we reformulate the problem to estimate $\mathbf{x}_{\mathrm{II}}$ from a known augmented parameter $\mathbf{y}$ with error covariance $\mathbf{P}_{\mathrm{y}}$

$$\mathbf{y} = [\mathbf{x}_{\mathrm{I}}' \; \mathbf{z}_{s_2}(t_k)']', \tag{18}$$

$$\mathbf{P}_{\mathrm{y}} = \begin{bmatrix} \mathbf{P}_{\mathrm{I}} & \mathbf{0}_{6\times 2} \\ \mathbf{0}_{2\times 6} & \mathbf{R} \end{bmatrix}. \tag{19}$$

Since $\mathbf{x}_{\mathrm{II}}$ and $\mathbf{y}$ have an implicit relationship, as shown below in (20)–(31), a GH measurement model is defined as

$$\mathbf{g}(\mathbf{x}_{\mathrm{II}}, \mathbf{y}) = \mathbf{w}_{\mathrm{g}}, \tag{20}$$

which is the same as (3), where

$$\mathbf{g}(\cdot) = [g_1(\cdot)\; g_2(\cdot)\; g_3(\cdot)\; g_4(\cdot)\; g_5(\cdot)\; g_6(\cdot)\; g_7(\cdot)\; g_8(\cdot)]', \tag{21}$$

with

$$g_1(\cdot) = r_{\mathrm{I}}(x_{\mathrm{II}} - x_{s_1}) - r_{\mathrm{II}}(x_{\mathrm{I}} - x_{s_1}), \tag{22}$$

$$g_2(\cdot) = r_{\rm I}(y_{\rm II} - x_{s_1}) - r_{\rm II}(y_{\rm I} - x_{s_1}), \tag{23}$$

$$g_3(\cdot) = r_{\rm I}(z_{\rm II} - x_{s_1}) - r_{\rm II}(z_{\rm I} - x_{s_1}), \tag{24}$$

$$g_4(\cdot) = r_{\rm I}\dot{x}_{\rm II} - r_{\rm II}\dot{x}_{\rm I}, \tag{25}$$

$$g_5(\cdot) = r_{\rm I}\dot{y}_{\rm II} - r_{\rm II}\dot{y}_{\rm I}, \tag{26}$$

$$g_6(\cdot) = r_{\rm I}\dot{z}_{\rm II} - r_{\rm II}\dot{z}_{\rm I}, \tag{27}$$

$$g_7(\cdot) = \arctan\left(\frac{x_{\rm II} - x_{s_2}}{y_{\rm II} - y_{s_2}}\right) - a_{s_2}(t_k), \tag{28}$$

$$g_8(\cdot) = \arctan\left(\frac{z_{\rm II} - z_{s_2}}{\sqrt{(x_{\rm II} - x_{s_2})^2 + (y_{\rm II} - y_{s_2})^2}}\right) - e_{s_2}(t_k), \tag{29}$$

and

$$r_{\rm I} = \sqrt{(x_{\rm I} - x_{s_1})^2 + (y_{\rm I} - y_{s_1})^2 + (z_{\rm I} - z_{s_1})^2}, \tag{30}$$

$$r_{\rm II} = \sqrt{(x_{\rm II} - x_{s_1})^2 + (y_{\rm II} - y_{s_1})^2 + (z_{\rm II} - z_{s_1})^2}. \tag{31}$$

This model considers three constraints, first, the position $(x_{\rm II}, y_{\rm II}, z_{\rm II})$ in $\mathbf{x}_{\rm II}$ and the position $(x_{\rm I}, y_{\rm I}, z_{\rm I})$ in $\mathbf{x}_{\rm I}$ are on the same LOS with reference to camera 1. This is described in (22)–(24). Second, the velocities $(\dot{x}_{\rm II}, \dot{y}_{\rm II}, \dot{z}_{\rm II})$ and $(\dot{x}_{\rm I}, \dot{y}_{\rm I}, \dot{z}_{\rm I})$ are proportional to their ranges with reference to camera 1, which is given in (25)–(27). Third, the position $(x_{\rm II}, y_{\rm II}, z_{\rm II})$ is on the LOS of camera 2's measurement, as given in (28)–(29). The third constraint is the within-camera measurement model given in (10). The GH measurement model adds the constraints in (22)–(27). This ensures that the useful state information obtained from camera 1 is fully transferred to the subsequent estimation in a proper manner.

III. DYNAMIC ESTIMATION ALGORITHMS

This section presents six dynamic estimation algorithms for the problem addressed in this paper:

- EKF: extended Kalman filter without implicit constraint;
- pMLIT: partial maximum likelihood information transfer;
- fMLIT: full maximum likelihood information transfer without considering cross-correlation;
- fMLIT-EGHF: full maximum likelihood information transfer using extended Gauss–Helmert filter;
- fMLIT-UGHF: full maximum likelihood information transfer using unscented Gauss-Helmert filter; and
- fMLIT-UGHFapp: full maximum likelihood information transfer using unscented Gauss-Helmert filter with approximation.

Like all dynamic estimation algorithms, they consist of two steps, prediction and update. The six algorithms share the same prediction step given in (13) and (14). The update step is also the same for within-camera estimation, which follows the extended Kalman filter (EKF)[1] as

$$\hat{\mathbf{x}}(t_k|t_k) = \hat{\mathbf{x}}(t_k|t_{k-1}) + \mathbf{W}(t_k)\nu(t_k), \tag{32}$$

$$\mathbf{P}(t_k|t_k) = \mathbf{P}(t_k|t_{k-1}) - \mathbf{W}(t_k)\mathbf{S}(t_k)\mathbf{W}(t_k)', \tag{33}$$

where

$$\nu(t_k) = \mathbf{z}_{s_i}(t_k) - \mathbf{h}[\hat{\mathbf{x}}(t_k|t_{k-1}), \mathbf{x}_{s_i}], \tag{34}$$

$$\mathbf{W}(t_k) = \mathbf{P}(t_k|t_{k-1})\mathbf{H}(t_k)\mathbf{S}(t_k)^{-1}, \tag{35}$$

$$\mathbf{S}(t_k) = \mathbf{R} + \mathbf{H}(t_k)\mathbf{P}(t_k|t_{k-1})\mathbf{H}(t_k)', \tag{36}$$

and

$$\mathbf{H}(t_k) = \left.\frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{x}_{s_i})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}(t_k|t_{k-1})}, \tag{37}$$

with $\mathbf{h}(\cdot)$ given in (11). The differences between the six algorithms are at the across-camera update step. This is the main focus of this section, and will be given next.

A. Extended Kalman Filter without Implicit Constraint

This algorithm updates the target across camera state using the measurement model given in (10). The implicit constraints given in (22)–(27) are totally ignored. Its update step is according to the EKF given in (32)–(33).

As we discussed in Section II, this algorithm suffers from large errors due to unobservability. We include it here as baseline for comparison purpose. This algorithm may be in use in some real applications due to its easy implementability. It is worth to evaluate its performance to understand its limitations.

B. Partial Maximum Likelihood Information Transfer (pMLIT)

This algorithm updates the across-camera state using the partial information in the predicted state $\mathbf{x}_{\rm I}$ (position only). It converts the predicted position to the LOS information (azimuth and elevation) with reference to the first camera, and then combines with the LOS measurement from the second camera to estimate the position in $\mathbf{x}_{\rm II}$ using the maximum likelihood (ML) estimation. The velocity in $\mathbf{x}_{\rm I}$ is not utilized, and the velocity in $\mathbf{x}_{\rm II}$ is not estimated. The details are given in sequel.

[1] The EKF is selected for within-camera estimation. This is because the camera measurement error is small. The other nonlinear filters like unscented Kalmen filter and particle filter cannot improve estimation accuracy when the measurement errors are small. Furthermore, the EKF is more efficient, and is good for the high sampling/measurement rate of a camera.

First, the LOS and its error covariance from the first camera are computed by

$$\mathbf{z}_{\mathrm{IA}} = [a_{\mathrm{I}}\ e_{\mathrm{I}}]' = \mathbf{h}(\mathbf{x}_{\mathrm{I}}, \mathbf{x}_{\mathrm{s}_1}), \tag{38}$$

$$\mathbf{R}_{\mathrm{IA}} = \mathbf{H}(t_k)\mathbf{P}_{\mathrm{I}}\mathbf{H}(t_k)', \tag{39}$$

where $\mathbf{h}(\cdot)$ and $\mathbf{H}(\cdot)$ are given in (10) and (37), respectively.

Second. the position in $\mathbf{x}_{\mathrm{II}}$ is estimated. The LOS of the second camera is the measurement

$$\mathbf{z}_{\mathrm{s}_2}(t_k) = [a_{\mathrm{s}_2}(t_k)\ e_{\mathrm{s}_2}(t_k)]'. \tag{40}$$

The iterated least squares (ILS) [1] is used to obtain the ML estimate of the target position by the model

$$\mathbf{z}_{\mathrm{p}} = \mathbf{h}_{\mathrm{p}}(\mathbf{x}_{\mathrm{p}}, \mathbf{x}_{\mathrm{s}_1}, \mathbf{x}_{\mathrm{s}_2}) + \mathbf{w}_{\mathrm{p}}, \tag{41}$$

where

$$\mathbf{z}_{\mathrm{p}} = [\mathbf{z}'_{\mathrm{IA}}\ \mathbf{z}_{\mathrm{s}_2}(t_k)']', \tag{42}$$

$$\mathbf{x}_{\mathrm{p}} = \mathbf{x}_{\mathrm{II}}(1:3) = [x_{\mathrm{II}}\ y_{\mathrm{II}}\ z_{\mathrm{II}}]', \tag{43}$$

$$\mathbf{h}_{\mathrm{p}}[\mathbf{x}_{\mathrm{p}}, \mathbf{x}_{\mathrm{s}_1}, \mathbf{x}_{\mathrm{s}_2}] = \begin{bmatrix} \arctan\left(\frac{x_{\mathrm{II}}-x_{\mathrm{s}_1}}{y_{\mathrm{II}}-y_{\mathrm{s}_1}}\right) \\ \arctan\left(\frac{z_{\mathrm{II}}-z_{\mathrm{s}_1}}{\sqrt{(x_{\mathrm{II}}-x_{\mathrm{s}_1})^2+(y_{\mathrm{II}}-y_{\mathrm{s}_1})^2}}\right) \\ \arctan\left(\frac{x_{\mathrm{II}}-x_{\mathrm{s}_2}}{y_{\mathrm{II}}-y_{\mathrm{s}_2}}\right) \\ \arctan\left(\frac{z_{\mathrm{II}}-z_{\mathrm{s}_2}}{\sqrt{(x_{\mathrm{II}}-x_{\mathrm{s}_2})^2+(y_{\mathrm{II}}-y_{\mathrm{s}_2})^2}}\right) \end{bmatrix}, \tag{44}$$

and $\mathbf{w}_{\mathrm{p}}$ is the error of $\mathbf{z}_{\mathrm{p}}$ with the covariance

$$\mathbf{R}_{\mathrm{p}} = \begin{bmatrix} \mathbf{R}_{\mathrm{IA}} & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \mathbf{R} \end{bmatrix}. \tag{45}$$

The position ML estimate is obtained by the ILS algorithm as

$$\hat{\mathbf{x}}_{\mathrm{p}}^{j+1} = \hat{\mathbf{x}}_{\mathrm{p}}^{j} + \mathbf{P}_{\mathrm{p}}^{j+1}(\mathbf{J}^j)'\mathbf{R}_{\mathrm{p}}^{-1}[\mathbf{z}_{\mathrm{p}} - \mathbf{h}_{\mathrm{p}}(\hat{\mathbf{x}}_{\mathrm{p}}^{j}, \mathbf{x}_{\mathrm{s}_1}, \mathbf{x}_{\mathrm{s}_2})], \tag{46}$$

$$\mathbf{P}_{\mathrm{p}}^{j+1} = [(\mathbf{J}^j)'\mathbf{R}_{\mathrm{p}}^{-1}\mathbf{J}^j]^{-1}, \tag{47}$$

where j is the iteration index, and

$$\mathbf{J}^j = \left.\frac{\partial \mathbf{h}_{\mathrm{p}}(\mathbf{x}_{\mathrm{p}}, \mathbf{x}_{\mathrm{s}_1}, \mathbf{x}_{\mathrm{s}_2})}{\partial \mathbf{x}_{\mathrm{p}}}\right|_{\mathbf{x}_{\mathrm{p}}=\hat{\mathbf{x}}_{\mathrm{p}}^{j}}. \tag{48}$$

The state $\mathbf{x}_{\mathrm{II}}$ estimate and its error covariance are then given by

$$\hat{\mathbf{x}}_{\mathrm{II}} = [\hat{\mathbf{x}}'_{\mathrm{p}}\ 0\ 0\ 0]', \tag{49}$$

$$\mathbf{P}_{\mathrm{II}} = \begin{bmatrix} \mathbf{P}_{\mathrm{p}} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathrm{diag}(\sigma_{\dot{x}}^2\ \sigma_{\dot{y}}^2\ \sigma_{\dot{z}}^2) \end{bmatrix}, \tag{50}$$

where $\sigma_{\dot{x}}$, $\sigma_{\dot{y}}$, and $\sigma_{\dot{z}}$ are predefined velocity error standard deviations in x, y, and z coordinates, respectively.

This algorithm can overcome the range bias problem in the EKF described in the Section III-A. It carries the estimated azimuth and elevation information from the first camera to the subsequent tracking process. However, the velocity estimates are lost, and the velocities in $\mathbf{x}_{\mathrm{II}}$ in (49) are set to 0 without the contribution from the previous state estimate. Thus the information from the previous state is only partially transferred to the subsequent tracking process.

C. Full Maximum Likelihood Information Transfer (fMLIT) without Considering Cross-Correlation

This algorithm estimates $\mathbf{x}_{\mathrm{II}}$ and its error covariance $\mathbf{P}_{\mathrm{II}}$ using the fMLIT. However, it ignores the cross-covariance between the position and velocity error. The details are given below.

First, the algorithm estimates the position in $\mathbf{x}_{\mathrm{II}}$ using the same method as the pMLIT described in the Section III-B.

Second, $\mathbf{x}_{\mathrm{II}}$ (including velocity estimation) and its error covariance $\mathbf{P}_{\mathrm{II}}$ are estimated by

$$\hat{\mathbf{x}}_{\mathrm{II}} = [\hat{\mathbf{x}}'_{\mathrm{p}}\ \lambda\mathbf{x}'_{\mathrm{I},(4:6)}]', \tag{51}$$

$$\mathbf{P}_{\mathrm{II}} = \begin{bmatrix} \mathbf{P}_{\mathrm{p}} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \lambda^2\mathbf{P}_{\mathrm{I},(4:6,4:6)} \end{bmatrix}, \tag{52}$$

where

$$\lambda = \frac{\hat{r}_{\mathrm{II}}}{r_{\mathrm{I}}}, \tag{53}$$

where r_{I} and $\hat{r}_{\mathrm{II}}$ are computed using (30) and (31), respectively (see Fig. 3).

This is an approximate solution to compute the velocity in $\mathbf{x}_{\mathrm{II}}$. The velocity error covariance is also approximately proportional with factor λ^2. The cross-covariance between the errors of the position and velocity is not taken into consideration, setting it to $\mathbf{0}_{3\times 3}$ in $\mathbf{P}_{\mathrm{II}}$.

D. Full Maximum Likelihood Information Transfer Using Extended Guass–Helmert Filter (fMLIT-EGHF)

This algorithm updates the across-camera state based on the GHM described in Section II-B. Since the GHM is nonlinear, we approximate the nonlinear transformation by the first-order Taylor expansion. Following the name of the EKF, using the same approximation approach, we designate the algorithm as extended GHF (EGHF).

The update step is performed by the following iteration

$$\mathbf{P}_{\mathrm{II}}^{j+1} = \left[(\mathbf{A}^j)'[\mathbf{B}^j\mathbf{P}_{\mathrm{y}}(\mathbf{B}^j)']^{-1}\mathbf{A}^j\right]^{-1,} \tag{54}$$

$$\hat{\mathbf{x}}_{\mathrm{II}}^{j+1} = \hat{\mathbf{x}}_{\mathrm{II}}^{j} - \mathbf{P}_{\mathrm{II}}^{j+1}(\mathbf{A}^j)'[\mathbf{B}^j\mathbf{P}_{\mathrm{y}}(\mathbf{B}^j)']^{-1}\mathbf{g}(\hat{\mathbf{x}}_{\mathrm{II}}^{j}, \mathbf{y}), \tag{55}$$

where j is the iteration index,

$$\mathbf{A}^j = \left.\frac{\partial \mathbf{g}(\mathbf{x}_{\rm II}, \mathbf{y})}{\partial \mathbf{x}_{\rm II}}\right|_{\mathbf{x}_{\rm II}=\hat{\mathbf{x}}_{\rm II}^j}, \tag{56}$$

$$\mathbf{B}^j = \left.\frac{\partial \mathbf{g}(\mathbf{x}_{\rm II}, \mathbf{y})}{\partial \mathbf{y}}\right|_{\mathbf{x}_{\rm II}=\hat{\mathbf{x}}_{\rm II}^j}. \tag{57}$$

The initial $\hat{\mathbf{x}}_{\rm II}^0$ is set as in (51). The Jaconbians $\mathbf{A}$ and $\mathbf{B}$ are given in Appendix B.

This algorithm is a solution for the implicit GHM given in the Section II-B, using the first-order Taylor expansion to approximate the nonlinear function (20). However, in addition to the well-known disadvantages of the first-order Taylor approximation, such as not being suitable for highly nonlinear models with large errors, the EGHF involves matrix inversion operations. This may cause numerical issues when the matrices are ill-conditioned. This will be discussed later in the simulation tests.

E. Full Maximum Likelihood Information Transfer using Unscented Gauss–Helmert Filter (fMLIT-UGHF)

This algorithm is also a GHF to provide a solution to the GH measurement model. To better approximate the nonlinear transformation, the unscented transform is used to replace the first-order Taylor expansion in the fMLIT-EGHF. The algorithm is designated as fMLIT-UGHF.

First, $(2n_{\rm y}+1)$ weighted sigma points of $\mathbf{y}$ are generated as

$$\mathbf{y}_1 = \mathbf{y}, \tag{58}$$

$$\mathbf{y}_i = \mathbf{y} + \left[\sqrt{(n_{\rm y}+\kappa)\mathbf{P}_{\rm y}}\right]_{i-1} \quad i = 2, \ldots, n_{\rm y}+1, \tag{59}$$

$$\mathbf{y}_i = \mathbf{y} - \left[\sqrt{(n_{\rm y}+\kappa)\mathbf{P}_{\rm y}}\right]_{i-n_{\rm y}-1} \quad i = n_{\rm y}+2, \ldots, 2n_{\rm y}+1. \tag{60}$$

Their corresponding weights are

$$w_1 = \frac{\kappa}{n_{\rm y}+\kappa}, \tag{61}$$

$$w_i = \frac{1}{2(n_{\rm y}+\kappa)} \quad i = 2, \ldots, 2n_{\rm y}+1, \tag{62}$$

where $n_{\rm y} = 8$ is the dimension of $\mathbf{y}$, and κ is a scalar to determine the spread of sigma points.

Second, the following iteration step is performed for each sigma point, so that the sigma points of $\mathbf{x}_{\rm II}$ are obtained.

$$\hat{\mathbf{x}}_{{\rm II},i}^{j+1} = \hat{\mathbf{x}}_{{\rm II},i}^{j} - [(\mathbf{A}^j)'\mathbf{A}^j]^{-1}(\mathbf{A}^j)'\mathbf{g}(\hat{\mathbf{x}}_{{\rm II},i}^{j}, \mathbf{y}_i). \tag{63}$$

The initial $\hat{\mathbf{x}}_{{\rm II},i}^0$ is set by (51) based on $\mathbf{y}_i$.

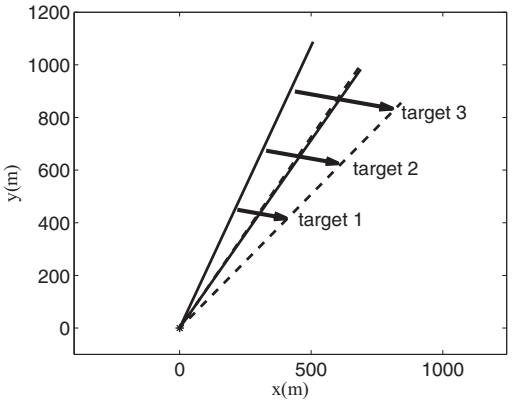


Fig. 6. Test scenario with two adjacent cameras and three targets (the two cameras, at 1 m apart, appear overlapped due to the necessary scale of the figure).

Third, the updated state and its error covariance are computed by the sigma points $\mathbf{x}_{{\rm II},i}$ with $i = 1, \ldots, 2n_{\rm y}+1$

$$\hat{\mathbf{x}}_{\rm II} = \sum_{i=1}^{2n_{\rm y}+1} w_i \hat{\mathbf{x}}_{{\rm II},i}, \tag{64}$$

$$\mathbf{P}_{\rm II} = \sum_{i=1}^{2n_{\rm y}+1} w_i (\hat{\mathbf{x}}_{{\rm II},i} - \hat{\mathbf{x}}_{\rm II})(\hat{\mathbf{x}}_{{\rm II},i} - \hat{\mathbf{x}}_{\rm II})'. \tag{65}$$

The fMLIT-UGHF still has inverse matrix operation, but less occurrence than the fMLIT-EGHF.

F. Full Maximum Likelihood Information Transfer using Unscented Guass–Helmert Filter with approximation (fMLIT-UGHFapp)

This algorithm is an approximation version of the fMLIT-UGHF. It removes the iteration process of the fMLIT-UGHF in (63). The sigma points of $\hat{\mathbf{x}}_{\rm II}$ are the initial ones $\hat{\mathbf{x}}_{{\rm II},i}^0$. If the initial sigma points are accurate enough, this approximation will not affect the final accuracy much. Furthermore, this approximation version has no matrix inversion operations, so the numerical issues due to the inversion of possibly ill-conditioned matrices are avoided. It also more efficient as it has no iteration process.

IV. SIMULATION RESULTS

This section evaluates the performance of the six algorithms described in Section III. The test scenario is shown in Fig. 6. Two adjacent cameras 1 and 2 are located at (0, 0, 50) m and (1, 0, 50) m, respectively, 1 m apart. Each camera has a field of view of 10° and 5.6° horizontal and vertical, respectively. The pointing angles of cameras 1 and 2 are 30° and 38° (clockwise from true North), respectively. Both of them are looking 2° up in elevation (from the horizontal). Their measurement error standard deviations of azimuth and elevation are $\sigma_b = \sigma_e = 0.046$ mrad, which is equivalent to 1 pixel

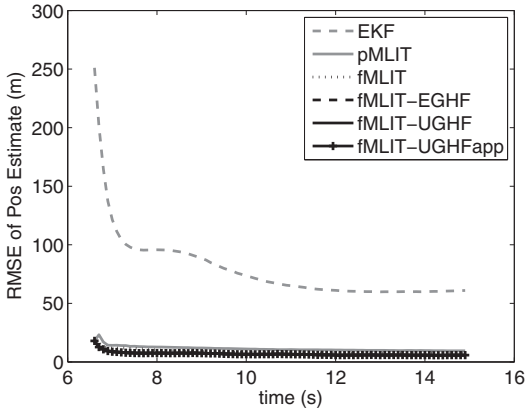


Fig. 7. The RMSE of position estimates versus time from 100 runs for the six algorithms after camera 2 starts to detect **target 1** (with starting range 500 m).

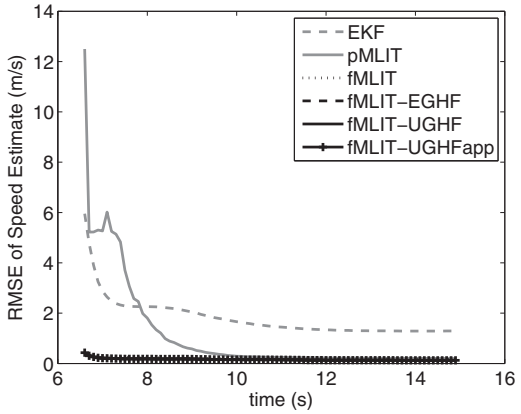


Fig. 8. The RMSE of speed estimates versus time from 100 runs for the six algorithms after camera 2 starts to detect **target 1** (with starting range 500 m).

(the cameras are assumed to have 8 megapixels). The two cameras provide their detections every 0.1 s. We simulate the targets 1, 2, and 3 with starting ranges at 500 m, 750 m, and 1,000 m, respectively. They move across the two cameras horizontally with the same heading 100° (clockwise from true North) and the same constant speed of 12.5 m/s. The observation durations for the targets 1, 2, and 3 are 15 s, 22 s, and 30 s, respectively. A near-range target has better observability than a far-range target, as the two LOSs from the two cameras have larger angle between them when the target crosses the cameras. So target 1 should have better estimation accuracy than target 2, and target 3 is the worst. The PSD in (9) is set as (see [2] for the units)

$$q_{\rm p} = 0.01^2\,{\rm m}^2/{\rm s}^3. \tag{66}$$

The velocity error standard deviations in (50) are set as

$$\sigma_{\dot{x}} = \sigma_{\dot{y}} = \sigma_{\dot{z}} = 10\,{\rm m/s}. \tag{67}$$

The scalar κ in (59)–(62) is set to 1. We will present the estimation accuracy using the root mean square errors (RMSE) of the position and speed (magnitude of the velocity vector), the statistical consistency analysis using the normalized estimation error squared (NEES) [1], and the algorithms' performances when the probability of detection (PD) is less than unity in sequel.

A. Estimation Accuracy

The position and velocity estimates from 100 Monte Carlo runs for each targets will be presented in this subsection.

First, we present the estimation accuracy of the six algorithms for target 1 with starting range 500 m. The position and speed estimate RMSEs versus time from 100 Monte Carlo runs are shown in Figs. 7 and 8, respectively. The time starts from the moment camera 2 starts detecting the target (since the state is not fully observable before that). It can be seen that the EKF and pMLIT are obviously worse than the other four fMLIT algorithms, as they do not take all constrains into consideration when the target crosses the cameras' FOVs. We remove these two algorithms from the performance figures, so that the other algorithms' performance can be enlarged and seen clearly. Since the EKF and pMLIT have poor performance clearly, their performance will not be presented in the sequel. The position and speed RMSEs versus time for the four fMLIT algorithms are shown in Figs. 9 and 10, respectively. We can see the fMLIT is worse than the other three algorithms. This is because the fMLIT is an approximate solution to compute the position and velocity separately, the cross-correlation of the estimation error between the position and velocity is totally ignored, whereas, the other three are solutions to the GH model. We do not observe much differences in accuracy among the fMLIT-EGHF, fMLIT-UGHF, and fMLIT-UGHFapp. It means the nonlinearity and errors in the problem are small enough, so that the first-order Taylor expansion (in the fMLIT-EGHF) and unscented transform (in the fMLIT-UGHF and fMLIT-UGHFapp) have very similar accuracy. Also, since target 1 is the nearest with the best observability, it does not cause numerical issues with ill-conditioned ma-

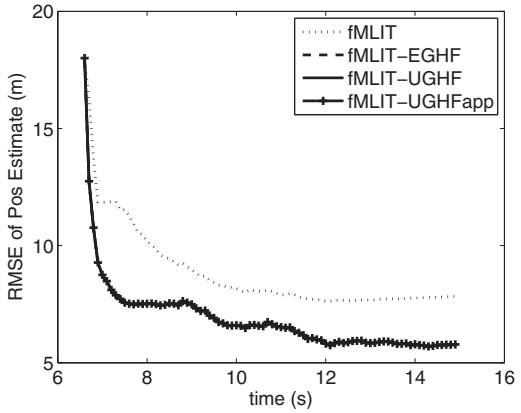


Fig. 9. The RMSE of position estimates versus time from 100 runs for the four fMLIT algorithms after camera 2 starts to detect **target 1** (with starting range 500 m).

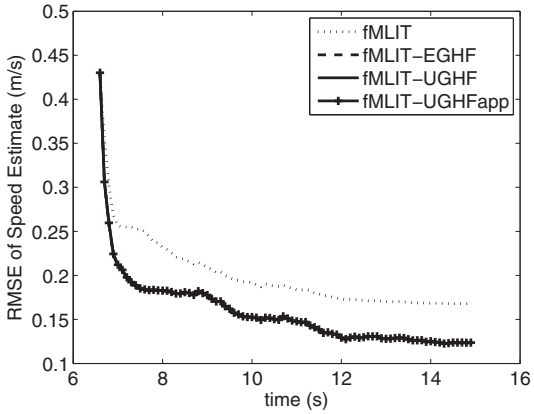


Fig. 10. The RMSE of speed estimates versus time from 100 runs for the four fMLIT algorithms after camera 2 starts to detect **target 1** (with starting range 500 m).

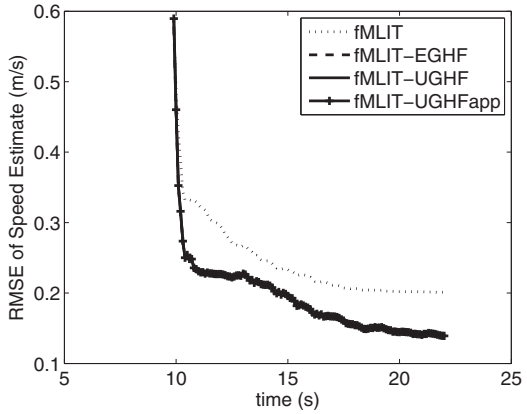


Fig. 12. The RMSE of speed estimates versus time from 100 runs for the four fMLIT algorithms after camera 2 starts to detect **target 2** (with starting range 750 m).

trices in the filtering process, so that the three GHFs all perform well. The similar results of the fMLIT-UGHF and fMLIT-UGHFapp also show the initial sigma points $\hat{\mathbf{x}}^0_{\mathrm{II,i}}$ in (63) are accurate enough, as similar accuracies are obtained with or without the iteration given in (63).

Second, we show the estimation accuracy of target 2 with starting range 750 m. The position and speed RMSEs versus time for the four fMLIT algorithms are shown in Figs. 11 and 12, respectively. Similarly to target 1, the fMLIT is worse than the other three GHFs, and the three GHFs have very similar results.

Next, we present the estimation performance for target 3 with starting range 1,000 m. Fig. 13 shows the position RMSEs versus time for the four fMLIT algorithms. It can be seen that the fMLIT-EGHF diverges. Divergence happened after the across-camera event in one run only. This is due to the state estimate error coveriance $\mathbf{P}_{\mathrm{II}}$ in (54), which is not positive-definite in the fMLIT-EGHF estimation. We investigated the condition number of $[\mathbf{B}\mathbf{P}_{\mathrm{y}}\mathbf{B}']$ in (54). It is 1×10^{18}. The inversion operation on such an ill-conditioned matrix encountered numerical issues, and caused $\mathbf{P}_{\mathrm{II}}$ to lose its positive definiteness. This resulted in estimation divergence subsequently. Target 3 is at the farthest range among the three targets, and its observability is marginal. The two LOSs of the two cameras are nearly parallel when the target crosses between the cameras. This leads to ill-conditioned matrices during the fMLIT-EGHF iteration process. We remove the fMLIT-EGHF, and show the position and speed estimate RMSEs for the remaining three algorithms in Figs. 14 and 15, respectively. It can be seen that the fMLIT-UGHF and fMLIT-UGHFapp have similar accuracy, and fMLIT is worse. We also investigate $\mathbf{P}_{\mathrm{II}}$ in fMLIT-UGHF. It is positive-definite. This is because $\mathbf{P}_{\mathrm{II}}$ is computed with sigma points using (65), and its positive-definiteness is maintained. However, the sigma points computation also has an inversion operation in (63). The condition number of the matrix $[\mathbf{A}'\mathbf{A}]$ to be inverted in (63) reached 1×10^{18}. Thus, the fMLIT-UGHF has a potential risk of numerical issues, although not as much as the fMLIT-EGHF.

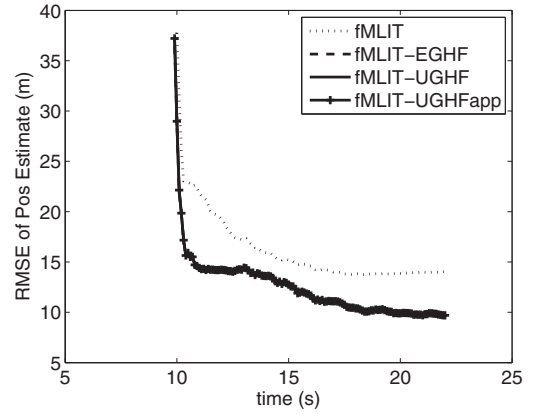


Fig. 11. The RMSE of position estimates versus time from 100 runs for the four fMLIT algorithms after camera 2 starts to detect **target 2** (with starting range 750 m).

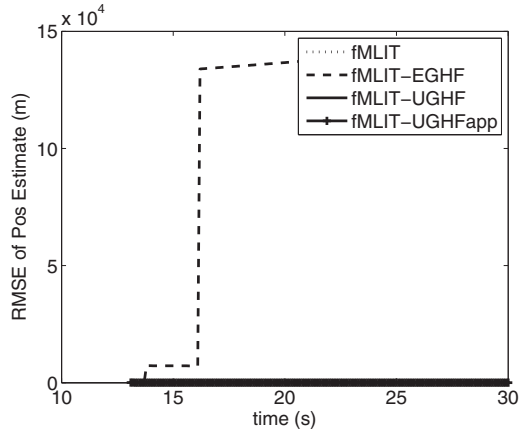


Fig. 13. The RMSE of position estimates versus time from 100 runs for the four fMLIT algorithms after camera 2 starts to detect **target 3** (with starting range 1,000 m).

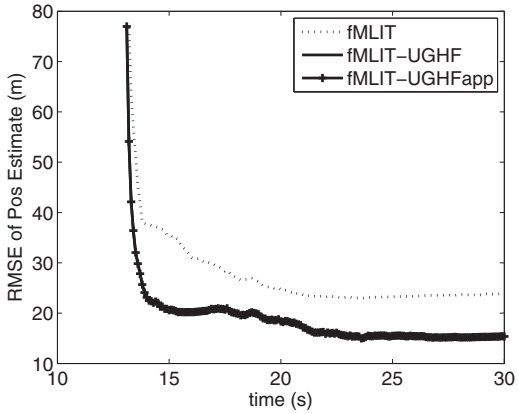


Fig. 14. The RMSE of position estimates versus time from 100 runs for the fMLIT, fMLIT-UGHF, and fMLIT-UGHFapp after camera 2 starts to detect **target 3** (with starting range 1,000 m).

B. Statistical Consistency Analysis

The statistical consistency analysis is conducted using the NEES [1] at the across-camera event and is computed by

$$\epsilon^i(t_k) = \tilde{\mathbf{x}}^i(t_k)' \mathbf{P}^i(t_k)^{-1} \tilde{\mathbf{x}}^i(t_k), \tag{68}$$

where $i = 1 \ldots 100$ is the run index,

$$\tilde{\mathbf{x}}^i(t_k) = \mathbf{x}(t_k) - \hat{\mathbf{x}}^i(t_k), \tag{69}$$

and $\hat{\mathbf{x}}^i(t_k)$ and $\mathbf{x}(t_k)$ are the ith run state estimate and ground truth at time t_k, respectively, assuming the target crosses between the cameras at time t_k. The NEESs of 100 runs are recorded. The NEES of the state (with dimension 6) is a 6 degrees of freedom chi-square random variable. Its two-sided 95% probability region is [1.24, 14.45]. The estimation is statistically consistent, if 95% of NEESs are within this interval.

Figs. 16–18 show the NEESs versus the run index at across-cameras for targets 1–3, respectively. The numbers of NEESs that are outside the 95% region for the four fMLIT algorithms are counted and listed in Table I. It can be seen, the fMLIT is not statically consistent for

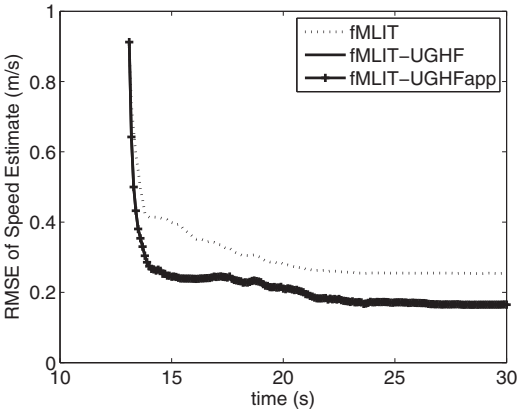


Fig. 15. The RMSE of speed estimates versus time from 100 runs for the fMLIT, fMLIT-UGHF, and fMLIT-UGHFapp after camera 2 starts to detect **target 3** (with starting range 1,000 m).

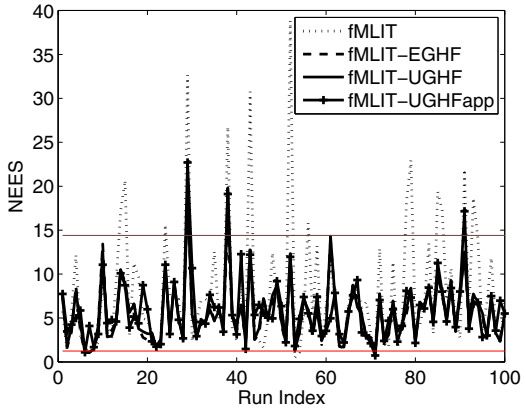


Fig. 16. The across-cameras NEES of **target 1** (with starting range 500 m) in 100 runs for the four fMLIT algorithms.

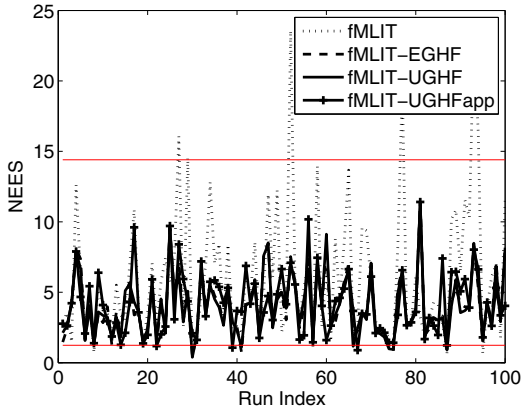


Fig. 17. The across-cameras NEES of **target 2** (with starting range 750 m) in 100 runs for the four fMLIT algorithms.

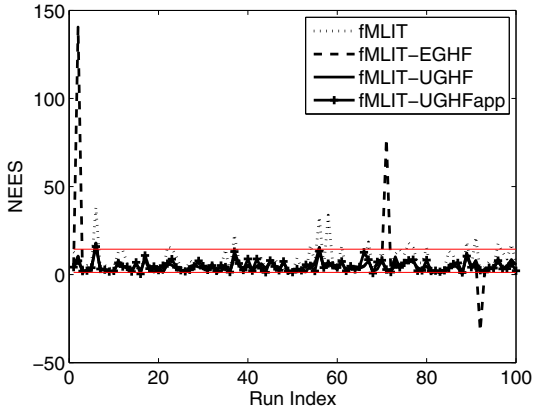


Fig. 18. The across-cameras NEES of **target 3** (with starting range 1,000 m) in 100 runs for the four fMLIT algorithms.

Table I
Number of NEES out of 95% boundary (100 runs)

Target	1	2	3
fMLIT	18	11	24
fMLIT-EGHF	5	5	11
fMLIT-UGHF	5	6	9
fMLIT-UGHFapp	5	4	4

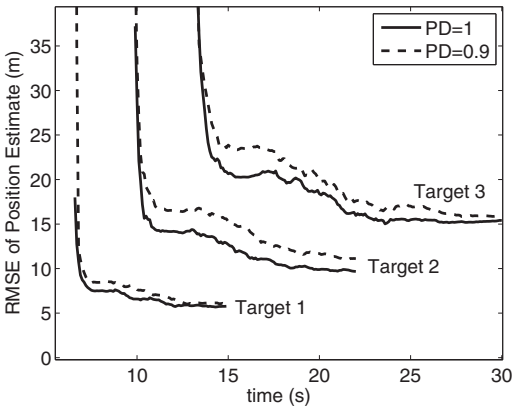


Fig. 19. RMSE of position estimate in 100 runs for the fMLIT-UGHF.

all three targets. The number of NEES outside the 95% region is 11 to 24, much larger than 5. For target 1, the three GHFs are statistically consistent, as they all have five NEESs outside the 95% region, i.e., they meet the criterion. For target 2, the three GHFs are also consistent. For target 3, only the fMLIT-UGHFapp is statistically consistent. The fMLIT-EGHF and fMLIT-UGHF do not meet the consistency criterion. The worse performance of the fMLIT-EGHF and fMLIT-UGHF is due to their numerical issues.

C. Performance under Imperfect Detection

The PD is usually imperfect in real camera applications. We investigate the estimation accuracy of the three GHFs (namely, fMLIT-EGHF, fMLTL-UGHF, and fMLTL-UGHFapp) when PD = 0.9 for the three targets shown in Fig. 6. The RMSE of position estimates in 100 Monte Carlo runs are studied. The fMLIT-EGHF diverges for all three targets when PD = 0.9. The results of the fMLIT-UGHF and fMLIT-UGHFapp are shown in Figs. 19 and 20, respectively. The corresponding RMSEs when PD = 1 are also shown in the figures for comparison. The position RMSEs increased 1–2 m when PD = 0.9 in both fMLIT-UGHF and fMLIT-UGHFapp. The fMLIT-UGHF and fMLIT-UGHFapp

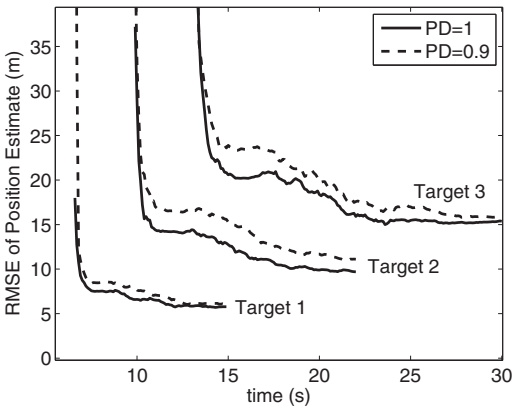


Fig. 20. RMSE of position estimate in 100 runs for the fMLIT-UGHFapp.

generate very similar results. Figs. 19 and 20 are shown separately because the results of the fMLIT-UGHF and fMLIT-UGHFapp would be indistinguishable on a combined figure at the same scale—they are, as in Figs. 9–15, practically the same.

In summary, it can be seen from the simulation results that the fMLIT-UGHFapp is the most robust algorithm. Its accuracy is similar to the fMLIT-UGHF and without numerical issues. Its estimation is consistent statistically. It also has lower computational cost than the fMLIT-UGHF as no iteration process is needed at each sigma point. The fMLIT-UGHF is good in theory, but is limited by numerical issues in practice. It is better than the fMLIT-EGHF which suffers from serious numerical problems. In general, the unscented transform (in the fMLIT-UGHF) should be better than the first-order Taylor approximation (in the fMLIT-EGHF) for a highly nonlinear model with large error. In this application, the nonlinearity and the degree of error do not affect their performance. However, we observed another disadvantage of the first-order Taylor approximation—it is easier to encounter numerical issues in the information transfer when the observablity is marginal. The other three algorithms (EKF, pMLIT, and fMLIT) are worse than those using GHFs.

V. CONCLUSIONS

In this paper, we formulated an implicit GHM to transfer the full state information when a target crosses between cameras. This model is based on the principle that the target heading estimated from the first camera is observable (assuming the target is in a NCV motion), so that the heading should be consistent when the target moves across cameras, and the speed is therefore proportional to the range from the cameras. These implicit constraints were added to the original measurement model at the crossover time as the GHM. We developed three GHFs, namely, fMLIT-EGHF, fMLIT-UGHF, and fMLIT-UGHFapp, as solutions for the GHM (with different approximations of the nonlinear constraint). The three GHFs outperform the other three algorithms (EKF, pMLIT, and fMLIT), which are not using the GHM, when numerical issues are not encountered. The three GHFs are statistically consistent when the target is near ($\leq$750 m) and PD = 1. However, when the target is at 1,000 m range, only fMLIT-UGHFapp is still statistically consistent, and the fMLIT-EGHF and fMLIT-UGHF cannot meet the consistency criterion due to numerical problems in the matrix inversion. The fMLIT-UGHFapp is an approximate implementation of the fMLIT-UGHF to avoid the matrix inversion. It has similar accuracy to the fMLIT-EGHF (when the latter does not diverge) and fMLIT-UGHF, and its estimates are statistically consistent. Therefore, the fMLIT-UGHFapp is the most robust algorithm in this problem.

To apply the proposed algorithm to real applications, other issues need to be addressed. First of all, cameras

should be calibrated accurately. Second, a suitable track to measurement association algorithm should be applied when multiple targets and false alarms exist. Third, maneuvering targets need to be handled. The GHM in this paper is based on the assumption of constant velocity motion. This assumption is only necessary in the neighborhood of the transfer between cameras. Based on our study, the velocity estimate from a single camera can converge within 1 s observation interval. If the target is not maneuvering for 1 s before camera crossing, the GHM is valid. Otherwise, the pMLIT without speed constraint should be applied. All these issues are separate topics in themselves and quite extensive. Further study on them will be conducted in the future.

APPENDIX A

OBSERVABILITY OF HEADING FROM A SEQUENCE OF ANGULAR MEASUREMENTS

This appendix proves that the target heading is observable for a constant velocity (CV) target from a sequence of angular measurements (azimuths and elevations) of a stationary sensor. The 3D problem can be simplified to a 2D problem in the plane formed by the target path (a straight line) and the sensor location point. The azimuth and elevation measurements can then be converted to bearing lines in this plane. If we can prove the target heading in this plane is observable, the heading is also observable in 3D Cartesian space.

Fig. 21 shows this 2D plane with the sensor at point O and three bearing lines OA, OB, and OC in time sequence with a fixed time interval T. Assuming the target path is on the straight line ABC, we have $|AB| = |BC|$ as the target is moving in CV. We will prove that the target heading is observable by the following two steps:

- At an arbitrary point A on the first bearing line, there is an unique straight path ABC with $|AB| = |BC|$ (namely, with two equal cuts by the three bearing lines).
- All the (CV target paths with two equal cuts (by the three bearing lines) are parallel.

These will prove that all possible target trajectories share the same unique heading.

First, we prove that if A is fixed, then ABC is the unique straight path with $|AB| = |BC|$ for the three LOS as above. We draw a line AD parallel to OC and it intersects OB at D. We prove $|AD| = |OC|$. Since the triangles ABD and OBC are similar and $|AB| = |BC|$, we have

$$\frac{|AB|}{|BC|} = \frac{|AD|}{|OC|} = 1, \tag{A1}$$

$$|AD| = |OC|. \tag{A2}$$

We then draw two arbitrary straight lines AB_1C_1 and AB_2C_2 below and above ABC, respectively. We will

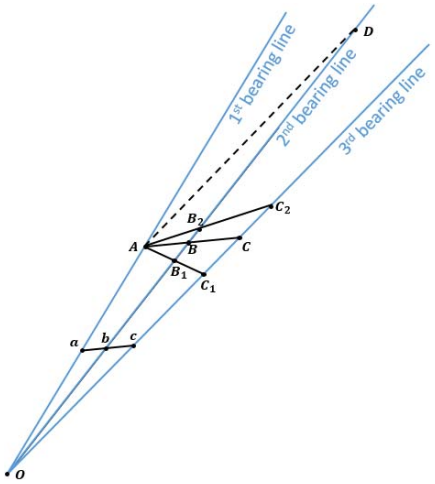


Fig. 21. Proof of the unique target heading obtained from three bearing lines for a constant velocity target.

prove these two arbitrary lines cannot meet the equalities (implied by CV assumption) $|AB_1| = |B_1C_1|$ or $|AB_2| = |B_2C_2|$. For AB_1C_1, we prove $|AB_1| > |B_1C_1|$ by

$$\frac{|AB_1|}{|B_1C_1|} = \frac{|AD|}{|OC_1|} > \frac{|AD|}{|OC|} = 1, \tag{A3}$$

$$\frac{|AB_1|}{|B_1C_1|} > 1, \tag{A4}$$

$$|AB_1| > |B_1C_1|. \tag{A5}$$

For AB_2C_2, we prove $|AB_2| < |B_2C_2|$ by

$$\frac{|AB_2|}{|B_2C_2|} = \frac{|AD|}{|OC_2|} < \frac{|AD|}{|OC|} = 1, \tag{A6}$$

$$\frac{|AB_2|}{|B_2C_2|} < 1, \tag{A7}$$

$$|AB_2| < |B_2C_2|. \tag{A8}$$

Since any arbitrary line below or above ABC does not have two equal cuts by the three bearing lines, it follows that ABC is unique for a CV target starting at A.

Second, we will prove all the possible target trajectories with equal cuts are parallel. Let's draw another arbitrary line abc parallel to ABC, and a is an arbitrary point besides A on the first bearing line. We will prove $|ab| = |bc|$ by

$$\frac{|ab|}{|AB|} = \frac{|Ob|}{|OB|}, \tag{A9}$$

$$|ab| = \frac{|Ob||AB|}{|OB|}, \tag{A10}$$

$$\frac{|bc|}{|BC|} = \frac{|Ob|}{|OB|}, \tag{A11}$$

$$|bc| = \frac{|Ob||BC|}{|OB|} = \frac{|Ob||AB|}{|OB|} = |ab|. \tag{A12}$$

Based on our previous proof, the straight line with two equal cuts is unique if a is fixed. Thus abc is the unique line starting at a with two equal cuts. Since abc is parallel to ABC, they share the same heading. Furthermore, point a is an arbitrary point on the first bearing line, and this proves that all possible CV trajectories (with two equal cuts) are parallel and share the same heading. This proves that the heading is observable.

Note the assumption of the above proof that the three bearing lines are different. If a target approaching to the sensor (or moving away from the sensor) directly, the three bearings coincide. The heading angle is known, but cannot differentiate if the target is approaching or moving away from O. That needs additional information, such as target size increasing/decreasing in a sequence of images (or intensity for point detection), to tell the direction.

APPENDIX B

JACOBIANS IN THE fMLIT-EGHF AND fMLIT-UGHF

The Jacobians $\mathbf{A}$ and $\mathbf{B}$ used in the fMLIT-EGHF and fMLIT-UGHF are derived below.

$$\mathbf{A} = \frac{\partial \mathbf{g}(\mathbf{x}_{\mathrm{II}}, \mathbf{y})}{\partial \mathbf{x}_{\mathrm{II}}} \begin{bmatrix} A_{11} & A_{12} & A_{13} & 0 & 0 & 0 \\ A_{21} & A_{22} & A_{23} & 0 & 0 & 0 \\ A_{31} & A_{32} & A_{33} & 0 & 0 & 0 \\ A_{41} & A_{42} & A_{43} & r_{\mathrm{I}} & 0 & 0 \\ A_{51} & A_{52} & A_{53} & 0 & r_{\mathrm{I}} & 0 \\ A_{61} & A_{62} & A_{63} & 0 & 0 & r_{\mathrm{I}} \\ A_{71} & A_{72} & 0 & 0 & 0 & 0 \\ A_{81} & A_{82} & A_{83} & 0 & 0 & 0 \end{bmatrix}, \tag{B1}$$

$$\mathbf{B} = \frac{\partial \mathbf{g}(\mathbf{x}_{\mathrm{II}}, \mathbf{y})}{\partial \mathbf{y}} \begin{bmatrix} B_{11} & B_{12} & B_{13} & 0 & 0 & 0 & 0 & 0 \\ B_{21} & B_{22} & B_{23} & 0 & 0 & 0 & 0 & 0 \\ B_{31} & B_{32} & B_{33} & 0 & 0 & 0 & 0 & 0 \\ B_{41} & B_{42} & B_{43} & -r_{\mathrm{II}} & 0 & 0 & 0 & 0 \\ B_{51} & B_{52} & B_{53} & 0 & -r_{\mathrm{II}} & 0 & 0 & 0 \\ B_{61} & B_{62} & B_{63} & 0 & 0 & -r_{\mathrm{II}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}, \tag{B2}$$

with

$$A_{11} = r_{\mathrm{I}} - (x_{\mathrm{I}} - x_{\mathrm{s}_1})(x_{\mathrm{II}} - x_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B3}$$

$$A_{12} = -(x_{\mathrm{I}} - x_{\mathrm{s}_1})(y_{\mathrm{II}} - y_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B4}$$

$$A_{13} = -(x_{\mathrm{I}} - x_{\mathrm{s}_1})(z_{\mathrm{II}} - z_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B5}$$

$$A_{21} = -(y_{\mathrm{I}} - y_{\mathrm{s}_1})(x_{\mathrm{II}} - x_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B6}$$

$$A_{22} = r_{\mathrm{I}} - (y_{\mathrm{I}} - y_{\mathrm{s}_1})(y_{\mathrm{II}} - y_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B7}$$

$$A_{23} = -(y_{\mathrm{I}} - y_{\mathrm{s}_1})(z_{\mathrm{II}} - z_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B8}$$

$$A_{31} = -(z_{\mathrm{I}} - z_{\mathrm{s}_1})(x_{\mathrm{II}} - x_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B9}$$

$$A_{32} = -(z_{\mathrm{I}} - z_{\mathrm{s}_1})(y_{\mathrm{II}} - y_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B10}$$

$$A_{33} = r_{\mathrm{I}} - (z_{\mathrm{I}} - z_{\mathrm{s}_1})(z_{\mathrm{II}} - z_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B11}$$

$$A_{41} = -\dot{x}_{\mathrm{I}}(x_{\mathrm{II}} - x_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B12}$$

$$A_{42} = -\dot{x}_{\mathrm{I}}(y_{\mathrm{II}} - y_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B13}$$

$$A_{43} = -\dot{x}_{\mathrm{I}}(z_{\mathrm{II}} - z_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B14}$$

$$A_{51} = -\dot{y}_{\mathrm{I}}(x_{\mathrm{II}} - x_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B15}$$

$$A_{52} = -\dot{y}_{\mathrm{I}}(y_{\mathrm{II}} - y_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B16}$$

$$A_{53} = -\dot{y}_{\mathrm{I}}(z_{\mathrm{II}} - z_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B17}$$

$$A_{61} = -\dot{z}_{\mathrm{I}}(x_{\mathrm{II}} - x_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B18}$$

$$A_{62} = -\dot{z}_{\mathrm{I}}(y_{\mathrm{II}} - y_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B19}$$

$$A_{63} = -\dot{z}_{\mathrm{I}}(z_{\mathrm{II}} - z_{\mathrm{s}_1})/r_{\mathrm{II}}, \tag{B20}$$

$$A_{71} = \frac{y_{\mathrm{II}} - y_{\mathrm{s}_2}}{(x_{\mathrm{II}} - x_{\mathrm{s}_2})^2 + (y_{\mathrm{II}} - y_{\mathrm{s}_2})^2}, \tag{B21}$$

$$A_{72} = -\frac{x_{\mathrm{II}} - x_{\mathrm{s}_2}}{(x_{\mathrm{II}} - x_{\mathrm{s}_2})^2 + (y_{\mathrm{II}} - y_{\mathrm{s}_2})^2}, \tag{B22}$$

$$A_{81} = -\frac{(x_{\mathrm{II}} - x_{\mathrm{s}_2})(z_{\mathrm{II}} - z_{\mathrm{s}_2})}{\sqrt{(x_{\mathrm{II}} - x_{\mathrm{s}_2})^2 + (y_{\mathrm{II}} - y_{\mathrm{s}_2})^2}} \times \frac{1}{(x_{\mathrm{II}} - x_{\mathrm{s}_2})^2 + (y_{\mathrm{II}} - y_{\mathrm{s}_2})^2 + (z_{\mathrm{II}} - z_{\mathrm{s}_2})^2}, \tag{B23}$$

$$A_{82} = -\frac{(y_{\mathrm{II}} - y_{s_2})(z_{\mathrm{II}} - z_{s_2})}{\sqrt{(x_{\mathrm{II}} - x_{s_2})^2 + (y_{\mathrm{II}} - y_{s_2})^2}} \times \frac{1}{(x_{\mathrm{II}} - x_{s_2})^2 + (y_{\mathrm{II}} - y_{s_2})^2 + (z_{\mathrm{II}} - z_{s_2})^2}, \tag{B24}$$

$$A_{83} = \frac{\sqrt{(x_{\mathrm{II}} - x_{s_2})^2 + (y_{\mathrm{II}} - y_{s_2})^2}}{(x_{\mathrm{II}} - x_{s_2})^2 + (y_{\mathrm{II}} - y_{s_2})^2 + (z_{\mathrm{II}} - z_{s_2})^2}, \tag{B25}$$

$$B_{11} = (x_{\mathrm{II}} - x_{s_1})(x_{\mathrm{I}} - x_{s_1})/r_{\mathrm{I}} - r_{\mathrm{II}}, \tag{B26}$$

$$B_{12} = (x_{\mathrm{II}} - x_{s_1})(y_{\mathrm{I}} - y_{s_1})/r_{\mathrm{I}}, \tag{B27}$$

$$B_{13} = (x_{\mathrm{II}} - x_{s_1})(z_{\mathrm{I}} - z_{s_1})/r_{\mathrm{I}}, \tag{B28}$$

$$B_{21} = (y_{\mathrm{II}} - y_{s_1})(x_{\mathrm{I}} - x_{s_1})/r_{\mathrm{I}}, \tag{B29}$$

$$B_{22} = (y_{\mathrm{II}} - y_{s_1})(y_{\mathrm{I}} - y_{s_1})/r_{\mathrm{I}} - r_{\mathrm{II}}, \tag{B30}$$

$$B_{23} = (y_{\mathrm{II}} - y_{s_1})(z_{\mathrm{I}} - z_{s_1})/r_{\mathrm{I}}, \tag{B31}$$

$$B_{31} = (z_{\mathrm{II}} - z_{s_1})(x_{\mathrm{I}} - x_{s_1})/r_{\mathrm{I}}, \tag{B32}$$

$$B_{32} = (z_{\mathrm{II}} - z_{s_1})(y_{\mathrm{I}} - y_{s_1})/r_{\mathrm{I}}, \tag{B33}$$

$$B_{33} = (z_{\mathrm{II}} - z_{s_1})(z_{\mathrm{I}} - z_{s_1})/r_{\mathrm{I}} - r_{\mathrm{II}}, \tag{B34}$$

$$B_{41} = \dot{x}_{\mathrm{II}}(x_{\mathrm{I}} - x_{s_1})/r_{\mathrm{I}}, \tag{B35}$$

$$B_{42} = \dot{x}_{\mathrm{II}}(y_{\mathrm{I}} - y_{s_1})/r_{\mathrm{I}}, \tag{B36}$$

$$B_{43} = \dot{x}_{\mathrm{II}}(z_{\mathrm{I}} - z_{s_1})/r_{\mathrm{I}}, \tag{B37}$$

$$B_{51} = \dot{y}_{\mathrm{II}}(x_{\mathrm{I}} - x_{s_1})/r_{\mathrm{I}}, \tag{B38}$$

$$B_{52} = \dot{y}_{\mathrm{II}}(y_{\mathrm{I}} - y_{s_1})/r_{\mathrm{I}}, \tag{B39}$$

$$B_{53} = \dot{y}_{\mathrm{II}}(z_{\mathrm{I}} - z_{s_1})/r_{\mathrm{I}}, \tag{B40}$$

$$B_{61} = \dot{z}_{\mathrm{II}}(x_{\mathrm{I}} - x_{s_1})/r_{\mathrm{I}}, \tag{B41}$$

$$B_{62} = \dot{z}_{\mathrm{II}}(y_{\mathrm{I}} - y_{s_1})/r_{\mathrm{I}}, \tag{B42}$$

$$B_{63} = \dot{z}_{\mathrm{II}}(z_{\mathrm{I}} - z_{s_1})/r_{\mathrm{I}}, \tag{B43}$$

where r_{I} and r_{II} are given in (30) and (31), respectively (see Fig. 3).

REFERENCES

[1] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan
Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software. New York, NY, USA: Wiley, 2001.

[2] Y. Bar-Shalom, P. K. Willett, and X. Tian
Tracking and Data Fusion: A Handbook of Algorithms. Storrs, CT, USA: YBS Publishing, 2011.

[3] L. Chen, H. Ai, C. Shang, Z. Zhuang, and B. Bai
"Online multi-object tracking with convolutional neural networks,"
In IEEE Int. Conf. Image Process, Sep. 2017, pp. 645–649.

[4] W. Forstner
"On weighting and choosing constraints for optimally reconstructing the geometry of image triplets,"
In the 6th Eur. Conf. Comput. Vision, Jun. 2000, pp. 669–684.

[5] M. A. Haseeb, J. Guan, D. Ristic-Durrant, and A. Graser
"DisNet: A novel method for distance estimation from monocular camera,"
In 10th Planning Perception Navigation Intell. Vehicles, Oct. 2018.

[6] C. Jauffret, D. Pillon, and A. Pignol
"Bearings-only TMA without observer maneuver,"
In 11th Int. Conf. Inf. Fusion, Jul. 2008.

[7] K. Kanatani and H. Niitsuma
"Optimal computation of 3-D similarity: Gauss-Newton vs. Gauss-Helmert,"
Comput. Statist. Data Anal., vol. 56, no. 4, pp. 4470–4483, Dec. 2012.

[8] T. Klinger, F. Rottensteiner, and C. Heipke
"Probabilistic multi-person localisation and tracking in image sequences,"
ISPRS J. Photogrammetry Remote Sens., vol. 127, pp. 73–88, May 2017.

[9] K.-H. Lee and J.-N. Hwang
"On-road pedestrian tracking across multiple driving recorders,"
IEEE Trans. Multimedia, vol. 17, no. 9, pp. 1429–1438, Sep. 2015.

[10] E. M. Mikhail and F. E. Ackermann
Observations and Least Squares. New York, NY, USA: Harper and Row, 1976.

[11] S. Y. Nam and G. P. Joshi
"Unmanned aerial vehicle localization using distributed sensors,"
Int. J. Distrib. Sensor Netw., vol. 13, no. 9, pp. 1–8, Aug. 2017.

[12] F. Neitzel
"Generalization of total least-squares on example of unweighted and weighted 2D similarity transformations,"
J. Geodesy, vol. 84, no. 12, pp. 751–762, Dec. 2010.

[13] C. Perwass, C. Gebken, and G. Sommer
"Geometry and kinematics with uncertain data,"
In the 9th Eur. Conf. Comput. Vis. Part I, May 2006, pp. 225–237.

[14] B. Schaffrin and K. Snow
"Total least-squares regularization of Tykhonov type and an ancient racetrack in Corinth,"
Linear Algebra Appl., vol. 432, no. 8, pp. 2061–2076, Apr. 2010.

[15] S. Soatto, R. Frezza, and P. Perona
"Motion estimation via dynamic vision,"
IEEE Trans. Autom. Control, vol. 41, no. 3, pp. 393–413, Mar. 1996.

[16] Z. Tang, J.-N. Hwang, Y.-S. Lin, and J.-H. Chuang
"Multiple-kernel adaptive segmentation and tracking (MAST) for robust object tracking,"

In IEEE Int. Conf. Acoust., Speech Signal Process., pp. 1115–1119, Mar. 2016.

[17] Z. Tang and J.-N. Hwang
"MOANA: An online learned adaptive appearance model for robust multiple object tracking in 3D,"
IEEE Access, vol. 7, no. 1, pp. 31934–31945, Mar. 2019.

[18] L. Wen, Z. Lei, M.-C. Chang, H. Qi, and S. Lyu
"Multi-camera multitarget tracking with space-time-view hyper-graph,"
Int. J. Comput. Vis., vol. 122, no. 2, pp. 313–333, Sep. 2016.

[19] M. Yang, Y. Wu, and Y. Jia
"A hybrid data association framework for robust online multi-object tracking,"
IEEE Trans. Image Process., vol. 26, no. 12, pp. 5667–5679, Dec. 2017.

[20] R. Yang, Y. Bar-Shalom, H. A. Jack Huang, and G. W Ng
"Interacting multiple model unscented Gauss-Helmert filter for bearings-only tracking with state-dependent propagation delay,"
In 17th Int. Conf. Inf. Fusion, Jul. 2014.

[21] R. Yang, Y. Bar-Shalom, H. A. Jack Huang, and G. W. Ng
"UGHF for acoustic tracking with state-dependent propagation delay,"
IEEE Trans. Aerosp. Electron. Syst., vol. 51, no. 3, pp. 1747–1761, Jul. 2015.

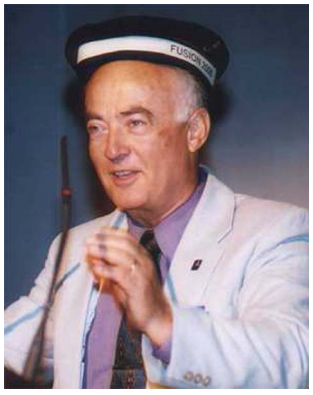
[22] R. Yang, Y. Bar-Shalom, and G. W. Ng
"Bearings-only tracking with fusion from heterogenous passive sensors: ESM/EO and acoustic,"
In the 18th Int. Conf. Inf. Fusion, Jul. 2015.

[23] R. Yang, Y. Bar-Shalom, and G. W. Ng
"Bearings-only tracking with fusion from heterogenous passive sensors: ESM/EO and acoustic,"
J. Adv. Inf. Fusion, vol. 12, no. 3, pp. 3–17, Jul. 2017.

[24] R. Yang, Y. Bar-Shalom, C. Jauffret, A.-C. Perez, and G. W. Ng
"Maneuvering target tracking using continuous wave bistatic sonar with propagation delay,"
J. Adv. Inf. Fusion, vol. 13, no. 1, pp. 36–49, Jun. 2018.

[25] T. Yuan, Y. Bar-Shalom, and X. Tian
Heterogeneous track-to-track fusion.
In 14th Int. Conf. Inf. Fusion, Jul. 2011.

Rong Yang received the B.E. degree in information and control from Xi'an Jiao Tong University, China, in 1986, the M.Sc. degree in electrical engineering from National University of Singapore, in 2000, and the Ph.D. degree in electrical engineering from Nanyang Technological University, Singapore, in 2012. She is currently a Principal Member of Technical Staff at DSO National Laboratories, Singapore. Her research interests include passive tracking, low observable target tracking, GMTI tracking, hybrid dynamic estimation, and data fusion. She was Publicity and Publication Chair of FUSION 2012 and received the FUSION 2014 Best Paper Award (First runner up).



Yaakov Bar-Shalom (F'84) received the B.S. and M.S. degrees from the Technion, in 1963 and 1967, and the Ph.D. degree from Princeton University, in 1970, all in EE. Currently he is Board of Trustees Distinguished Professor in the ECE Dept. and Marianne E. Klewin Professor at the University of Connecticut. His current research interests are in estimation theory, target tracking, and data fusion. He has published over 650 papers and book chapters. He coauthored/edited eight books, including Tracking and Data Fusion (YBS Publishing, 2011). He has been elected Fellow of IEEE for "contributions to the theory of stochastic systems and of multitarget tracking". He served as 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 AESS. He is corecipient of the M. Barry Carlton Award for the best paper in the IEEE TAE Systems in 1995 and 2000. In 2002, he received 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 awarded 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 is 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. He has the following Wikipedia page: https://en.wikipedia.org/wiki/Yaakov Bar-Shalom.

Unbiased Conversion of Passive Sensor Measurements Using Closest Point of Approach

MICHAEL KOWALSKI
YAAKOV BAR-SHALOM
PETER WILLETT
TIM FAIR

In passive sensor target tracking, there are applications that require converting the angle-only measurements into Cartesian space. Multisensor methods can be used to convert the raw measurements into Cartesian measurements by finding the intersection of lines of sight. This method contains significant nonlinearity in its conversion; therefore, it is subject to corresponding errors such as a conversion bias and an improper covariance. The proposed method uses a second-order Taylor series expansion to accurately account for the conversion nonlinearities. This results in an explicit (noniterative) expression of the Cartesian position based on two line-of-sight measurements in three dimensions. This paper investigates the severity of the conversion biases from nonlinearity and the efficiency of the unbiased conversion with regard to compensating for them.

Manuscript received October 31, 2021; revised November 27, 2021; released for publication July 7, 2022.

Associate Editor: Florian Meyer.

M. Kowalski, Y. Bar-Shalom, and P. Willett are with the Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269 USA (e-mail: michael.p.kowalski@uconn.edu; ybs@uconn.edu; peter.willett@uconn.edu).

T. Fair is with the Toyon Research Corporation, Sterling, VA 93117 USA (e-mail: tfair@toyon.com).

1557-6418/22/$17.00

I. INTRODUCTION

Passive sensors are especially challenging compared to those that are active. Although some passive sensors also deliver amplitude information—and, indeed, in some situations, can incorporate processing such as via wavefront curvature or based on target image expectations to infer range—they are commonly assumed to present only angular measurements (see [3], [4]). Passive sensor measurements require a data fusion step if a three-dimensional plot is desired, as it often is by a downstream tracker. However, as we shall see, a typical processing chain aimed at such plots produces an unwanted but modelable bias.[1] This paper addresses such modeling, but we note that while there are sophisticated ways to mitigate bias (e.g., nonlinear least-squares in [2] and maximum-likelihood (ML) estimation in [13]), the goal here is for a simple delivery of Cartesian measurements to a simple Cartesian tracker such as a Kalman filter; a nonlinear dynamic estimation approach such as a particle filter does not require such pre-processing, but is often much more computationally demanding. However, in any case, even therein improvements can be made in terms of initialization (using the angle measurements to obtain Cartesian positions and velocities), and sensor registration through bias estimation is a widely researched topic that commonly uses converted measurements from spherical coordinates and range/direction-sines coordinates [16].

Triangulation is often used to convert two angle-only measurements into Cartesian position. In [14], triangulation is used to initialize an ML approach to converting angle-only measurements into Cartesian. However, the ML method uses a search to obtain an (iterative) estimate of the Cartesian position. In the case of an ML search, it is practically impossible to produce a Jacobian matrix of the converted Cartesian coordinates with respect to the original angle-only measurements as the "location" of the ML estimate's convergence point is not analytically related to the angle-only measurements. This Jacobian is useful for applications in other target tracking applications such as bias estimation. The novel [10] was the first to attempt the second-order Taylor series expansion for angle-only measurements in passive sensors and use them for estimation of sensor bias, which is separate from conversion bias. The work relied on triangulation to form bias pseudo-measurements. Conversion via triangulation is flawed particularly as the conversion is overly reliant on azimuth rather than the total angle. The transformation fails when the lines of sight of the two sensors are equal in azimuth as the denominator of the conversion equation becomes zero. This would result in "blind spots" of the transformation where the error and covariance consistency would dete-

[1] Other effects such as refraction also produce offsets that might be called biases, but these can be treated in other ways and we do not discuss them here.

riorate greatly, as seen in [9]. The law of sines is used in [5] to generate ranges in order to convert the angle measurements into Cartesian similarly to triangulation. In [15], the closest point of approach between the two line-of-sight (LOS) rays is used to create composite Cartesian coordinates. This paper investigates the closest point of approach method to produce a (noniterative) expression of the Cartesian position based on two LOS measurements in three dimensions. However, the conversion of angle measurements into Cartesian is a nonlinear transformation that requires an unbiased conversion [3]. As such, this paper investigates the bias of the explicit conversion expression and provides a solution to overcome it.

An approach to deriving an unbiased conversion for a nonlinear transformation is used in [17] to convert sine space coordinates into Cartesian by using a second-order Taylor series expansion. In this paper, the approach is replicated for the conversion of passive sensor measurements to Cartesian. The method is also similarly evaluated to ensure that the second-order conversion is necessary.

In particular, it is necessary to account for the nonlinear measurement conversion by accounting for the bias and also by converting the measurement noise covariance. The conversion bias is well documented in the literature [3], [12] for spherical to Cartesian conversion where the noise pattern resembles a curved lens in Cartesian space rather than a sphere. It is necessary to adjust the converted noise covariance to more accurately represent the converted measurements. Similarly, the same process must be made for the conversion of angle-only measurements into Cartesian. Previously, in [9], the conversion bias and covariance consistency errors were improved by using the second-order Taylor series expansion. A simulation was constructed that produced a conversion bias, and the unbiased conversion reduced it to less than one-tenth of its original value. This conversion was then used in [10] to form bias pseudo-measurements, which could then be used to estimate sensor biases without needing to estimate the target state. In [11], debiased polar measurements were used in a Kalman Filter and are shown to improve tracking performance relative to a mixed coordinate Extended Kalman Filter (EKF). The RMS values were reduced as a result of the reduction of errors due to nonlinearity. This is because an EKF is subject to the same nonlinearities when converting its state to sensor coordinates when using a mixed measurement model, albeit in the opposite "direction".

A cubature integration method can also be used to approximate the moments of the nonlinear conversion. In [6], several filters involving cubature methods were examined with respect to areas of severe nonlinearity. The cubature Kalman filters and a Kalman filter using converted measurements were found to be commensurate in performance when wrapping was used.

This paper is outlined as follows: Section II contains the definitions for passive sensing used by this method. The proposed method of closest point of approach is presented in Section III. The methods of analyzing the conversion are in Section IV. The parameters for the simulation are included in Section IV-A. The methods include analysis of the bias discussed in Section IV-B and analysis of the covariance discussed in Section IV-C. Section V concludes the paper.

Notation used in this work includes vectors as bold symbols such as $\mathbf{x}$. Gradients are defined using the ∇ symbol. The superscript t and subscript s specify the target and sensor indices, respectively. The superscripts c, db, and m specify that a variable is converted, debiased, and measured, respectively. The notation $'$ means the vector or matrix is transposed. The subscripts x, y, and z are used to specify that a variable refers to the respective Cartesian coordinate. The notation (k) specifies the time index. $\mathcal{N}$ specifies a Gaussian random variable with the mean and covariance in parentheses.

II. PROBLEM FORMULATION

Passive three-dimensional angle-only sensors are used for this paper. Passive sensors only give angle measurements pointing in the direction of the target. In three dimensions, this is made up of two angles, azimuth, and elevation. The sensors are assumed to be synchronous and the network consists of N_s sensors. At a timestep k, the position of sensor s in Cartesian space, assumed to be known, is

$$\mathbf{x}_s(k) = [x_s(k),\ y_s(k),\ z_s(k)]'. \tag{1}$$

For simplicity, there is only a single target t, and its Cartesian position is similarly

$$\mathbf{x}^t(k) = [x^t(k),\ y^t(k),\ z^t(k)]'. \tag{2}$$

The sensors generate measurements of the target from their own reference frame:

$$\mathbf{x}_s^t(k) = \mathbf{x}^t(k) - \mathbf{x}_s(k). \tag{3}$$

Using the positions derived in (3), the sensors generate elevation and azimuth measurements. Azimuth is denoted as α and elevation is denoted as ϵ. The measurements use atan2, which (MATLAB notation) is the four-quadrant inverse tangent.

$$\boldsymbol{\xi}_s(k) = \begin{bmatrix} \alpha_s(k) \\ \epsilon_s(k) \end{bmatrix} = \begin{bmatrix} \text{atan2}\left(\frac{y_s^t(k)}{x_s^t(k)}\right) \\ \text{atan2}\left(\frac{z_s^t(k)}{\sqrt{x_s^t(k)^2 + y_s^t(k)^2}}\right) \end{bmatrix}. \tag{4}$$

The measurements are combined with noises $w_s^\alpha(k)$ and $w_s^\epsilon(k)$ to produce the final measurement model, where superscript m signifies that the angles are the measurements generated by the sensors. The noise for each sensor is assumed uncorrelated independent white

Gaussian with variances $(\sigma_s^\alpha)^2$ and $(\sigma_s^\epsilon)^2$ and zero mean.[2] The noisy measurements are

$$\boldsymbol{\xi}_s^m(k) = \begin{bmatrix} \alpha_s^m(k) \\ \epsilon_s^m(k) \end{bmatrix} = \begin{bmatrix} \alpha_s(k) \\ \epsilon_s(k) \end{bmatrix} + \begin{bmatrix} w_s^\alpha(k) \\ w_s^\epsilon(k) \end{bmatrix}, \quad (5)$$

$$w_s^\alpha(k) \sim \mathcal{N}(0, (\sigma_s^\alpha)^2) \quad w_s^\epsilon(k) \sim \mathcal{N}(0, (\sigma_s^\epsilon)^2). \quad (6)$$

The noise variances are assumed to be known by the system. These measurements are assumed to be synchronous, but this model can be modified for the asynchronous case. The true target state is treated as a parameter because we consider a single point in time, and the method proposed here seeks to be agnostic with regard to target dynamics. A filter (which operates across time) normally introduces process noise models for tracking, but our method converts the individual measurements to be used in a tracking filter with the converted noise errors being zero-mean, with consistent covariance, and independent across time. As such, this method can be used with any target motion or process noise model.

In this paper, we assume sensors deliver synchronized measurements. In practice, a high frame rate mitigates asynchronicity. But, it is admitted that significantly asynchronous measurements will cause additional errors as they would need to be propagated to the same time for conversion, and this would require integration of bias estimation into the dynamic estimation procedure. That is beyond the scope of this paper, but is an intriguing topic for future work.

III. CONVERSION USING CLOSEST POINT OF APPROACH

A. The Conversion

The method for conversion investigated in this work is using the closest point of approach to generate a single Cartesian measurement using angle measurements from two sensors. This method finds the points on the two LOS rays that are closest to each other and then labels the midpoint between the two points as the composite Cartesian position. When converting lines of sight into Cartesian, it is important to take into account observability. Previously, in [9], the triangulation conversion was examined, but this was found to have observability problems when solely the azimuth components were similar, regardless of whether the lines of sight were parallel or not. This reduces the practicality of the approach because the reference frame would have to be rotated to avoid having the azimuth components be the same. The conversion proposed in this work is an improvement because it avoids this severe reliance on one of the two angle measurements, improving observability and avoiding the need for rotation. This process is shown in Fig. 1. The superscript c is used to signify that it is a conversion:

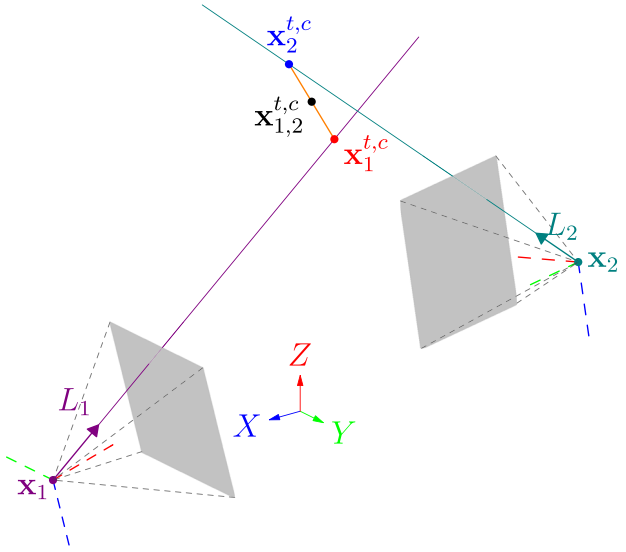


Fig. 1. Using CPA (closest point of approach) to convert azimuth measurements into three-dimensional Cartesian measurements.

$$\mathbf{x}_1^{t,c}(k) = \begin{bmatrix} x_1^{t,c}(k) \\ y_1^{t,c}(k) \\ z_1^{t,c}(k) \end{bmatrix} = \mathbf{x}_1(k) + \frac{\mathbf{L}_1(k)(\mathbf{L}_1(k)'\mathbf{B}_{1,2}(k)) - (\mathbf{L}_1(k)'\mathbf{L}_2(k))(\mathbf{L}_2(k)'\mathbf{B}_{1,2}(k))}{1-(\mathbf{L}_1(k)'\mathbf{L}_2(k))^2}, \quad (7)$$

$$\mathbf{x}_2^{t,c}(k) = \begin{bmatrix} x_2^{t,c}(k) \\ y_2^{t,c}(k) \\ z_2^{t,c}(k) \end{bmatrix} = \mathbf{x}_2(k) + \frac{\mathbf{L}_2(k)(\mathbf{L}_1(k)'\mathbf{L}_2(k))(\mathbf{L}_1(k)'\mathbf{B}_{1,2}(k)) - (\mathbf{L}_2(k)'\mathbf{B}_{1,2}(k))}{1-(\mathbf{L}_1(k)'\mathbf{L}_2(k))^2}, \quad (8)$$

$$\mathbf{x}_{1,2}^{t,c}(k) = \tfrac{1}{2}\left(\mathbf{x}_1^{t,c}(k) + \mathbf{x}_2^{t,c}(k)\right). \quad (9)$$

The conversion relies on the Cartesian vectors of the lines of sight defined as $\mathbf{L}$ and the line between the sensors defined as $\mathbf{B}$. These are calculated as

$$\mathbf{L}_1(k) = \begin{bmatrix} \cos(\alpha_1(k))\cos(\epsilon_1(k)) \\ \sin(\alpha_1(k))\cos(\epsilon_1(k)) \\ \sin(\epsilon_1(k)) \end{bmatrix}, \quad (10)$$

$$\mathbf{L}_2(k) = \begin{bmatrix} \cos(\alpha_2(k))\cos(\epsilon_2(k)) \\ \sin(\alpha_2(k))\cos(\epsilon_2(k)) \\ \sin(\epsilon_2(k)) \end{bmatrix}, \quad (11)$$

[2] In passive sensor angle measurements, the noise is commonly approximated with a Gaussian, but it is not the most accurate model for real sensors. Research in passive sensors has used alternatives such as the Kent distribution in [8] or a wrapped distribution as in [6]. For the purposes of this work, the noise is approximated as Gaussian. A Gaussian assumption yields a simple algorithm, so we use it; but, if a higher fidelity/complexity solution is required, the approaches in [10] and [4] should be consulted.

$$\mathbf{B}_{1,2}(k) = \mathbf{x}_2(k) - \mathbf{x}_1(k) = \begin{bmatrix} x_2(k) - x_1(k) \\ y_2(k) - y_1(k) \\ z_2(k) - z_1(k) \end{bmatrix}. \quad (12)$$

The derivation for these equations is included in the Appendix. In ML applications [10], the two Cartesian coordinate measurements can be used separately rather than merged for the parameter estimate.

In the presence of noise, this method is imperfect as the two lines of sight will not intersect exactly in three-dimensional space. As such, estimation methods are commonly used to estimate the true target state, which will be discussed later in the paper. However, in circumstances when estimation methods are undesirable, it is possible to use this method, and the effect of the noise can be approximated with a conversion. Furthermore, the noise can cause a bias in the converted measurement, but this bias can be calculated and removed.

It is important to note that this is an explicit expression of the conversion. Unlike iterative methods such as ML, it is possible to calculate a Jacobian of the converted Cartesian positions with respect to the original LOS measurements. As such, the derivatives can be calculated and, in turn, used to calculate the bias and covariance. The explicit expression and these derivatives are useful in other applications such as the generation of pseudo-measurements for bias estimation [10].

B. The Bias of the Conversion and Its Compensation

To approximate the noise covariance and debias the converted measurements, a Taylor series expansion is used. A second-order expansion is used here; however, further orders can be used for a more accurate conversion. For simplicity, the y and z expansions as well as the individual derivatives are moved to the Appendix. The superscript m denotes that the converted Cartesian measurements are made with the noisy LOS measurements:

$$\begin{aligned} x_{1,2}^{t,c,m}(k) \approx\ & x^t(k) + \frac{\partial x_{1,2}^{t,c}}{\partial \alpha_1} w_1^{\alpha}(k) + \frac{\partial x_{1,2}^{t,c}}{\partial \alpha_2} w_2^{\alpha}(k) \\ & + \frac{\partial x_{1,2}^{t,c}}{\partial \epsilon_1} w_1^{\epsilon}(k) + \frac{\partial x_{1,2}^{t,c}}{\partial \epsilon_2} w_2^{\epsilon}(k) \\ & + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1^2} w_1^{\alpha}(k)^2 + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2^2} w_2^{\alpha}(k)^2 \\ & + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_1^2} w_1^{\epsilon}(k)^2 + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_2^2} w_2^{\epsilon}(k)^2 \\ & + \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2} w_1^{\alpha}(k) w_2^{\alpha}(k) + \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1} w_1^{\alpha}(k) w_1^{\epsilon}(k) \\ & + \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2} w_1^{\alpha}(k) w_2^{\epsilon}(k) + \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1} w_2^{\alpha}(k) w_1^{\epsilon}(k) \\ & + \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2} w_2^{\alpha}(k) w_2^{\epsilon}(k) + \frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2} w_1^{\epsilon}(k) w_2^{\epsilon}(k). \end{aligned} \quad (13)$$

The expected value of the expanded term contains the conversion bias.

$$\begin{aligned} E[x_{1,2}^{t,c,m}(k)] \approx\ & x^t(k) + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1^2} (\sigma_1^{\alpha})^2 \\ & + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2^2} (\sigma_2^{\alpha})^2 + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_1^2} (\sigma_1^{\epsilon})^2 \\ & + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_2^2} (\sigma_2^{\epsilon})^2, \end{aligned} \quad (14)$$

$$\begin{aligned} c_{x,1,2} =\ & 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1^2} (\sigma_1^{\alpha})^2 + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2^2} (\sigma_2^{\alpha})^2 \\ & + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_1^2} (\sigma_1^{\epsilon})^2 + 0.5 \frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_2^2} (\sigma_2^{\epsilon})^2, \end{aligned} \quad (15)$$

$$E[x_{1,2}^{t,c,m}(k)] \approx x^t(k) + c_{x,1,2}. \quad (16)$$

By calculating and subtracting the bias, it is possible to avoid this error, producing the debiased measurements denoted by the superscript db.

$$x_{1,2}^{t,c,m,db}(k) = x_{1,2}^{t,c,m}(k) - c_{x,1,2}, \quad (17)$$

$$E[x_{1,2}^{t,c,m,db}(k)] \approx x^t(k). \quad (18)$$

The converted state is rewritten into a simple form that contains the truth and converted zero-mean Gaussian noise. This results in the following measurement equation for the converted measurements:

$$\begin{aligned} \mathbf{x}_{1,2}^{t,c,m,db}(k) &= \begin{bmatrix} x_{1,2}^{t,c,m,db}(k) \\ y_{1,2}^{t,c,m,db}(k) \\ z_{1,2}^{t,c,m,db}(k) \end{bmatrix} \\ &= \begin{bmatrix} x^t(k) \\ y^t(k) \\ z^t(k) \end{bmatrix} + \begin{bmatrix} w_{1,2}^{x,t,c,db}(k) \\ w_{1,2}^{y,t,c,db}(k) \\ w_{1,2}^{z,t,c,db}(k) \end{bmatrix}. \end{aligned} \quad (19)$$

The noise after conversion is now zero-mean white Gaussian, as is desirable for tracking and applications,

$$\mathbf{w}_{1,2}(k) = \begin{bmatrix} w_{1,2}^{x,t,c,db}(k) \\ w_{1,2}^{y,t,c,db}(k) \\ w_{1,2}^{z,t,c,db}(k) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \mathbf{R}_{1,2}^{t,c,db} \right). \quad (20)$$

C. The Covariance of the Converted Errors

The covariance matrix for the converted measurements can be calculated with the same Taylor series expansion. The full derivation of the following equations and the appropriate derivatives are found in the Appendix. The covariance matrix (with the superscripts

and subscripts removed for simplicity) is

$$\mathbf{R}_{1,2}^{t,c,db} = \begin{bmatrix} V(x(k)) & CV(x(k),y(k)) & CV(x(k),z(k)) \\ CV(x(k),y(k)) & V(y(k)) & CV(y(k),z(k)) \\ CV(x(k),z(k)) & CV(y(k),z(k)) & V(z(k)) \end{bmatrix}, \tag{21}$$

where V is the variance of the variable defined by

$$V(x_{1,2}^{t,c,m,db}(k)) = E[x_{1,2}^{t,c,m,db}(k)^2] - E[x_{1,2}^{t,c,m,db}(k)]^2. \tag{22}$$

By using the terms in the Taylor series expansion, the equation is changed into a usable form expressed as

$$\begin{aligned} &V(x_{1,2}^{t,c,m,db}(k)) = \\ &\left(\frac{\partial x_{1,2}^{t,c}}{\partial \alpha_1}\right)^2 (\sigma_1^\alpha)^2 + \left(\frac{\partial x_{1,2}^{t,c}}{\partial \alpha_2}\right)^2 (\sigma_2^\alpha)^2 \\ &+ \left(\frac{\partial x_{1,2}^{t,c}}{\partial \epsilon_1}\right)^2 (\sigma_1^\epsilon)^2 + \left(\frac{\partial x_{1,2}^{t,c}}{\partial \epsilon_2}\right)^2 (\sigma_2^\epsilon)^2 \\ &+ \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)^2 (\sigma_1^\alpha)^2(\sigma_2^\alpha)^2 + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)^2 (\sigma_1^\alpha)^2(\sigma_1^\epsilon)^2 \\ &+ \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)^2 (\sigma_1^\alpha)^2(\sigma_2^\epsilon)^2 + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)^2 (\sigma_2^\alpha)^2(\sigma_1^\epsilon)^2 \\ &+ \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)^2 (\sigma_2^\alpha)^2(\sigma_2^\epsilon)^2 + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)^2 (\sigma_1^\epsilon)^2(\sigma_2^\epsilon)^2, \end{aligned} \tag{23}$$

and the covariance CV is defined by

$$\begin{aligned} &CV(x_{1,2}^{t,c,m,db}(k), y_{1,2}^{t,c,m,db}(k)) \\ &\quad = E[(x_{1,2}^{t,c,m,db}(k) - E[x_{1,2}^{t,c,m,db}(k)]) \\ &\qquad \times (y_{1,2}^{t,c,m,db}(k) - E[y_{1,2}^{t,c,m,db}(k)])], \end{aligned} \tag{24}$$

where, similarly, the Taylor series expansion transforms the equation into

$$\begin{aligned} &\mathrm{Cov}(x_{1,2}^{t,c,m,db}(k), y_{1,2}^{t,c,m,db}(k)) \\ &= \left(\frac{\partial x_{1,2}^{t,c}}{\partial \alpha_1}\right)\left(\frac{\partial y_{1,2}^{t,c}}{\partial \alpha_1}\right)(\sigma_1^\alpha)^2 + \left(\frac{\partial x_{1,2}^{t,c}}{\partial \alpha_2}\right)\left(\frac{\partial y_{1,2}^{t,c}}{\partial \alpha_2}\right)(\sigma_2^\alpha)^2 \\ &\quad + \left(\frac{\partial x_{1,2}^{t,c}}{\partial \epsilon_1}\right)\left(\frac{\partial y_{1,2}^{t,c}}{\partial \epsilon_1}\right)(\sigma_1^\epsilon)^2 + \left(\frac{\partial x_{1,2}^{t,c}}{\partial \epsilon_2}\right)\left(\frac{\partial y_{1,2}^{t,c}}{\partial \epsilon_2}\right)(\sigma_2^\epsilon)^2 \\ &\quad + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)\left(\frac{\partial^2 y_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)(\sigma_1^\alpha)^2(\sigma_2^\alpha)^2 \\ &\quad + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)\left(\frac{\partial^2 y_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)(\sigma_1^\alpha)^2(\sigma_1^\epsilon)^2 \\ &\quad + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)\left(\frac{\partial^2 y_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)(\sigma_1^\alpha)^2(\sigma_2^\epsilon)^2 \\ &\quad + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)\left(\frac{\partial^2 y_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)(\sigma_2^\alpha)^2(\sigma_1^\epsilon)^2 \\ &\quad + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)\left(\frac{\partial^2 y_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)(\sigma_2^\alpha)^2(\sigma_2^\epsilon)^2 \\ &\quad + \left(\frac{\partial^2 x_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)\left(\frac{\partial^2 y_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)(\sigma_1^\epsilon)^2(\sigma_2^\epsilon)^2. \end{aligned} \tag{25}$$

At this point, the previous unbiased conversion and covariance evaluation can be defined as the second-order conversion. In the simulations, this will be compared to a first-order conversion, in which the same conversion via the closest point of approach is made, but debiasing is not performed and the second-order components are ignored in the covariance calculation.

IV. SIMULATIONS AND RESULTS

A. Simulation Parameters

To analyze the conversion, we study a long-range orbital scenario in which an orbiting target passes through the field of view of two sea-level sensors. The target starts at 7000 km from the center of the earth, or at an altitude of 622 km above sea level on the equator directly on the x axis in ECI (Earth-centered inertial) coordinates. The target begins with a velocity necessary for maintaining an orbit, 7.546 km/s, with the vector pointing at an angle of 2.678 radians (clockwise from the Y axis) in the Y–Z plane of ECI. Both sensors are stationary at sea level. Sensor 1 is at -1° in latitude and -3° in longitude, and sensor 2 is at 1° in latitude and 3° in longitude. The sensors move via the rotation of the earth with respect to ECI coordinates. The sensors are oriented facing directly up, meaning that the local vertical is the boresight and 0 azimuth position. This scenario is designed to have a target that passes the sensors, causing angle measurements that begin relatively perpendicular and become parallel over time. A less observable system is present when the measurements are parallel. Less observable systems benefit more from improved nonlinear conversion. The simulation setup is shown in Fig. 2.

The measurement noise values and simulation parameters are given in Table I and are kept the same for both sensors for simplicity. The performance of the conversion is analyzed over 100 000 Monte Carlo runs.

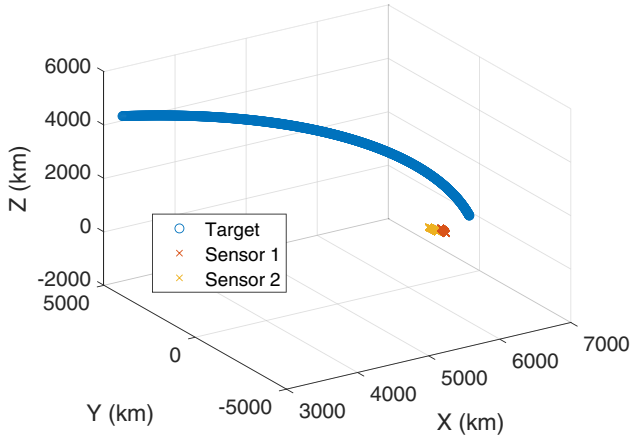


Fig. 2. Sensor and target setup in ECI.

B. Cartesian Position Bias Evaluation

The value of this conversion is first investigated by evaluating the significance of the bias. The conventional conversion is the first-order conversion, which is based on the first-order Taylor series expansion. The bias in the conversion is defined as

$$\mu_x = x^t(k) - E[x^{t,c,m}_{1,2}(k)], \tag{26}$$

$$\mu_y = y^t(k) - E[y^{t,c,m}_{1,2}(k)], \tag{27}$$

$$\mu_z = z^t(k) - E[z^{t,c,m}_{1,2}(k)]. \tag{28}$$

The significance of the bias is defined as the norm of the bias divided by the standard deviation of the noise in Cartesian coordinates. This metric is based on the metric proposed in [11] and used in [7]. The noise is roughly converted by multiplying the range from the closest sensor by the sine of the standard deviation of the azimuth,

$$\beta = \frac{\sqrt{\mu_x^2 + \mu_y^2 + \mu_z^2}}{\sin(\sigma_s^\alpha)\sqrt{(x_s^t(k))^2 + (y_s^t(k))^2 + (z_s^t(k))^2}}. \tag{29}$$

The biases and their significance over the simulation time steps are seen in Fig. 3. The bias increases significantly as the measurements become more parallel. The significance of the bias is quite low, but for long-distance applications, it is advantageous to include debiasing. If the bias significance is less than 0.3 in most applications, it is considered to be negligible as it is less than a 10% in-

Table I
Simulation Parameters, $N_s = 2, N_t = 1, K = 1000$ s, and $N_{\text{MC}} = 100\,000$ runs

	Sensor measurement noise	
	Azimuth noise standard deviation	Elevation noise standard deviation
Sensor 1	1 mrad	1 mrad
Sensor 2	1 mrad	1 mrad

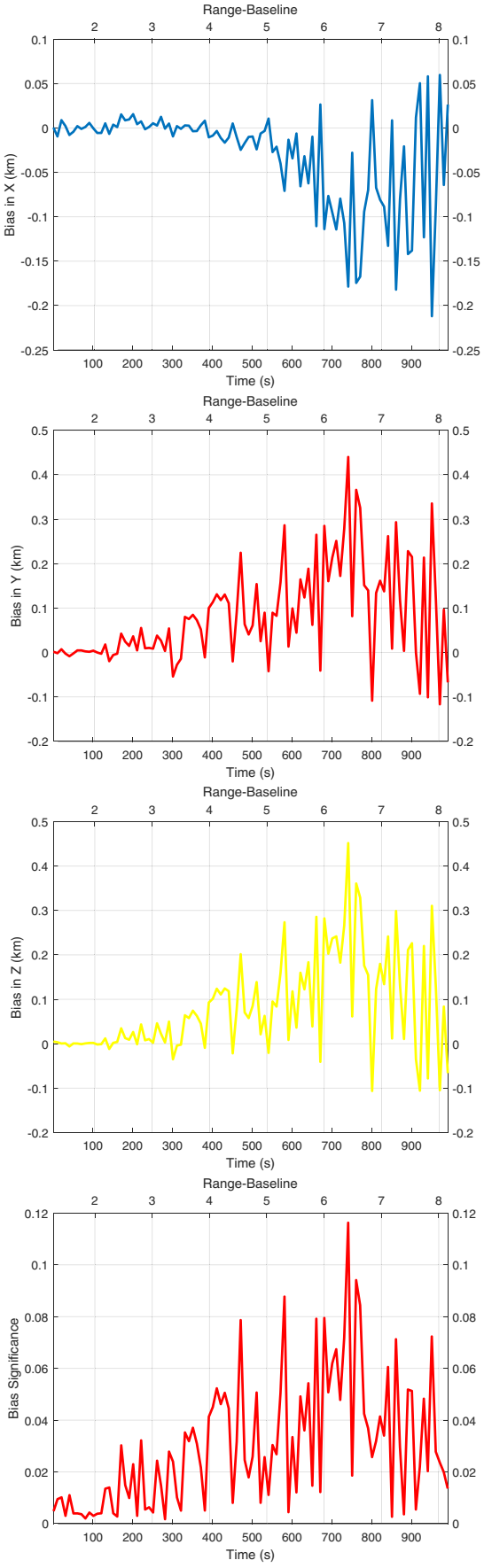


Fig. 3. Conversion bias and its significance over time.

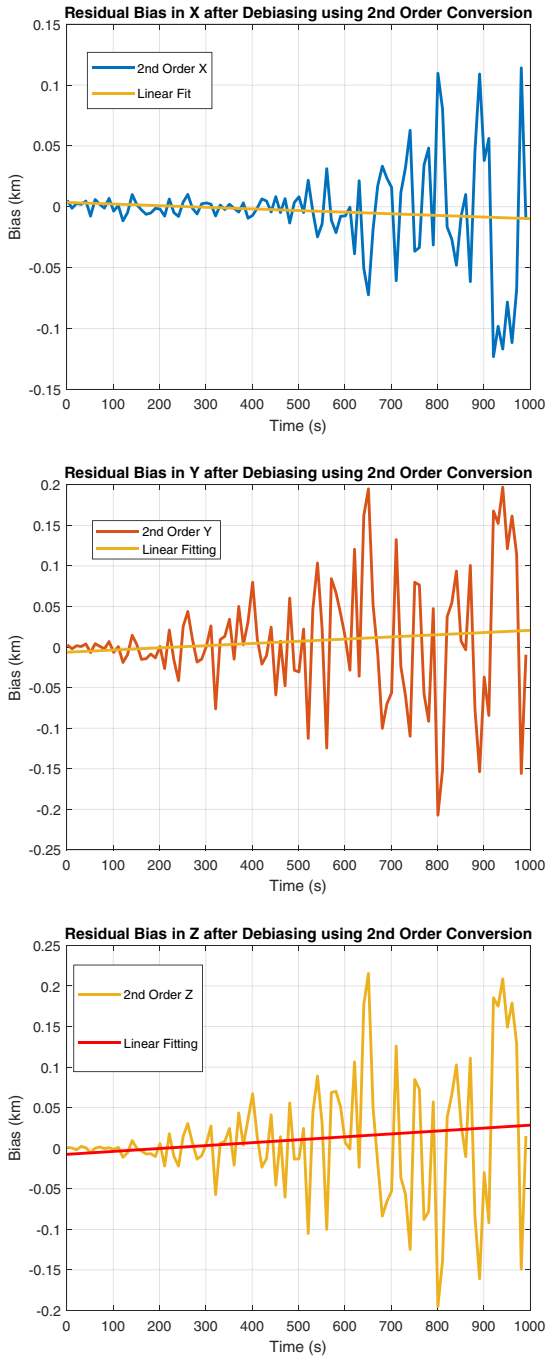


Fig. 4. Results of debiasing using the second-order conversion.

crease in the mean square error. The results of debiasing are seen in Fig. 4.

After debiasing, the mean of the results is appropriately centered on zero, meaning a significant improvement in measurement consistency is made. Higher order Taylor expansions may lead to more accurate debiasing.

C. Covariance Analysis

The accuracy of the covariance matrix from equation (21) is also analyzed for the second-order conversion. A Monte Carlo simulation is made to achieve this. For each

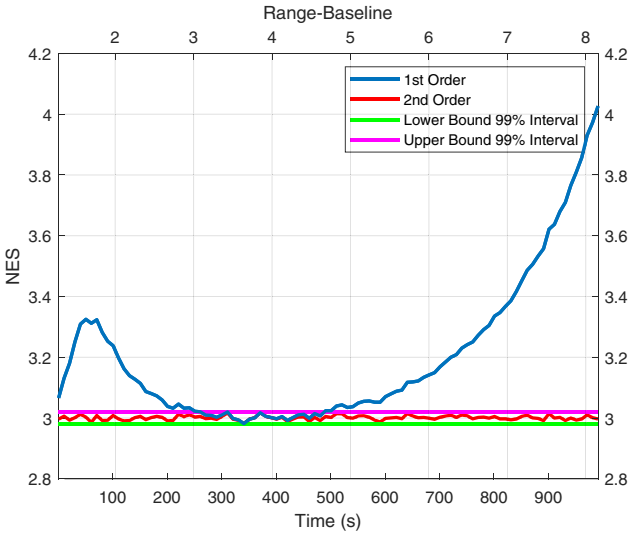


Fig. 5. NES First and Second Order Comparison of covariance NES.

Monte Carlo run, denoted by superscript n out of N_{MC}, a converted measurement is obtained and compared to the truth. The metric for analyzing the covariance is the Normalized Error Squared (NES). This is defined as

$$\tilde{\mathbf{x}}^t_{1,2}(k) = \mathbf{x}^{t,c,m,db,n}_{1,2}(k) - \mathbf{x}^t(k), \tag{30}$$

$$\mathrm{NES}(k) = \tfrac{1}{N_{\mathrm{MC}}} \textstyle\sum_{n=1}^{N_{\mathrm{MC}}} \tilde{\mathbf{x}}^t_{1,2}(k)' \mathbf{R}^{t,c,db,n}_{1,2}(k)^{-1} \tilde{\mathbf{x}}^t_{1,2}(k). \tag{31}$$

Comparisons of the NESs, both for the first- and for the second-order conversions to each other and to the 99% confidence region, are shown in Fig. 5. The NES for the first-order conversion is significantly higher than the confidence interval, meaning that the covariance calculated is not accurately containing the measurement points. In this case, the covariance is too small. The second-order conversion very accurately remedies this problem and results in a consistent NES. This calculated covariance matrix can be safely used to represent the converted Cartesian measurements.

D. Comparison With ML Conversion

In the previous sections, the proposed method is shown to remove nearly all bias and calculates an accurate, albeit pessimistic, covariance matrix. However, it is important to consider comparing the method with the already present method of generating composite measurements using ML. This method is presented in [14] and involves implementation of the ML using Iterated Least-Squares (ILS). Intuitively, one can deduce that the ML method should produce more accurate results for Cartesian coordinates from fusing two angle-only measurements because it is efficient compared to the Cramér–Rao Lower Bound (CRLB). However, the proposed method has an advantage in that it is an explicit (noniterative) expression of the Cartesian posi-

Table II
Comparison of Computation Times

	Proposed explicit conversion method	ML one-iteration conversion method	ML ten-iterations conversion method
$K = 100$ s	0.0084 s	0.0165 s	0.0966 s
$K = 1000$ s	0.0755 s	0.1401 s	0.9228 s

tion based on two line-of-sight measurements in three dimensions as opposed to a search to obtain the (iterative) MLE of the Cartesian position. This also means the Jacobian of the converted Cartesian coordinates with respect to the original angle-only measurements can be calculated. In the case of ML, it is practically impossible to produce the Jacobian matrix as the "location" of the ML estimate's convergence point is not analytically related to the angle-only measurements. Furthermore, the proposed method is significantly faster than the ML estimate as no matrix multiplication is required, and the conversion is done in one step rather than requiring multiple iterations. In this section, the performance of the proposed method is compared to the ML method to verify the improvement in computation speed and examine the difference in standard deviation. Two experiments are made for comparison: one with the proposed method compared to one iteration of ILS in the ML method, and the other where the proposed method is compared to ten iterations of ILS in the ML method. The single iteration is the fastest the ML method can perform, but may lose some accuracy compared to using ten iterations to converge. Additionally, the methods are compared with more or fewer measurements. The same parameters are used from the previous simulations. Computation time is evaluated using MATLAB and is averaged over 100 Monte Carlo runs. The results are displayed in Table II and in Figs. 6 and 7. These figures present the time history of converted measurement errors along the trajectory (no filtering is carried out here).

The table shows that the computation time is significantly reduced by using the proposed method. The explicit conversion takes half the time compared to the ML method for one iteration, and hence naturally nearly 20 times less time for ten iterations. Additionally, the explicit conversion is nearly identical in performance to the ML method in terms of standard deviation and residual bias. The ML method is slightly better for situations with poor azimuth observability and situations with very good observability. The graph for the single-iteration ML method is neglected as it is nearly identical in performance to the ten-iteration ML method. The results show that the ML method can be favored in situations where computation speed is not a factor and an explicit expression of the conversion is not needed, but the explicit conversion is a very small reduction in performance if needed. Both methods have negligible bias, and

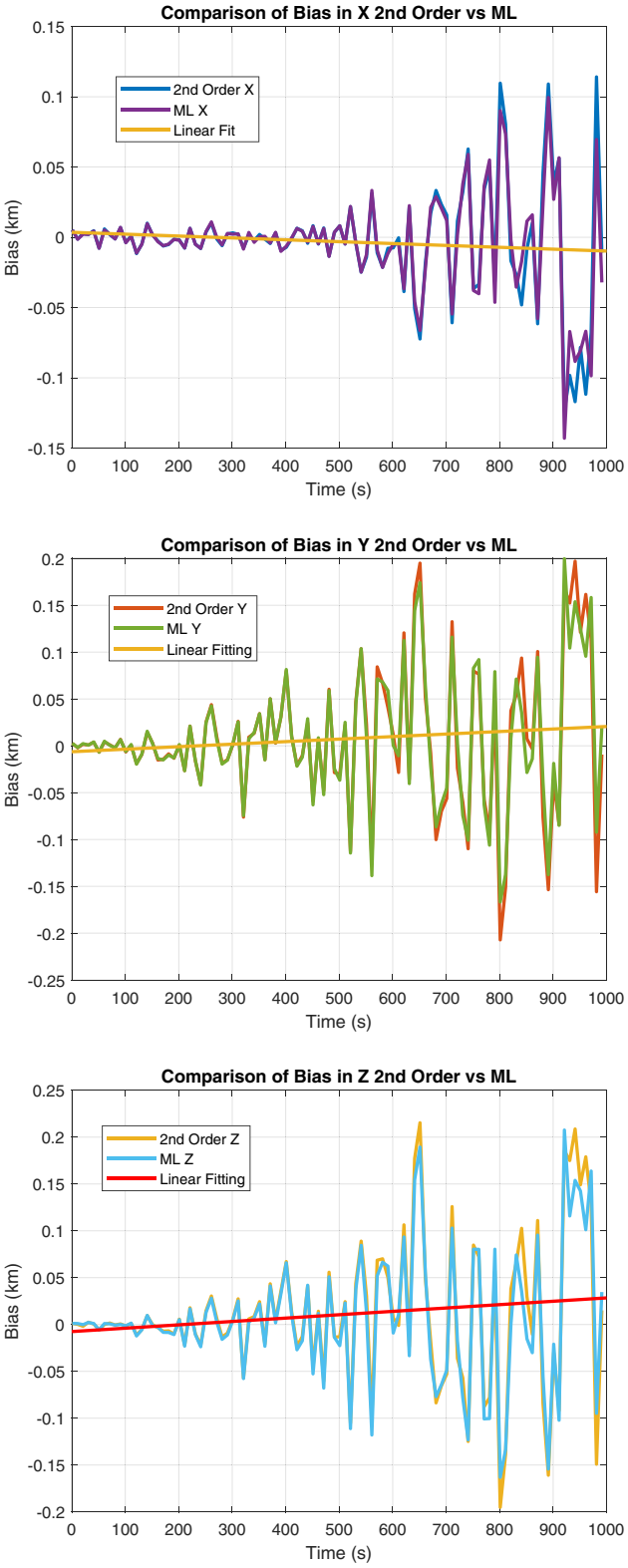


Fig. 6. Comparison of the biases in the proposed method and the ML method.

the residual bias is nearly the same. The results also imply that tracking methods, such as the EKF, can use these converted measurements without significant degradation compared to a mixed measurement filter tracking in Cartesian with angle-only measurments. Previous research in [11] has shown that in cases of high conversion

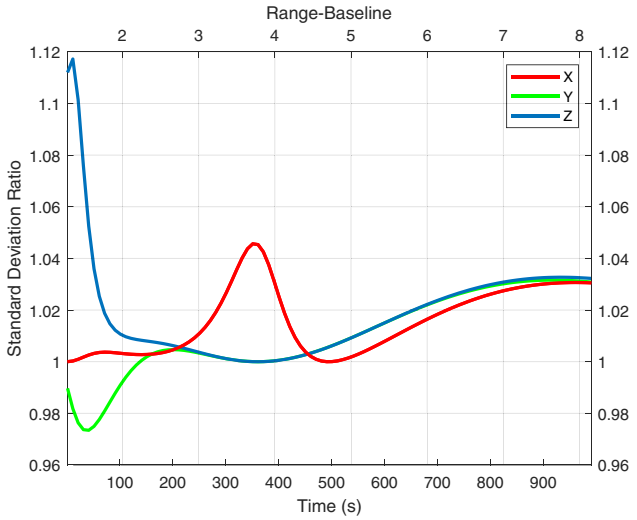


Fig. 7. Ratio of Cartesian coordinate standard deviations from the proposed method over the ML method using ten iterations.

bias significance (if done in the conventional way), the filter with measurements using the unbiased conversion performs better than the mixed measurement filter, as errors from nonlinearity can be present in the Jacobians.

V. CONCLUSION

When the observability of a target drops for passive sensors, it is necessary to account for higher order dynamics present in the conversion from angle-only measurements into Cartesian measurements. The bias present in the conversion can interfere with passive sensing applications such as target tracking and bias estimation. By using a second-order Taylor series expansion, it is possible to effectively remove the bias from converted measurements. Additionally, a more accurate model of the covariance of the converted noise is achieved. This method is useful as it is nearly equal in accuracy to ML methods, but also it is significantly faster and includes an explicit (noniterative) expression of the Cartesian position that can be used for applications. The bias present is relatively minimal and can be easily removed as well, meaning it is a robust conversion.

Future work with converted angle-only measurements would be to test this approach in situations incorporating additional real-world considerations. These would include asynchronous measurements and data association. Data association, in particular, is important as the method presented relies on correct associations for the formation of converted measurements. Asynchronous measurements must be propagated at the same time; thus, an additional source of error will be introduced. Therefore, the next step would be to integrate this approach to a target association method and determine how much error is likely from association errors, and how much error is obtained from the propagation of asynchronous measurements. Future work will also include analysis of the converted measurements with respect to an EKF, as this approach focuses on the conversion of individual measurements. Although the converted measurements have been shown to be effective relative to the ML solution, the inclusion of all of the nuances of tracking in an EKF such as process noise and target evolution models must be analyzed with respect to the coordinate conversion.

APPENDIX

A. *Introduction of the Conversion*

In order to calculate the unbiased conversion, it is necessary to derive the expressions for the converted measurements and then the derivatives of the converted measurements with respect to the angle measurements. First, the basics of the measurements are presented

$$x_s^t(k) = x^t(k) - x_s(k), \tag{32}$$

$$y_s^t(k) = y^t(k) - y_s(k), \tag{33}$$

$$z_s^t(k) = z^t(k) - z_s(k), \tag{34}$$

$$\alpha_s(k) = \text{atan2}\left(\frac{y_s^t(k)}{x_s^t(k)}\right), \tag{35}$$

$$\epsilon_s(k) = \text{atan2}\left(\frac{z_s^t(k)}{\sqrt{y_s^{t2}(k) + x_s^{t2}(k)}}\right). \tag{36}$$

The closest point of approach method involves converting the LOS measurements into a Cartesian ray. The ray for a sensor s is defined as

$$\mathbf{L}_s = \begin{bmatrix} \cos(\alpha_s)\cos(\epsilon_s) \\ \sin(\alpha_s)\cos(\epsilon_s) \\ \sin(\epsilon_s) \end{bmatrix}. \tag{37}$$

The sensor positions are assumed to be known, and the line between them is used to determine the closest point of approach

$$\mathbf{B}_{1,2} = \mathbf{x}_2 - \mathbf{x}_1 = \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \end{bmatrix}. \tag{38}$$

The closest point of approach for each sensor using the derivation in [1] and multiplied by the LOS ray to find the Cartesian positions on each LOS that are closest to each other. These are shifted by the sensor positions to place them within the same reference frame.

$$\mathbf{x}_1^{t,c}(k) = \begin{bmatrix} x_1^{t,c}(k) \\ y_1^{t,c}(k) \\ z_1^{t,c}(k) \end{bmatrix} = \mathbf{x}_1(k) + \frac{\mathbf{L}_1(k)(\mathbf{L}_1(k)'\mathbf{B}_{1,2}(k)) - (\mathbf{L}_1(k)'\mathbf{L}_2(k))(\mathbf{L}_2(k)'\mathbf{B}_{1,2}(k))}{1-(\mathbf{L}_1(k)'\mathbf{L}_2(k))^2}, \tag{39}$$

$$\mathbf{x}_2^{t,c}(k) = \begin{bmatrix} x_2^{t,c}(k) \\ y_2^{t,c}(k) \\ z_2^{t,c}(k) \end{bmatrix}$$

$$= \mathbf{x}_2(k) + \frac{\mathbf{L}_2(k)(\mathbf{L}_1(k)'\mathbf{L}_2(k))(\mathbf{L}_1(k)'\mathbf{B}_{1,2}(k)) - (\mathbf{L}_2(k)'\mathbf{B}_{1,2}(k))}{1-(\mathbf{L}_1(k)'\mathbf{L}_2(k))^2}, \quad (40)$$

$$\mathbf{x}_{1,2}^{t,c}(k) = \tfrac{1}{2}\left(\mathbf{x}_1^{t,c}(k) + \mathbf{x}_2^{t,c}(k)\right). \quad (41)$$

The conversion relies on the Cartesian vectors of the lines of sight defined as **L** and the line between the sensors defined as **B**. These are calculated as

$$\mathbf{L}_1(k) = \begin{bmatrix} \cos(\alpha_1(k))\cos(\epsilon_1(k)) \\ \sin(\alpha_1(k))\cos(\epsilon_1(k)) \\ \sin(\epsilon_1(k)) \end{bmatrix}, \quad (42)$$

$$\mathbf{L}_2(k) = \begin{bmatrix} \cos(\alpha_2(k))\cos(\epsilon_2(k)) \\ \sin(\alpha_2(k))\cos(\epsilon_2(k)) \\ \sin(\epsilon_2(k)) \end{bmatrix}, \quad (43)$$

$$\mathbf{B}_{1,2}(k) = \mathbf{x}_2(k) - \mathbf{x}_1(k) = \begin{bmatrix} x_2(k) - x_1(k) \\ y_2(k) - y_1(k) \\ z_2(k) - z_1(k) \end{bmatrix}. \quad (44)$$

For simplicity in the appendix the following substitutions are made:

$$N_1 = \mathbf{L}_1'\mathbf{B}_{1,2} - (\mathbf{L}_1'\mathbf{L}_2)(\mathbf{L}_2'\mathbf{B}_{1,2}), \quad (45)$$

$$N_2 = (\mathbf{L}_1'\mathbf{L}_2)(\mathbf{L}_1'\mathbf{B}_{1,2}) - \mathbf{L}_2'\mathbf{B}_{1,2}, \quad (46)$$

$$N_{x,1} = L_{x,1}N_1, \quad (47)$$

$$N_{y,1} = L_{y,1}N_1, \quad (48)$$

$$N_{z,1} = L_{z,1}N_1, \quad (49)$$

$$N_{x,2} = L_{x,2}N_2, \quad (50)$$

$$N_{y,2} = L_{y,2}N_2, \quad (51)$$

$$N_{z,2} = L_{z,2}N_2, \quad (52)$$

$$D = 1 - (\mathbf{L}_1'\mathbf{L}_2)^2, \quad (53)$$

which results in the equations

$$x_1^{t,c} = x_1 + \frac{N_{x,1}}{D}, \quad (54)$$

$$x_2^{t,c} = x_2 + \frac{N_{x,2}}{D}, \quad (55)$$

$$y_1^{t,c} = y_1 + \frac{N_{y,1}}{D}, \quad (56)$$

$$y_2^{t,c} = y_2 + \frac{N_{y,2}}{D}, \quad (57)$$

$$z_1^{t,c} = z_1 + \frac{N_{z,1}}{D}, \quad (58)$$

$$z_2^{t,c} = z_2 + \frac{N_{z,2}}{D}, \quad (59)$$

$$x_{1,2}^{t,c} = \frac{(x_1^{t,c} + x_2^{t,c})}{2}, \quad (60)$$

$$y_{1,2}^{t,c} = \frac{(y_1^{t,c} + y_2^{t,c})}{2}, \quad (61)$$

$$z_{1,2}^{t,c} = \frac{(z_1^{t,c} + z_2^{t,c})}{2}. \quad (62)$$

By making these substitutions, it is possible to use calculus rules to more efficiently represent the derivatives of the composite measurements.

B. Debiasing Calculation

It is necessary to calculate the bias for each Cartesian coordinate by using a Taylor series expansion to include the noise variables. For terseness, the symbol ζ is used to represent any of the Cartesian coordinates.

$$\zeta = x,\ y,\ z, \quad (63)$$

$$\begin{aligned} \zeta_{1,2}^{t,c,m}(k) \approx{}& \zeta^t(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_1} w_1^\alpha(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_2} w_2^\alpha(k) \\ &+ \frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_1} w_1^\epsilon(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_2} w_2^\epsilon(k) \\ &+ 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1^2} w_1^\alpha(k)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2^2} w_2^\alpha(k)^2 \\ &+ 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1^2} w_1^\epsilon(k)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_2^2} w_2^\epsilon(k)^2 \\ &+ \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2} w_1^\alpha(k) w_2^\alpha(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1} w_1^\alpha(k) w_1^\epsilon(k) \\ &+ \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2} w_1^\alpha(k) w_2^\epsilon(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1} w_2^\alpha(k) w_1^\epsilon(k) \\ &+ \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2} w_2^\alpha(k) w_2^\epsilon(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2} w_1^\epsilon(k) w_2^\epsilon(k). \end{aligned} \quad (64)$$

The bias is defined as the difference between the truth and the expected value of the converted measurement. The first-order terms are eliminated from the bias, but the second-order terms contribute to the mean:

$$\begin{aligned} E[\zeta_{1,2}^{t,c,m}(k)] \approx{}& \zeta^t(k) + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1^2}(\sigma_1^\alpha)^2 \\ &+ 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2^2}(\sigma_2^\alpha)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1^2}(\sigma_1^\epsilon)^2 \\ &+ 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_2^2}(\sigma_2^\epsilon)^2, \end{aligned} \quad (65)$$

$$c_{\zeta,1,2} = 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1^2}(\sigma_1^\alpha)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2^2}(\sigma_2^\alpha)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1^2}(\sigma_1^\epsilon)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_2^2}(\sigma_2^\epsilon)^2, \quad (66)$$

$$E[\zeta_{1,2}^{t,c,m}(k)] \approx \zeta^t(k) + c_{\zeta,1,2}, \quad (67)$$

$$\zeta_{1,2}^{t,c,m,db}(k) = \zeta_{1,2}^{t,c,m}(k) - c_{\zeta,1,2}, \quad (68)$$

$$E[\zeta_{1,2}^{t,c,m,db}(k)] \approx \zeta^t(k). \quad (69)$$

C. Variance Calculation

The variance is defined as the expected value of the measurement squared minus the mean squared:

$$\mathrm{Var}(\zeta_{1,2}^{t,c,m,db}(k)) = E[\zeta_{1,2}^{t,c,m,db}(k)^2] - E[\zeta_{1,2}^{t,c,m,db}(k)]^2. \quad (70)$$

The debiasing is included in order to find the variance of the debiased measurements

$$\begin{aligned}
&\mathrm{Var}(\zeta_{1,2}^{t,c,m,db}(k)) \\
&= E\Bigg[\Bigg(\zeta^t(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_1} w_1^\alpha(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_2} w_2^\alpha(k) \\
&\quad + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_1} w_1^\epsilon(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_2} w_2^\epsilon(k) + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1^2} w_1^\alpha(k)^2 \\
&\quad + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2^2} w_2^\alpha(k)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1^2} w_1^\epsilon(k)^2 \\
&\quad + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_2^2} w_2^\epsilon(k)^2 + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2} w_1^\alpha(k) w_2^\alpha(k) \\
&\quad + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1} w_1^\alpha(k) w_1^\epsilon(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2} w_1^\alpha(k) w_2^\epsilon(k) \\
&\quad + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1} w_2^\alpha(k) w_1^\epsilon(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2} w_2^\alpha(k) w_2^\epsilon(k) \\
&\quad + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2} w_1^\epsilon(k) w_2^\epsilon(k) - c_{\zeta,1,2}\Bigg)^2\Bigg] - \zeta^t(k)^2.
\end{aligned} \quad (71)$$

The debiasing will remove some of the terms, so they are separated from the equation:

$$\begin{aligned}
&V(\zeta_{1,2}^{t,c,m,db}(k)) = \\
&\Bigg(\zeta^t(k) + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1^2}(\sigma_1^\alpha)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2^2}(\sigma_2^\alpha)^2 \\
&+0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1^2}(\sigma_1^\epsilon)^2 + 0.5\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_2^2}(\sigma_2^\epsilon)^2 - c_{\zeta,1,2}\Bigg)^2 \\
&-\zeta^t(k)^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_1}\right)^2(\sigma_1^\alpha)^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_2}\right)^2(\sigma_2^\alpha)^2 \\
&+ \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_1}\right)^2(\sigma_1^\epsilon)^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_2}\right)^2(\sigma_2^\epsilon)^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)^2(\sigma_1^\alpha)^2(\sigma_2^\alpha)^2 + \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)^2(\sigma_1^\alpha)^2(\sigma_1^\epsilon)^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)^2(\sigma_1^\alpha)^2(\sigma_2^\epsilon)^2 + \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)^2(\sigma_2^\alpha)^2(\sigma_1^\epsilon)^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)^2(\sigma_2^\alpha)^2(\sigma_2^\epsilon)^2 + \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)^2(\sigma_1^\epsilon)^2(\sigma_2^\epsilon)^2.
\end{aligned} \quad (72)$$

The final variance equation is as follows:

$$\begin{aligned}
&V(\zeta_{1,2}^{t,c,m,db}(k)) = \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_1}\right)^2(\sigma_1^\alpha)^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_2}\right)^2(\sigma_2^\alpha)^2 \\
&+ \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_1}\right)^2(\sigma_1^\epsilon)^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_2}\right)^2(\sigma_2^\epsilon)^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)^2(\sigma_1^\alpha)^2(\sigma_2^\alpha)^2 + \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)^2(\sigma_1^\alpha)^2(\sigma_1^\epsilon)^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)^2(\sigma_1^\alpha)^2(\sigma_2^\epsilon)^2 + \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)^2(\sigma_2^\alpha)^2(\sigma_1^\epsilon)^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)^2(\sigma_2^\alpha)^2(\sigma_2^\epsilon)^2 + \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)^2(\sigma_1^\epsilon)^2(\sigma_2^\epsilon)^2.
\end{aligned} \quad (73)$$

D. Covariance Calculation

Similarly, the covariance is calculated for each combination of two Cartesian coordinates. As before, ζ is used to represent any particular Cartesian coordinate. γ is used to represent a different Cartesian coordinate. The calculation begins with the definition

$$\zeta = x,\ y,\ z, \quad (74)$$

$$\gamma = x,\ y,\ z, \quad (75)$$

$$\gamma \neq \zeta, \quad (76)$$

$$\begin{aligned}
&CV(\zeta_{1,2}^{t,c,m,db}(k), \gamma_{1,2}^{t,c,m,db}(k)) \\
&\quad = E[(\zeta_{1,2}^{t,c,m,db}(k) - E[\zeta_{1,2}^{t,c,m,db}(k)]) \\
&\quad\quad \times (\gamma_{1,2}^{t,c,m,db}(k) - E[\gamma_{1,2}^{t,c,m,db}(k)])] \\
&= E[\zeta_{1,2}^{t,c,m,db}(k)\gamma_{1,2}^{t,c,m,db}(k)] \\
&\quad\quad - E[\zeta_{1,2}^{t,c,m,db}(k)]E[\gamma_{1,2}^{t,c,m,db}(k)].
\end{aligned} \quad (77)$$

This is expanded with the second-order conversion

$$
\begin{aligned}
&\mathrm{Cov}(\zeta_{1,2}^{t,c,m,db}(k),\, \gamma_{1,2}^{t,c,m,db}(k)) \\
&= E\Bigg[\Bigg(\zeta^t(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_1} w_1^{\alpha}(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_2} w_2^{\alpha}(k) + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_1} w_1^{\epsilon}(k) \\
&\quad + \frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_2} w_2^{\epsilon}(k) + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1^2} w_1^{\alpha}(k)^2 + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2^2} w_2^{\alpha}(k)^2 \\
&\quad + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1^2} w_1^{\epsilon}(k)^2 + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_2^2} w_2^{\epsilon}(k)^2 \\
&\quad + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2} w_1^{\alpha}(k) w_2^{\alpha}(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1} w_1^{\alpha}(k) w_1^{\epsilon}(k) \\
&\quad + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2} w_1^{\alpha}(k) w_2^{\epsilon}(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1} w_2^{\alpha}(k) w_1^{\epsilon}(k) \\
&\quad + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2} w_2^{\alpha}(k) w_2^{\epsilon}(k) + \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2} w_1^{\epsilon}(k) w_2^{\epsilon}(k) - c_{\zeta,1,2}\Bigg) \\
&\quad \times \Bigg(\gamma^t(k) + \frac{\partial \gamma_{1,2}^{t,c}}{\partial \alpha_1} w_1^{\alpha}(k) + \frac{\partial \gamma_{1,2}^{t,c}}{\partial \alpha_2} w_2^{\alpha}(k) + \frac{\partial \gamma_{1,2}^{t,c}}{\partial \epsilon_1} w_1^{\epsilon}(k) \\
&\quad + \frac{\partial \gamma_{1,2}^{t,c}}{\partial \epsilon_2} w_2^{\epsilon}(k) + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1^2} w_1^{\alpha}(k)^2 + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2^2} w_2^{\alpha}(k)^2 \\
&\quad + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \epsilon_1^2} w_1^{\epsilon}(k)^2 + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \epsilon_2^2} w_2^{\epsilon}(k)^2 \\
&\quad + \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2} w_1^{\alpha}(k) w_2^{\alpha}(k) + \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1} w_1^{\alpha}(k) w_1^{\epsilon}(k) \\
&\quad + \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2} w_1^{\alpha}(k) w_2^{\epsilon}(k) + \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1} w_2^{\alpha}(k) w_1^{\epsilon}(k) \\
&\quad + \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2} w_2^{\alpha}(k) w_2^{\epsilon}(k) + \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2} w_1^{\epsilon}(k) w_2^{\epsilon}(k) - c_{\gamma,1,2}\Bigg)\Bigg] \\
&\quad - \zeta^t(k)\gamma^t(k). \qquad (78)
\end{aligned}
$$

The debiasing will remove some of the terms, so they are separated from the equation:

$$
\begin{aligned}
&\mathrm{Cov}(\zeta_{1,2}^{t,c,m,db}(k),\, \gamma_{1,2}^{t,c,m,db}(k)) \\
&= \Bigg(\zeta^t(k) + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1^2} (\sigma_1^{\alpha})^2 + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2^2} (\sigma_2^{\alpha})^2 \\
&\quad + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1^2} (\sigma_1^{\epsilon})^2 + 0.5 \frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_2^2} (\sigma_2^{\epsilon})^2 - c_{\zeta,1,2}\Bigg) \\
&\quad \times \Bigg(\gamma^t(k) + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1^2} (\sigma_1^{\alpha})^2 + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2^2} (\sigma_2^{\alpha})^2 \\
&\quad + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \epsilon_1^2} (\sigma_1^{\epsilon})^2 + 0.5 \frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \epsilon_2^2} (\sigma_2^{\epsilon})^2 - c_{\gamma,1,2}\Bigg)
\end{aligned}
$$

$$
\begin{aligned}
&+ \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_1}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \alpha_1}\right)(\sigma_1^{\alpha})^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_2}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \alpha_2}\right)(\sigma_2^{\alpha})^2 \\
&+ \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_1}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \epsilon_1}\right)(\sigma_1^{\epsilon})^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_2}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \epsilon_2}\right)(\sigma_2^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)(\sigma_1^{\alpha})^2(\sigma_2^{\alpha})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)(\sigma_1^{\alpha})^2(\sigma_1^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)(\sigma_1^{\alpha})^2(\sigma_2^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)(\sigma_2^{\alpha})^2(\sigma_1^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)(\sigma_2^{\alpha})^2(\sigma_2^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)(\sigma_1^{\epsilon})^2(\sigma_2^{\epsilon})^2. \qquad (79)
\end{aligned}
$$

The final covariance equation is as follows:

$$
\begin{aligned}
&\mathrm{Cov}(\zeta_{1,2}^{t,c,m,db}(k),\, \gamma_{1,2}^{t,c,m,db}(k)) \\
&= \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_1}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \alpha_1}\right)(\sigma_1^{\alpha})^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \alpha_2}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \alpha_2}\right)(\sigma_2^{\alpha})^2 \\
&+ \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_1}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \epsilon_1}\right)(\sigma_1^{\epsilon})^2 + \left(\frac{\partial \zeta_{1,2}^{t,c}}{\partial \epsilon_2}\right)\left(\frac{\partial \gamma_{1,2}^{t,c}}{\partial \epsilon_2}\right)(\sigma_2^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \alpha_2}\right)(\sigma_1^{\alpha})^2(\sigma_2^{\alpha})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_1}\right)(\sigma_1^{\alpha})^2(\sigma_1^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_1 \partial \epsilon_2}\right)(\sigma_1^{\alpha})^2(\sigma_2^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_1}\right)(\sigma_2^{\alpha})^2(\sigma_1^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \alpha_2 \partial \epsilon_2}\right)(\sigma_2^{\alpha})^2(\sigma_2^{\epsilon})^2 \\
&+ \left(\frac{\partial^2 \zeta_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)\left(\frac{\partial^2 \gamma_{1,2}^{t,c}}{\partial \epsilon_1 \partial \epsilon_2}\right)(\sigma_1^{\epsilon})^2(\sigma_2^{\epsilon})^2. \qquad (80)
\end{aligned}
$$

E. Derivatives of Converted Measurements

It is necessary to calculate the derivatives of the converted measurement with respect to the original measurements. This process begins with the first-order

derivatives. The derivatives are defined in terms of the substitutions made earlier. Furthermore, terms ϕ is included for terseness to represent any of the individual LOS measurements.

$$N_1 = \mathbf{L}_1'\mathbf{B}_{1,2} - (\mathbf{L}_1'\mathbf{L}_2)(\mathbf{L}_2'\mathbf{B}_{1,2}), \tag{81}$$

$$N_2 = (\mathbf{L}_1'\mathbf{L}_2)(\mathbf{L}_1'\mathbf{B}_{1,2}) - \mathbf{L}_2'\mathbf{B}_{1,2,} \tag{82}$$

$$\mathbf{N}_1 = N_1\mathbf{L}_1, \tag{83}$$

$$\mathbf{N}_2 = N_2\mathbf{L}_2, \tag{84}$$

$$D = 1 - (\mathbf{L}_1'\mathbf{L}_2)^2, \tag{85}$$

which results in the equations

$$\nabla\mathbf{N}_1 = \mathbf{L}_1'\nabla N_1 + N_1\nabla\mathbf{L}_1, \tag{86}$$

$$\nabla\mathbf{N}_2 = \mathbf{L}_2'\nabla N_2 + N_2\nabla\mathbf{L}_2, \tag{87}$$

$$\mathbf{x}_1^{t,c} = \mathbf{x}_1 + \frac{\mathbf{N}_1}{D}, \tag{88}$$

$$\mathbf{x}_2^{t,c} = \mathbf{x}_2 + \frac{\mathbf{N}_2}{D}, \tag{89}$$

$$\mathbf{x}_{1,2}^{t,c} = \frac{(\mathbf{x}_1^{t,c} + \mathbf{x}_2^{t,c})}{2}, \tag{90}$$

$$\nabla\mathbf{x}_{1,2}^{t,c} = \frac{1}{2}(\nabla\mathbf{x}_1^{t,c} + \nabla\mathbf{x}_2^{t,c}), \tag{91}$$

$$\nabla\mathbf{x}_1^{t,c} = \frac{D\nabla\mathbf{N}_1 - \mathbf{N}_1'\nabla D}{D^2}, \tag{92}$$

$$\nabla\mathbf{x}_2^{t,c} = \frac{D\nabla\mathbf{N}_2 - \mathbf{N}_2'\nabla D}{D^2}. \tag{93}$$

Gradients are used to simplify the equations, with an example being

$$\nabla\mathbf{x}_{1,2}^{t,c} = \begin{bmatrix} \frac{\partial x_{1,2}^{t,c}}{\partial\alpha_1} & \frac{\partial x_{1,2}^{t,c}}{\partial\epsilon_1} & \frac{\partial x_{1,2}^{t,c}}{\partial\alpha_2} & \frac{\partial x_{1,2}^{t,c}}{\partial\epsilon_2} \\ \frac{\partial y_{1,2}^{t,c}}{\partial\alpha_1} & \frac{\partial y_{1,2}^{t,c}}{\partial\epsilon_1} & \frac{\partial y_{1,2}^{t,c}}{\partial\alpha_2} & \frac{\partial y_{1,2}^{t,c}}{\partial\epsilon_2} \\ \frac{\partial z_{1,2}^{t,c}}{\partial\alpha_1} & \frac{\partial z_{1,2}^{t,c}}{\partial\epsilon_1} & \frac{\partial z_{1,2}^{t,c}}{\partial\alpha_2} & \frac{\partial z_{1,2}^{t,c}}{\partial\epsilon_2} \end{bmatrix}, \tag{94}$$

$$\nabla N_1 = \mathbf{B}_{1,2}'\nabla\mathbf{L}_1 - (\mathbf{L}_1'\mathbf{L}_2)(\mathbf{B}_{1,2}'\nabla\mathbf{L}_2) - (\mathbf{L}_2'\mathbf{B}_{1,2})(\mathbf{L}_1'\nabla\mathbf{L}_2 + \mathbf{L}_2'\nabla\mathbf{L}_1), \tag{95}$$

$$\nabla N_2 = -\mathbf{B}_{1,2}'\nabla\mathbf{L}_2 + (\mathbf{L}_1'\mathbf{L}_2)(\mathbf{B}_{1,2}'\nabla\mathbf{L}_1) + (\mathbf{L}_1'\mathbf{B}_{1,2})(\mathbf{L}_1'\nabla\mathbf{L}_2 + \mathbf{L}_2'\nabla\mathbf{L}_1), \tag{96}$$

$$\nabla D = -2(\mathbf{L}_1'\mathbf{L}_2)(\mathbf{L}_1'\nabla\mathbf{L}_2 + \mathbf{L}_2'\nabla\mathbf{L}_1), \tag{97}$$

$$\nabla\mathbf{L}_1 = \begin{bmatrix} \frac{\partial L_{x,1}}{\partial\alpha_1} & \frac{\partial L_{x,1}}{\partial\epsilon_1} & 0 & 0 \\ \frac{\partial L_{y,1}}{\partial\alpha_1} & \frac{\partial L_{y,1}}{\partial\epsilon_1} & 0 & 0 \\ \frac{\partial L_{z,1}}{\partial\alpha_1} & \frac{\partial L_{z,1}}{\partial\epsilon_1} & 0 & 0 \end{bmatrix}, \tag{98}$$

$$\nabla\mathbf{L}_2 = \begin{bmatrix} 0 & 0 & \frac{\partial L_{x,2}}{\partial\alpha_2} & \frac{\partial L_{x,2}}{\partial\epsilon_2} \\ 0 & 0 & \frac{\partial L_{y,2}}{\partial\alpha_2} & \frac{\partial L_{y,2}}{\partial\epsilon_2} \\ 0 & 0 & \frac{\partial L_{z,2}}{\partial\alpha_2} & \frac{\partial L_{z,2}}{\partial\epsilon_2} \end{bmatrix}, \tag{99}$$

$$\frac{\partial L_{x,1}}{\partial\alpha_1} = -\cos(\epsilon_1)\sin(\alpha_1), \tag{100}$$

$$\frac{\partial L_{x,1}}{\partial\epsilon_1} = -\sin(\epsilon_1)\cos(\alpha_1), \tag{101}$$

$$\frac{\partial L_{y,1}}{\partial\alpha_1} = \cos(\epsilon_1)\cos(\alpha_1), \tag{102}$$

$$\frac{\partial L_{y,1}}{\partial\epsilon_1} = -\sin(\epsilon_1)\sin(\alpha_1), \tag{103}$$

$$\frac{\partial L_{z,1}}{\partial\alpha_1} = 0, \tag{104}$$

$$\frac{\partial L_{z,1}}{\partial\epsilon_1} = \cos(\epsilon_1), \tag{105}$$

$$\frac{\partial L_{x,2}}{\partial\alpha_2} = -\cos(\epsilon_2)\sin(\alpha_2), \tag{106}$$

$$\frac{\partial L_{x,2}}{\partial\epsilon_2} = -\sin(\epsilon_2)\cos(\alpha_2), \tag{107}$$

$$\frac{\partial L_{y,2}}{\partial\alpha_2} = \cos(\epsilon_2)\cos(\alpha_2), \tag{108}$$

$$\frac{\partial L_{y,2}}{\partial\epsilon_2} = -\sin(\epsilon_2)\sin(\alpha_2), \tag{109}$$

$$\frac{\partial L_{z,2}}{\partial\alpha_2} = 0, \tag{110}$$

$$\frac{\partial L_{z,2}}{\partial\epsilon_2} = \cos(\epsilon_2), \tag{111}$$

The second-order derivatives are calculated to be

$$\phi = \alpha_1,\ \alpha_2,\ \epsilon_1,\ \epsilon_2, \tag{112}$$

$$\frac{\partial\nabla\mathbf{x}_{1,2}^{t,c}}{\partial\phi} = \frac{1}{2}(\frac{\partial\nabla\mathbf{x}_1^{t,c}}{\partial\phi} + \frac{\partial\nabla\mathbf{x}_2^{t,c}}{\partial\phi}), \tag{113}$$

$$\frac{\partial\nabla\mathbf{x}_1^{t,c}}{\partial\phi} = \left(\frac{D^2\left(\frac{\partial D}{\partial\phi}\nabla\mathbf{N}_1 + D\frac{\partial\nabla\mathbf{N}_1}{\partial\phi} - \mathbf{N}_1'\frac{\partial\nabla D}{\partial\phi} - \frac{\partial\mathbf{N}_1}{\partial\phi}'\nabla D\right)}{D^4}\right) + \left(\frac{-2D\frac{\partial D}{\partial\phi}(D\nabla\mathbf{N}_1 - \mathbf{N}_1'\nabla D)}{D^4}\right), \tag{114}$$

$$\frac{\partial\nabla\mathbf{x}_2^{t,c}}{\partial\phi} = \left(\frac{D^2\left(\frac{\partial D}{\partial\phi}\nabla\mathbf{N}_2 + D\frac{\partial\nabla\mathbf{N}_2}{\partial\phi} - \mathbf{N}_2'\frac{\partial\nabla D}{\partial\phi} - \frac{\partial\mathbf{N}_2}{\partial\phi}'\nabla D\right)}{D^4}\right) + \left(\frac{-2D\frac{\partial D}{\partial\phi}(D\nabla\mathbf{N}_2 - \mathbf{N}_2'\nabla D)}{D^4}\right), \tag{115}$$

$$\frac{\partial \nabla \mathbf{N}_1}{\partial \phi} = \mathbf{B}'_{1,2}\frac{\partial \nabla \mathbf{L}_1}{\partial \phi} - \left(\frac{\partial \mathbf{L}_1}{\partial \phi}'\mathbf{L}_2 + \mathbf{L}'_1\frac{\partial \mathbf{L}_2}{\partial \phi}\right)(\mathbf{B}'_{1,2}\nabla \mathbf{L}_2)$$
$$-(\mathbf{L}'_1\mathbf{L}_2)\left(\mathbf{B}'_{1,2}\frac{\partial \nabla \mathbf{L}_2}{\partial \phi}\right)$$
$$-\left(\frac{\partial \mathbf{L}_2}{\partial \phi}'\mathbf{B}_{1,2}\right)(\mathbf{L}'_1\nabla \mathbf{L}_2 + \mathbf{L}'_2\nabla \mathbf{L}_1)$$
$$-(\mathbf{L}'_2\mathbf{B}_{1,2})\left(\frac{\partial \mathbf{L}_1}{\partial \phi}'\nabla \mathbf{L}_2 + \frac{\partial \mathbf{L}_2}{\partial \phi}'\nabla \mathbf{L}_1\right.$$
$$\left.+\mathbf{L}'_1\frac{\partial \nabla \mathbf{L}_2}{\partial \phi} + \mathbf{L}'_2\frac{\partial \nabla \mathbf{L}_1}{\partial \phi}\right), \tag{116}$$

$$\frac{\partial \nabla \mathbf{N}_2}{\partial \phi} = -\mathbf{B}'_{1,2}\frac{\partial \nabla \mathbf{L}_2}{\partial \phi} + \left(\frac{\partial \mathbf{L}_1}{\partial \phi}'\mathbf{L}_2 + \mathbf{L}'_1\frac{\partial \mathbf{L}_2}{\partial \phi}\right)(\mathbf{B}'_{1,2}\nabla \mathbf{L}_1)$$
$$+(\mathbf{L}'_1\mathbf{L}_2)\left(\mathbf{B}'_{1,2}\frac{\partial \nabla \mathbf{L}_1}{\partial \phi}\right)$$
$$+\left(\frac{\partial \mathbf{L}_1}{\partial \phi}'\mathbf{B}_{1,2}\right)(\mathbf{L}'_1\nabla \mathbf{L}_2 + \mathbf{L}'_2\nabla \mathbf{L}_1)$$
$$+(\mathbf{L}'_1\mathbf{B}_{1,2})\left(\frac{\partial \mathbf{L}_1}{\partial \phi}'\nabla \mathbf{L}_2 + \frac{\partial \mathbf{L}_2}{\partial \phi}'\nabla \mathbf{L}_1\right.$$
$$\left.+\mathbf{L}'_1\frac{\partial \nabla \mathbf{L}_2}{\partial \phi} + \mathbf{L}'_2\frac{\partial \nabla \mathbf{L}_1}{\partial \phi}\right), \tag{117}$$

$$\frac{\partial \nabla D}{\partial \phi} = -2\left(\frac{\partial \mathbf{L}_1}{\partial \phi}'\mathbf{L}_2 + \mathbf{L}'_1\frac{\partial \mathbf{L}_2}{\partial \phi}\right)(\mathbf{L}'_1\nabla \mathbf{L}_2 + \mathbf{L}'_2\nabla \mathbf{L}_1)$$
$$-2(\mathbf{L}'_1\mathbf{L}_2)\left(\frac{\partial \mathbf{L}_1}{\partial \phi}'\nabla \mathbf{L}_2 + \frac{\partial \mathbf{L}_2}{\partial \phi}'\nabla \mathbf{L}_1\right.$$
$$\left.+\mathbf{L}'_1\frac{\partial \nabla \mathbf{L}_2}{\partial \phi} + \mathbf{L}'_2\frac{\partial \nabla \mathbf{L}_1}{\partial \phi}\right), \tag{118}$$

$$\frac{\partial \nabla \mathbf{L}_1}{\partial \phi} = \begin{bmatrix} \frac{\partial^2 L_{x,1}}{\partial \alpha_1 \partial \phi} & \frac{\partial^2 L_{x,1}}{\partial \epsilon_1 \partial \phi} & 0 & 0 \\ \frac{\partial^2 L_{y,1}}{\partial \alpha_1 \partial \phi} & \frac{\partial^2 L_{y,1}}{\partial \epsilon_1 \partial \phi} & 0 & 0 \\ \frac{\partial^2 L_{z,1}}{\partial \alpha_1 \partial \phi} & \frac{\partial^2 L_{z,1}}{\partial \epsilon_1 \partial \phi} & 0 & 0 \end{bmatrix}, \tag{119}$$

$$\frac{\partial \nabla \mathbf{L}_2}{\partial \phi} = \begin{bmatrix} 0 & 0 & \frac{\partial^2 L_{x,2}}{\partial \alpha_2 \partial \phi} & \frac{\partial^2 L_{x,2}}{\partial \epsilon_2 \partial \phi} \\ 0 & 0 & \frac{\partial^2 L_{y,2}}{\partial \alpha_2 \partial \phi} & \frac{\partial^2 L_{y,2}}{\partial \epsilon_2 \partial \phi} \\ 0 & 0 & \frac{\partial^2 L_{z,2}}{\partial \alpha_2 \partial \phi} & \frac{\partial^2 L_{z,2}}{\partial \epsilon_2 \partial \phi} \end{bmatrix}, \tag{120}$$

$$\frac{\partial^2 L_{x,1}}{\partial^2 \alpha_1} = -\cos(\epsilon_1)\cos(\alpha_1), \tag{121}$$
$$\frac{\partial^2 L_{x,1}}{\partial^2 \epsilon_1} = -\cos(\epsilon_1)\cos(\alpha_1), \tag{122}$$
$$\frac{\partial^2 L_{y,1}}{\partial^2 \alpha_1} = -\cos(\epsilon_1)\sin(\alpha_1), \tag{123}$$
$$\frac{\partial^2 L_{y,1}}{\partial^2 \epsilon_1} = -\cos(\epsilon_1)\sin(\alpha_1), \tag{124}$$
$$\frac{\partial^2 L_{z,1}}{\partial^2 \alpha_1} = 0, \tag{125}$$
$$\frac{\partial^2 L_{z,1}}{\partial^2 \epsilon_1} = -\sin(\epsilon_1), \tag{126}$$
$$\frac{\partial^2 L_{x,1}}{\partial \alpha_1 \partial \epsilon_1} = \sin(\epsilon_1)\sin(\alpha_1), \tag{127}$$
$$\frac{\partial^2 L_{y,1}}{\partial \alpha_1 \partial \epsilon_1} = -\sin(\epsilon_1)\cos(\alpha_1), \tag{128}$$
$$\frac{\partial^2 L_{z,1}}{\partial \alpha_1 \partial \epsilon_1} = 0, \tag{129}$$
$$\frac{\partial^2 L_{x,1}}{\partial \alpha_1 \partial \alpha_2} = 0, \tag{130}$$
$$\frac{\partial^2 L_{x,1}}{\partial \alpha_1 \partial \epsilon_2} = 0, \tag{131}$$
$$\frac{\partial^2 L_{y,1}}{\partial \alpha_1 \partial \alpha_2} = 0, \tag{132}$$
$$\frac{\partial^2 L_{y,1}}{\partial \alpha_1 \partial \epsilon_2} = 0, \tag{133}$$
$$\frac{\partial^2 L_{z,1}}{\partial \alpha_1 \partial \alpha_2} = 0, \tag{134}$$
$$\frac{\partial^2 L_{z,1}}{\partial \alpha_1 \partial \epsilon_2} = 0, \tag{135}$$
$$\frac{\partial^2 L_{x,1}}{\partial \epsilon_1 \partial \alpha_2} = 0, \tag{136}$$
$$\frac{\partial^2 L_{x,1}}{\partial \epsilon_1 \partial \epsilon_2} = 0, \tag{137}$$
$$\frac{\partial^2 L_{y,1}}{\partial \epsilon_1 \partial \alpha_2} = 0, \tag{138}$$
$$\frac{\partial^2 L_{y,1}}{\partial \epsilon_1 \partial \epsilon_2} = 0, \tag{139}$$
$$\frac{\partial^2 L_{z,1}}{\partial \epsilon_1 \partial \alpha_2} = 0, \tag{140}$$
$$\frac{\partial^2 L_{z,1}}{\partial \epsilon_1 \partial \epsilon_2} = 0, \tag{141}$$
$$\frac{\partial^2 L_{x,2}}{\partial^2 \alpha_2} = -\cos(\epsilon_2)\cos(\alpha_2), \tag{142}$$
$$\frac{\partial^2 L_{x,2}}{\partial^2 \epsilon_2} = -\cos(\epsilon_2)\cos(\alpha_2), \tag{143}$$
$$\frac{\partial^2 L_{y,2}}{\partial^2 \alpha_2} = -\cos(\epsilon_2)\sin(\alpha_2), \tag{144}$$
$$\frac{\partial^2 L_{y,2}}{\partial^2 \epsilon_2} = -\cos(\epsilon_2)\sin(\alpha_2), \tag{145}$$
$$\frac{\partial^2 L_{z,2}}{\partial^2 \alpha_2} = 0, \tag{146}$$

$$\frac{\partial^2 L_{z,2}}{\partial^2 \epsilon_2} = -\sin(\epsilon_2), \tag{147}$$

$$\frac{\partial^2 L_{x,2}}{\partial \alpha_2 \partial \epsilon_2} = \sin(\epsilon_2)\sin(\alpha_2), \tag{148}$$

$$\frac{\partial^2 L_{y,2}}{\partial \alpha_2 \partial \epsilon_2} = -\sin(\epsilon_2)\cos(\alpha_2), \tag{149}$$

$$\frac{\partial^2 L_{z,2}}{\partial \alpha_2 \partial \epsilon_2} = 0, \tag{150}$$

$$\frac{\partial^2 L_{x,2}}{\partial \alpha_1 \partial \alpha_2} = 0, \tag{151}$$

$$\frac{\partial^2 L_{x,2}}{\partial \alpha_1 \partial \epsilon_2} = 0, \tag{152}$$

$$\frac{\partial^2 L_{y,2}}{\partial \alpha_1 \partial \alpha_2} = 0, \tag{153}$$

$$\frac{\partial^2 L_{y,2}}{\partial \alpha_1 \partial \epsilon_2} = 0, \tag{154}$$

$$\frac{\partial^2 L_{z,2}}{\partial \alpha_1 \partial \alpha_2} = 0, \tag{155}$$

$$\frac{\partial^2 L_{z,2}}{\partial \alpha_1 \partial \epsilon_2} = 0, \tag{156}$$

$$\frac{\partial^2 L_{x,2}}{\partial \epsilon_1 \partial \alpha_2} = 0, \tag{157}$$

$$\frac{\partial^2 L_{x,2}}{\partial \epsilon_1 \partial \epsilon_2} = 0, \tag{158}$$

$$\frac{\partial^2 L_{y,2}}{\partial \epsilon_1 \partial \alpha_2} = 0, \tag{159}$$

$$\frac{\partial^2 L_{y,2}}{\partial \epsilon_1 \partial \epsilon_2} = 0, \tag{160}$$

$$\frac{\partial^2 L_{z,2}}{\partial \epsilon_1 \partial \alpha_2} = 0, \tag{161}$$

$$\frac{\partial^2 L_{z,2}}{\partial \epsilon_1 \partial \epsilon_2} = 0. \tag{162}$$

REFERENCES

[1] "Closest point between two rays." Accessed: Jul. 25, 2019. [Online]. Available: http://morroworks.palitri.com/Content/Docs/Rays%20closest%20point.pdf.

[2] V. Aidala
"Kalman filter behavior in bearings-only tracking applications,"
IEEE Trans. Aerosp. Electron. Syst., vol. AES-15, no. 1, pp. 29–39, Jan. 1979.

[3] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan
Estimation With Applications to Tracking and Navigation: Theory, Algorithms and Software. New York, NY, USA: Wiley, 2001.

[4] D. Belfadel, R. Osborne, and Y. Bar-Shalom
"Bias estimation and observability for optical sensor measurements with targets of opportunity,"
J. Adv. Inf. Fusion, vol. 9, no. 2, pp. 59–74, Dec. 2014.

[5] H. Chen and F. Han
"Bias estimation for multiple passive sensors,"
Proc. Int. Conf. Measurement, Inf. Control, 2012, pp. 1081–1084.

[6] D. F. Crouse
"Cubature/unscented/sigma point Kalman filtering with angular measurement models,"
Proc. 18th Int. Conf. Inf. Fusion, 2015, pp. 1550–1557.

[7] B. Davis and W. D. Blair
"Gaussian mixture approach to long range radar tracking with high range resolution," in
Proc. IEEE Aerosp. Conf., 2015, pp. 1–9.

[8] Á. F. García-Fernández, S. Maskell, P. Horridge, and J. Ralph
"Gaussian tracking with Kent-distributed direction-of-arrival measurements,"
IEEE Trans. Veh. Technol., vol. 70, no. 7, pp. 7249–7254, Jul. 2021.

[9] M. Kowalski, Y. Bar-Shalom, P. Willett, and T. Fair
"Unbiased conversion of passive sensor measurements,"
Proc. IEEE 8th Int. Workshop Comput. Adv. Multi-Sensor Adaptive Process., 2019, pp. 505–509.

[10] M. Kowalski, Y. Bar-Shalom, P. Willett, B. Milgrom, and R. Ben-Dov
"CRLB for multi-sensor rotational bias estimation for passive sensors without target state estimation,"
Proc. SPIE, vol. 11018, 2019, Art. no. 1101805.

[11] D. Lerro and Y. Bar-Shalom
"Tracking with debiased consistent converted measurements versus EKF,"
IEEE Trans. Aerosp. Electron. Syst., vol. 29, no. 3, pp. 1015–1022, Jul. 1993.

[12] M. Longbin, S. Xiaoquan, Z. Yiyu, S. Zhong-Kang, and Y. Bar-Shalom
"Unbiased converted measurements for tracking,"
IEEE Trans. Aerosp. Electron. Syst., vol. 34, no. 3, pp. 1023–1027, Jul. 1998.

[13] N. Okello and B. Ristic
"Maximum likelihood registration for multiple dissimilar sensors,"
IEEE Trans. Aerosp. Electron. Syst., vol. 39, pp. 1074–1083, Aug. 2003.

[14] R. Osborne and Y. Bar-Shalom
"Statistical efficiency of composite position measurements from passive sensors,"
IEEE Trans. Aerosp. Electron. Syst., vol. 49, no. 4, pp. 2799–2806, Oct. 2013.

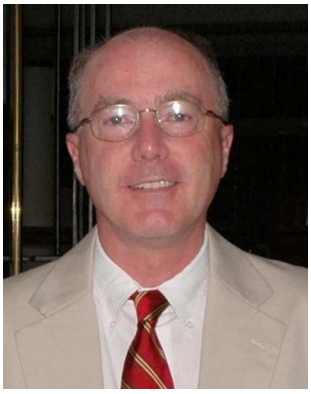
[15] Z. Sutton, P. Willett, T. Fair, and Y. Bar-Shalom
"MLPMH tracking in 3 dimensions using cluttered measurements from multiple 2-dimensional sensors,"
J. Adv. Inf. Fusion, vol. 16, no. 2, pp. 92–113, Dec. 2021.

[16] E. Taghavi, R. Tharmarasa, T. Kirubarajan, and M. McDonald
"Multisensor-multitarget bearing-only sensor registration,"
IEEE Trans. Aerosp. Electron. Syst., vol. 52, no. 4, pp. 1654–1666, Aug. 2016.

[17] X. Tian and Y. Bar-Shalom
"Coordinate conversion and tracking for very long range radars,"
IEEE Trans. Aerosp. Electron. Syst., vol. 45, no. 3, pp. 1073–1088, Jul. 2009.

Michael Kowalski received the B.Sc. and Ph.D. degrees in electrical engineering from the University of Connecticut, Storrs, CT, USA, in 2015 and 2020, respectively. He worked with his advisors Dr. Peter Willett and Dr. Yaakov Bar-Shalom in pursuit of his Ph.D. His research focuses on bias estimation as well as sensor fusion and target tracking. In August 2020, he joined the Georgia Tech Research Institute and is currently supporting the development of assessment tools and metrics in the Benchmark environment.

Yaakov Bar-Shalom (F'84) received the B.Sc. and M.Sc. degrees from the Technion, Haifa, Israel, in 1963 and 1967, respectively, and the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, USA, in 1970. He is currently a Board of Trustees Distinguished Professor with the ECE Department and Marianne E. Klewin Professor with the University of Connecticut. His current research interests are in estimation theory, target tracking, and data fusion. He has published more than 650 papers and book chapters. He coauthored/edited eight books, including *Tracking and Data Fusion* (YBS Publishing, 2011). He has been elected Fellow of IEEE for "contributions to the theory of stochastic systems and of multitarget tracking". He served as an Associate Editor for 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 has been a Distinguished Lecturer of the IEEE AESS. He is a corecipient of the M. Barry Carlton Award for the best paper in the IEEE TAESystems in 1995 and 2000. In 2002, he received 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 awarded 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 is 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. He has the following Wikipedia page: https://en.wikipedia.org/wiki/Yaakov Bar-Shalom.



Peter Willett (F'03) had been a faculty member in the Electrical and Computer Engineering Department, University of Connecticut, Storrs, CT, USA, since 1986. In 1998, he became a Professor. Since 2003, he has been an IEEE Fellow. His primary areas of research have been statistical signal processing, detection, machine learning, communications, data fusion, and tracking. He was the Chief Editor for the IEEE Aerospace and Electronic Systems Magazine from 2018 to 2020. He was the Editor-in-Chief for the IEEE Transactions on Aerospace and Electronic Systems from 2006 to 2011 and for the IEEE Signal Processing Letters from 2014 to 2016. He was also AESS Vice President for Publications from 2012 to 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 a member of the IEEE AESS Board of Governors and was Chair of the IEEE Signal Processing Society's Sensor-Array and Multichannel (SAM) technical committee.

Tim Fair, Toyon Senior Staff Analyst, Deputy Director Signal Processing, Principle Investigator, received a B.Sc. and M.Sc. degrees in electrical and computer engineering from the University of California, San Diego, CA, USA, with focus on signal and image processing, in 2008 and 2009, respectively. He has worked with SAIC and Johns Hopkins Applied Physics Lab, where he developed and analyzed detection, tracking, association, and discrimination algorithms for next-generation ISR platforms and Missile Defense Agency satellite systems and the Navy Aegis Ballistic Missile Defense program. Since joining Toyon in 2012, he has focused on algorithm development for image and video processing. As part of this work at Toyon, he has developed solutions in the fields of target detection and tracking, trajectory estimation, target pose estimation, and more recently on methods for applying deep learning and artificial intelligence in the military sector.

Optimal Threat-Based Radar Resource Management for Multitarget Joint Tracking and Classification

MAX IAN SCHÖPE
HANS DRIESSEN
ALEXANDER YAROVOY

The Radar Resource Management problem in a multitarget joint tracking and classification scenario is considered. The problem is solved using a previously introduced dynamic budget-balancing algorithm that models the sensor tasks as Partially Observable Markov Decision Processes. It is shown that tracking and classification tasks can be considered as a single task type. Furthermore, it is shown how the task resource allocations can be jointly optimized using a carefully formulated cost function based on the task threat variance. Multiple two-dimensional radar scenarios demonstrate how sensor resources are allocated depending on the current knowledge of the target position and class. In contrast to previous approaches, the novelty of this paper lies in combining tracking and classification performance into a single cost function, preventing heuristic trade-offs.

Manuscript received September 29, 2021; revised May 25, 2022; released for publication July 7, 2022.

The authors are with the Microwave Sensing, Signals and Systems (MS3), Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: m.i.schope@tudelft.nl; j.n.driessen@tudelft.nl; a.yarovoy@tudelft.nl).

1557-6418/22/$17.00

I. INTRODUCTION

Due to the recent developments in multifunction radar (MFR), such systems have become more flexible and allow an automatic adjustment of many of their parameters during runtime [18]. Possible situations where such an adaptation is desirable are, e.g., quickly changing weather conditions, target maneuvers, or interference. This automatic control of the radar parameters or resources is often named Radar Resource Management (RRM). It is generally considered as a part of so-called cognitive radar (see, e.g, [11], [14], [18], [23], [32]). Possible applications can be found in many domains, such as traffic monitoring, autonomous driving, air traffic control, or (maritime) surveillance.

A. Radar Resource Management

Many different overviews of RRM approaches are available, for instance, by Moo and Ding in [45], Hero and Cochran in [25], or Hintz in [26]. Most RRM research focuses on a single task, e.g., guaranteeing consistent track quality even under target maneuvers. This commonly means that the available time budget has to be scheduled for a specific task. However, MFR systems are usually considered to operate at their resource limit (w.r.t., e.g., time or energy) and deal with a large number of different tasks. This means that increasing the resource budget for one task automatically reduces it for the others and inevitably deteriorates their performance, making the RRM problem a balancing problem.

As a solution for multitask RRM problems, several heuristic solutions have been presented in the past (see, e.g., the overview in [29]). Applying heuristics too early in the design makes it difficult to understand which problem is supposed to be solved. Additionally, it is challenging to judge the optimality of that solution. Moreover, the heuristic solutions frequently schedule tasks based on different priorities (as shown, e.g., in [42] and [53]). Such approaches usually assume that each task has a specific fixed resource need, which frequently leads to a situation where tasks need to be dropped. If different tasks have the same fixed priority, then this can potentially lead to tasks being dropped at random. Additionally, prioritizing tasks is usually tricky, and often it is not clear how many priority levels are necessary.

It should be noted that RRM algorithms are not identical to multitarget tracking algorithms. In this approach, a multitarget tracker is applied to process the radar measurements and provide the estimated tracks to the RRM algorithm, which then optimizes future radar transmissions. Although the proposed RRM approach comprises computing expected track accuracies, it does not represent the actual multitarget tracking and will not automatically lead to maximized track accuracies for every task.

This paper approaches the RRM problem as a resource balancing procedure and as an optimal stochastic

control problem. This strategy relies on an explicit formulation of

- the sensing problem that the radar needs to solve w.r.t. dynamic and measurement models,
- the control actions that the radar sensor has available, reflecting the degrees-of-freedom of the MFR mentioned earlier,
- a cost function that expresses the system performance that the user would like to optimize w.r.t. the sensing aim.

To the authors' best knowledge, an overall solution to the RRM problem based on such a problem solution approach has not been presented so far. A genuinely optimal solution could supposedly lead to a significant performance improvement of adaptive sensors [24], but that still needs to be illustrated. However, an optimality-based approach using a modular framework could significantly reduce the design effort for new MFR systems compared to heuristic solutions, even if the performance does not improve.

B. RRM for Tracking and Classification

For a successful radar application, it is often necessary to distinguish between different types of targets. Therefore, classification is a vital task for every modern radar system and needs to be considered in RRM. A general high-level overview of classification techniques in CR and RRM is shown by Brüggenwirth *et al.* in [14]. Furthermore, Kreucher and Hero presented a very generic framework that is potentially capable of doing joint detection, tracking, and classification [34]. The explanation of the approach stays at a very high level and is only demonstrated through a detection and tracking scenario.

Most RRM approaches for classification are myopic and focus on a simple waveform or sensor mode selection, often for a single object. In [55], Sowelam and Tewfik present such an approach where the Kullback–Leibler information is maximized for the subsequent measurement. Based on this, the algorithm can decide if another measurement is necessary and which waveform must be chosen from a predefined library. Another example has been shown by Bell *et al.* and considers both tracking and classification [2]. The system is assumed to have separate tracking and classification modes, which each have a predefined waveform library to choose from. The proposed algorithm decides the following sensor action to be executed. As for objective functions, both task-driven and information-driven possibilities are discussed. While the task-driven approach requires different objective functions for the two sensing modes, the information-driven approach can compare the two different task types through information gain.

A popular approach is to introduce a measure of risk or threat. The idea is to summarize the interesting task quantities into a single scalar number that is easy to compare. In [39], Martin introduces a risk-based approach where the risk depends on the probability of making a wrong classification and the possibility of track loss multiplied with predefined cost values. The approach finds a solution for both tracking and classification in a myopic fashion. The measurements are always taken the same way, but the algorithm decides which target will be sensed. In [22] and [47], similar approaches are presented. From the perspective of this paper, such a cost function definition is not preferred, as predefined cost values cannot easily represent the risk in all possible situations. Such a formulation leads to a lack of flexibility in the approach. Bolderhij *et al.* present an approach for military radar applications that relies on a large amount of expert knowledge to decide the risk level [12]. Although many different situations are considered, this approach does not automatically balance the resources and cannot flexibly adapt to different situations. A more interesting approach is shown by Katsilieris *et al.* in [30] where joint tracking and classification are performed by running a tracking filter per target class in parallel. The classification is done by comparing the likelihood of a measurement belonging to the different tracks. The next sensing action is then chosen by evaluating the threat's uncertainty based on the target state.

Additionally, some authors have introduced RRM approaches with a hierarchical structure. This is usually done for two main reasons. Firstly, such a structure can decrease the computational complexity and increase the efficiency of an algorithm. Two notable examples of such approaches are the ones by Wintenby and Krishnamurthy in [60] and Castãnón in [15], which both use a hierarchical structure to solve the RRM problem using Lagrangian relaxation (LR). Secondly, a hierarchical structure can also be used to emulate the cognitive behavior of the human brain in order to improve radar performance. An example for such an approach has been proposed by Mitchell *et al.* in [44] and Mitchell in [43].

This paper treats the RRM problem as an optimal control problem. It is not the intention to mimic the behavior of human or animal brains. Furthermore, the functional performance is the focus of the proposed approach rather than a computationally efficient implementation. However, the RRM approach in this paper can potentially be applied in a hierarchical fashion, which might lead to improvements that are beneficial for practical implementation.

C. Markov Decision Processes in RRM for Tracking and Classification

Many RRM approaches for tracking and classification of multiple targets assume a Markov Decision Process (MDP) or Partially Observable MDP (POMDP) framework.

Wintenby and Krishnamurthy have presented an interesting RRM approach for tracking scenarios in [60]

where a Markov chain consisting of performance states is applied for each tracking task. The problem is then solved using a combination of LR and approximate dynamic programming. Furthermore, White and Williams assume a discretized state space and solve a fully observable MDP by applying dynamic programming [59] in combination with LR. Similar to LR, some approaches also apply the Quality of Service resource allocation method (Q-RAM) with POMDPs to solve the RRM tracking problem. Two examples of such an approach can be found in [17], [19].

For simple classification problems, it is relatively easy to implement a solution assuming underlying MDPs or POMDPs, as the number of considered states is usually relatively low. An advantage of using such a framework is the possibility of taking the expected future into account. Chong *et al.* present a straightforward general example of how to use classification in RRM with POMDPs [20]. Castãnón applies a nonmyopic POMDP approach for a classification scenario of almost 10 000 objects where the algorithm chooses from a set of sensor modes [15]. This approach does not take the position and velocity of the objects into account. Another interesting approach has been presented by La Scala *et al.* in [37] and nonmyopically solves the underlying POMDP in a detection and classification scenario. The algorithm selects the best waveforms from a predefined library. The authors promise that an extension to tracking can be achieved without much effort, but do not demonstrate that explicitly. Two other classification approaches that assume an MDP or POMDP framework are shown in [22], [27]. Since the number of states is relatively low in most classification problems, also machine learning techniques have been suggested for solving the underlying (PO)MDPs, e.g., by Langford and Zadrozny in [36] and Blatt and Hero in [10].

D. The Cost Function

An optimization-based RRM approach is preferred over a heuristic approach. However, this requires an explicit definition of a cost function that determines the sensor system's performance. It has been suggested previously that generic measures, such as the Information Gain or the Renyi divergence, applied to the posterior density of the full state, could be applied (see, e.g., [35], [58]). Unfortunately, one single cost function will not meet the expectations of different users in different applications using different sensor systems in different environments and for different targets (see, e.g., [21]).

Developing a useful cost function is critical for the good performance of the RRM algorithm. The development of specific cost functions requires close cooperation with future users and is an intricate development process on its own. Since the focus of this paper is to illustrate how the underlying framework and algorithmic solution can be applied in an example scenario, the development of a user-specific cost function is out of scope. Therefore, it is not claimed that the presented cost function formulation is necessarily leading to the best performance possible.

An example of a more specific operationally relevant cost function can be found in the approach by Narykov *et al.* in [46] where the adversarial risk is introduced as a cost function in a military impact assessment scenario.

E. Proposed Approach

This paper is based on the framework and algorithmic solution presented in [51] and [50], which was previously mainly illustrated in multitarget tracking scenario, i.e., without classification. Here, the framework and approach are applied to a joint tracking and classification problem.

Most approaches that focus on RRM for classification are concrete and apply heuristic rules to compare different task types, such as tracking and classification tasks. In most proposed approaches, the different tasks are assumed to be independent or very weakly dependent. This paper specifically focuses on cases where the tasks are joint. In addition to that, most available approaches are myopic and do not consider MDP or POMDP frameworks. The introduction of risk or threat measures is widespread and seems promising as it enables the objective comparison of different task types. In this paper, it is shown that the generic algorithm introduced in previous publications can be used to address the shortcomings of previously published literature. It is explained how the approach can easily be adjusted to include joint tracking and classification, using a single cost function for both task types. The underlying POMDP is solved nonmyopically, and the resulting policy is achieved by balancing all the considered actions in the action space.

The purpose of this paper is to introduce as few assumptions and simplifications as possible and formulate the RRM problem as a single optimization problem. However, the techniques shown here could potentially be applied in a hierarchical algorithm as well, taking into account higher and lower levels of optimization.

F. Novelty

This paper shows that it is possible to solve the RRM problem for multiple task types by using only a single cost function based on a definition of a mission threat. Such an approach has been suggested previously but has never been fully developed and demonstrated with the help of practical simulation scenarios.

G. Structure of the Paper

The remainder of this paper is structured as follows: Section II defines the general RRM problem, and Section III introduces the proposed solution for a tracking and classification scenario. Furthermore, Section IV

introduces the applied threat and cost function, while Section V gives details about the assumed radar scenario for the simulations. The results of those simulations are discussed in Sections VI–IX. Finally, Section X contains the conclusions.

II. GENERAL RRM PROBLEM DEFINITION

A. Motion Model

Each target can be described by a state based on its position and velocity in x and y directions in a two-dimensional Cartesian coordinate system. For Target n at time t, this state is defined as

$$\boldsymbol{s}_t^n = [x_t^n \quad y_t^n \quad \dot{x}_t^n \quad \dot{y}_t^n]^T, \tag{1}$$

where x_t^n, y_t^n, and $\dot{x}_t^n, \dot{y}_t^n$ are the position and velocity of Target n in x and y, respectively. The future target state at time $t + \Delta t$ can be calculated following a function:

$$\boldsymbol{s}_{t+\Delta t}^n = f_{\Delta t}\left(\boldsymbol{s}_t^n, \boldsymbol{w}_t^n\right), \tag{2}$$

where $s_{t+\Delta t}^n$ is the next state at time $t + \Delta t$ and $\boldsymbol{w}_t^n \in \mathbb{R}^4$ is the maneuverability noise for Target n at time t. The state evolution equation (2) directly defines the evolution probability density function (PDF), which is given as

$$p\left(\boldsymbol{s}_{t+\Delta t}^n | \boldsymbol{s}_t^n\right). \tag{3}$$

B. Measurement Model

It is assumed that the considered sensor is taking noisy measurements of the state $\boldsymbol{s}_t^n$ by executing a sensor action $\boldsymbol{a}_t^n \in \mathbb{R}^m$, where m denotes the amount of adjustable action parameters. The measurement $\boldsymbol{z}_t^n$ of Target n at time t is expressed by the measurement function $\mathfrak{h}$ as

$$\boldsymbol{z}_t^n = \mathfrak{h}\left(\boldsymbol{s}_t^n, \boldsymbol{v}_t^n, \boldsymbol{a}_t^n\right), \tag{4}$$

where $\boldsymbol{v}_t^n \in \mathbb{R}^q$ is the measurement noise for Target n, and q is the amount of measurement parameters. The measurement equation (4) directly defines the measurement PDF, which can be written as

$$p\left(\boldsymbol{z}_t^n | \boldsymbol{s}_t^n, \boldsymbol{a}_t^n\right). \tag{5}$$

C. Tracking Algorithm

As this paper considers joint tracking and classification scenarios, a tracking filter needs to be chosen that aims at computing the posterior density. A simple Kalman filter (KF) would be the exact solution for a linear system. In contrast, nonlinear systems require approximate solutions given by, e.g., an extended KF (EKF) or a particle filter.

D. Budget Optimization Problem

The radar sensor has a limited maximum budget $\Theta_{\max}$ of any kind. For action $\boldsymbol{a}_t^n$ that is executed for each task n, a specific amount of budget (such as time or energy allocations) is required. An overload situation occurs when the current tasks require more of the total budget than is available. In such a situation, the available budget needs to be allocated to all tasks by minimizing the cost (e.g., related to the uncertainty of the current situation).

At time t, the optimization problem for N different tasks can be written as

$$\begin{aligned} \underset{\boldsymbol{a}_t}{\text{minimize}} \quad & \sum_{n=1}^{N} c(\boldsymbol{a}_t^n, \boldsymbol{s}_t^n) \\ \text{subject to} \quad & \sum_{n=1}^{N} \Theta_t^n(\boldsymbol{a}_t^n) \leq \Theta_{\max}, \end{aligned} \tag{6}$$

where $\Theta_t^n \in [0, 1]$ is the budget for task n at time t, $c(\cdot)$ is the applied cost function, and $\Theta_{\max} \in [0, 1]$ is the maximum available budget (0: no budget assigned, 1: all budget assigned). It is critical for the performance of the algorithm to define a relevant cost function. However, this is not the focus of this paper. An example of another operationally relevant cost function has been discussed by Katsilieris *et al.* [31].

III. PROPOSED APPROACH FOR THE RRM PROBLEM

A. Joint Tracking and Classification

This paper assumes that each target is of a specific predefined class that is initially unknown to the radar system. A Bayesian classifier will be applied to make the classification decision. Suppose a class feature could be measured directly and the features were independent of each other. In that case, the classification problem can be solved, e.g., by applying a naive Bayes classifier using these class measurements directly.

If the class features cannot be observed directly, then the behavior of the target often contains information about the underlying target type. In that case, joint tracking and classification can be applied. Similar approaches have been presented, e.g., in [1], [40]. Based on the measurements taken by the radar sensor, a track can be created with the help of a tracking filter (e.g., EKF or particle filter). The track then describes the movements of the observed objects. The problem that needs to be solved contains both discrete (class) and continuous variables (e.g., position and velocity) of the targets are considered. The following equations are based on Bayesian theory (see, for instance, [4], [49], [57]).

Taking into account the class of the target, the state evolution equation in (2) changes to

$$\boldsymbol{s}_{t+\Delta t}^n = f_{\Delta t}\left(\boldsymbol{s}_t^n, c^n, \boldsymbol{w}_t^n\right), \tag{7}$$

where $c^n \in \boldsymbol{C}$ is a scalar and denotes the class of Target n, which is not changing over time. The measurement function is defined similarly to (4):

$$\boldsymbol{z}_t^n = \mathfrak{h}\left(\boldsymbol{s}_t^n, \boldsymbol{v}_t^n, \boldsymbol{a}_t^n, f_c^n\right), \tag{8}$$

where f_c^n is a directly measurable feature of Target n, represented as a scalar value. The PDF function for feature f_c^n can explicitly be written as

$$p\left(z_{t,f}^n | \text{SNR}, c^n\right) = \frac{}{\sqrt{2\pi\sigma_f^2(\text{SNR})}} \exp\left(-\frac{(z_{t,f}^n - f_c^n)^2}{\sigma_f^2(\text{SNR})}\right), \tag{9}$$

where $z_{t,f}^n$ is the measurement of the feature of Target n at time t, and $\sigma_f^2(\text{SNR})$ is the feature measurement variance which depends on the signal-to-noise ratio (SNR). One could think, for instance, of the RCS or the micro-Doppler spectrum. For simplicity, the values used in this paper do not have any physical origin and are merely chosen for demonstration purposes. This feature is assumed to be directly connected to the object's class and not dependent on the state of the target. Therefore, the measurement $\boldsymbol{z}_t^n$ consists of a state (position and Doppler) and a class (feature) component. The PDFs of state, process, or maneuverability noise and measurement noise can then depend on the underlying target class:

$$\begin{aligned} &p\left(\boldsymbol{s}_{t+\Delta t}^n | c^n\right), \\ &p\left(\boldsymbol{w}_t^n | c^n\right), \\ &p\left(\boldsymbol{v}_t^n | c^n\right) \\ &p\left(f_c^n | c^n\right). \end{aligned} \tag{10}$$

The goal of this joint tracking and classification approach is to recursively calculate the posterior joint PDF

$$p\left(\boldsymbol{s}_t^n, c^n | \boldsymbol{Z}_t^n\right) = p\left(\boldsymbol{s}_t^n | c^n, \boldsymbol{Z}_t^n\right) P(c^n | \boldsymbol{Z}_t^n), \tag{11}$$

where $\boldsymbol{Z}_t^n = [\boldsymbol{z}_t^n, \boldsymbol{z}_{t-\Delta t}^n, \boldsymbol{z}_{t-2\Delta t}^n, \ldots, \boldsymbol{z}_0^n]$ are all measurements taken for Target n until time t and $P(c^n|\boldsymbol{Z}_t^n)$ are the prior class probabilities, which are known from the last iteration. Using the Bayesian evolution and update equations, the conditional posterior density can be written as

$$p\left(\boldsymbol{s}_{t+\Delta t}^n | c^n, \boldsymbol{Z}_t^n\right) = \int_S p\left(\boldsymbol{s}_{t+\Delta t}^n | \boldsymbol{s}_t^n, c^n\right) p\left(\boldsymbol{s}_t^n | c^n, \boldsymbol{Z}_t^n\right) d\boldsymbol{s}_t^n, \tag{12}$$

where

$$p\left(\boldsymbol{s}_t^n | c^n, \boldsymbol{Z}_t^n\right) = \frac{p\left(\boldsymbol{z}_t^n | \boldsymbol{s}_t^n, c^n\right) p\left(\boldsymbol{s}_t^n | c^n, \boldsymbol{Z}_{t-\Delta t}^n\right)}{p\left(\boldsymbol{z}_t^n | c^n, \boldsymbol{Z}_{t-\Delta t}^n\right)}. \tag{13}$$

The normalizing constant in the denominator is calculated with

$$p\left(\boldsymbol{z}_t^n | c^n, \boldsymbol{Z}_{t-\Delta t}^n\right) = \int_S p\left(\boldsymbol{z}_t^n | \boldsymbol{s}_t^n, c^n, \boldsymbol{Z}_{t-\Delta t}^n\right) p\left(\boldsymbol{s}_t^n | c^n, \boldsymbol{Z}_t^n\right) d\boldsymbol{s}_t^n. \tag{14}$$

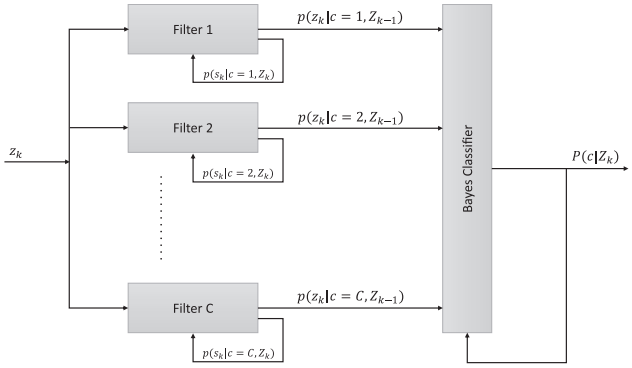


Figure 1. Joint tracking and classification process.

As the measurement consists of a state dependent and a state independent part, which is based only on the class, this expression can also be written as

$$p\left(\boldsymbol{z}_t^n | c^n, \boldsymbol{Z}_{t-\Delta t}^n\right) = p\left(\boldsymbol{z}_t^{n,s,c} | c^n, \boldsymbol{Z}_{t-\Delta t}^{n,s}\right) p\left(\boldsymbol{z}_t^{n,c} | c^n\right), \tag{15}$$

where $\boldsymbol{z}^{n,s,c}$ denotes the state and class dependent measurements and $\boldsymbol{z}^{n,c}$ the class dependent measurement of feature f_c^n for Target n at time t. The posterior class probability is calculated via

$$P\left(c^n | \boldsymbol{Z}_t^n\right) = \frac{p\left(\boldsymbol{z}_t^n | c^n, \boldsymbol{Z}_{t-\Delta t}^n\right) P\left(c^n | \boldsymbol{Z}_{t-\Delta t}^n\right)}{p\left(\boldsymbol{z}_t^n | \boldsymbol{Z}_{t-\Delta t}^n\right)}. \tag{16}$$

The likelihood of the current measurement given all the previous measurements is defined as

$$p\left(\boldsymbol{z}_t^n | \boldsymbol{Z}_{t-\Delta t}^n\right) = \sum_{c=1}^{C} p(\boldsymbol{z}_t^n | c^n, \boldsymbol{Z}_{t-\Delta t}^n) P\left(c^n | \boldsymbol{Z}_{t-\Delta t}^n\right), \tag{17}$$

where C is the number of assumed target classes. The recursive process that is described through (7)–(17) requires C different tracking filters, each conditioned to a specific class. Based on the likelihood of the current measurement being associated with one of the tracks, the class probability is updated. The process is summarized in Fig. 1.

B. Distribution of the Sensor Budgets Using LR

The approach presented here is based on the algorithmic solution presented in [51] and [50]. It applies LR to relax the problem by including the constraint into the cost function. This results in the so-called Lagrangian dual (LD). The original optimization problem is then decoupled into suboptimization problems, one for each task. This leads to the Lagrangian dual problem (LDP), which can be formulated as

$$Z_D = \max_{\lambda_t}\left(\min_{\boldsymbol{a}_t}\left(\sum_{n=1}^{N}\left(c(\boldsymbol{a}_t^n, \boldsymbol{s}_t^n) + \lambda_t \cdot \Theta_t^n\right)\right) - \lambda_t \cdot \Theta_{\max}\right), \tag{18}$$

where $\lambda_t \in \mathbb{R}$ is the Lagrange multiplier for the resource budget constraint. The sum in the LDP allows the algorithm to solve the minimization problem in parallel for each Target n before updating the Lagrangian multiplier

in an iterative process using the subgradient method. This process is explained following, where an internal index l is used for the iterations within the LR process:

1) $l = 0$: Set the initial Lagrange multiplier ($\lambda = \lambda_0$).
2) For every task n, minimize the LD with respect to the actions. Keep the resulting $\boldsymbol{a}_l^n$ and Θ_l^n.
3) Choose the subgradient for the Lagrange multiplier as $\mu_l^\lambda = \sum_{n=1}^{N} \Theta_l^n - \Theta_{\max}$.
4) Is $\mu_l^\lambda \approx 0$ reached with desired precision? If yes, then stop the process. The current λ_l, $\boldsymbol{a}_l^n$ and Θ_l^n are the LR solution for λ_t at time t.
5) Set $\lambda_{l+1} = \max\{0, \lambda_l + \gamma_l \mu_l^\lambda\}$, where γ_l is the LR step size at time l. In this step, the LD is iteratively maximized with respect to λ.
6) Go to step 2 and set $l = l + 1$.

Further information regarding LR can be found in Appendix A, as well as in [50]–[52].

C. Definition of a POMDP

A POMDP is defined as an MDP whose state cannot be observed directly. Instead, the state can be observed through noisy measurements, leading to a probability distribution over the possible states called the belief state. Knowing the structure of the underlying MDP and having noisy measurements available, the POMDP framework allows solving optimization problems nonmyopically, which means calculating the expected cost in future time steps. For the following equations, the time is assumed to be discretized in intervals k with length T, the time between two consecutive measurement operations.

A POMDP is commonly defined by the following parameters (see, e.g., [48] and [20]):

State space $\boldsymbol{S}$: All possible states that can be reached within the process, see (1). At time step k the state is defined as $\boldsymbol{s}_k$. The belief-state defines a probability distribution over all possible states based on the previous measurements and is defined as $\boldsymbol{b}_k$.

Action space $\boldsymbol{A}$: All possible actions within the process. Each executed action leads to a certain cost defined by the cost function. The action at time step k is written as $\boldsymbol{a}_k$.

Observation space $\boldsymbol{Z}$: All possible observations that can be made within the process. An observation at time step k it is defined as $\boldsymbol{z}_k$.

Transition probability $\Psi(\boldsymbol{s}_k, \boldsymbol{s}_{k+1}, \boldsymbol{a}_k)$: The probability function $p(\boldsymbol{s}_{k+1}|\boldsymbol{s}_k, \boldsymbol{a}_k)$ that defines the probability of transitioning from state $\boldsymbol{s}_k$ to state $\boldsymbol{s}_{k+1}$ given action $\boldsymbol{a}_k$. Note: In this paper, the transition probability does not depend on the action.

Probability of observation $\boldsymbol{O}(\boldsymbol{z}_k, \boldsymbol{s}_{k+1}, \boldsymbol{a}_k)$: The probability function $p(\boldsymbol{z}_k|\boldsymbol{s}_{k+1}, \boldsymbol{a}_k)$ that defines the probability to make a certain observation $\boldsymbol{z}_k$ when action $\boldsymbol{a}_k$ is executed with the resulting state being $\boldsymbol{s}_{k+1}$.

Cost function $c(\boldsymbol{s}_k, \boldsymbol{a}_k)$: The immediate cost of executing action $\boldsymbol{a}_k$ in state $\boldsymbol{s}_k$.

Discount factor γ: A possible factor that discounts future costs. Note: in this paper, the discount factor is always set to $\gamma = 1$.

D. Policy Rollout for POMDPs

A variety of different POMDP solution methods exist. A short general discussion of possible approaches can be found in Appendix B, or the overviews by Ross *et al.* in [48] and Chong *et al.* in [20].

In this paper, the policy rollout (PR) technique is applied, which takes Monte Carlo samples of the expected future. This means that it stochastically explores the possible future actions and their related costs. Per possible action $\boldsymbol{a}$ in the action space $\boldsymbol{A}$, a so-called rollout is used to evaluate the expected cost. Expected observations and belief states are generated from a given initial belief state and a given candidate action within such a rollout. The candidate action is executed first, while a so-called base policy (BP) π_{base} is used for every following step in the rollout, until the horizon $\mathscr{H}$ is reached. The cost of all steps within a rollout is summed up. This procedure is repeated M times, and finally, the cost of all rollouts is averaged. The resulting number is then called the expected cost of the evaluated action. The candidate action with the lowest expected cost is chosen to be executed in the next time step. It has been shown that PR leads to a policy that is at least as good as the BP with a very high probability if enough samples are provided [6]. Choosing a good BP and a large enough number of samples is therefore crucial to the algorithm's performance. In this case, the number of samples is equivalent to the number of rollouts M per action. Therefore, one sample is the evaluation of one possible future. It is no trivial task to find a good BP for the considered scenario. As an example, previously experienced situations could be used for it, e.g., as a lookup table. Additionally, the BP could also be improved with new information over time, which could be considered in RL, for instance. Unfortunately, it is not very likely to experience the exact same situation multiple times if a huge state space is assumed, so the usefulness of RL techniques is questionable for typical radar scenarios. Another straightforward choice of the BP could be an equal resource allocation to all the tasks. PR has been covered extensively, e.g., by Bertsekas in [5]–[7].

The PR can be expressed as shown in (19) and (21). The Q-value is defined as

$$Q^{\pi_{\text{base}}}(\boldsymbol{b}_k, \boldsymbol{a}_k) = C_B(\boldsymbol{b}_k, \boldsymbol{a}_k) + E\left[V^{\pi_{\text{base}}}(\boldsymbol{b}_{k+1})|\boldsymbol{b}_k, \boldsymbol{a}_k\right], \tag{19}$$

where $C_B(\boldsymbol{b}_k, \boldsymbol{a}_k) = \sum_{s \in S} \boldsymbol{b}_k(\boldsymbol{s}) c(\boldsymbol{s}, \boldsymbol{a}_k)$ being the expected cost given belief state $\boldsymbol{b}_k$, $E[\cdot]$ is the expectation and $V^{\pi_{\text{base}}}(\cdot)$ is the so-called value function assuming the

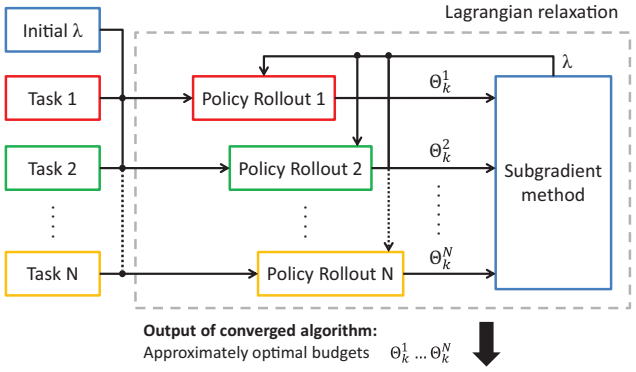


Figure 2. High-level block scheme of the AODB algorithm [50].

chosen BP. The value function can be expressed as

$$V^{\pi_{\text{base}}}(\boldsymbol{b}_{k_0}) = E\left[\sum_{k=k_0}^{k_0+\mathscr{H}} C_B(\boldsymbol{b}_k, \boldsymbol{a}_k)|\boldsymbol{b}_{k_0}\right]. \tag{20}$$

The best policy can then be found by applying

$$\pi_k(\boldsymbol{b}_k) = \arg\min_{a_k \in A}(Q^{\pi_{\text{base}}}(\boldsymbol{b}_k, \boldsymbol{a}_k)). \tag{21}$$

PR does not necessarily lead to the optimal policy. It instead aims at improving the chosen BP π_{base}.

E. Approximately Optimal Dynamic Budget Balancing

This paper uses the Approximately Optimal Dynamic Budget Balancing (AODB) algorithm as introduced in [51] and [50], which applies a combination of PR and LR. The general structure of our proposed algorithm is illustrated in Fig. 2. The outputs of the algorithm are the converged budgets for each task. The PR is applied per task, which in this paper means that for each observed object, the expected cost for each action is calculated, taking into account the current class probabilities for all possible classes.

IV. FORMULATION OF THE COST FUNCTION

The assumed cost function in this paper is based on a definition of threat. This definition depends heavily on the considered scenario and the wishes and expectations of the user. There are practically an unlimited amount of possibilities for constructing such a function. In this section, it is considered that the threat $\phi(c, \boldsymbol{s})$ depends on the class and the state of a target. The cost function will be defined by the variance in the threat knowledge of a target. This means that the cost will be very high for unclassified targets, as all class-dependent threat values are equally likely. Once the knowledge of the target class increases, this variance in threat will decrease also. An explicit example formulation of the threat and the cost function will be introduced later, together with the simulation scenarios. First, the focus is on transforming the PDF from the state domain to the threat domain. As the cost calculation is done for each target separately, the target-related superscript n is dropped to simplify the notations in the following subsections.

A. Unscented Transform

The running target tracks supply a PDF of the target state. Since the transformation of the state PDF to the threat PDF is nonlinear, a sampling approach is chosen. A possible implementation is to sample the threat PDF with a certain number of random samples in the state PDF. For an accurate result, many samples are necessary, which can make this approach very slow. Therefore, in this paper, the samples in the state space of the target are chosen with the help of the unscented transform that is also applied in the unscented KF [28]. For a D-dimensional PDF, $2D + 1$ sigma points are necessary. The procedure for calculating the current threat at a certain moment in time is as follows:

1) Calculate the Cholesky decomposition of the belief state covariance matrix of the target:

$$\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{P}, \tag{22}$$

where $\boldsymbol{P}$ is the belief state covariance matrix of the target.

2) Calculate the so-called sigma points:

$$\begin{aligned} \mathbb{X}^0 &= \hat{\boldsymbol{s}}, \\ \mathbb{X}^i &= \hat{\boldsymbol{s}} + \sqrt{D+\kappa}\ \text{col}_i\boldsymbol{L} \quad i = 1, \ldots, D, \\ \mathbb{X}^{i+D} &= \hat{\boldsymbol{s}} - \sqrt{D+\kappa}\ \text{col}_i\boldsymbol{L} \quad i = 1, \ldots, D, \end{aligned} \tag{23}$$

where $\hat{\boldsymbol{s}}$ is the belief state mean of the target, $\kappa = 3-D$ and $\text{col}_i\boldsymbol{L}$ denotes the i-th column of matrix $\boldsymbol{L}$.

3) Now, each of these samples has to be transformed into the threat domain by using the threat function $\phi(c, \boldsymbol{s})$.

$$\mathbb{Y}_c^i = \phi(\hat{c}, \mathbb{X}^i)) \quad i = 0, \ldots, 2D, \tag{24}$$

where $\hat{c}$ is the believed class of the target.

4) From the samples in the threat domain, the threat PDF is defined by the mean and covariance:

$$\begin{aligned} \hat{\phi}_c &= \sum_{i=0}^{2D} w^i \mathbb{Y}_c^i, \\ \Sigma_{\phi,c} &= \sum_{i=0}^{2D} w^i (\mathbb{Y}_c^i - \hat{\phi}_c)(\mathbb{Y}_c^i - \hat{\phi}_c)^T, \end{aligned} \tag{25}$$

where $\hat{\phi}_c$ is the mean and $\Sigma_{\phi,c}$ the covariance of the threat PDF based on class c and w^i are weights for the samples given as

$$w^i = \begin{cases} \frac{\kappa}{D+\kappa}, & \text{if } i = 0 \\ \frac{}{2(D+\kappa)}, & \text{otherwise} \end{cases}. \tag{26}$$

B. Combination of Threat PDFs

Since the threat PDF depends on the class c, is has to be calculated for each class separately. Based on the resulting PDFs for C different classes, a total PDF can be constructed. The total mean of the threat $\hat{\phi}_{\text{tot}}$ for the target is defined as

$$\begin{aligned}\hat{\phi}_{\text{tot}} &= \int_{\Phi} \phi p(\phi|\boldsymbol{z}) d\phi \\ &= \int_{\Phi} \phi \sum_{c=1}^{C} \mathcal{N}(\phi; \hat{\phi}_c, \Sigma_{\phi,c}) P(c|\boldsymbol{Z}) d\phi \\ &= \sum_{c=1}^{C} P(c|\boldsymbol{Z}) \int_{\Phi} \phi \mathcal{N}(\phi; \hat{\phi}_c, \Sigma_{\phi,c}) d\phi \\ &= \sum_{c=1}^{C} P(c|\boldsymbol{Z}) \hat{\phi}_c,\end{aligned} \tag{27}$$

where $\boldsymbol{z}$ is a recent measurement of the target state, $\mathcal{N}(\phi; \hat{\phi}_c, \Sigma_{\phi,c})$ denotes a normal distribution with mean $\hat{\phi}_c$ and variance $\Sigma_{\phi,c}$, and $P(c, \boldsymbol{Z})$ is the posterior class probability based on all previous measurements $\boldsymbol{Z}$. The variance can be calculated using

$$\begin{aligned}\Sigma_{\phi,\text{tot}} &= \int_{\Phi} (\phi - \hat{\phi}_{\text{tot}})^2 p(\phi|\boldsymbol{z}) d\phi \\ &= \int_{\Phi} (\phi^2 - 2\phi\hat{\phi}_{\text{tot}} + \hat{\phi}_c)^2) p(\phi|\boldsymbol{z}) d\phi \\ &= \int_{\Phi} \phi^2 p(\phi|\boldsymbol{z}) d\phi - \hat{\phi}_{\text{tot}}^2.\end{aligned} \tag{28}$$

Using

$$\begin{aligned}\int_{\Phi} \phi^2 p(\phi|\boldsymbol{z}) d\phi &= \int_{\Phi} \phi^2 \sum_{c=1}^{C} \mathcal{N}(\phi; \hat{\phi}_c, \Sigma_{\phi,c}) P(c|\boldsymbol{Z}) d\phi \\ &= \sum_{c=1}^{C} P(c|\boldsymbol{Z}) \int_{\Phi} \phi^2 \mathcal{N}(\phi; \hat{\phi}_c, \Sigma_{\phi,c}) d\phi \\ &= \sum_{c=1}^{C} P(c|\boldsymbol{Z}) (\Sigma_{\phi,c} + \hat{\phi}_c^2),\end{aligned} \tag{29}$$

it can also be written as

$$\Sigma_{\phi,\text{tot}} = \sum_{c=1}^{C} P(c|\boldsymbol{Z}) (\Sigma_{\phi,c} + \hat{\phi}_c^2) - \hat{\phi}_{\text{tot}}^2. \tag{30}$$

C. Variance of Threat

The previous subsection described transforming the PDF from the state and class domain to the threat domain by considering multiple possible target classes. Given this threat PDF, a different cost function could be constructed. A simple and unambiguous choice is to simply evaluate the total threat variance $\Sigma_{\phi,\text{tot}}$. The underlying assumption is that the radar system cannot influence the target state but only the uncertainty about the knowledge of the target state by adjusting its sensing actions. Following this cost function, the most resources will be assigned to the targets where the biggest decrease in uncertainty (decrease in threat variance) is expected.

The hypothesis is that this will lead to more resources being assigned to objects of an uncertain class. Once all the objects are classified, the uncertainty in the threat will drop significantly and only depend on the uncertainty in the track. This emphasizes the jointness of the proposed tracking and classification approach, as the uncertainty in both the tracking and classification classes is directly taken into account through this cost function. For the remainder of this paper, the cost function will thus be defined as

$$\mathcal{C}(\boldsymbol{a}, \boldsymbol{s}_{k|k-1}, \boldsymbol{P}_{k|k-1}, c) = \Sigma_{\phi,\text{tot}}, \tag{31}$$

where $\boldsymbol{s}_{k|k-1}$ is the predicted state and $\boldsymbol{P}_{k|k-1}$ is the predicted error-covariance for the considered target given by the tracking filter. Therefore, the predicted belief state is used as input for the cost calculation.

V. ASSUMED RADAR SCENARIO

For the following simulation sections, a simplified radar scenario for tracking and classification is assumed. An EKF is applied as a tracking algorithm and similar definitions are used as already shown in [50]. As mentioned in (1), the targets move in a two-dimensional Cartesian coordinate system. The algorithm is jointly optimizing the revisit interval T and the dwell time τ. The former is the time between two consecutive measurements, and the latter is the time the radar sensor spends on a target. For a Target n, a pair of T_n and τ_n defines a budget allocation, also called action $\boldsymbol{a}_n \in \mathbb{R}^2$. The actions influence both the classification and the tracking performance. The outcome of the algorithm is budget allocations per target that theoretically fit into the time frame.

Furthermore, as the situation changes over time, the resource allocation needs to be adjusted to it. In this paper, this is done in regular predefined update intervals.

A. Assumed Radar System

The assumed radar system is able to take measurements in range r and angle θ. Additionally, it can take measurements of a certain target feature f. There exists a measurement noise with variances $\sigma_{r,0}^2$, $\sigma_{\theta,0}^2$, and $\sigma_{f,0}^2$, which refer to a reference measurement. The parameters of that reference measurement are shown in Table I.

B. Target Dynamics

The targets are assumed to move with a constant velocity in x and y direction with an added maneuverability noise, which is class-dependent. Instead of (2), the next

TABLE I
Parameters of the Reference Measurement

Parameter	Value
SNR (SNR_0)	1
RCS (ς_0)	10 m^2
Dwell time (τ_0)	1s
Range (r_0)	50 km
$\sigma^2_{r,0}$	625 m^2
$\sigma^2_{\theta,0}$	4×10^{-4} rad^2
$\sigma^2_{f,0}$	4

state can therefore be written as

$$\boldsymbol{s}^n_{k_n+1} = \boldsymbol{F}_n \boldsymbol{s}^n_{k_n} + \boldsymbol{w}^{n,c}_{k_n}, \tag{32}$$

with $\boldsymbol{F}_n \in \mathbb{R}^{4\times4}$ defined as

$$\boldsymbol{F}_n = \begin{bmatrix} 1 & 0 & T_n & 0 \\ 0 & 1 & 0 & T_n \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{33}$$

and the maneuverability noise $\boldsymbol{w}^{n,c}$ with covariance

$$\boldsymbol{Q}_{n,c} = \begin{bmatrix} T_n^3/3 & 0 & T_n^2/2 & 0 \\ 0 & T_n^3/3 & 0 & T_n^2/2 \\ T_n^2/2 & 0 & T_n & 0 \\ 0 & T_n^2/2 & 0 & T_n \end{bmatrix} \sigma^2_{w,c}, \tag{34}$$

where $\sigma^2_{w,c}$ is the maneuverability noise variance for class c.

C. SNR Model

As previously mentioned, measurements are taken in range r and angle θ. Due to the nonlinearity between measurements and target states, at $\boldsymbol{s}^n_{k_n}$ a measurement transformation function $\boldsymbol{h}(\boldsymbol{s}^n_{k_n}) \in \mathbb{R}^3$ is defined. The measurement equation in (4) therefore becomes

$$\boldsymbol{z}^n_{k_n} = \boldsymbol{h}(\boldsymbol{s}^n_{k_n}, c^n) + \boldsymbol{v}^n_{k_n}, \tag{35}$$

with

$$\boldsymbol{h}(\boldsymbol{s}^n_{k_n}) = \left[\sqrt{(x^n_{k_n})^2 + (y^n_{k_n})^2}\,,\ \text{atan2}\left(y^n_{k_n}, x^n_{k_n}\right)\,,\ f(c^n)\right]^T, \tag{36}$$

and $\boldsymbol{v}^n_{k_n} \in \mathbb{R}^3$ being the measurement noise for Target n. The feature of class c^n is denoted as $f(c^n)$ and atan2$(\cdot)$ denotes the two-argument arctangent as commonly used in programming languages.

The range and azimuth components of $\boldsymbol{v}^n_{k_n}$ are considered to be independent:

$$\boldsymbol{v}^n_{k_n} = [v^{r,n}_{k_n} \quad v^{\theta,n}_{k_n} \quad v^{f,n}_{k_n}]^T, \tag{37}$$

with variances $\sigma^2_{r,n}, \sigma^2_{\theta,n}$, and $\sigma^2_{f,n}$.

Since the relationship between the measurements and the states is nonlinear, an EKF is applied in the following simulations. The observation matrix $\boldsymbol{H}^n_{k_n}$ is defined as the Jacobian of the measurement transformation function $\boldsymbol{h}$:

$$\boldsymbol{H}^n_{k_n} = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{s} \partial f}\right|_{\boldsymbol{s}^n_{k_n}, f^n}. \tag{38}$$

For the assumed radar systems, it has dimensions $\boldsymbol{H}^n_{k_n} \in \mathbb{R}^{3\times5}$.

In line with the radar scenario described [50], the SNR is calculated by using (39), which is based a paper by Koch [33]:

$$\text{SNR}_{k_n}(\varsigma_n, \tau_n, r^n_{k_n}) = \text{SNR}_0 \cdot \left(\frac{\varsigma_n}{\varsigma_0}\right) \cdot \left(\frac{\tau_n}{\tau_0}\right) \cdot \left(\frac{r^n_{k_n}}{r_0}\right)^{-4} \cdot e^{-2\Delta\alpha}. \tag{39}$$

where $\Delta\alpha$ is the relative beam positioning error, ς_n is the constant radar cross section (RCS) of the Target n, $r^n_{k_n}$ is the distance of Target n at time step k_n and ς_0, τ_0 and r_0 are the corresponding values for a reference target. The dwell time is used equivalently to the transmitted energy mentioned by Koch. The relative beam positioning error is calculated as

$$\Delta\alpha = \frac{\left(\theta_{k_n} - \hat{\theta}_{k_n}\right)^2}{\Gamma^2}, \tag{40}$$

where θ_{k_n} is the real target angle and $\hat{\theta}_{k_n}$ is the predicted target angle in azimuth at time k_n and Γ is the one-sided beam-width in azimuth.

The variance of the range and azimuth measurement noise for Target n can then be defined as (see, e.g., [41])

$$\sigma^2_{\bullet,n} = \frac{\sigma^2_{\bullet,0}}{\text{SNR}_{k_n}(\varsigma_n, \tau_n, r^n_{k_n})}, \tag{41}$$

where $\bullet \in (r, \theta, f)$ and $\sigma^2_{\bullet,0}$ is the measurement noise variance for a reference target 0 as defined in Table I.

Assuming independent measurements, the measurement covariance can be written as

$$\boldsymbol{R}^n_{k_n} = \begin{bmatrix} \sigma^2_{r,n} & 0 & 0 \\ 0 & \sigma^2_{\theta,n} & 0 \\ 0 & 0 & \sigma^2_{f,n} \end{bmatrix}. \tag{42}$$

D. Target Classes

Each target is assumed to belong to a specific class. The class is defined before the simulation scenario starts and cannot be changed. Therefore, it stays the same during the entire scenario. The measurement variance regarding the class feature f^n of object n is calculated as shown in (41). The corresponding variance value for the reference measurement f_0 is shown together with the other simulation parameters in the specific subsection. For the simulations discussed below, different target classes are considered that influence the maneuverability of the targets. These maneuverabilities are implemented in the trajectory simulations of the targets and are also considered in the resource optimization algorithm. As discussed earlier, one tracking filter per target class is applied, each tuned to one of the classes.

D. Optimization Problem

There are N tracked targets in the environment. Based on the general definition in (6), the RRM problem can thus be expressed as

$$\begin{aligned} \min_{T,\tau} \quad & \sum_{n=1}^{N} E\left[\mathcal{C}\left(\boldsymbol{s}^n_{k_n|k_n-1}(T_n,\tau_n), \boldsymbol{P}^n_{k_n|k_n-1}(T_n,\tau_n), c^n\right)\right] \\ \text{s. t.} \quad & \sum_{n=1}^{N} \frac{\tau_n}{T_n} \leq \Theta_{\max}, \end{aligned} \tag{43}$$

where $E[\cdot]$ denotes the expected value. The cost that is optimized is therefore based on the current prediction of the tracking filter, which is based on the measurement actions T_n and τ_n for Target n. Both the revisit times T_n, as well as the dwell times τ_n are optimized. The state measurements are influenced by both T and τ, while the state independent feature measurement is only influenced by the dwell time.

For all shown simulations, the implemented BP is simply to apply the evaluated action in every step of the PR. Therefore, $\pi_{\text{base}} = \boldsymbol{a}$.

F. Threat Definition

Since a two-dimensional scenario is assumed, the dimension parameter in the unscented transform is $D = 2$. As mentioned before, the choice of the "correct" threat definition depends on the scenario and the user's wishes. As an example, in the following, the threat ϕ is defined as

$$\phi(c, \boldsymbol{s}) = \frac{\rho_c \cdot \left(0.1 + \exp\left(-\frac{r-r'}{\eta}\right)\right)}{1 + \exp\left(-\frac{r-r'}{\eta}\right)}, \tag{44}$$

where ρ_c is a scalar factor unique for each class, $r = \sqrt{x^2 + y^2}$ is the range of the target from the sensor, $r' = 18$ km is a reference range and $\eta = 5000$ is a parameter to fine-tune the threat function slope. A possible example of such a threat is shown in Fig. 3. This formulation assumes that targets at a long distance pose a very low threat, while the threat increases the closer the target advances toward the sensor location. At a certain distance, the maximum threat value is reached. In addition to that, some classes generally pose a higher threat than others. One could think of an automotive scenario where a vehicle is moving toward the sensor location. When it is far away, the threat would be low as it would probably turn away at some point. However, once it comes closer, the threat increases until it is not very likely to turn away anymore, which means that the maximum threat level is reached, and a collision is almost inevitable. For instance, regarding the different classes, one could think of a truck having a higher base threat level than a cyclist.

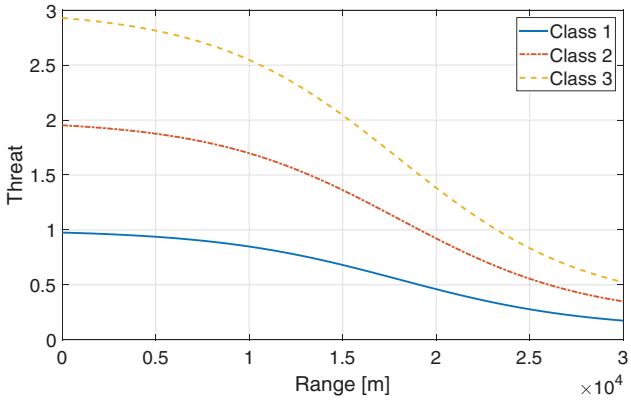


Figure 3. Example threat function for three different targets. The reference range r' is 18 km and the tuning parameter η is 5000. The class parameters ρ are set to values 1, 2, and 3, respectively.

TABLE II
Simulation Parameters for Simulation Scenario A

Parameter	Value
Precision of solution (δ)	0.01
Action space discretization steps ($\Delta T, \Delta\tau$)	Adaptive
Action space limits revisit interval ($T_{\min}, T_{\max}$)	$T \in [0.1\text{s} \ldots 5\text{ s}]$
Action space limits dwell time ($\tau_{\min}, \tau_{\max}$)	$\tau \in [0.1\text{ s} \ldots \infty]$
Number of rollouts (M)	10
Rollout horizon ($\mathscr{H}$)	10
Maximum available budget ($\Theta_{\max}$)	1
Budget update interval (t_B)	5 s
Beam positioning error ($\Delta\alpha$)	0
Probability of detection (P_D)	1
Threat reference range (r')	18 km
Threat slope parameter (η)	5000

VI. SIMULATION SCENARIO A

In this section, the dynamic tracking example as presented in [50] is used to show the impact of the chosen cost function based on the threat. Essentially, the AODB algorithm from [50] is applied with the cost function as defined in (31). The radar sensor is placed at the origin of the coordinate system. Initially, there are four targets in the scene. After 25 s, a fifth target is detected, and a new track is started. All targets move with constant velocities. Here, it is assumed that the class of the targets is not of interest, so no classification is applied during the simulation scenario. The simulation parameters are summarized in Table II, while the target parameters are shown in Table III. The trajectories of the targets during the simulation scenario are shown in Fig. 4, and the resulting budget allocation of the simulations is shown in Fig. 5.

Since classification is not considered in this example, the uncertainty in threat comes directly from the tracking accuracy. This is reflected in Fig. 5 by the fact that the budgets overall show very similar behavior to the dashed lines. Those lines indicate the results from [50], where the error-covariance of the tracking filter was used directly to optimize the resource allocation. For example,

TABLE III
Initial Target Parameters for Simulation Scenario A

Parameter \ Target n	1	2	3	4	5
x_0^n [km]	12	12	7	19	7.9
y_0^n [km]	10	15	11	2	8.3
$\dot{x}_0^n$ [m s^{-1}]	9	−30	45	−35	−20
$\dot{y}_0^n$ [m s^{-1}]	−15	15	30	0	−25
ς^n [m^2]	25	25	64	64	64

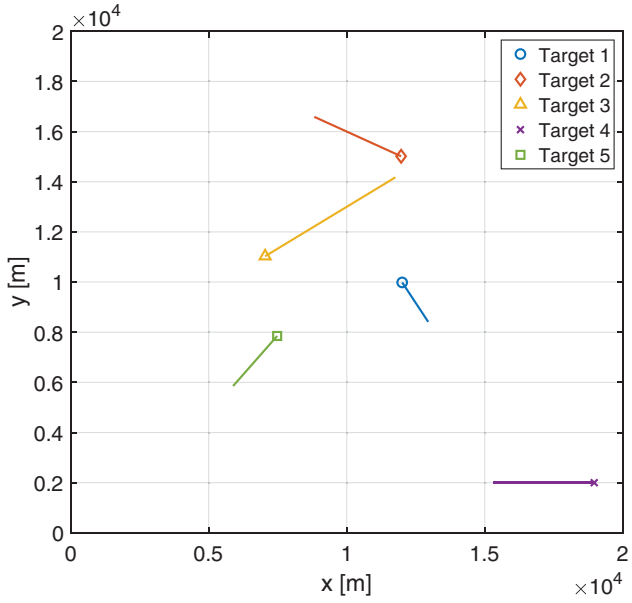


Figure 4. Trajectories of the targets for Simulation Scenario A. The symbols mark the starting positions.

Target 4 receives the largest budget allocation during the first 70% of the scenario, while Target 5 always receives the smallest, which is in line with the previous results. It should be noted that the algorithm decides the resource allocations on the expected threat variance reduction rather than the actual threat variance values. Nevertheless, the target with the highest threat variance will offer

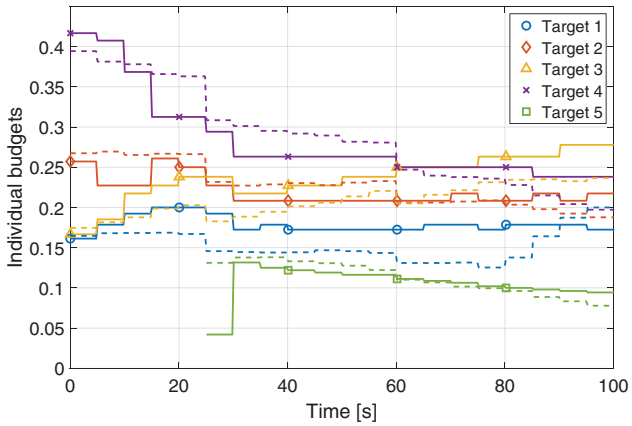


Figure 5. Resulting budget distribution for Simulation Scenario A. The dashed lines denote the results for tracking without classification as shown in [50].

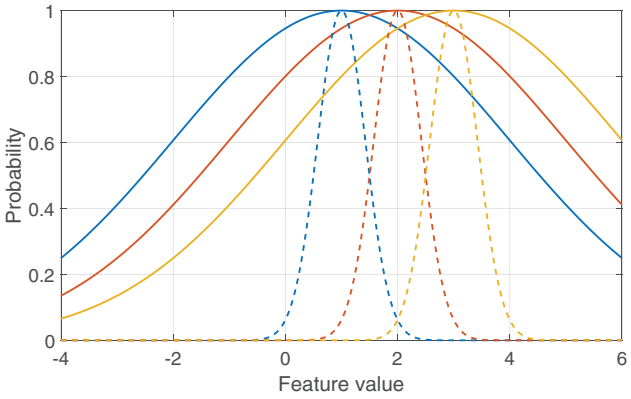


Figure 6. The PDFs of the class features given different SNR values. The solid lines correspond to SNR = 1 and the dashed lines to SNR = 50.

the biggest opportunity to reduce this threat variance in most cases.

VII. SIMULATION SCENARIO B

In this section, a dynamic joint tracking and classification scenario are presented. The radar sensor is placed at the origin of the coordinate system. There are two possible classes and two observed targets. The first target is of Class 1, and the second one of Class 2. The radar sensor is aware of all possible target classes, but it does not know which target is of which class. Therefore, the initial class probabilities are equal for both classes for each target. The available budget is set to 1, implying that the radar system fully focuses on these two tracking tasks. Aspects like false alarms are not taken into account. However, even if ambiguities in the measurement-to-track assignment were considered, it would still be necessary to compute the joint posterior of the state variables that are used for the optimization. Therefore, the approach would not be significantly different from what is presented here. For future use in a real radar system, those aspects certainly need to be taken into account.

The targets move with a class-typical maneuverability noise, which can be seen from trajectories in Fig. 7(a). The simulation parameters are identical with the ones for simulation A as mentioned in Table II. The target and class parameters are shown in Tables IV and V, respectively. As mentioned before, the values used in this section do not have any physical origin and are merely chosen for demonstration purposes. Figure 6 shows the PDFs of the features given two different SNR values. The simulation results are presented in Fig. 7(b)–(e).

In the beginning, Target 2 gets a larger amount of dwell time assigned than Target 1, which leads to a quick classification. Target 2 is closer to the radar sensor than Target 1, which means that not knowing the class leads to a higher threat variance. Additionally, the feature measurements for Target 2 are more accurate than for Target 1 due to the smaller distance and, therefore, higher SNR. Subsequently, after 5 s, the dwell time and

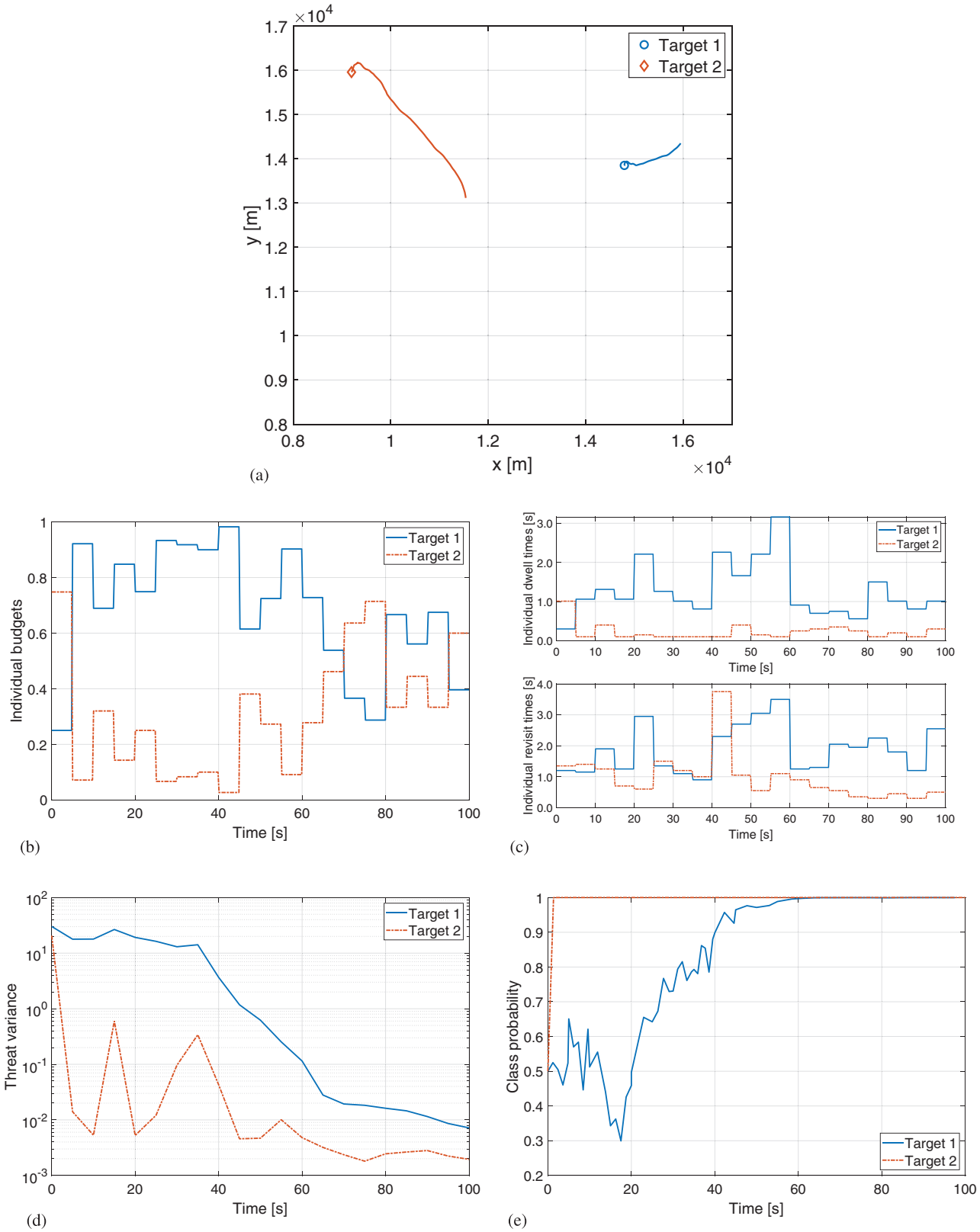


Figure 7. Simulation results for Simulation Scenario B. (a) Trajectories of the targets for Simulation Scenario B. The symbols mark the starting positions. (b) Resulting budget distribution. (c) Resulting dwell time and revisit time distribution. (d) Resulting optimized cost (threat variance). (e) Resulting class probabilities. A value of 1 means that a target was correctly classified.

TABLE IV
Initial Target Parameters for Simulation Scenario B

Parameter \ Target n	1	2
x_0^n [km]	14.8	9.2
y_0^n [km]	13.9	15.9
$\dot{x}_0^n$ [m s^{-1}]	2	2
$\dot{y}_0^n$ [m s^{-1}]	−1	1
ς_n [m^2]	5	5
c^n	1	2

TABLE VI
Initial Target Parameters for Simulation Scenario C

Parameter \ Target n	1	2	3
x_0^n [km]	10.1	12.3	12.1
y_0^n [km]	17.1	17.5	15.3
$\dot{x}_0^n$ [m s^{-1}]	1	−2	1
$\dot{y}_0^n$ [m s^{-1}]	2	2	−2
ς_n [m^2]	5	5	5
c^n	1	2	3

with it the relative budget for Target 2 drops, while Target 1 gets significantly more dwell time and budget assigned. During the bigger part of the scenario, the sensor focuses on classifying Target 1, which is more difficult due to its larger distance from the sensor. While Target 1 gets slowly classified, its budget starts to decrease after about 45 s. The budget for Target 2 increases at the same time. After both targets are successfully classified at about 70 s, the assigned budgets for both targets stay around 0.5, although Target 1 still gets a higher dwell time. The targets are both classified and theoretically deserve a similar amount of attention. However, as Target 1 was classified later, there is still slightly more uncertainty left about its class, leading to a higher dwell time allocation. This behavior emphasizes that the resource allocation is based on the expected threat variance reduction. Figure 7(d) shows how the early classification of Target 2 leads to a significant direct decrease in threat variance, while the classification of Target 1 takes longer, and the cost therefore also drops slower.

VIII. SIMULATION SCENARIO C

This simulation scenario is similar to Scenario B. The radar sensor is again placed at the origin of the coordinate system, and the available maximum budget is set to $\Theta_{\max} = 0.5$. The reason for a lower maximum budget could be, e.g., that an operator of the radar system manually assigned some of the total budget to other tasks. Additionally, the length of the simulation is 500 s, which is longer than in Scenario B. All other general simulation parameters are the same as shown in Table II. The class parameters are summarized in Table VII, and the initial target parameters are shown in Table VI. This time, there are three targets in the environment, and the targets can be of three possible classes. Figure 8(a) shows the trajectories of the simulated targets. The simulation results are shown in Fig. 8(b)–(e).

At the beginning of the scenario, it can be seen that Target 3 gets the largest relative budget assigned. Subsequently, it gets classified very quickly. It can be seen that the algorithm makes a wrong decision about the class of Targets 1 and 2. Figure 7(d) shows that making the first classification decisions leads to a large reduction of the calculated threat variance for all targets within the first 100 s. It can be seen that while the algorithm slowly classifies Target 2 between about 100 s and 300 s, the threat variance increases and then drops again. The reason is that the class probabilities are shifting during that phase, and there has no clear decision been made yet. The same happens to Target 1, as its class probability values are also changing at that time. Between about 320 s and 420 s Target 1 is classified correctly, which also leads to an increased threat variance. After 400 s, all targets are correctly classified, and the threat variances decrease rapidly.

In Fig. 8(b), it can be seen that the budget allocations roughly follow the threat variances. The target with the highest threat variance usually receives the largest budget. Similarly, the dwell times are assigned approximately proportional to the threat variance.

IX. SIMULATION SCENARIO D

This simulation is similar to Simulation C, except for all targets being of the same class. The available budget is set to $\Theta_{\max} = 0.7$. The length of the simulation is 100 s. All other simulation parameters are identical with Simulation A as shown in Table II. The target and class parameters for this simulation scenario are shown in Tables VIII and IX. The simulation results are shown in Fig. 9(a)–(d).

TABLE V
Class Parameters for Simulation Scenario B

Parameter \ Class	1	2
Class feature f_c	1	2
Threat parameter ρ_c	1	9
Maneuverability $\sigma_{w,c}$ [m s^{-2}]	2	5

Table VII
Class Parameters for Simulation Scenario C

Parameter \ Class	1	2	3
Class feature f_c	1	2	3
Threat parameter ρ_c	1	9	19
Maneuverability $\sigma_{w,c}$ [m s^{-2}]	2	5	9

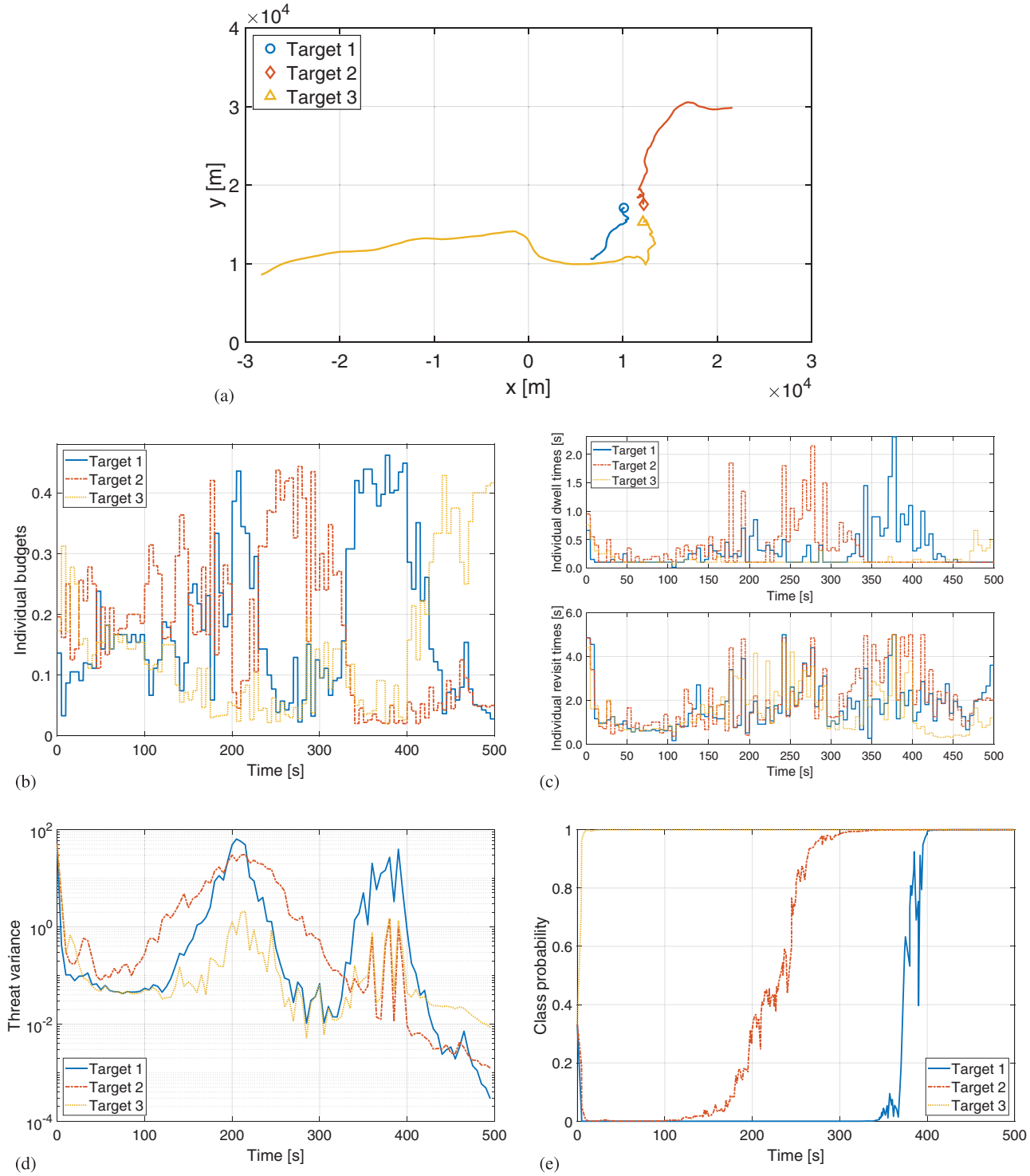


Figure 8. Simulation results for Simulation Scenario C. (a) Trajectories of the targets for Simulation Scenario C. The symbols mark the starting positions. (b) Resulting budget distribution. (c) Resulting dwell time and revisit time distribution. (d) Resulting optimized cost (threat variance). (e) Resulting class probabilities. A value of 1 means that a target was correctly classified.

Figure 9(c) shows that the closer the targets, the earlier they get classified. This has to do with the higher SNR at shorter ranges and the fact that the proposed algorithm assigns larger budgets to "dangerous" targets to classify them quickly. This can be seen in Fig. 9(b), where Target 2 receives the largest budget between 5 and 10 s. After the classification of the targets is completed, the budget distribution in Fig. 9(b) shows a clear influence from the distance of the targets. The reason is that there is no significant uncertainty in the class

TABLE VIII
Initial Target Parameters for Simulation Scenario D

Parameter \ Target n	1	2	3
x_0^n [km]	4	8	12
y_0^n [km]	4	8	12
$\dot{x}_0^n$ [m s^{-1}]	−1	−1	−2
$\dot{y}_0^n$ [m s^{-1}]	1	−1	−2
ς_n [m^2]	5	5	5
c^n	3	3	3

TABLE IX
Class Parameters for Simulation Scenario D

Parameter \ Class	1	2	3
Class feature f_c	1	2	3
Threat parameter ρ_c	1	9	19
Maneuverability $\sigma_{w,c}$ [m s^{-2}]	2	6	10

anymore, so the position uncertainty becomes more important.

Additionally, Fig. 9(d) shows that the proposed approach delivers the lowest expected cost compared to three other resource allocation methods. It can also be seen that once the target classes are determined, the costs of the different approaches are relatively small. The reason, therefore, is that the cost function heavily depends on the assumed class probabilities, especially when the target class is not determined yet. The differences between the approaches depend on the cost function and feature definition.

X. CONCLUSION

This paper introduced a novel RRM approach for joint tracking and classification using a previously introduced framework. In contrast to most available approaches, two different task types are combined into one. It is shown that it is possible to solve the RRM problem for multiple task types by using only a single cost function based on a definition of mission threat. Such approaches have been suggested previously but have never been fully developed and demonstrated with the help of fully worked out practical simulation scenarios.

Firstly, the joint tracking and classification framework have been introduced, which builds on the previous RRM framework as shown in [50].

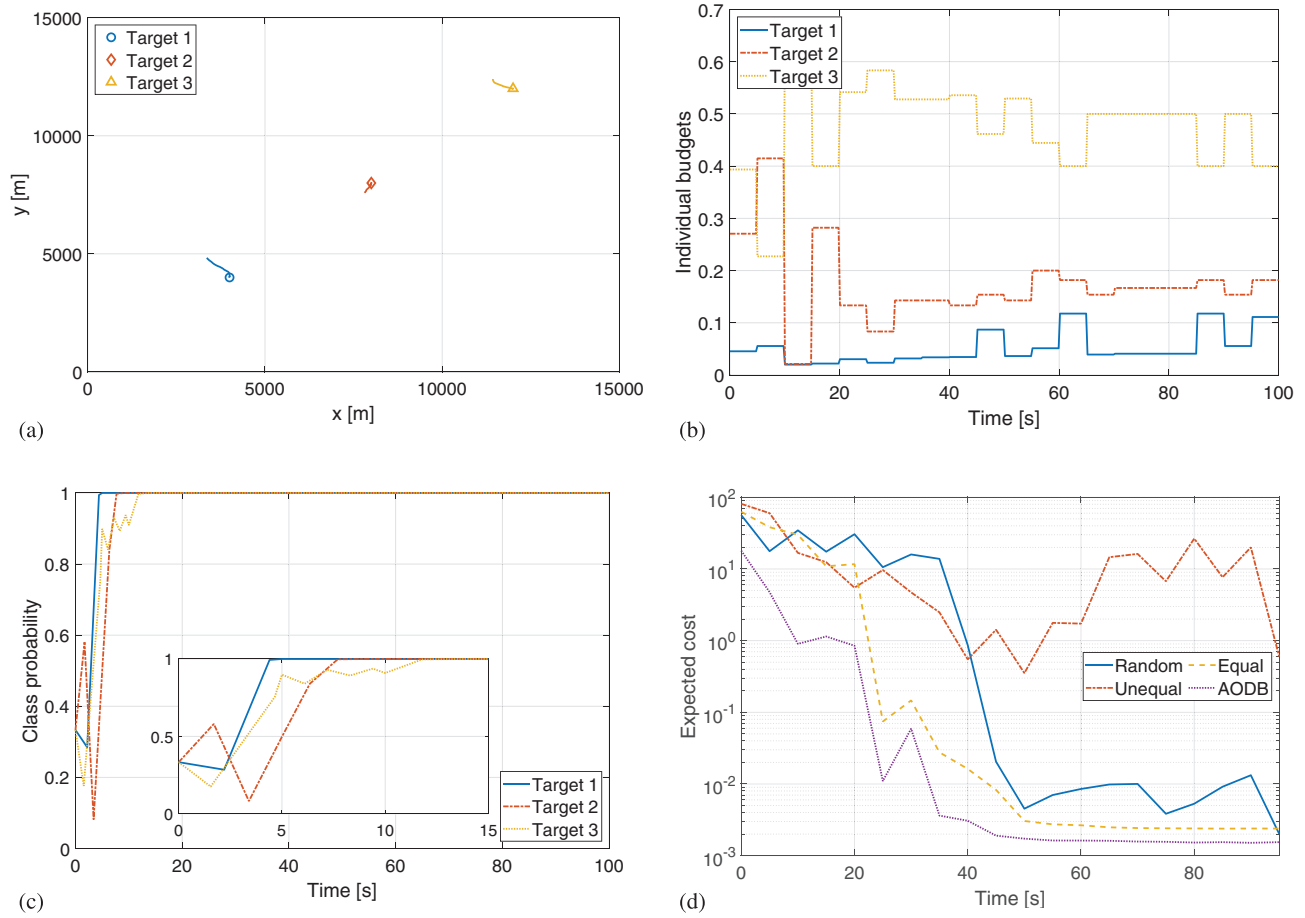


Figure 9. Simulation results for Simulation Scenario D. (a) Trajectories of the targets. The symbols mark the starting positions. (b) Resulting budget distribution. (c) Resulting class probabilities. A value of 1 means that a target was correctly classified. The box on the bottom middle shows the class probabilities zoomed in on the first 15 s. (d) Comparison of the expected optimized cost for different resource allocation methods. The values are averaged over 10 different runs.

Secondly, it has been explained how to move from the state to the threat domain and combine the costs of different target classes. The idea of threat is to transform the state of each task into an easily comparable scalar number.

Finally, an explicit definition of a possible mission threat has been introduced. The presented threat definition is based on the position and a class-dependent parameter. It has been shown how the threat looks like for multiple classes in a two-dimensional environment.

Through an analysis of the dynamic tracking Scenarios A–D, it has been shown that the algorithm works in different situations. It calculates the resource allocations based on the class probabilities and the tracking state accuracy. The algorithm tries to classify targets of unknown classes faster, especially when they are close to the radar sensor. On the other hand, the classification is done over time while tracking if the targets are further away and have a smaller threat variance. Once the targets are classified, the resource allocations depend primarily on the track uncertainty. This means that the target tracks get the resources assigned based on the expected decrease in uncertainty. Compared to other approaches, the presented approach leads to a quicker classification of dangerous objects. Within the presented simulations, a single measurable feature was chosen for demonstration purposes. In a practical implementation of the approach, multiple features (e.g., RCS and velocity) could be taken into account to further accelerate the classification procedure. Nevertheless, the presented simulations confirm the applicability of the proposed algorithm.

The proposed algorithm fills the timeline on average, leading to some overlap of tasks. Specifically, this can happen when predefined start or end times are required, or the tasks cannot be split up into multiple parts. A possible defensive solution to this problem is to assume a lower available budget for the tasks to keep a part of the timeline free. However, this would lead to a less optimal result. For practical implementation, the impact of these overlaps would need to be investigated, and an explicit scheduler would need to be implemented at a lower level, but this is out of the scope of this paper.

Future research has to be conducted w.r.t. the definition of the threat and cost function. Additionally, it has to be investigated further how these different threat and cost formulations influence the budget allocations. It will be especially interesting to look at how the classification process exactly depends on the different input parameters, such as the measurement variance. Finally, a practical implementation of the algorithm requires the investigation of an explicit scheduler and its impact on the results.

APPENDIX A
LAGRANGIAN RELAXATION PRINCIPLE

LR simplifies complicated constrained optimization problems by removing constraints and adding them as penalty terms into the original problem multiplied by so-called Lagrange multipliers. Consequently, a new optimization problem is created with fewer constraints than the original problem. LR maximizes the minimum of the cost function by adjusting the Lagrange multipliers. This problem is called the LDP, which is usually a lower estimate of the original problem if the initial Lagrange multipliers are chosen properly (see, for example, [8]).

LR and Lagrange multipliers have been extensively covered in literature (for example, in [5], [8], [9], [13], or [38]). As an example of how LR is applied, we consider the general optimization problem with N input variables that is shown below:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{A} \\ & \boldsymbol{h}(\boldsymbol{x}) \geq \boldsymbol{B}, \end{aligned} \tag{45}$$

where

$$\begin{aligned} \boldsymbol{x} &= [x_1, \ldots, x_N]^T \in \mathbb{R}^N, \\ \boldsymbol{g}(\boldsymbol{x}) &= [g_1(\boldsymbol{x}), \ldots, g_m(\boldsymbol{x})]^T \in \mathbb{R}^m, \\ \boldsymbol{h}(\boldsymbol{x}) &= [h_1(\boldsymbol{x}), \ldots, h_p(\boldsymbol{x})]^T \in \mathbb{R}^p, \\ \boldsymbol{A} &= [A_1, \ldots, A_m]^T \in \mathbb{R}^m, \\ \boldsymbol{B} &= [B_1, \ldots, B_p]^T \in \mathbb{R}^p. \end{aligned}$$

Including the constraints into the optimization problem is done by adding a penalty term for each removed constraint, multiplied by Lagrange multipliers, which are defined as $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]^T \in \mathbb{R}^m$ and $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_N]^T \in \mathbb{R}^p$. The Lagrangian is defined as

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) &+ \sum_{i=1}^{m} \lambda_i (g_i(\boldsymbol{x}) - A_i) \\ &+ \sum_{j=1}^{p} \mu_j (B_j - h_j(\boldsymbol{x})). \end{aligned} \tag{46}$$

The relaxed problem is then called Lagrangian dual function and can be expressed as

$$d(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \underset{x}{\text{minimize}}\, L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}). \tag{47}$$

The meaning of this expression is to find the maximum of the Lagrangian dual function with respect to the Lagrange multipliers, as

$$Z_D = \underset{\lambda, \mu}{\text{maximize}}\, d(\boldsymbol{\lambda}, \boldsymbol{\mu}). \tag{48}$$

Therefore, the objective function is minimized over $\boldsymbol{x}$ while also being maximized over the Lagrange multipliers. The goal is to come as close to the original problem as possible. Iterative approaches are typically used to find the optimal Lagrange multipliers and, consequently, the tightest lower bound to the original problem. One of those techniques, the subgradient method, will be explained in the following subsection.

A. Subgradient Method

The subgradient method is an iterative process, which starts with an initial value for the Lagrange multipliers (e.g., 1). At each iteration k, the minimum of the relaxed problem is calculated using that value (Lagrangian dual function, see (46)). Then, subgradients are chosen for each of the constraint as $\boldsymbol{s}_{\lambda}^{k} = [s_{\lambda,1}^{k}, \ldots, s_{\lambda,m}^{k}]^{T} \in \mathbb{R}^{m}$ and $\boldsymbol{s}_{\mu}^{k} = [s_{\mu,1}^{k}, \ldots, s_{\mu,p}^{k}]^{T} \in \mathbb{R}^{p}$. Assuming the constraints mentioned in (45), the subgradients are defined as

$$\begin{aligned} \boldsymbol{s}_{\lambda}^{k} &= (\boldsymbol{g}(\boldsymbol{x}^{k}) - \boldsymbol{A}) \\ \boldsymbol{s}_{\mu}^{k} &= (\boldsymbol{B} - \boldsymbol{h}(\boldsymbol{x}^{k})). \end{aligned} \tag{49}$$

Next, the Lagrange multipliers are updated with a step size γ^{k}. If an inequality constraint is given, then the penalty term may not become negative. The updated Lagrange multipliers are therefore calculated as

$$\begin{aligned} \boldsymbol{\lambda}^{k+1} &= \max\{0, \boldsymbol{\lambda}^{k} + \gamma^{k}\boldsymbol{s}_{\lambda}^{k}\} \\ \boldsymbol{\mu}^{k+1} &= \max\{0, \boldsymbol{\mu}^{k} + \gamma^{k}\boldsymbol{s}_{\mu}^{k}\}. \end{aligned} \tag{50}$$

There are many possible step size approaches, such as constant or decreasing step sizes like γ_0/k or $/\gamma^{k}$, for example. The procedure started again with the new Lagrange multipliers. A new Lagrangian dual function is found, and afterward, new subgradients are again calculated. The exact result is found when the gradients $\boldsymbol{s}_{\lambda}^{k}$ and $\boldsymbol{s}_{\mu}^{k}$ reach $\mathbf{0}$. Since this value can never be reached exactly using this method, the process is repeated until the gradient reaches 0 with the desired precision.

To summarize, a short overview of the subgradient algorithm for the above mentioned optimization problem is given here:

1) $k = 0$: Set the Lagrangian multipliers to initial value $(\boldsymbol{\lambda}^{\mathbf{0}} = \boldsymbol{\lambda}_{\mathbf{0}}, \boldsymbol{\mu}^{\mathbf{0}} = \boldsymbol{\mu}_{\mathbf{0}})$.
2) Calculate solution for $d(\boldsymbol{\lambda}, \boldsymbol{\mu})$ and save $\boldsymbol{x}^{k}$.
3) Choose subgradients for Lagrangian multipliers $\boldsymbol{s}_{\lambda}^{k}$ and $\boldsymbol{s}_{\mu}^{k}$ (see (49)).
4) Check if $\boldsymbol{s}_{\lambda}^{k} \approx \mathbf{0}$ and $\boldsymbol{s}_{\mu}^{k} \approx \mathbf{0}$ with desired precision. If it is, then stop the process.
5) Adjust Lagrangian multipliers as shown in (50).
6) Go to step 2 and set $k = k + 1$.

APPENDIX B
SOLUTION METHODS FOR POMDPS

POMDPs can be solved for finite or infinite horizons. In order to reduce complexity, limited horizons $\mathscr{H}$ are often considered. The value of $\mathscr{H}$ represents the number of measurement time steps into the future that are considered in the optimization. Once a new budget allocation is calculated, the horizon $\mathscr{H}$ will be moved to the current moment in time. This approach is therefore also called a receding horizon.

In [16], Charlish and Hoffmann have presented an excellent summary of the general solution of a POMDP. The following equations are based on their explanations. The goal is to find the actions that minimize the total cost (value $V_{\mathscr{H}}$ over horizon $\mathscr{H}$). Starting at time step k_0, this can be expressed as

$$V_{\mathscr{H}} = E\left[\sum_{k=k_0}^{k_0+\mathscr{H}} c(\boldsymbol{s}_k, \boldsymbol{a}_k)\right]. \tag{51}$$

Using $C_B(\boldsymbol{b}_k, \boldsymbol{a}_k) = \sum_{s\in S} \boldsymbol{b}_k(\boldsymbol{s})c(\boldsymbol{s}, \boldsymbol{a}_k)$ being the expected cost given belief state $\boldsymbol{b}_k$, $V_{\mathscr{H}}$ can be written as a so-called value function of the belief state $\boldsymbol{b}_{k_0}$ at time step k_0:

$$V_{\mathscr{H}}(\boldsymbol{b}_{k_0}) = E\left[\sum_{k=k_0}^{k_0+\mathscr{H}} C_B(\boldsymbol{b}_k, \boldsymbol{a}_k)|\boldsymbol{b}_{k_0}\right]. \tag{52}$$

For belief state $\boldsymbol{b}_0$ and taking action $\boldsymbol{a}_0$, the optimal value function is defined according to Bellman's equation [3] as

$$V_{\mathscr{H}}^{*}(\boldsymbol{b}_0) = \min_{a_0\in A}\left(C_B(\boldsymbol{b}_0, \boldsymbol{a}_0) + \gamma \cdot E\left[V_{\mathscr{H}-1}^{*}(\boldsymbol{b}_1)|\boldsymbol{b}_0, \boldsymbol{a}_0\right]\right). \tag{53}$$

For very long or infinite horizons, the discount factor can be set to $\gamma < 1$. Using this equation, the optimal policy can be expressed as

$$\pi_0^{*}(\boldsymbol{b}_0) = \arg\min_{a_0\in A}\left(C_B(\boldsymbol{b}_0, \boldsymbol{a}_0) + \gamma \cdot E\left[V_{\mathscr{H}-1}^{*}(\boldsymbol{b}_1)|\boldsymbol{b}_0, \boldsymbol{a}_0\right]\right). \tag{54}$$

For each $\boldsymbol{b}_k$ and $\boldsymbol{a}_k$, the optimal so-called Q-value is then defined as

$$Q_{\mathscr{H}-k}(\boldsymbol{b}_k, \boldsymbol{a}_k) = C_B(\boldsymbol{b}_k, \boldsymbol{a}_k) + \gamma \cdot E\left[V_{\mathscr{H}-k-1}^{*}(\boldsymbol{b}_{k+1})|\boldsymbol{b}_k, \boldsymbol{a}_k\right]. \tag{55}$$

Another way to find the optimal policy is to find the action $\boldsymbol{a}_k$ that minimizes the optimal Q-value:

$$\pi_k^{*}(\boldsymbol{b}_k) = \arg\min_{a_k\in A}(Q_{\mathscr{H}-k}(\boldsymbol{b}_k, \boldsymbol{a}_k)). \tag{56}$$

Therefore, it is necessary to calculate the Q-value for all possible actions, which is generally infeasible.

Generally, POMDPs can be solved both online as well as offline. Which type of solution is applied depends on the size of the state space. The so-called state-space explosion limits the usefulness of exact offline techniques.

Many offline methods are based on the so-called value iteration (VI). This technique iteratively calculates the cost/reward values of all possible states. An exact approach is, e.g., the One-Pass algorithm [54]. Examples for approximate point-based algorithms are PBVI, or Perseus [56]. Exact methods often lead to highly complex optimization problems, while approximate point-based methods require many grid points within the state space to converge toward the exact solution. The advantage of offline solutions is that the POMDP is fully solved only once. Unfortunately, this type of method is already infeasible for a very small state space.

Contrary to that, online algorithms only solve a small currently relevant part of the POMDP. This makes them less accurate but much easier and faster to compute. Some of the online approaches involve approximate tree methods (see, for example, the overview in [48]) or Monte Carlo sampling (e.g., PR).

Acknowledgments

This research was partially funded by Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO) through the "Integrated Cooperative Automated Vehicles" project (i-CAVE).

REFERENCES

[1] D. Angelova and L. Mihaylova
"Joint target tracking and classification with particle filtering and mixture Kalman filtering using kinematic radar information,"
Digit. Signal Process., vol. 16, no. 2, pp. 180–204, 2006.

[2] K. Bell, C. Kreucher, and M. Rangaswamy
"An evaluation of task and information driven approaches for radar resource allocation,"
in *Proc. IEEE Radar Conf.*, 2021, pp. 1–6.

[3] R. E. Bellman
Dynamic Programming. Princeton, NJ, USA: Princeton Univ. Press, 1957.

[4] J. M. Bernardo and A. F. M. Smith
Bayesian Theory. Hoboken, NJ, USA: Wiley, 1994.

[5] D. P. Bertsekas
Constrained Optimization and Lagrange Multiplier Methods. Nashua, NH, USA: Athena Scientific, 1996.

[6] D. P. Bertsekas and D. A. Castanon
"Rollout algorithms for stochastic scheduling problems,"
J. Heuristics, vol. 5, no. 1, pp. 89–108, Apr. 1999.

[7] D. P. Bertsekas, J. N. Tsitsiklis, and C. Wu
"Rollout algorithms for combinatorial optimization,"
J. Heuristics, vol. 3, no. 3, pp. 245–262, Dec. 1997.

[8] D. Bertsimas and J. N. Tsitsiklis
Introduction to Linear Optimization. Nashua, NH, USA: Athena Scientific, 1997.

[9] S. S. Blackman and R. Popoli
Design and Analysis of Modern Tracking Systems. London, UK: Artech House, 1999.

[10] D. Blatt and A. O. Hero
"From weighted classification to policy search,"
in *Proc. 18th Int. Conf. Neural Inf. Process. Syst.*, 2005, pp. 139–146.

[11] M. Bockmair, C. Fischer, M. Letsche-Nuesseler, C. Neumann, M. Schikorr, and M. Steck
"Cognitive radar principles for defence and security applications,"
IEEE Aerosp. Electron. Syst. Mag., vol. 34, no. 12, pp. 20–29, Dec. 2019.

[12] F. Bolderheij, F. G. J. Absil, and P. van Genderen
"A risk-based object-oriented approach to sensor management,"
in *Proc. 8th Int. Conf. Inf. Fusion*, 2005, pp. 1–8.

[13] S. Boyd and L. Vandenberghe
Convex Optimization. Cambridge, UK: Cambridge Univ. Press, 2004.

[14] S. Brüggenwirth, M. Warnke, S. Wagner, and K. Barth
"Cognitive radar for classification,"
IEEE Aerosp. Electron. Syst. Mag., vol. 34, no. 12, pp. 30–38, Dec. 2019.

[15] D. A. Castañón
"Approximate dynamic programming for sensor management,"
in *Proc. 36th IEEE Conf. Decis. Control*, 1997, pp. 1202–1207.

[16] A. Charlish and F. Hoffmann
"Anticipation in cognitive radar using stochastic control,"
in *Proc. IEEE Radar Conf.*, 2015, pp. 1692–1697.

[17] A. Charlish and F. Hoffmann
"Cognitive radar management,"
in *Novel Radar Techniques and Applications: Waveform Diversity and Cognitive Radar, and Target Tracking and Data Fusion* Vol. 2. London, UK: SciTech Publishing, pp. 157–193, 2017.

[18] A. Charlish, F. Hoffmann, and I. Schlangen
"The development from adaptive to cognitive radar resource management,"
IEEE Aerosp. Electron. Syst. Mag., vol. 35, no. 6, pp. 8–19, Jun. 2020.

[19] A. Charlish, K. Woodbridge, and H. Griffiths
"Phased array radar resource management using continuous double auction,"
IEEE Trans. Aerosp. Electron. Syst., vol. 51, no. 3, pp. 2212–2224, Jul. 2015.

[20] E. K. P. Chong, C. M. Kreucher, and A. O. Hero
"Partially observable Markov decision process approximations for adaptive sensing,"
Discrete Event Dyn. Syst., vol. 19, no. 3, pp. 377–422, Sep. 2009.

[21] T. H. de Groot
"Mission-driven resource management for reconfigurable sensing systems," Ph.D. thesis, Delft Univ. Technol., Delft, Netherlands, 2015.

[22] M. E. Gomes-Borges, D. Maltese, P. Vanheeghe, and E. Duflos
"A Risk-based sensor management using random finite sets and POMDP,"
in *Proc. 20th Int. Conf. Inf. Fusion*, 2017, pp. 1–9.

[23] S. Haykin
"Cognitive radar: A way of the future,"
IEEE Signal Process. Mag., vol. 23, no. 1, pp. 30–40, Jan. 2006.

[24] A. O. Hero, D. A. Castañón, D. Cochran, and K. Kastella
Foundations and Applications of Sensor Management, New York, NY, USA: Springer, 2008.

[25] A. O. Hero and D. Cochran
"Sensor management: Past, present, and future,"
IEEE Sensors J., vol. 11, no. 12, pp. 3064–3075, Dec. 2011.

[26] K. Hintz
Sensor Management in ISR, Boston, MA, USA: Artech House, 2020.

[27] D. Hitchings and D. A. Castañón
"Receding horizon stochastic control algorithms for sensor management,"
in *Proc. Amer. Control Conf.*, pp. 6809–6815, 2010.

[28] S. Julier and J. Uhlmann
"Unscented filtering and nonlinear estimation,"
Proc. IEEE, vol. 92, pp. 401–422, Mar. 2004.

[29] F. Katsilieris
"Sensor management for surveillance and tracking: An operational perspective," Ph.D. thesis, Delft Univ. Technol., Delft, Netherlands, 2015.

[30] F. Katsilieris, H. Driessen, and A. Yarovoy
"Threat-based sensor management for joint target tracking and classification,"
in *Proc. 18th Int. Conf. Inf. Fusion*, 2015, pp. 435–442.

[31] F. Katsilieris, H. Driessen, and A. Yarovoy
"Threat-based sensor management for target tracking,"
IEEE Trans. Aerosp. Electron. Syst., vol. 51, no. 4, pp. 2772–2785, Oct. 2015.

[32] R. Klemm, H. Griffiths, and W. Koch
Novel Radar Techniques and Applications, Volume 2 - Waveform Diversity and Cognitive Radar, and Target Tracking and Data Fusion. London, UK: Scitech Publishing, 2017.

[33] W. Koch
"On adaptive parameter control for phased-array tracking," *Signal Data Process. Small Targets*, vol. 3809, pp. 444–455, 1999.

[34] C. Kreucher and A. O. Hero
"Non-myopic approaches to scheduling agile sensors for multistage detection, tracking and identification," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process.*, 2005, pp. 885–888.

[35] C. Kreucher, K. Kastella, and A. O. Hero
"Sensor management using an active sensing approach," *Signal Process.*, vol. 85, no. 3, pp. 607–624, Mar. 2005.

[36] J. Langford and B. Zadrozny
"Reducing T-step reinforcement learning to classification. Unpublished, 2003. [Online]. Available: https://hunch.net/jl/projects/reductions/RL to class/colt submission.ps.

[37] B. La Scala, W. Moran, and R. Evans
"Optimal adaptive waveform selection for target detection," in *Proc. Int. Conf. Radar*, 2003, pp. 492–496.

[38] F. L. Lewis, D. L. Vrabie, and V. L. Syrmos
Optimal Control, Hoboken, NJ, USA:Wiley, 2012.

[39] S. Martin
"Risk-based sensor resource management for field of view constrained sensors," in *Proc. 18th Int. Conf. Inf. Fusion*, 2015, pp. 2041–2048.

[40] S. Maskell
"Joint tracking of manoeuvring targets and classification of their manoeuvrability," *EURASIP J. Adv. Signal Process.*, vol. 2004, no. 15, 2004, Art. no. 613289.

[41] H. Meikle
Modern Radar Systems, Norwood, MA, USA: Artech House, 2008.

[42] H. S. Mir and A. Guitouni
"Variable dwell time task scheduling for multifunction radar," *IEEE Trans. Automat. Sci. Eng.*, vol. 11, no. 2, pp. 463–472, Apr. 2014.

[43] A. E. Mitchell
"Advancements and applications of the fully adaptive radar framework," Ph.D. thesis, Elect. Comput. Eng., Ohio State Univ., Columbus, OH, USA, 2018.

[44] A. E. Mitchell, G. E. Smith, K. L. Bell, A. J. Duly, and M. Rangaswamy
"Hierarchical fully adaptive radar," *IET Radar, Sonar Navigation*, vol. 12, pp. 1371–1379, Dec. 2018.

[45] P. W. Moo and Z. Ding
Adaptive Radar Resource Management, London, UK: Academic Press, 2015.

[46] A. Narykov, E. Delande, and D. E. Clark
"A formulation of the adversarial risk for multiobject filtering," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 57, pp. 2082–2092, Aug. 2021.

[47] D. Papageorgiou and M. Raykin
"A risk-based approach to sensor resource management," in *Advances in Cooperative Control and Optimization*. Heidelberg, Germany: Springer, 2007, pp. 129–144.

[48] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa
"Online planning algorithms for POMDPs," *J. Artif. Intell. Res.*, vol. 32, no. 1, pp. 663–704, Jul. 2008.

[49] V. Savchuk and C. P. Tsokos
"Bayesian theory and methods with applications," in *Atlantis Studies in Probability and Statistics*. Dordrecht, Netherlands: Atlantis Press, 2011.

[50] M. I. Schöpe, H. Driessen, and A. Yarovoy
"A constrained POMDP formulation and algorithmic solution for radar resource management in multi-target tracking," *ISIF J. Adv. Inf. Fusion*, vol. 16, no. 1, pp. 31–47, Jun. 2021.

[51] M. I. Schöpe, H. Driessen, and A. Yarovoy
"Multi-task sensor resource balancing using Lagrangian relaxation and policy rollout," in *Proc. 23rd Int. Conf. Inf. Fusion*, 2020, pp. 1–8.

[52] M. I. Schöpe, H. Driessen, and A. Yarovoy
"Optimal balancing of multi-function radar budget for multi-target tracking using Lagrangian relaxation," in "*Proc. 22nd Int. Conf. Inf. Fusion*," 2019, pp. 1–8.

[53] M. Shaghaghi and R. S. Adve
"Task selection and scheduling in multifunction multichannel radars," in *Proc. IEEE Radar Conf.*, 2017, pp. 969–974.

[54] R. D. Smallwood and E. J. Sondik
"The optimal control of partially observable Markov processes over a finite horizon," *Operations Res.*, vol. 21, no. 5, pp. 1071–1088, 1973.

[55] S. Sowelam and A. Tewfik
"Optimal waveform selection for radar target classification," in *Proc. Int. Conf. Image Process.*, 1997, pp. 476–479.

[56] M. T. J. Spaan and N. Vlassis
"Perseus: Randomized point-based value iteration for POMDPs," *J. Artif. Intell. Res.*, vol. 24, pp. 195–220, 2005.

[57] L. D. Stone, R. L. Streit, T. L. Corwin, and K. L. Bell
Bayesian Multiple Target Tracking, Boston, MA, USA: Artech House, 2013.

[58] K. A. B. White, J. L. Williams, and P. Hoffensetz
"Radar sensor management for detection and tracking," in "*Proc. 11th Int. Conf. Inf. Fusion*," 2008, pp. 1–8.

[59] K. A. B. White and J. L. Williams
"Lagrangian relaxation approaches to closed loop scheduling of track updates," *Signal Data Process. Small Targets*, vol. 8393, pp. 8393–8393, 2012.

[60] J. Wintenby and V. Krishnamurthy
"Hierarchical resource management in adaptive airborne surveillance radars," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, no. 2, pp. 401–420, Apr. 2006.

Max Ian Schöpe received the B.Eng. degree in Electrical Engineering from the University of Applied Sciences, Hamburg, Germany in February 2015. Subsequently, he started the M.Sc. program in Telecommunication and Sensing Systems at Delft University of Technology, Delft, The Netherlands, which he finished in October 2017. From 2017 to 2021, he was a Ph.D. candidate with the Microwave Sensing, Signals and Systems (MS3) group of TU Delft. His main research interests are in the area of radar resource management using a partially observable Markov decision process framework.

Hans Driessen received the M.Sc. and Ph.D. degrees from the Department of Electrical Engineering, Delft University of Technology, Delft, The Netherlands, in 1987 and 1992, respectively. Since then, he has been employed with Thales Nederland BV, Hengelo, The Netherlands. He has been and is still involved in various (international) research projects and radar development programs in the areas of signal/data processing and radar management. Since January 2015, he additionally holds a part-time position as an Associate Professor of EEMCS faculty at the TU Delft in the Microwave Signals Sensor and Systems group in the field of radar systems, waveforms, and processing. His research interests are in the practical application of detection, estimation, information, and control theory to various problems in sensor systems.

Alexander G. Yarovoy (FIEEE' 2015) graduated from Kharkov State University, Ukraine, in 1984 with a Diploma with honors in radiophysics and electronics. He received the Candidate Phys. & Math. Sci. and Doctor Phys. & Math. Sci. degrees in radiophysics in 1987 and 1994, respectively. In 1987, he joined the Department of Radiophysics, Kharkov State University, Kharkiv, Ukraine as a Researcher and became a Full Professor there in 1997. From September 1994 to 1996, he was with the Technical University of Ilmenau, Ilmenau, Germany as a Visiting Researcher. Since 1999, he has been with the Delft University of Technology, Delft, The Netherlands. Since 2009, he has been leading there a chair of Microwave Sensing, Systems, and Signals. His main research interests are in high-resolution radar, microwave imaging, and applied electromagnetics (in particular, UWB antennas). He has authored and coauthored more than 450 scientific or technical papers, 6 patents, and 14 book chapters. He is the recipient of the European Microwave Week Radar Award for the paper that best advances the state-of-the-art in radar technology in 2001 (together with L. P. Ligthart and P. van Genderen) and in 2012 (together with T. Savelyev). In 2010, together with D. Caratelli, Prof. Yarovoy got the best paper award from the Applied Computational Electromagnetic Society (ACES). He served as the General TPC Chair of the 2020 European Microwave Week (EuMW'20), as the Chair and TPC Chair of the 5th European Radar Conference (EuRAD'08), as well as the Secretary of the 1st European Radar Conference (EuRAD'04). He also served as the cochair and TPC Chair of the 10th International Conference on GPR (GPR2004). He served as an Associated Editor of the International Journal of Microwave and Wireless Technologies from 2011 to 2018 and as a Guest Editor of five special issues of the IEEE Transactions and other journals. In the period 2008–2017, Prof. Yarovoy served as Director of the European Microwave Association (EuMA).

INTERNATIONAL SOCIETY OF INFORMATION FUSION

ISIF Website: http://www.isif.org

2022 BOARD OF DIRECTORS*

2020–2022	2021–2023	2022–2024
Pieter De Villiers	Alta De Waal	Felix Govaers
Murat Efe	Fredrik Gustafsson	Lyudmila Mihaylove
Wolfgang Koch	Uwe Hanebeck	Paul Thomas

*Board of Directors are elected by the members of ISIF for a three year term.

PAST PRESIDENTS

Simon Maskell, 2021	Wolfgang Koch, 2013	W. Dale Blair, 2005
Paulo Costa, 2020	Roy Streit, 2012	Chee Chong, 2004
Paulo Costa, 2019	Joachim Biermann, 2011	Xiao-Rong Li, 2003
Lyudmila Mihaylova, 2018	Stefano Coraluppi, 2010	Yaakov Bar-Shalom, 2002
Lyudmila Mihaylova, 2017	Elisa Shahbazian, 2009	Pramod Varshney, 2001
Jean Dezert, 2016	Darko Musicki, 2008	Yaakov Bar-Shalom, 2000
Darin Dunham, 2015	Erik Blasch, 2007	Jim Llinas, 1999
Darin Dunham, 2014	Pierre Valin, 2006	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 17 Number 1 June 2022 ISSN 1557-6418