June 2010 Volume 5 Number 1 ISSN 1557-6418

JAIF 10

Journal of Advances in Information Fusion

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

Regular Papers Page

ISIF

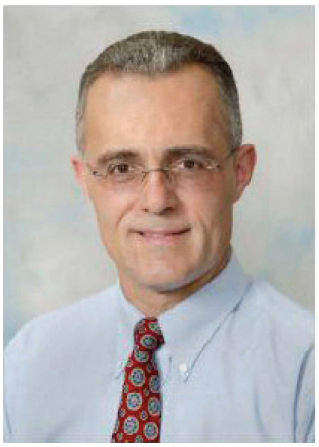
INTERNATIONAL SOCIETY OF INFORMATION FUSION

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.

From the ISIF Administrative Editor and Vice President for Publications

June 2010



Timelines and Timeliness for Publication of JAIF

Timely publication is a criterion used by citation indexing services to measure the quality of a technical journal. For JAIF, being a semi-annual publication, this means that at least four papers need to be finished with the review process, typesetting, and published every six months.

The ability to meet publication deadlines in a timely manner is important not only for the authors, but it also allows a journal to meet one of the requirements for inclusion in academic citation indexing services. Examples of such services include the ISI citation index of Thomson Scientific, and Scopus of Elsevier.

ISIF is very interested in seeing JAIF indexed in as many indexing services as possible. This cannot be accomplished without the cooperation and diligence of the editorial staff, referees, and authors to complete all assigned tasks for a manuscript in a timely manner.

To help ensure that JAIF is published in a timely manner, all persons involved with the reviewing, decision, revisions, proofing and final preparation of a manuscript should consider the following guidelines.

The table presents an illustrative timeline of events for 14 stages from submission to publication for all manuscripts submitted to JAIF.

From the table, it can be seen that a manuscript with all parties meeting the required deadlines and one review cycle will take approximately 34 weeks to go from submission to publication. Therefore, in order for JAIF to achieve timely publication, all persons in the review cycle must make every possible effort to meet their requested deadlines.

To aid in the timely preparation of manuscripts for publication in Stages 8 through 14, reminders will be sent to all delinquent authors and editors after one week. These reminders will indicate that the paper needs to be prepared for typesetting as soon as possible. If no action is taken in response to the reminder within one week, the manuscript will be excluded from publication in the next available issue and moved to a subsequent issue.

ISIF is looking forward to the fullest cooperation of all involved in the publication of papers in JAIF.

Stage	Action Item	Times for each item plus the required time to complete item
1	Author submits manuscript.	T_0
2	Editor in Chief (EIC) assigns Area Editor.	$T_1 = T_0 + 1$ week
3	Area Editor assigns Associate Editor.	$T_2 = T_1 + 1$ week
4	Associate Editor invites three referees.[i]	$T_3 = T_2 + 2$ weeks
5	Referees review manuscript.	$T_4 = T_3 + 8$ weeks
6	Associate Editor makes recommendation based on referee review, and Editor in Chief approves recommendation and forwards to author.	$T_5 = T_4 + 1$ week
7	If the manuscript is not rejected, author revises manuscript based on the recommended changes of the referees and resubmits the manuscript for further review.[ii]	$T_6 = T_5 + 8$ weeks
8	For all accepted manuscripts, author includes recommended changes of the referees, and prepares manuscript files for publication and typesetting.	$T_7 = T_6 + 3$ weeks
9	Administrative Editor verifies that final manuscript files are complete.	$T_8 = T_7 + 1$ week
10	Assistant EICor Assistant Administrative Editor performs a copy edit review of accepted manuscripts.[iii]	$T_9 = T_8 + 4$ weeks
11	Author includes final copy edits directly to manuscript files, and copy editor sends file to typesetter for publication.	$T_{10} = T_9 + 2$ weeks
12	Typesetter prepares manuscript for publication.	$T_{11} = T_{10} + 2$ weeks
13	Author Reviews final typeset version of manuscript.	$T_{12} = T_{11} + 2$ weeks
14	Administrative Editor posts manuscript.	$T_{13} = T_{12} + 1$ week

[i]If a referee does not respond within one week of receiving an invitation, an alternate referee should be contacted.
[ii]This step and the previous two steps of the review cycle will be repeated for conditionally accepted manuscripts until final acceptance without any further review.
[iii]In this step of the review cycle the author is involved with, and approves, the copy edit recommendations.

R. S. Lynch
JAIF Administrative Editor

Y. Bar-Shalom
ISIF VP Publications

Multi-step Look-Ahead Policy for Autonomous Cooperative Surveillance by UAVs in Hostile Environments

X. TIAN
Y. BAR-SHALOM
K. R. PATTIPATI

In this paper a real-time cooperative path decision algorithm for UAV surveillance is proposed. The surveillance mission includes multiple objectives: i) navigate the UAVs safely in a hostile environment; ii) search for new targets in the surveillance region; iii) classify the detected targets; iv) maintain tracks on the detected targets. To handle these competing objectives, a *layered decision framework* is proposed, in which different objectives are deemed relevant at different decision layers according to their priorities. Compared to previous work, in which multiple objectives are integrated into a single global objective function, this layered decision framework allows detailed specification of the desired performance for each objective and guarantees that an objective with high priority will be better satisfied by eliminating possible compromises from other less important ones. In addition, specific path decision strategies that are suited to the individual objectives can be used at different decision layers. An important objective of the path decision algorithm is to navigate the UAV safely in the hostile environment. To achieve this, it is shown necessary to increase the time horizon of the path decisions. In order to overcome the computational explosion of an optimal multi-step look-ahead path decision strategy, a Rollout Policy is proposed. This policy has moderate complexity and, when used in the layered decision framework, it is able to find safe paths effectively and efficiently. When the number of UAVs is large, the formation of UAV decision groups based on a nearest neighbor rule is proposed to control the complexity of the path decision algorithm. Further flexibility of assigning different objectives to the UAVs is also discussed. Simulation results show that the proposed path decision algorithm can guide the group of UAVs efficiently and safely for the multi-objective surveillance mission.

Manuscript received January 10, 2008; revised October 7, 2008; released for publication October 30, 2008.

Refereeing of this contribution was handled by Stephane Paradis.

Authors' address: Dept. of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269-2157, E-mail: ({xin.tian, ybs, Krishna}@engr.uconn.edu).

1557-6418/10/$17.00

1. INTRODUCTION

Recently a considerable amount of research effort has been directed toward the navigation and cooperative control of groups of unmanned (or uninhabited) aerial vehicles (UAVs). The advantages of UAVs include greater mobility, removal of risk to human operators, potentially lower cost, smaller size/weight, and the possibility of effective coordination. These features make them ideal for repetitive or dangerous tasks in both military and civilian applications [17]. A number of UAV management algorithms that serve various applications can be found in the literature. Ref. [14] addresses the problem of cooperatively controlling multiple UAVs so that they reach a predetermined target location simultaneously, while maximizing the survivability of the UAVs against exposed threats and adhering to the fuel constraints. A hierarchical decision mechanism is proposed in which at the team level the estimated time until arrival is computed and at the UAV level path planning is performed. In [5] a similar approach, which includes a Voronoi diagram in path planning, is used for the simultaneous intercept problem in the presence of dynamic threats. Similar approaches can be found in [6, 13, 15]. In [8] another hybrid control structure is proposed for the simultaneous intercept problem. Ref. [22] studies the task assignment problem for a group of UAVs.

We focus on the surveillance application of UAVs. The scenario considered involves a group of UAVs that search and track ground moving targets in a hostile environment. The objectives in the surveillance mission include: i) navigate the UAVs safely in a hostile surveillance environment; ii) search for new targets; iii) classify the detected targets; iv) maintain tracks on the detected targets. The conventional method of handling multiple objectives is to construct a combined objective function, e.g., the weighted sum approach used in [16, 23]. Our previous work [18] also uses the weighted sum approach, where the path decision problem is formulated as a nonlinear programming problem and solved by optimizing the global objective function over the continuous control variables (turn rates of the UAVs). However, there are several drawbacks to this. First, since different objectives have different meanings, the weighted sum of the objective functions is difficult to interpret and validate. Second, when the path decisions are made by maximizing a combined global objective function, it is hard to specify the requirements for the individual objectives. For example, for the objective tracking, it would be more reasonable to require the RMS errors of a given target to be no larger than a predefined level as opposed to requiring the errors to be as small as possible. Third, the simultaneous impact of multiple objectives on the path decisions could compromise the satisfaction of one or another objective in an unpredictable manner. As shown in Section 4, the survival probabilities of the UAVs can drop significantly when a combined global objective function is used for

path decisions. Ref. [11] proposes an algorithm for the design of the weights for the weighted sum approach in order to achieve a desirable tradeoff in the different objectives. However, the computational requirements of this algorithm are too involved for complex cooperative tasks and may preclude the possibility of multi-step look-ahead policies.

In this paper, a novel approach—*layered decision framework*—is proposed to handle multiple objectives in a surveillance mission. In the layered decision framework, instead of combining different objectives into a single objective function, multiple objectives are in separate decision layers according to their priorities. The control options[1] are evaluated first in the top decision layer, which results in a subset of controls that yields satisfactory results for the primary objective. Then, this subset of controls are passed down to the next decision layer for further selection. Proceeding through the decision layers, the control options are sifted and reduced to the final decision with the best overall performance. Major benefits of this approach include: i) it allows the specification of desired performance for each individual objective in different layers; ii) an objective with a higher priority will be better satisfied by eliminating possible compromises from other less important ones; iii) depending on the nature of the objective, suitable path decision strategies can be used at each decision layer, which may lead to significant savings in computation. iv) computation can be saved when the path decisions can be made through some of the decision layers, because there is no need to evaluate the remaining less important ones.

In the surveillance problem considered, the objective of safe navigation is assigned the highest priority; this is based on the premise that the safety of a UAV is more important than gathering one extra measurement. The problem of navigating a single UAV in a hostile environment while chasing a target has been studied in [25], in which the UAV tries to stay within a defined proximity of its target while avoiding restricted regions and obstacles. A gradient search algorithm with a geometry based strategy is used for the path decisions. In the present paper, the threats come not only from fixed positions, but also from moving targets. A different approach based on a Rollout Policy [3] is proposed for path decisions. This is used in the decision layer for safe navigation and is shown to be able to find safe path decisions effectively with moderate complexity. For other surveillance objectives including—search, classification and tracking, following [18], specific objective functions are constructed based on certain information criteria. These objective functions, along with suitable path decision strategies, form the rest of the decision layers.

[1]The control variables are discretized into a set of control options. As shown later, this also facilitates the multi-step look-ahead path decision policy.

Two additional features are also incorporated into the path decision algorithm, which make it more efficient and flexible for a large number of UAVs. One is the formation of decision groups based on a nearest neighbor rule. By doing this, the complexity of the algorithm increases linearly with the number of UAVs. Another is to assign different objectives to UAVs, since it is common to require the UAVs to focus on different tasks in the surveillance region.

The paper is organized as follows. Section 2 formulates the surveillance problem. Section 3 is devoted to the layered decision framework for surveillance with multiple objectives. In Section 4, the problem of safe navigation in a hostile environment is studied. The multi-step look-ahead path decision strategy is proposed using a Rollout Policy, and it is shown to be effective in solving the problem of safe navigation. In Section 5, the construction of small decision groups and how to assign different objectives to the UAVs are discussed. Simulation results are also presented to show the effectiveness of the algorithm. Section 6 presents the conclusions.

2. THE SURVEILLANCE MODELS AND OBJECTIVE FUNCTIONS

In this paper, the surveillance scenario follows mostly [19]. To make this paper self-contained, all models used are described in this section, including UAV specifications, models for threats in the surveillance region, as well as, tracking, search and classification models. It is worth mentioning that, for the sake of simplicity, we assume the surveillance mission takes place in a 2-D plane, namely, altitudes of the UAVs are not taken into account. This, however, does not compromise the main ideas of the paper, which are the *layered decision framework* and the *multi-step look-ahead path decision policy* for UAV navigation.

2.1. UAV Characteristics

Assume that fixed wing unmanned aerial vehicles are used for surveillance. The UAVs can fly only within a speed interval and have limited maneuverability. Following the formulation in [18], it is assumed that the UAVs move with a constant speed V_{uav} and the maximum turn rate the UAVs can take is Φ_{max}. Unlike in [18], the control of the UAVs is discretized into D levels, namely the UAVs can only take turn rates from a finite set. For example, when $D = 3$ the control set is $\{-\Phi_{max}, 0, \Phi_{max}\}$. It is assumed that the path decisions are made every T seconds. For cooperation, the UAVs need to exchange information of their states and measurements from the onboard sensors. In this paper, a centralized data processing framework is used, that is all the information from the UAV network is available for data fusion and path decisions. While the proposed path decision algorithm works best in a centralized setting, it can be used in a distributed system by treating

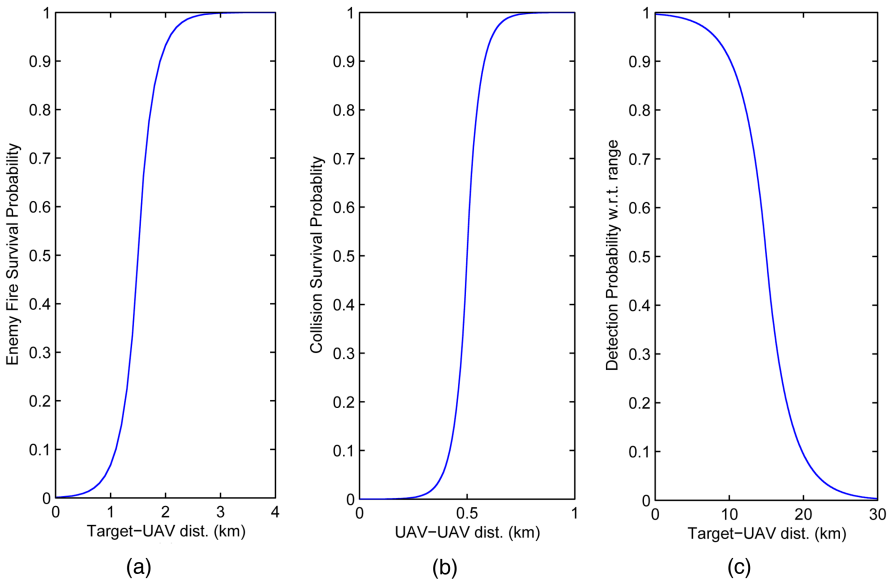


Fig. 1. Detection and survival probabilities as functions of distance to target.

each individual UAV as a duplication of the decision center. The issue of synchronizing information among distributed agents (UAVs) in a distributed system is beyond the scope of this paper.

2.2. The Model of the Threats in the Surveillance Region

In the surveillance, UAV losses may happen due to hostile fire and collisions among the UAVs. As in [18], these potential threats are incorporated into the survival probabilities of the UAVs. The survival probability of UAV s equals the product of target-fire survival probability $\pi_S^1(s)$, stationary-attack survival probability $\pi_S^2(s)$, and collision survival probability $\pi_S^3(s)$, i.e.,

$$\pi_S(s) = \pi_S^1(s)\pi_S^2(s)\pi_S^3(s) \tag{1}$$

where $\pi_S^1(s)$, is the product of target-fire survival probabilities of UAV s in view of each target j, i.e.,

$$\pi_S^1(s) = \prod_j \pi_S^1(s,j). \tag{2}$$

Similarly, for the attacks from stationary threats, survival probability of the UAV is

$$\pi_S^2(s) = \prod_l \pi_S^2(s,l) \tag{3}$$

and $\pi_S^3(s)$ is the product of collision survival probabilities corresponding to all other UAVs,

$$\pi_S^3(s) = \prod_{i:i\neq s} \pi_S^3(s,i). \tag{4}$$

The nature of these survival probabilities is application-dependent. In this paper, the probabilities are modeled as functions of distance as shown in Fig. 1: (a) for π_S^1 and π_S^2 and (b) for π_S^3. For safe navigation, the survival probabilities of the UAVs should be above a threshold,[2] e.g., 0.9, which is a design parameter of the algorithm.

2.3. The Tracking Model

Using a 2-D model, the kinematic state of the target is defined as

$$X = [x \quad \dot{x} \quad y \quad \dot{y}]'. \tag{5}$$

The target motion is modeled by the Discrete White Noise Acceleration (DWNA) model [1]. The UAVs are assumed to be equipped with Ground Moving Target Indicator (GMTI) radars, which measure the locations of moving ground targets as well as their radial velocities (Doppler). A 2-D measurement model is used

$$r_m = r + w_r \tag{6}$$

$$\alpha_m = \alpha + w_\alpha \tag{7}$$

$$\dot{r}_m = \dot{r} + w_{\dot{r}} \tag{8}$$

in which w_r, w_α and $w_{\dot{r}}$ are Gaussian noise with standard deviations σ_r, σ_α and $\sigma_{\dot{r}}$ respectively. Applying the Polar to Cartesian conversion [1], the measurement is converted to

$$Z_m = [x_m \quad y_m \quad \dot{r}_m] \tag{9}$$

where

$$x_m = r_m \cos\alpha_m \tag{10}$$

$$y_m = r_m \sin\alpha_m. \tag{11}$$

[2]This threshold serves as a soft boundary, the path decision algorithm should be able to keep the survival probabilities above or close to this safety bound.

The noise in the converted measurement is zero-mean[3] with covariance matrix

$$R = \begin{bmatrix} R_{1,1} & R_{1,2} & 0 \\ R_{1,2} & R_{2,2} & 0 \\ 0 & 0 & \sigma_{\dot{r}}^2 \end{bmatrix} \tag{12}$$

where

$$R_{1,1} = r_m^2\sigma_\alpha^2 \sin\alpha_m^2 + \sigma_r^2 \cos\alpha_m^2 \tag{13}$$

$$R_{2,2} = r_m^2\sigma_\alpha^2 \cos\alpha_m^2 + \sigma_r^2 \sin\alpha_m^2 \tag{14}$$

$$R_{1,2} = (\sigma_r^2 - r_m^2\sigma_\alpha^2)\sin\alpha_m \cos\alpha_m. \tag{15}$$

The observation matrix corresponding to (9) is (see, e.g., [24])

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & \cos\alpha & 0 & \sin\alpha \end{bmatrix} \tag{16}$$

where α_m can be used in H, as shown in [24].

The detection probability is a function of range and range rate of the target with respect to the GMTI radar. Denoted as π_D, the detection probability is given by

$$\pi_D = \pi_D^1(r)\pi_D^2(\dot{r}) \tag{17}$$

$\pi_D^1(r)$ is shown in Fig. 1(c). For a GMTI radar, if the range rate for a target falls below a threshold $\dot{r}_{\min}$ then the target will not be detected. Hence,

$$\pi_D^2(\dot{r}) = 1 - \text{Prob}\{-\dot{r}_{\min} < \dot{r} < \dot{r}_{\max}\}. \tag{18}$$

For tracking, it is assumed that the UAVs obtain measurements from the detected targets every T seconds. Following [18], at decision time kT, the expected track update for target j at time $(k+1)T$ is

$$\hat{I}_j(k+1 \mid k+1) = I_j(k+1 \mid k) + \sum_{s=1}^{N} \hat{\pi}_D(s,j,k+1)\hat{H}(s,j,k+1)' \times \hat{R}(s,j,k+1)^{-1}\hat{H}(s,j,k+1) \tag{19}$$

in which I_j denotes the information matrix from the track of target j, namely, $I_j = P_j^{-1}$; N is the number of the UAVs; s is the index of the UAV. The "hat" marks indicate the values are expectations that depend on the relative positions of target j to the UAVs at $k+1$. Clearly, $\hat{I}_j(k+1 \mid k+1)$ is a function of the collective path decisions (controls) of the UAVs at k. To evaluate the expected quality of the track, the expected mean square position error can be used, since it is directly related to the RMS position error (components 1 and 3 of the state vector). For target j one has,

$$\widehat{\text{MSE}}(j,k+1) = \hat{P}_j(k+1 \mid k+1)_{(1,1)} + \hat{P}_j(k+1 \mid k+1)_{(3,3)}. \tag{20}$$

[3]Since the condition for the unbiasedness conversion [1] is satisfied, the noises in x_m and y_m can be assumed to be zero-mean.

The construction of the objective function for tracking is based on (20). Further details will be discussed in Section 3.

2.4. The Model for Search

Studies on the problem of cooperative search using multiple autonomous UAVs can be found in [10, 9, 16]. For different applications, formulations of the problem may change. In the surveillance problem considered, the surveillance region is divided into a number of sectors. It is assumed that each UAV scans a fixed number N_s of such sectors in each period of its operation. As in [18], in each sector, the arrival of new targets is modeled as a Poisson process.

Let $P_{m,n}(k)$ denote the probability that there is no new target in sector $\{m,n\}$ and $\lambda_{m,n}$ denote the Poisson parameter (expected spatial density of new targets) of this sector. At the kth decision time one has

$$P_{m,n}(k) = P_{m,n}(k-1)\exp^{-\lambda_{m,n}T}. \tag{21}$$

If the sector is scanned by UAV s at k with a detection probability of $\tilde{\pi}_D(m,n,s,k)$, it follows (assuming there are no false alarms) that the updated probability $P_{m,n}(k^+)$ is given by

$$P_{m,n}(k^+) = \begin{cases} \dfrac{P_{m,n}(k)}{P_{m,n}(k) + [1-P_{m,n}(k)][1-\tilde{\pi}_D(m,n,s,k)]} \\ \qquad \text{if scanned and no target was detected} \\ 1 \qquad \text{if scanned and a target was detected} \end{cases}. \tag{22}$$

An intuitive interpretation of (22) is as follows. When scanned, no target is detected with probability $P_{m,n}(k) + [1-P_{m,n}(k)][1-\tilde{\pi}_D(m,n,s,k)]$. So, the updated probability $P_{m,n}(k^+)$ is as given in the first probability of (22). If a target is detected, no new target is in that sector with probability 1. From (22), the payoff of a specific scan can be calculated as

$$\begin{aligned} \Delta(m,n,s,k) &= \text{E}[P_{m,n}(k^+)] - P_{m,n}(k) \\ &= [1-P_{m,n}(k)]\tilde{\pi}_D(m,n,s,k). \end{aligned} \tag{23}$$

For a single UAV its scan decision can be made by selecting the most profitable (largest Δ given by (23)) N_s sectors, which favors the sectors that are more likely to have new targets (low $P_{m,n}(k)$) and the potential new targets are more likely to be detected (high $\tilde{\pi}_D(m,n,s,k)$). In the multiple UAV case, the optimal scan decision is a complicated assignment problem. However, a near-optimal solution can be found using simple heuristics. Since the UAVs tend to operate in different regions (to produce good coverage to the whole surveillance area), their N_s best sectors to scan are very unlikely to overlap, which allows the UAVs to make their scan decisions independently; rare conflicts can be resolved by making

their scan decisions sequentially in the order of the UAV indices.

2.5. The Model for Target Classification

An important objective of surveillance is to classify the detected targets. Studies of optimal search with joint detection and classification can be found in [20, 6, 12]. An integrated algorithm for tracking and classification with data association is presented in [2]. In this paper, following [19], the classification and tracking are considered as different problems as it is assumed that the GMTI radar mounted on the UAV does not provide classification information; instead, it is assumed that classification information is provided by a closed circuit digital (CCD) camera. A classifier associated with the camera processes the data from the camera. The outputs are class decisions and their associated class confusion matrix. The class confusion matrix gives the probabilities of class decision output given the actual class of the target. It is assumed that the class confusion matrix is only a function of target-UAV distance, i.e., spatial diversity does not improve classification results. Let $\zeta(j,s,k)$ denote the output of the classifier on UAV s at the kth decision time for target j and $C(j,s,k)$ be the corresponding class confusion matrix. Element $c_{ab}(j,s,k)$ of $C(j,s,k)$ is given by

$$c_{ab}(j,s,k) = \mathrm{P}(\zeta(j,s,k) = b \mid \kappa_j = a) \tag{24}$$

in which κ_j denotes the true class of the target.

To facilitate classification, when a new target is detected, the UAV closest to that target is assigned to perform the classification. The UAV will start to use the classification sensor when it gets close enough to the target. Notice that classification is a special case when the UAVs are focusing on different objectives. Such needs are common in multiple UAV surveillance. For example, some of the UAVs may focus on tracking while others focus on search. In Section 5.2, the problem of assigning different objectives to the UAVs will be discussed.

For a detected target, a class probability vector is used as the state for classification. Let μ_j denote the class probability vector for target j which can be initialized, e.g., as a uniform distribution over all possible classes. If the output of the classifier is $\zeta(j,s,k) = b$, then μ_j is updated as [2]

$$\mu_j^+ = \frac{C_b(j,s,k) \otimes \mu_j}{C_b(j,s,k)'\mu_j} \tag{25}$$

where $C_b(j,s,k)$ is the bth column of the class confusion matrix and $\otimes$ is the Schur-Hadamard product (term by term). The classification of target j is completed when

$$\max\{\mu_j\} > \tau_{\mathrm{CLS}} \tag{26}$$

in which τ_{CLS} is a confidence threshold, e.g., 0.95.

TABLE I
Decision Layers in the Path Decision Algorithm for Surveillance (s is the index of the UAVs and j is the index of targets)

Objective	Decision Layer (priority)	Satisfactory Level	Evaluation Criterion for the Accomplishment
Safe Navigation	1	τ_{PS}	$\min\{\pi_S(s), \tau_{\mathrm{PS}}\}$
Classification	2	τ_{CLS}	$\min\{\max\{\mu_j\}, \tau_{\mathrm{CLS}}\}$
Tracking	3	$\tau_{\mathrm{MSE}}(j)$	$\max\{\mathrm{MSE}(j), \tau_{\mathrm{MSE}}(j)\}$
Search	4	τ_{PNNT}	$\min\{P_{m,n}, \tau_{\mathrm{PNNT}}\}$

3. LAYERED DECISION FRAMEWORK FOR SURVEILLANCE MISSION WITH MULTIPLE OBJECTIVES

In this paper, a layered decision framework is used for handling multiple objectives, in which each objective occupies a decision layer according to its priority. A decision layer consists of: i) the objective; ii) a function that evaluates the degree of accomplishment of the objective; iii) a satisfactory level, at which point no further improvement on the objective is necessary. Table I shows an example of arrangement of the decision layers.

In the layered decision framework, an objective with a higher priority will be considered first. The key idea is that once a satisfactory level is reached, the "satisfied" objective will have no effect on the path decisions, thus freedom in the path decisions can be passed on to the next decision layer. To illustrate this, consider a simple case of a group of $N = 2$ UAVs tracking two targets while performing search in the surveillance region (classification is omitted in this example). Suppose the control of each UAV is discretized into $D = 3$ levels. Thus, at every decision epoch, the number of control options for the UAV group is $D^N = 9$. For simplicity, the example will stay with one-step look-ahead path decision (multi-step will be introduced later) and all the data in this example are for the purposes of illustration only.[4]

In this example, the control options are first evaluated by the top decision layer of safe navigation. Table II shows the m-best control options ($m = 5$ in this case) indicated by a check mark.[5]

When $m = 1$, this is the control option that yields the best result for the current objective and it is chosen directly as the path decision, since there is no freedom in control left for the remaining decision layers. If $m > 1$, these m best control options will be passed on to the next decision layer of tracking. As shown in Table III, the

[4] In actual simulations, the differences between different control options are much smaller than those shown in this example. However, by always following the best control option, the UAVs will navigate to desired positions by capturing the gradient information of the objective functions.

[5] In Tables II–III, control index (C_1, C_2) denotes a combination of the controls taken by the two UAVs, $C_1 \in \{1,2,3\}$ for UAV 1, $C_2 \in \{1,2,3\}$ for UAV 2.

TABLE II
Decision Layer 1: Control decisions for Safe Navigation with $N = 2$ UAVs and $\tau_{PS} = 0.9$

Control Index (C_1, C_2)	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)	(3,1)	(3,2)	(3,3)
$\hat{\pi}_S$	0.92	0.95	0.87	0.99	0.82	0.81	0.93	0.83	0.91
(Expected π_S at $k+1$)	1	0.98	0.91	0.92	0.87	0.95	1	0.91	0.92
$\bar{\pi}_S$	0.9	0.9	0.87	0.9	0.82	0.81	0.9	0.83	0.9
$(\min\{\hat{\pi}_S, \tau_{PS}\})$	0.9	0.9	0.9	0.9	0.87	0.9	0.9	0.9	0.9
Control Evaluation	0.81	0.81	0.783	0.81	0.713	0.729	0.81	0.747	0.81
$\left(\prod \bar{\pi}_S(s)\right)$	√	√		√			√		√

TABLE III
Decision Layer 2: Control decisions for Tracking with $\tau_{MSE} = 25$ m^2

Control Index (C_1, C_2)	(1,1)	(1,2)	(2,1)	(3,1)	(3,3)
$\widehat{\text{MSE}}(\text{m}^2)$	17	20	30	32	23
(Expected MSE at $k+1$)	23	22	27	29	31
$\overline{\widehat{\text{MSE}}}(\text{m}^2)$	25	25	30	32	25
$(\max\{\widehat{\text{MSE}}, \tau_{MSE}\})$	25	25	27	29	31
Control Evaluation	50	50	57	61	56
$\left(\sum \overline{\text{MSE}}(j)\right)$	√	√			

output of the second decision layer is a further reduced set of controls indicated by columns with a check mark. If the size of this reduced control set is greater than 1, it will be passed to the next decision layer for further selection.

The path decision algorithm ends when the best control option is found. The uniqueness of the final path decision can be guaranteed by simply setting the "satisfactory level" of the last decision layer to the "ideal" one. In this example, the last decision layer is "Search," thus τ_{PNNT} can be set to 1, which is an "ideal" level that can never be simultaneously achieved at all the sectors due to the limited scan capability of the UAV group. A similar procedure as in Tables II and III can be used for "Search" and it is omitted here for conciseness.

Compared to the weighted sum approach, the layered decision framework has the following advantages:

- Multiple objectives in the surveillance are clearly delineated. Thus, objectives with higher priorities are free from possible compromises from the less important ones. Section 4 will show that this is particularly important for the objective of safe navigation.
- For each objective, the "satisfactory" levels allow the path decision algorithm to be sensitive to the entities (e.g., targets in the tracking layer, sectors in the search layer) that demand more attention. Take tracking for example. The objective function is a combination of sub-objectives related to the tracks of all the targets. The use of the satisfactory level τ_{MSE} eliminates the impact of those sufficiently accurate tracks and allows the inaccurate tracks have more influences on the path decisions.
- The layered decision framework allows different path decision strategies to be used for the objectives. For example, depending on the nature of the objectives, they may or may not benefit from multi-step look-ahead strategies. Significant computation cost can be saved by decomposing the objectives among multiple decision layers.
- When a path decision is determined by the first few decision layers, the remaining layers do not need to be evaluated.

4. MULTI-STEP LOOK-AHEAD PATH DECISION STRATEGY FOR UAV NAVIGATION

An important objective for the path decision algorithm is to navigate the UAV group safely in the surveillance region. As specified in Section 2.2, the threats to the UAVs are modeled in terms of survival probabilities (1). In [18] the survival probabilities of the UAVs are incorporated into the global objective function through the track update as

$$\hat{I}_j(k+1 \mid k+1) = I_j(k+1 \mid k) + \sum_{s=1}^{N} \hat{\pi}_S(s,k+1)\hat{\pi}_D(s,j,k+1) \times \hat{H}(s,j,k+1)'\hat{R}(s,j,k+1)^{-1}\hat{H}(s,j,k+1) \tag{27}$$

which is a variation of (19). If the UAV survival probabilities, $\hat{\pi}_S(s,k+1)$, drop, there will be a reduction in the expected information gain. As a result, the path decision algorithm tends to avoid drops in the survival probabilities of the UAVs. While this formulation intuitively makes sense, it turns out to be incapable of preventing the UAV survival probabilities from significant drops. There are two reasons for this problem. First, tracking and safe navigation are two competing objectives. Particularly when a UAV is tracking a single target it tends to get close to the target, while safe navigation requires the UAV to keep adequate distance from the target. The combination of competing objectives into a single global objective function can lead to unpredictable compromises. Second, due to limited maneuverability of the UAV, a one-step look-ahead path decision strategy can result in late detections of potential safety risks. In the rest of this section, a multi-step

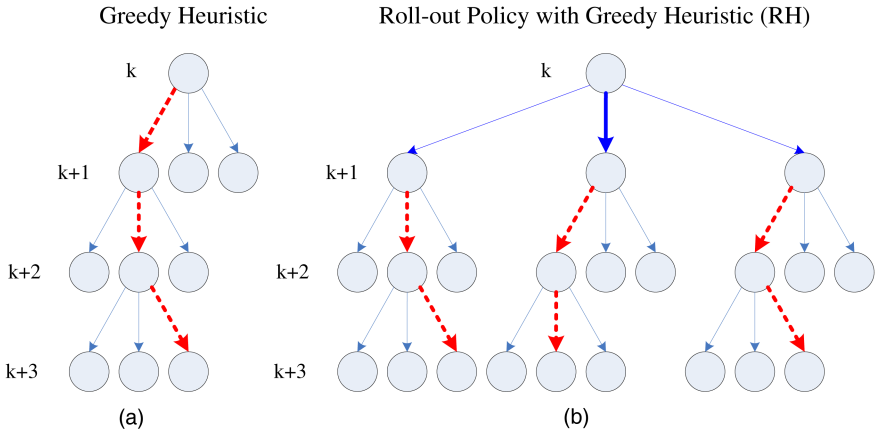


Fig. 2. Greedy Heuristic and Rollout Policy.

look-ahead path decision strategy is proposed based on the Rollout Policy [3]. When used in the decision layer of safe navigation, it is shown to produce significantly improved results.

4.1. Multi-step Look-ahead Path Decision and Rollout Policy

By discretizing the controls of the UAVs, a multi-step look-ahead path decision for the UAV group can be modeled as a combinatorial optimization problem. However, the problem is NP-hard, e.g., for a UAV group that consists of N UAVs, the optimal solution for a K-step look-ahead path decision needs to consider D^{NK} possible paths, which can be far too expensive for a real-time algorithm even with modest N and K. Instead of seeking the optimal solution, a suboptimal solution requiring less computation is much more desirable. The Rollout policy proposed in [3] is a suboptimal solution to the combinatorial optimization problems. Based on a heuristic solution to the problem (called a base heuristic), the Rollout policy is guaranteed to find a solution that is no worse than the base heuristic. Successful applications of the Rollout policy can be found in [4, 21], in which it works surprisingly well by producing near-optimal solutions.

In [3], the Rollout Policy was introduced in a Dynamic Programming (DP) context. Consider a problem with a finite set of feasible solutions and a cost function $g(u)$, $u \in U$. Each u has K components, namely, $u = (u_1, u_2, \ldots, u_K)$. In the K-step look-ahead path decision algorithm, the components $u_1, \ldots, u_K$ correspond to the controls at different times. An i-tuple $(u_1, u_2, \ldots, u_i)$, $i < K$, consisting of i components of the solution is called an i-solution. The optimal solution $u^* = (u_1^*, u_2^*, \ldots, u_K^*)$ can be obtained via DP, which gives

$$u_i^* = \arg\left\{ \min_{u_i \in U_i(u_1^*, u_2^*, \ldots, u_{i-1}^*)} J^*(u_1^*, u_2^*, \ldots, u_{i-1}^*, u_i) \right\}, \quad i = 1, 2, \ldots, K \qquad (28)$$

where J^* is the optimal cost-to-go function for any i-solution. However, the evaluation of J^* is, in most cases, not feasible. In the Rollout policy, a base heuristic algorithm H is used. From any i-solution $u = (u_1, u_2, \ldots, u_i)$, the heuristic algorithm H can generate a complete K-solution $u = (u_1, u_2, \ldots, u_K)$ whose cost is denoted by $h(u_1, u_2, \ldots, u_i)$. The suboptimal solution $\tilde{u} = (\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_K)$ is found by replacing J^* in (28) with the heuristic cost-to-go function h, namely,

$$\tilde{u}_i = \arg\left\{ \min_{u_i \in U_i(\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_{i-1})} h(\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_{i-1}, u_i) \right\}, \quad i = 1, 2, \ldots, K. \qquad (29)$$

For the K-step look-ahead path decision strategy, at k, the control that produces the best control sequence from $k+1$ to $k+K$ is selected; the Greedy heuristic, which is equivalent to the one-step look-ahead path decision, is used as the base heuristic to generate the control sequences. Fig. 2 illustrates the greedy heuristic and its corresponding Rollout policy in a 3-step look-ahead path decision strategy for a single UAV.

Assume that at each node, there are 3 controls (turn rates) available for the UAV. Using the Greedy heuristic, the control that leads to the next "node" with the best immediate result will be selected. Fig. 2(a) shows the path (control sequence) from k to $k+K$ $(= k+3)$ generated by Greedy heuristic (highlighted by the thick dashed arrows). In the Rollout policy, instead of starting from k, the greedy heuristic starts from $k+1$ to generate the remaining paths to $k+3$. The control at k that produces the best path to $k+3$ (highlighted by the thick dashed arrows in Fig. 2(b)) will be selected as the control decision. Note that the evaluations of the paths from k to $k+K$ are based on the information available at k and the procedure is repeated at every decision time with updated information. Compared to the exhaustive search which requires one to evaluate $\sum_{i=1}^{K} D^{N \cdot i}$ "nodes," the Rollout policy only evaluates $D^N + (K-1)D^{2N}$ nodes. The computational cost increases linearly with the decision horizon K.

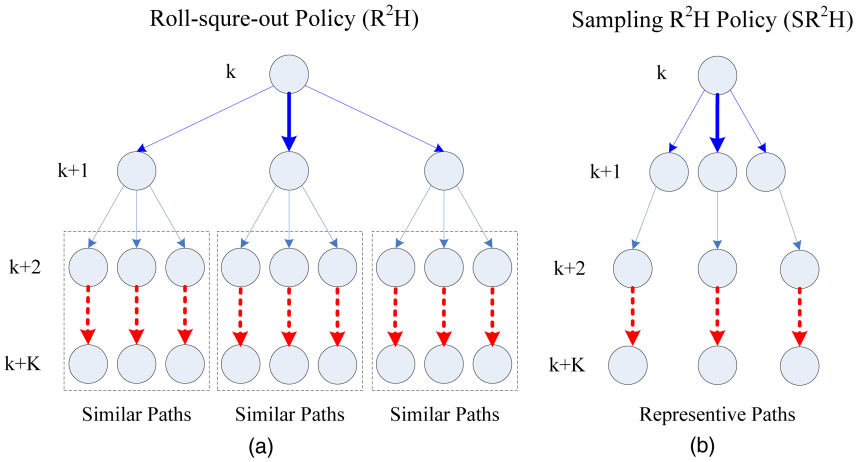


Fig. 3. Roll-square-out Policy and Sampling Roll-square-out Policy.

A variation of the Rollout policy RH is the Roll-square-out policy [3] (denoted as R^2H). As shown in Fig. 3(a), in R^2H the greedy heuristic (indicated by the dashed arrows) starts from $k+2$. R^2H (which needs to evaluate $D^N + D^{2N} + (K-2)D^{3N}$ "nodes") is more expensive than RH, while its results are guaranteed to be no worse than RH. In view of the specific feature of the path decision problem, it is reasonable to assume two close paths will produce similar performance. In Fig. 3(a), the dotted squares mark out 3 similar solution sets. By taking representative sample paths from the similar solution sets, R^2H can be simplified to a sampling R^2H policy (SR^2H). As illustrated in Fig 3(b) SR^2H is much less expensive than R^2H, which requires one to evaluate only $2D^N + (K-2)D^{2N}$ nodes. SR^2H is useful in UAV path decisions, since it increases the volume of the search space for optimal paths (the K-step look-ahead increases the time horizon of the search).

4.2. The Decision Layer for Safe Navigation

The proposed multi-step look-ahead path decision strategy (see Section 4.1) can be used in any decision layer in the layered decision framework (see Section 3). Instead of seeking one best control at k, at each decision layer, the path decision algorithm looks for m best controls which will be passed on to the next decision layer for further selection. An important issue in a K-step look-ahead path decision algorithm is to evaluate and compare the control sequences from k to $k+K$. Figs. 2–3 show that the evaluation of a control sequence from k to $k+K$ requires the evaluations of the nodes from $k+1$ to $k+K$. In the layer of safe navigation, a node at $k+i$ can be evaluated by

$$\hat{J}_S(k+i) = \sum_s \ln(\min\{\hat{\pi}_S(s,k+i), \tau_{\text{PS}}\}) \qquad (30)$$

where s is the index of the UAVs and τ_{PS} is the satisfactory level introduced in Table II. Accordingly, the evaluation of a control sequence from k to $k+K$ is given by

$$\sum_{i=1}^{K} \hat{J}_S(k+i). \qquad (31)$$

In addition, a control sequence is considered to be "safe" if the expected survival probabilities of the UAVs are above τ_{PS} along the path, namely,

$$\min_s\{\hat{\pi}_S(s,k+i)\} \geq \tau_{\text{PS}} \qquad \forall \quad i = 1,\ldots,K. \qquad (32)$$

Therefore, all "safe" control sequences have the same value (31), namely

$$\sum_{i=1}^{K} \hat{J}_S(k+i) = KN\ln(\tau_{\text{PS}}). \qquad (33)$$

Based on the above definitions, at the kth decision time, the procedure for a K-step look-ahead path decision algorithm for safe navigation is as follows:

- Use the Rollout Policy to generate control sequences from k to $k+K$.
- If "safe" control sequences that satisfy (33) are detected, pass the corresponding controls at k to the next decision layer.
- If no "safe" sequence is found, use the Sampling Rollout strategy to generate control sequences from k to $k+K$.
- If "safe" sequences are detected, pass the corresponding controls at k to the next decision layer.
- If still no "safe" sequence is found, the value of u_k that leads to the "best" control sequence (evaluated using (31)) is selected. The evaluations in the remaining decision layers are not needed.

4.3. Simulation Results for UAV Safe Navigation: Rollout vs. One-step Look-ahead

Consider first a "toy example" in which one UAV searches for and tracks one target. For simplicity, clas-

TABLE IV
Decision Layers in the Simulation

Objective	Decision Layer (priority)	Satisfactory Level	Evaluation Criterion for the Accomplishment	Strategy for Path Decision
Safe Navigation	1	$\tau_{\text{PS}} = 0.9$	$\min\{\pi_S(s), \tau_{\text{PS}}\}$	multi-step
Tracking	2	$\tau_{\text{MSE}} = 0$ m^2	$\max\{\text{MSE}(j), \tau_{\text{MSE}}\}$	one-step
Search	3	$\tau_{\text{PNNT}} = 1$	$\min\{P_{m,n}, \tau_{\text{PNNT}}\}$	one-step

sification is not included here. Table IV shows the decision layers of the path decision algorithm.[6] Note that τ_{MSE} in the tracking layer is set to zero, which means once the target is detected the UAV will "focus" on tracking. The surveillance region is 40 $km \times$ 40 km and is divided into 10×10 sectors. The target starts from $[2000, 14200]$ m with initial velocity $[10, -2]$ m/s. The process noise of the target has intensity $\sqrt{\tilde{q}} = 0.01$ m/s^2. It is assumed that $V_{\text{UAV}} =$ 40 m/s and the control set is $\{-3, 0, 3\}$ deg/s. The on board GMTI radar has measurement standard deviations of [10 m, 1 mrad, 1 m/s]. There are 3 stationary threats located at $[5000, 15000]$ m, $[7000, 7000]$ m and $[20000, 10000]$ m (indicated by the "asterisks"). The circles show the boundaries of the corresponding restricted zones within which the survival probability of the UAV from the threat is below the satisfactory level τ_{PS}. Specifications of the UAV survival probability and the target detection probability follow those in Section 2. Fig. 4 shows trajectories of the UAV and the target in one simulation. In this case the UAV has to circle around the target which is slower while avoiding certain regions.

For comparison, the combined objective approach, in which the survival probability of the UAV is incorporated into the expected update of the track in (27), is also tested. Notice that in the layered decision framework, safe navigation is treated separately from the objective of tracking; thus, unlike (27), the objective of the expected track update given in (19) does not deal with survival probabilities of the UAVs. A modified version of (27)

$$\begin{aligned}\hat{I}_j(k+1 \mid k+1) &= \min_s\{\hat{\pi}_S(s,k+1)\} I_j(k+1 \mid k) \\ &+ \sum_{s=1}^{N} \hat{\pi}_S(s,k+1)\hat{\pi}_D(s,j,k+1) \\ &\times \hat{H}(s,j,k+1)' \hat{R}(s,j,k+1)^{-1} \hat{H}(s,j,k+1)\end{aligned} \tag{34}$$

is tested as well, which places greater penalty to the drops in the survival probabilities.

[6]If the tactical value of the information is very high, safe navigation can be moved to a layer with lower priority.

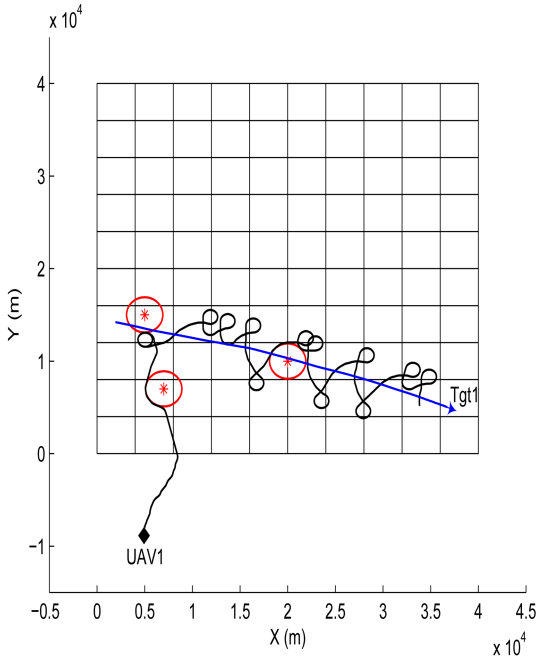


Fig. 4. UAV trajectory in one simulation using the layered decision framework (9-step look-ahead decisions for safe navigation).

Figs. 5–6 show the minimum survival probability of the UAV over 100 MC runs, in which "combined objective 1" refers to the approach that uses the expected update in (27) as the objective function and "combined objective 2" refers to the approach that uses the expected update (34) as the objective function. As shown in Fig. 5, the one-step look-ahead path decision strategy can not meet the requirement for safe navigation, no matter which objective function for path decision is used. In Fig. 6, although a 9-step look-ahead path decision strategy is used, significant drops in the survival probability of the UAV are still observed in the two combined objective approaches. However the 9-step look-ahead path decision strategy with the layered decision framework is able to keep the survival probability of the UAV close to the satisfactory threshold $\tau_{\text{PS}} = 0.9$. The rare drop to 0.8 occurred only once in the 100 runs. Fig. 7 compares the RMS position errors of the algorithms. Notice that, around the 100th decision time, the layered decision framework has larger RMS position errors than those of the combined objective function approaches, but the drops in the survival probability are avoided, as shown in Fig. 6. This is an example where an objective with higher priority

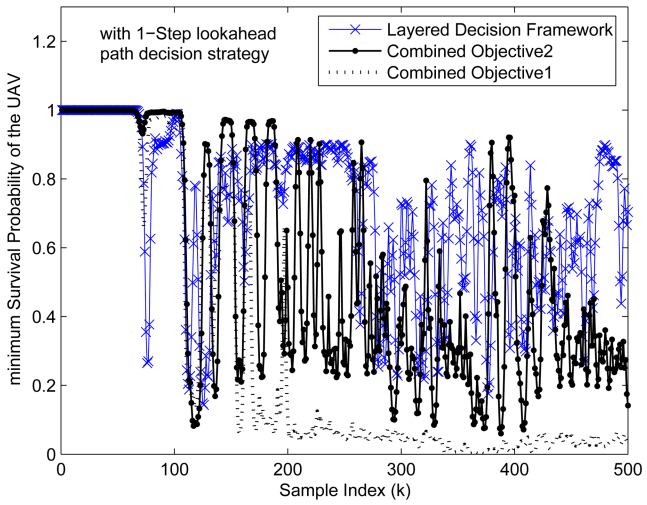


Fig. 5. Minimum survival probability (one-step look-ahead, 100 MC runs).

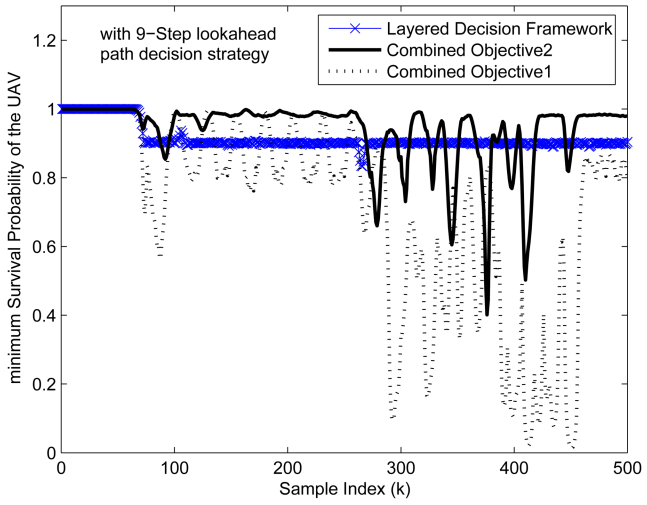


Fig. 6. Minimum survival probability (9-step look-ahead, 100 MC runs).

(safe navigation) will not be compromised by objectives with lower priorities (tracking and search), which is a desirable feature of the layered decision framework. Also notice that, most of the time, the three approaches have no significant differences in the RMS position errors.

5. MULTIPLE UAV COOPERATIVE PATH DECISION ALGORITHM FOR SURVEILLANCE MISSIONS

The multi-step look-ahead path decision algorithm proposed in Section 4 has no limitation on the number of UAVs. However, its complexity increases geometrically with respect to the number of UAVs. To keep the complexity of the path decision algorithm under control, clustering of UAVs into small decision groups will be discussed. Another feature also incorporated is to allow the UAVs to focus on different tasks in the surveillance mission.

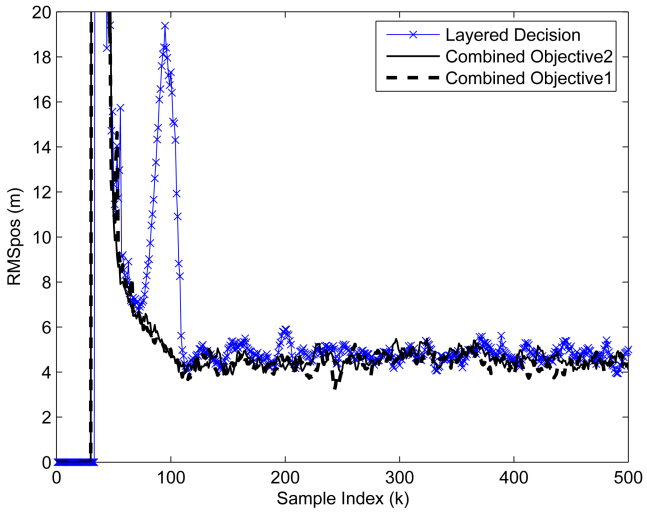


Fig. 7. RMS position error of the track (100 MC runs).

5.1. Formation of Decision Groups Based on a Nearest Neighbor Rule

As discussed in Section 4.1, using the Roll-policy, the K-step look-ahead path decision algorithm needs to evaluate $D^N + (K-1)D^{2N}$ nodes. However, as the number of UAVs increases, the complexity of the algorithm increases geometrically. To avoid this explosion in complexity, the formation of small path decision groups is proposed. For the cooperative path decision problem, it is reasonable to assume that the larger the distance between two UAVs, the less their path decisions are coupled. Thus, to control the number of UAVs involved in each path decision, it is reasonable to: i) set a maximum distance Dist_{max} beyond which the two UAVs' path decisions are decoupled; ii) construct small groups for path decisions with maximum number of N_g UAVs based on a Nearest Neighbor Rule (NNR). The NNR can be found in chapter 10 of [7], where it was used for the problem of hierarchical clustering. Fig. 8 is an example of the formation of decision groups for a group of 7 UAVs with $N_g = 3$. The procedure is as follows:

1. Find the "nearest neighbors"[7] of all the ungrouped UAVs. (As shown in Fig. 8, the arrows start from the UAVs point to their "nearest neighbors.")
2. The two UAVs that have the shortest distance to each other form a basic decision group ({UAV 3, UAV 5} in this example).
3. This decision group increases by including a UAV whose "nearest neighbor" is in the decision group. (In this example, both UAV 7 and 4's "nearest neighbors" are inside the basic decision group {UAV 3,

[7]The "nearest neighbor" of a UAV is defined as the closest UAV within a range of Dist_{max}. Notice that, in the example, the distance of UAV 6 to all the other UAVs is above Dist_{max}. Therefore, UAV 6 has no "nearest neighbor." Consequently, it forms a decision group by itself.

TABLE V
Decision Layers in the Simulation

Objective	Decision Layer (priority)	Satisfactory Level	Evaluation Criterion for the Accomplishment	Strategy for Path Decision
Safe Navigation	1	$\tau_{\text{PS}} = 0.9$	$\min\{\pi_S(s), \tau_{\text{PS}}\}$	multi-step
Classification	2	$\tau_{\text{CLS}} = 0.95$	$\min\{\max\{\mu_j\}, \tau_{\text{CLS}}\}$	one-step
Tracking	3	τ_{MSE}	$\max\{\text{MSE}(j), \tau_{\text{MSE}}\}$	one-step
Search	4	$\tau_{\text{PNNT}} = 1$	$\min\{P_{m,n}, \tau_{\text{PNNT}}\}$	one-step

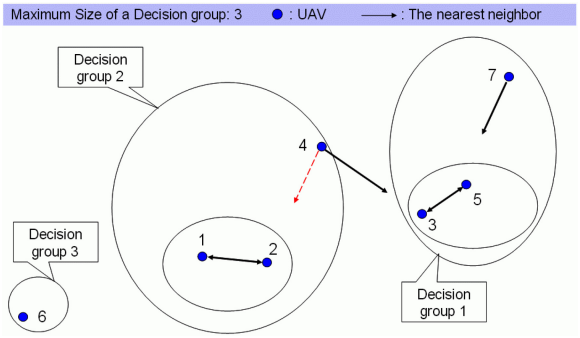


Fig. 8. Formation of decision groups.

UAV 5}; UAV 7 is first selected since it has a shorter distance to UAV 5 than the distance between UAV 4 and UAV 3.)

4. Repeat 3 on the current decision group, until it reaches the capacity limit N_g or there is no unassigned UAV that should be added based on the NNR. (In this example, {UAV 3, UAV 5, UAV 7} form the first decision group.)
5. Repeat this procedure from step 1 for the ungrouped UAVs, until all the UAVs are assigned to their respective decision groups.

For the path decisions in a decision group, the UAVs outside it are assumed to use their latest known controls throughout the path decision procedure; thus, their existence will not increase the complexity of the path decisions in this decision group. By incorporating the mechanism of decision group, the complexity of the path decision algorithm only increases linearly with the number of UAVs.

5.2. Cooperative Path Decision for UAVs with Different Objectives

In practical applications, it might be desirable to allow the UAVs to focus on different tasks. The function of assigning different objectives to the UAVs can be conveniently incorporated into the layered decision framework using satisfactory level matrices. This is illustrated by an example of multi-UAV surveillance with heterogeneous objectives, where some of the UAVs are dedicated to tracking, while the other UAVs focus more on other surveillance tasks. In this case, instead of using a satisfactory level in the decision layer of tracking, a satisfactory level matrix τ_{MSE} is used, whose element $\tau_{\text{MSE}}(s,j)$ specifies the satisfactory level of the track accuracy of target j to UAV s. Thus the desired track accuracy of a target can be different for different UAVs. In the path decision algorithm, once track j is sufficiently accurate to UAV s, that is, $\widehat{\text{MSE}}(j,k) \le \tau_{\text{MSE}}(s,j)$, a default turn rate (0 rad/s) will be used for UAV s when evaluating the sub-objective function (20) for target j. This makes the sub-objective MSE(j,k) indifferent to the control evaluations of UAV s, so that freedom in the path decision of UAV s can be saved for other "unsatisfied" objectives.

5.3. Simulation Results

The proposed multiple UAV cooperative path decision algorithm is tested in a similar surveillance scenario as in Section 4.3 but with 4 UAVs and 4 targets. The decision layers of the path decision algorithm are shown in Table V. The UAVs start out searching for targets in the surveillance region. When a target is detected, the UAV that is closest to the target will carry out the classification. Meanwhile the UAV group tracks the target cooperatively. As in Section 5.2, the satisfactory level matrix for tracking τ_{MSE} is a $N \times M$ matrix, where N is the number of the UAVs and M is the number of targets. The components in τ_{MSE} can be set dynamically during the surveillance mission. In the simulation, for the sake of simplicity, a predefined matrix

$$\tau_{\text{MSE}} = \begin{bmatrix} 0 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \end{bmatrix} \text{m}^2 \qquad (35)$$

is used. By setting $\tau_{\text{MSE}}(1,1) = 0$ m^2, UAV 1 will focus on the tracking of target 1 once it is detected, except when there is a target for it to classify. Fig. 9 shows the trajectories in single run of the simulation. Notice that at the early stage of the simulation, UAV 1 moves farther from target 1 to classify target 4, then it always stays close to target 1, while the other 3 UAVs will not try to stay as close to target 1 due to their relatively low requirements in tracking accuracy.

Fig. 10 shows the minimum survival probabilities of the UAVs. Like the results of the single UAV tracking case in Section 4.3, drops in the survival probabilities are very rare. The drops to about 0.75 occurred only twice over the 100 MC runs. Fig. 11 is the RMS position

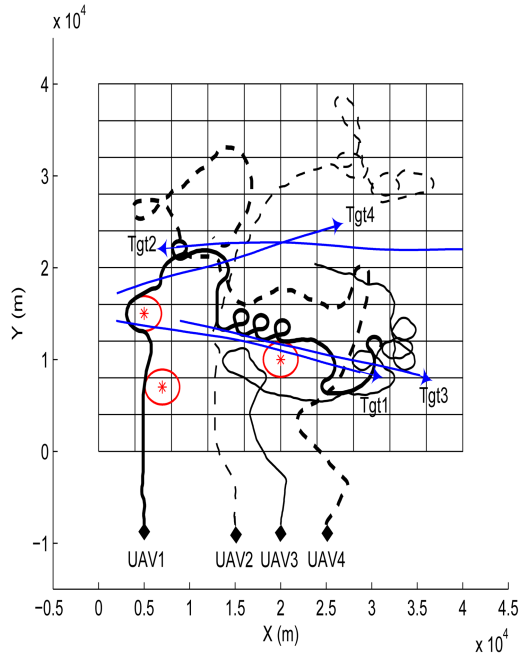


Fig. 9. UAV trajectories in one simulation (three exclusive zones are around the "asterisks").

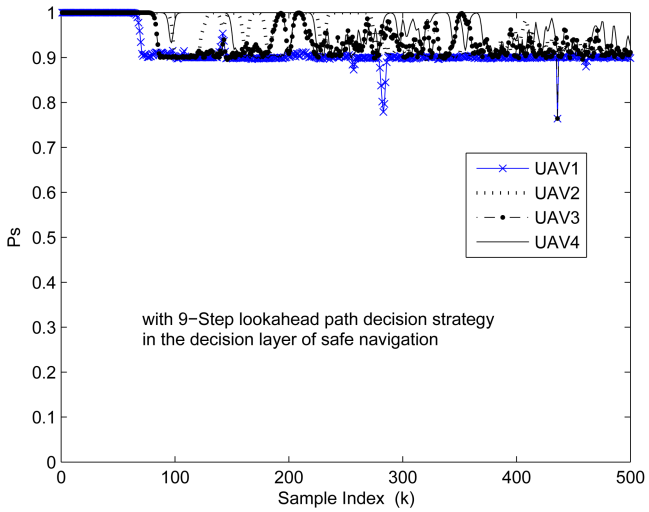


Fig. 10. Minimum survival probabilities of the UAVs (100 MC runs).

error of the targets. The initial zero RMS position errors indicate that the targets were not detected. Targets 1 and 3 were detected around time $k = 20$. Target 4 was detected around $k = 40$ and Target 2 was detected around $k = 70$ (There were some slight variations from run to run). It can be seen that target 1 is more accurately tracked due to the effort of UAV 1. The RMS position errors of the other targets satisfied the desired accuracy of the other UAVs (10 m as defined in (35)) soon after their detections, thus when the objectives with higher priorities (classification and tracking) have been accomplished, UAV 2–4's path decisions are optimized for search as long as the control decisions are "safe" for the UAVs.

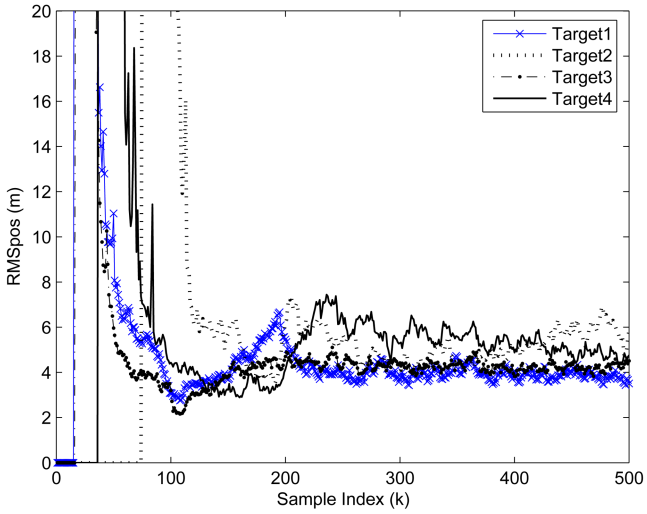


Fig. 11. RMS position error of the tracks (100 MC runs).

To summarize, the proposed path decision algorithm for UAV group is able, with moderate complexity, to i) guide a group of UAVs cooperatively for surveillance missions with multiple objectives, and ii) achieve balanced performance according to the various objective specifications.

6. CONCLUSIONS

For a surveillance mission by a group of UAVs with multiple objectives, generally the UAVs are guided by the gradient information from a certain "combination" of the objective functions. In this paper, the control of the UAV is discretized into a finite set, which amounts to sampling the objective functions over the continuous control space. Comparisons of the sample values are able to capture the gradient information in the objective functions, thus guiding the UAV group for the surveillance task.

More importantly, the discretization of control variables provides extra freedom in dealing with multiple objectives in the surveillance mission. Accordingly, a layered decision framework is proposed. Instead of using a single global objective function that is a weighted sum of all the objectives, different objectives are treated in separate decision layers in the order of their priorities. Compared to the weighted sum approach, the layered decision framework has the following advantages: i) multiple objectives in the surveillance mission are isolated; thus objectives with higher priorities are free from possible compromises from the less important ones; ii) for each objective, the specification of "satisfactory" levels allow the algorithm to be more sensitive to the entities (targets in tracking, sectors in search) that demand more attention; iii) the layered decision framework allows different path decision strategies to be used for the objectives, which makes the algorithm efficient.

The discretized controls also allow the extension of the time horizon of the path decisions, which is particu-

larly important for the safe navigation of the UAVs. Accordingly, a multi-step look-ahead path decision strategy based on the Rollout policy is proposed. When used in the decision layer of safe navigation, this approach produces significantly improved results.

To keep the algorithm computationally feasible for large groups of UAVs, clustering of UAVs into small decision groups is discussed. Further flexibility of assigning different tasks to the UAVs is also incorporated into the path decision algorithm. Simulation results show that the proposed multi-step look-ahead path decision algorithm can effectively guide the UAV group for multi-objective surveillance missions and its performance is superior to the one-step look-ahead combined-objective approach.

REFERENCES

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

[2] Y. Bar-Shalom, T. Kirubarajan and C. Gokberk
Tracking with classification-aided multiframe data association.
IEEE Transactions on Aerospace and Electronic Systems, **41**, 3 (July 2005), 868–878.

[3] D. P. Bertsekas, J. N. Tsitsiklis and C. Wu
Rollout algorithms for combinatorial optimization.
Journal of Heuristics, **3**, 3 (Nov. 1997), 245–262.

[4] D. P. Bertsekas and D. A. Castanon
Rollout algorithms for stochastic scheduling problems.
Journal of Heuristics, **5**, 1 (Apr. 1999), 89–108.

[5] R. W. Beard, T. W. McLain, M. A. Goodrich and E. P. Anderson
Coordinated Target assignment and intercept for unmanned air vehicles.
IEEE Transactions on Robotics and Automation, **18**, 6 (Dec. 2002), 911–922.

[6] P. R. Chandler
UAV coperative control.
In *Proceedings of American Control Conference*, June 2001, 50–55.

[7] R. O. Duda, P. E. Hart and D. G. Stork
Pattern Classification (2nd ed.).
New York: Wiley, 2001.

[8] T. Furukawa, F. Bourgault, H. F. Durrant-Whyte and G. Dissanayake
Dynamic allocation and control of coordinated UAVs to engage multiple targets in a time-optimal manner.
In *Proceedings of IEEE International Conference on Robotics and Automation*, Apr. 2004, 2353–2358.

[9] T. Furukawa, F. Bourgault, B. Lavis and H. F. Durrant-Whyte
Recursive Bayesian search-and-tracking using coordinated UAVs for lost targets.
In *Proceedings of Conference on Robotics and Automation*, May 2006, 2521–2526.

[10] G. M. Hoffmann, S. L. Waslander and C. J. Tomlin
Distributed cooperative search using information-theoretic costs for particle filters with quadrotor applications.
In *Proceedings of AIAA Guidance, Navigation and Control Conference*, Keystone, CO, Aug. 2006.

[11] V. P. Jilkov, X. R. Li and D. DelBalzo
Best combination of multiple objectives for UAV search & track path optimization.
In *Proceedings of the 10th International Confernce on Information Fusion*, July 2007.

[12] D. V. Kalbaugh
Optimal search among false contacts.
SIAM Journal of Applied Math, **52**, 6 (Dec. 1992), 1722–1750.

[13] S. M. Li, et al.
Automonous hierachical control of multiple unmanned combat vehicles.
In *Proceedings of American Control Conference*, May 2002, 274–279.

[14] T. W. McLain, P. R. Chandler and M. Pachter
A decomposition strategy for optimal coordination of unmanned air vehicles.
In *Proceedings of American Control Conference*, June 2000, 369–373.

[15] T. W. McLain, P. R. Chandler, S. Rasmussen and M. Pachter
Cooperative control of UAV rendezvous.
In *Proceedings of American Control Conference*, June 2001, 2309–2314.

[16] M. M. Polycarpou, Y. Yang and K. M. Passino
A cooperative search framework for distributed agents.
In *Proceedings of the 2001 IEEE International Symposium on Intelligent Control*, Sept. 2001.

[17] D. A. Schoenwald
UAVs: In space, air, water, and on the ground.
IEEE Control Systems Magazine, **20**, 6 (Dec. 2000), 15–18.

[18] A. Sinha, T. Kirubarajan and Y. Bar-Shalom
A distributed approach to autonomous surveillance by multiple cooperative UAVs.
In *Proceedings of SPIE Signal and Data Processing of Small Targets*, Oct. 2005, #5913-64.

[19] A. Sinha, T. Kirubarajan and Y. Bar-Shalom
Autonomous search, tracking and classification by multiple cooperative UAVs.
In *Proceedings of SPIE Conference on Signal Processing, Sensor Fusion, and Target Recognition*, Apr. 2006, #6235-09.

[20] L. D. Stone and J. A. Stanshine
Optimal search using uninterrupted contact investigation.
SIAM Journal on Applied Mathematics, **20**, 2 (Mar. 1971), 241–163.

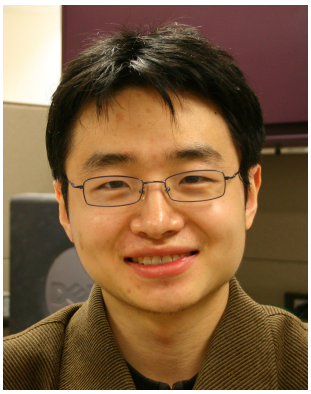
[21] F. Tu and K. R. Pattipati
Rollout strategy for sequential fault diagnosis.
IEEE Transactions of System, Man and Cybernetics, Part A, 33, 1 (Jan. 2003), 86–99.

[22] J. Yan, L. Yan, A. Minai and M. Polycarpou
Balancing search and target response in cooperative unmanned aerial vehichle (UAV) teams.
IEEE Transactions on Systems, Man and Cybernetics, Part B, **36**, 3 (June 2006), 571–587.

[23] Y. Yang, A. Minai and M. Polycarpou
Decentralized cooperative search by networked UAVs in an uncertain environment.
In *Proceedings of American Control Conference*, June 2004, 5558–5563.

[24] S. Yeom, T. Kirubarajan and Y. Bar-Shalom
Track segment association, fine-step IMM and initialization with Doppler for improved track performance.
IEEE Transactions on Aerospace and Electronic Systems, **40**, 2 (Jan. 2004), 293–309.

[25] U. Zengin and A. Dogan
Real-time target tracking for autonomous UAVs in adverarial environments: A gradient search algorithm.
IEEE Transactions on Robotics, **23**, 2 (Apr. 2007), 294–307.



Xin Tian was born in 1980. He received the B.E. degree in 2002 and M.E. degree in 2005, both from the Department of Information and Communication Engineering, Xi'an Jiaotong University, China. He is now a Ph.D. candidate in the Department of Electrical and Computer Engineering, University of Connecticut.

His research interests include statistical signal processing, tracking algorithms, and information fusion.

Yaakov Bar-Shalom (S'63—M'66—SM'80—F'84) was born on May 11, 1941. He received the B.S. and M.S. degrees from the Technion, Israel Institute of Technology, in 1963 and 1967 and the Ph.D. degree from Princeton University in 1970, all in electrical engineering.

From 1970 to 1976 he was with Systems Control, Inc., Palo Alto, CA. Currently he is Board of Trustees Distinguished Professor in the Dept. of Electrical and Computer Engineering and Marianne E. Klewin Professor in Engineering at the University of Connecticut. He is also Director of the ESP (Estimation and Signal Processing) Lab.

His current research interests are in estimation theory and target tracking and has published over 370 papers and book chapters in these areas and in stochastic adaptive control. He coauthored the monograph *Tracking and Data Association* (Academic Press, 1988), the graduate texts *Estimation and Tracking: Principles, Techniques and Software* (Artech House, 1993), *Estimation with Applications to Tracking and Navigation: Algorithms and Software for Information Extraction* (Wiley, 2001), the advanced graduate text *Multitarget-Multisensor Tracking: Principles and Techniques* (YBS Publishing, 1995), and edited the books *Multitarget-Multisensor Tracking: Applications and Advances* (Artech House, Vol. I, 1990; Vol. II, 1992; Vol. III, 2000).

He has been elected Fellow of IEEE for "contributions to the theory of stochastic systems and of multitarget tracking." He has been consulting to numerous companies and government agencies, and originated the series of Multitarget-Multisensor Tracking short courses offered via UCLA Extension, at Government Laboratories, private companies and overseas.

During 1976 and 1977 he served as Associate Editor of the IEEE Transactions on Automatic Control and from 1978 to 1981 as Associate Editor of Automatica. He was Program Chairman of the 1982 American Control Conference, General Chairman of the 1985 ACC, and Co-Chairman of the 1989 IEEE International Conference on Control and Applications. During 1983–87 he served as Chairman of the Conference Activities Board of the IEEE Control Systems Society and during 1987–89 was a member of the Board of Governors of the IEEE CSS. He was a member of the Board of Directors of the International Society of Information Fusion (1999–2004) and served as General Chairman of FUSION 2000, President of ISIF in 2000 and 2002 and Vice President for Publications in 2004–08.

In 1987 he received the IEEE CSS Distinguished Member Award. Since 1995 he is a Distinguished Lecturer of the IEEE AESS and has given numerous keynote addresses at major national and international conferences. He is corecipient of the M. Barry Carlton Award for the best paper in the IEEE Transactions on Aerospace and Electronic Systems in 1995 and 2000 and the 1998 University of Connecticut AAUP Excellence Award for Research. 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. He is the recipient of the 2008 IEEE Dennis J. Picard Medal for Radar Technologies and Applications.

Krishna R. Pattipati (S'77—M'80—SM'91—F'95) received the B.Tech. degree in electrical engineering with highest honors from the Indian Institute of Technology, Kharagpur, in 1975, and the M.S. and Ph.D. degrees in systems engineering from the University of Connecticut in 1977 and 1980, respectively.

From 1980–86 he was employed by ALPHATECH, Inc., Burlington, MA. Since 1986, he has been with the University of Connecticut, where he is a Professor of Electrical and Computer Engineering. His current research interests are in the areas of adaptive organizations for dynamic and uncertain environments, multi-user detection in wireless communications, signal processing and diagnosis techniques for complex system monitoring, and multi-object tracking. Dr. Pattipati has published over 370 articles, primarily in the application of systems theory and optimization (continuous and discrete) techniques to large-scale systems. He has served as a consultant to Alphatech, Inc., Aptima, Inc., and IBM Research and Development. He is a cofounder of Qualtech Systems, Inc., a small business specializing in intelligent diagnostic software tools.

Dr. Pattipati was selected by the IEEE Systems, Man, and Cybernetics Society as the Outstanding Young Engineer of 1984, and received the Centennial Key to the Future award. He was elected a Fellow of the IEEE in 1995 for his contributions to *discrete-optimization algorithms for large-scale systems and team decision making*. Dr. Pattipati has served as the Editor-in-Chief of the *IEEE Transactions on SMC: Part B—Cybernetics* during 1998–2001, Vice-President for Technical Activities of the IEEE SMC Society (1998–1999), and as Vice-President for Conferences and Meetings of the IEEE SMC Society (2000–2001). He was corecipient of the Andrew P. Sage award for the Best SMC Transactions Paper for 1999, Barry Carlton award for the Best AES Transactions Paper for 2000, the 2002 and 2008 NASA Space Act Awards for "A Comprehensive Toolset for Model-based Health Monitoring and Diagnosis," the 2003 AAUP Research Excellence Award and the 2005 School of Engineering Teaching Excellence Award at the University of Connecticut. He also won the best technical paper awards at the 1985, 1990, 1994, 2002, 2004 and 2005 IEEE AUTOTEST Conferences, and at the 1997 and 2004 Command and Control Conferences.

Fuse-before-Track in Large Sensor Networks

STEFANO CORALUPPI
MARCO GUERRIERO
PETER WILLETT
CRAIG CARTHEL

Recent years have seen a trend towards unmanned multi-sensor surveillance networks with large numbers of cheap and limited-performance sensors. While these networks hold significant potential for surveillance, it is of interest to address fundamental limitations in large-scale implementations. We first introduce a simple analytical tracker performance model. Analysis of this model suggests that scan-based tracking performance improves with increasing numbers of sensors, but only to a certain point beyond which degradation is observed. Correspondingly, we address model-based optimization of the local sensor detection threshold and the number of sensors. Next, we propose a two-stage tracking approach (*fuse-before-track*) as a possible approach to overcoming the difficulties in large-sensor surveillance, and we illustrate promising performance results with simulated surveillance data.

Manuscript received December 18, 2008; first revision July 13, 2009, second revision October 1, 2009; released for publication October 20, 2009.

Refereeing of this contribution was handled by Ben Slocumb.

Authors' addresses: Stefano Coraluppi and Craig Carthel, NATO Undersea Research Centre (NURC), Viale S. Bartolomeo 400, 19126 La Spezia, Italy, E-mail: coraluppi@nurc.nato.int, carthel@nurc.nato.int; Marco Guerriero and Peter Willett, University of Connecticut–ECE Department, Storrs, CT 06269, E-mail: marco.guerriero@engr.uconn.edu, willett@engr.uconn.edu

1. INTRODUCTION

While multi-sensor systems hold great potential for surveillance performance, the technical challenges are significant, and include the need for effective calibration as well as a statistically-valid characterization of environmental uncertainties and contact measurement errors. Additionally, automatic tracking and fusion processing with active sensors must contend with high false contact rates and target fading effects. Issues in multi-sensor surveillance and numerous design approaches are discussed in [4, 9, 24].

In [12–13], we present model-based, simulation-based, and sea-trial tracking performance results with a track-oriented, modular multi-hypothesis tracking scheme. Of particular interest is the tradeoff between centralized and multi-stage processing: we have found that, when faced with significant target fading effects and for modest false contact rates, distributed processing can outperform centralized processing. This somewhat surprising result is based on the fundamental sub-optimality of all tracking algorithms that must contend with measurement origin uncertainty. This explains the seeming contradiction with results in the nonlinear filtering and distributed detection literature, in particular the well-known optimality of centralized processing schemes [9, 25].

Ultimately, for sufficiently low-SNR target scenarios, effective real-time automatic tracking is extremely challenging regardless of the choice of data processing architecture. One approach is to relax the real-time requirement, and to leverage powerful batch processing techniques [3]. However, such schemes are not easily amenable to real-time surveillance requirements, and generally assume non-maneuvering targets. An alternative approach in challenging scenarios is to consider enlarging the surveillance network, possibly through bootstrapping approaches that include sub-band processing techniques [19], whereby a sensor is effectively "replaced" with a number of slightly-degraded sensors.

The latter approach (enlarging the surveillance network) implicitly assumes that an increased number of like-performing, calibrated, and registered sensors are always to be preferred, i.e., more sensors are always better than fewer. Is this true in general or are there performance limits as the number of sensors becomes large? This is the issue that we address in this paper.

We start by introducing in Section 2 a simple analytical model for tracker performance. We study tracker performance as a function of local detection threshold, number of sensors, and track management criteria. The model supports the conclusion that there are performance bounds on achievable performance in large sensor networks.

Can we do better if we consider a more complex, multi-stage processing architecture? In Section 3, we describe the *fuse-before-track* (FbT) architecture and

provide motivation for its use in large sensor networks. The key insight that motivates the FbT architecture is that it couples the advantages of batch processing in the fusion step, followed by the advantages of scan-based processing (real-time processing with a maneuvering target model) in the tracking step.

It is important to note that we do not argue that FbT will outperform a *sufficiently complex* centralized processing scheme. Rather, our claim is that multistage processing with a relatively *simple* tracking module can achieve good performance results. Further, as we will see in the Monte Carlo study, this performance is achieved with a significantly lower computational effort than in centralized (single-stage) processing.

The first stage in the FbT architecture is a static fusion (or contact fusion) stage [4, 9]. In Section 4, we study perhaps the simplest approach to contact fusion that was suggested in [19] and is known as contact sifting. We analytically characterize the performance of this stage of processing, and experimentally validate the performance model. We also briefly discuss an alternative approach to contact fusion.

Section 5 presents a Monte Carlo study of multisensor tracking performance for a representative multitarget surveillance scenario. The results suggest that the FbT architecture has merit and deserves further attention by the target tracking community. Section 6 provides conclusions and directions for future work. This paper is an extended and improved version of [14–16].

2. TRACKER PERFORMANCE MODELING

Tracker performance modelling is addressed at length in [4, 8]. Extensions that address target fading effects and distributed tracking architectures are given in [10, 12–13]. For our purposes here, we introduce a simple tracker performance model that identifies a compact relationship between scan rate and performance. Scan rate is directly proportional to the number of sensors and, thus, the model will support the subsequent analysis on performance as a function of the number of sensors.

2.1. Tracker Model

Modeling parameters:

- *Target*: kinematic "nearly constant position" motion model in two dimensions with maneuvering index q $\mathrm{m}^2\mathrm{s}^{-1}$; fixed target SNR d;
- *Sensor*: each with scan every Δt sec; positional measurements with covariance R; surveillance region of size A m^2, detection cells of size C m^2, detection threshold D;
- *Tracker*: declare track on N consecutive associated detections, terminate track on K consecutive coasts (missed updates), association probability gate P_{G} and gating parameter γ (with two-dimensional measurements, $\gamma = 9.2$ corresponds to $P_{\mathrm{G}} = 0.99$; see details in [8]).

The following derived quantities are of interest.

- *Detection probability* P_{D}, where we assume Rayleigh-distributed amplitude statistics [9]:

$$P_{\mathrm{D}} = \exp\left(-\frac{D}{1+d}\right). \tag{1}$$

- *False alarm density per square meter*, where we again assume Rayleigh-distributed amplitude statistics:

$$\lambda = \frac{\exp(-D)}{C}. \tag{2}$$

- *False alarm rate per hour*:

$$\lambda_{\mathrm{FAR}} = \frac{3600 \cdot \lambda A}{\Delta t}. \tag{3}$$

- *Probability of correct association*: we assume that the statistical nearest neighbor is used for track update, that the target-originated contact is one standard deviation in each dimension from the true target location, and that the track has steady-state filter covariance based on consecutive detection events. Thus, letting S be the *innovation covariance* [4, p. 49] and letting V be the *validation region volume* [4, p. 96], we have:

$$P_{\mathrm{CA}} = \exp(-\lambda V) \tag{4}$$

$$V = \pi |S|^{1/2} \tag{5}$$

$$S = P(-) + R \tag{6}$$

$$P(-) = P(+) + Q \tag{7}$$

$$P(+) = P(-) - P(-)(P(-) + R)^{-1} P(-) \tag{8}$$

$$Q = \begin{bmatrix} q\Delta t & 0 \\ 0 & q\Delta t \end{bmatrix}. \tag{9}$$

Note that $P(-)$ denotes the filter prediction covariance, while $P(+)$ denotes the filter update covariance. Further, note that $P(-)$ is the solution to the (steady-state) *algebraic Riccati equation* (ARE) [17].

- *Probability of track update and miss* (i.e., track *coast*), where a track update requires that the current scan include target detection, successful gating to the target track, and correct association:

$$P_{\mathrm{U}} = P_{\mathrm{D}} P_{\mathrm{G}} P_{\mathrm{CA}} \tag{10}$$

$$P_{\mathrm{M}} = 1 - P_{\mathrm{U}}. \tag{11}$$

- *Average track confirmation time* (note that the expected value of the geometric distribution with parameter p is given by $1/p$), where track confirmation requires N consecutive, associated target detections:

$$\tau_C = \frac{1}{P_{\mathrm{U}}^{N-1}} \left(N - 1 + \frac{1}{P_{\mathrm{D}}}\right) \Delta t. \tag{12}$$

- *Average track hold time*, where track termination is achieved after K consecutive scans without an associated target detection:

$$\tau_H = \frac{1}{P_{\mathrm{M}}^{K-1}} \left(K - 1 + \frac{1}{P_{\mathrm{M}}} \right) \Delta t. \tag{13}$$

Note that equations (12–13) both rely on nested geometric probability distributions—that is to say, the sojourn time prior to tentative track initiation has a geometric distribution, as does the total track initiation time. It is easy to show (by linearity of the expectation operator) that equations (12–13) hold.

- *Track detection probability*, given by the fraction of time during which a target has a corresponding confirmed track:

$$P_{\mathrm{D}}^{T} = \frac{\tau_H}{\tau_C + \tau_H}. \tag{14}$$

- *Probability of false update*, given by one minus the probability that no false contacts exist in the association gate and again assuming steady-state filter covariance:

$$P_{\mathrm{FU}} = 1 - \exp(-\lambda V_\gamma) \tag{15}$$

$$V_\gamma = \gamma \pi |S|^{1/2}. \tag{16}$$

- *Probability of false track*, given by the probability that a false contact leads to a sequence of associated false contacts:

$$P_{\mathrm{FT}} = P_{\mathrm{FU}}^{N-1}. \tag{17}$$

- *False track rate per hour:*

$$\lambda_{\mathrm{FTR}} = \frac{3600 \cdot P_{\mathrm{FT}} \lambda A}{\Delta t}. \tag{18}$$

For simplicity and to minimize the number of modeling parameters, we have assumed track confirmation on N consecutive detections rather than a more general M-of-N track initiation criterion. The equations above generalize easily to the M-of-N criterion, using the binomial distribution with parameters $N-1$ and P_{U} for in the target-present case, and parameters $N-1$ and P_{FU} in the target-absent case.

We have invoked several modeling simplifications, including the impact of false updates on the true track formation and maintenance. This effect is estimated empirically in [8, pp. 207–208], as the impact is difficult to capture analytically. Here, we assume for simplicity that the impact of a false update in terms of track degradation is comparable to that of a track miss.

An illustration of the Markov chain model that corresponds to the modeling above is given in Fig. 1.

The tracker performance model introduced here shares some commonalities with the *system operating characteristics* (SOC) curve introduced in [5]. One of the differences is that the metrics of interest differ. In [5], a single, fixed time window in considered, and track detection and false track probabilities are computed. Rather, here the track detection probability is a measure

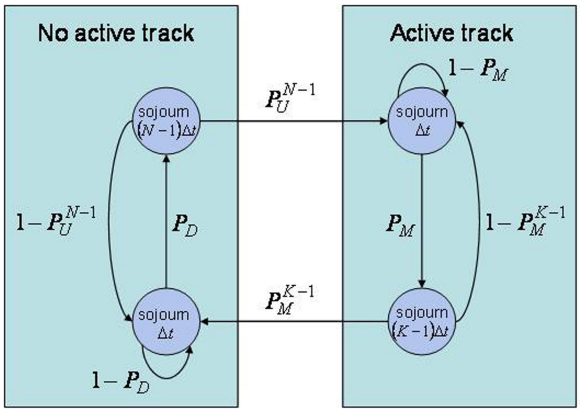


Fig. 1. Markov chain model for tracker logical state, in the target-present case. (A similar Markov chain applies to the target-absent case.)

TABLE I
Model Simulation Parameters

Parameter	Setting
Maneuverability index q	100 $\mathrm{m^2 s^{-1}}$
Target SNR d	10 dB
Scan interval $\Delta t'$	60 sec
Measurement covariance matrix R	$\begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}$ $\mathrm{m^2}$
Surveillance region A	10^8 $\mathrm{m^2}$
Detection cell C	100 $\mathrm{m^2}$
Detection threshold D	5.0–9.0 dB
Track initiation N	3
Track termination K	3
Association gate γ	9.2
Gate probability P_{G}	0.99

of *track hold*, which answers the following question: "For a given target, what is the probability that there is a corresponding track at any given time?" The false track rate identifies the number of false objects generated by the tracker per unit time.

2.2. Tracker Performance Analysis

We are interested to examine input and output performance curves (λ_{FAR} vs. P_{D}, and λ_{FTR} vs. P_{D}^{T}, respectively) as a function of the detection threshold D, and as a function of the number of sensors. The latter can be addressed by setting the scan rate to $\Delta t = \Delta t'/Z$, where Z is the number of equally-performing sensors and $\Delta t'$ is the single-sensor rate. Parameters are set as indicated in Table I, and performance curves are in Fig. 2. (Note that by *object* we mean either contact or track.)

Key conclusions are as follows:

- Tracking provides a roughly two-order-of-magnitude reduction in false objects, with comparable object detection performance.
- With a low constraint on false object rate, it is best to use few sensors.

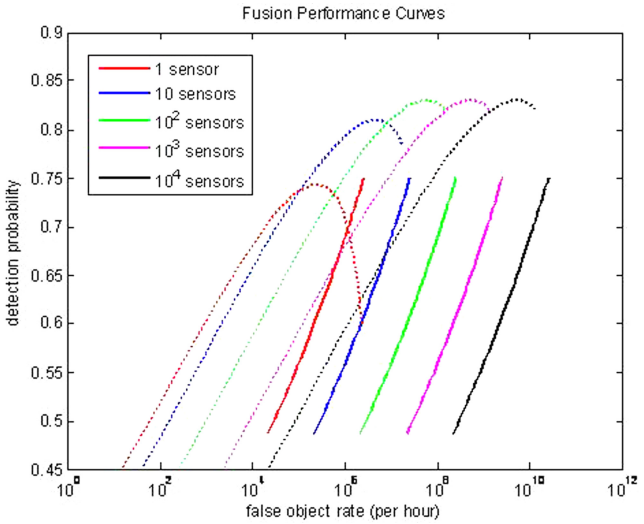


Fig. 2. Performance curves for several network-size assumptions. Solid lines characterize sensor performance, while dotted lines characterize tracker performance as characterized by the analytical model in Section 2.1.

- With a larger constraint on false object rate, it is best to use more sensors.
- For any given number of sensors, unlike the behavior of the (monotonic) input ROC curve, a maximum in track-level detection is achieved for a non-zero SNR detection threshold.

2.3. Optimal Detection Threshold and Number of Sensors

An approach to improve centralized tracking performance is to optimize the local sensor detection threshold (D) as well as the number of sensors to be processed (Z), as a function of a constraint on the false track rate. We anticipate that this will lead to a performance curve that is the envelope of the family of curves shown in Fig. 2.

For a given λ_{FTR}, we wish to optimize the local sensor detection threshold (D) as well as the sensor scan rate Δt (from which we infer the number of sensors). This optimization problem can be recast as the following *constrained* maximization problem:

$$\begin{aligned} &\max_{\Delta t, DT} \quad P_{\text{D}}^{T}(\Delta t, D) \\ &\text{s.t} \quad \lambda_{\text{FTR}}(\Delta t, D) = \alpha. \end{aligned} \tag{19}$$

Note that the dependence of P_{D}^{T} on Δt is complex, since P_{U} and P_{M} both depend on Δt. This optimization problem does not lend itself to an analytical solution. Using the same parameter settings as in Table I, the solution to equation (19) leads to the envelope of the family of curves in Fig. 2, as illustrated in Fig. 3. (Optimization is performed using the function `fmincon` in MATLAB.)

It is instructive to examine the optimal scan interval Δt^{opt} and the optimal detection threshold D^{opt} as a function of λ_{FTR}; these are illustrated in Figs. 4–5. It is interesting to note that, with increasing λ_{FTR}, the optimal P_{D}^{T} is achieved with a reduction in both Δt and D: we both increase the number of sensors and lower the detection threshold.

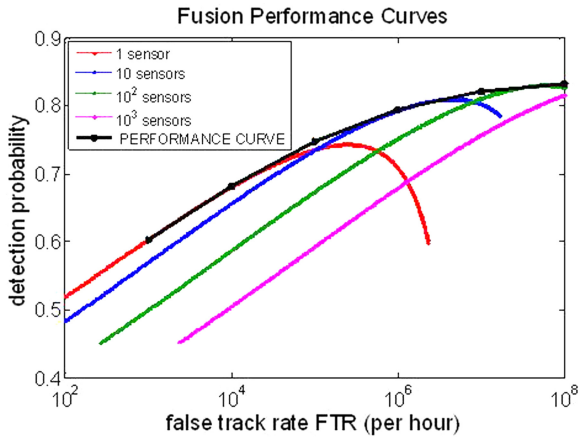


Fig. 3. Performance curve obtained solving the optimization problem (22).

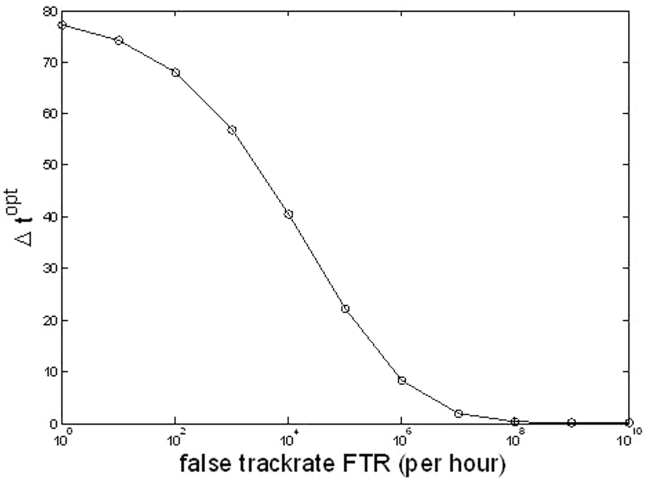


Fig. 4. Optimal scan interval (inversely proportional to number of sensors) as a function of λ_{FTR}.

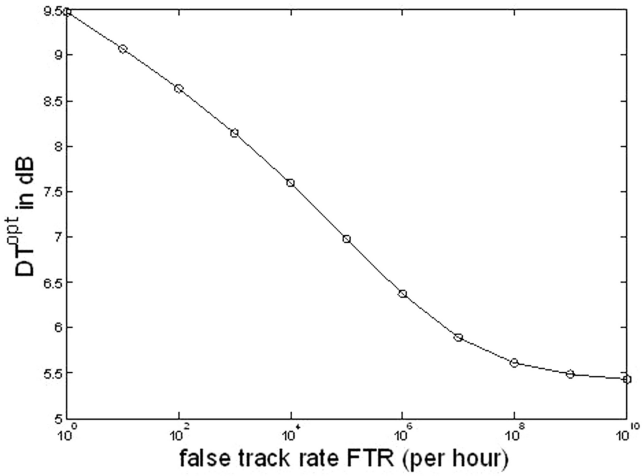


Fig. 5. Optimal detection threshold as a function of λ_{FTR}.

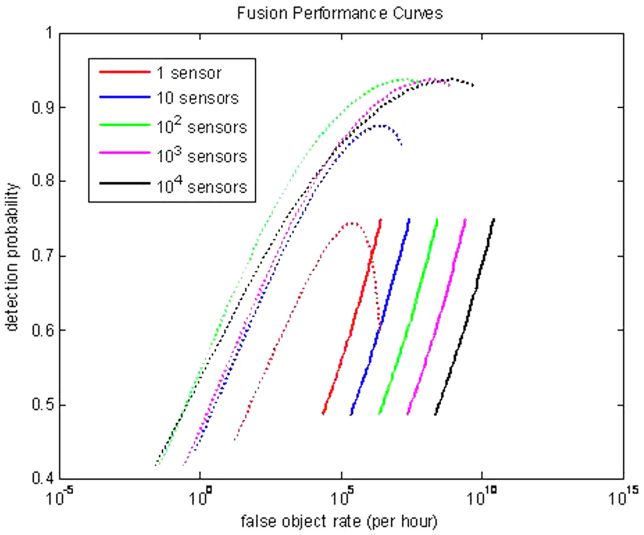


Fig. 6. Performance curves for several network size assumptions with adaptive track-management parameters. Solid lines characterize sensor performance, while dotted lines characterize tracker performance as characterized by the analytical model in Section 2.1.

Note that our analysis has been limited to the assumptions of equally-performing sensors with identical detection thresholds. Relaxing either of these assumptions introduces the need for a more complex tracker performance model.

2.4. Tuning the Track Management Parameters

A possible objection to the results in Sections 2.2–2.3 is that we use the same track initiation and termination criteria throughout. More generally, one might wish to adapt the parameters N and K to the data rate and to the detection threshold. A fully adaptive selection of these parameters is quite complex. We may, however, seek to vary N and K as a function of the data rate only. In particular, neglecting the dependence of P_{FU} on the data rate, we can achieve a comparable false track rate by setting $N(Z)$ in the case of $Z > 1$ sensors as follows, where $N(1) = N_0$. We set the false track rate to be independent of the number of sensors:

$$\lambda_{\mathrm{FTR}} = \frac{3600 \cdot P_{\mathrm{FU}}^{N(Z)-1} \lambda A}{\Delta t / Z} = \frac{3600 \cdot P_{\mathrm{FU}}^{N_0-1} \lambda A}{\Delta t}.$$

Neglecting the dependence of P_{FU} on the scan rate leads to the following simple relationship between the track confirmation window length and the number of sensors:

$$N(Z) = N_0 + \frac{\log Z}{\log(1/P_{\mathrm{FU}})}. \tag{20}$$

We scale the parameter K in a comparable manner. As a result of adaptively-selected track-management parameters, the curves in Fig. 2 are modified to those shown in Fig. 6. The corresponding values of N (and K) are given in Table II.

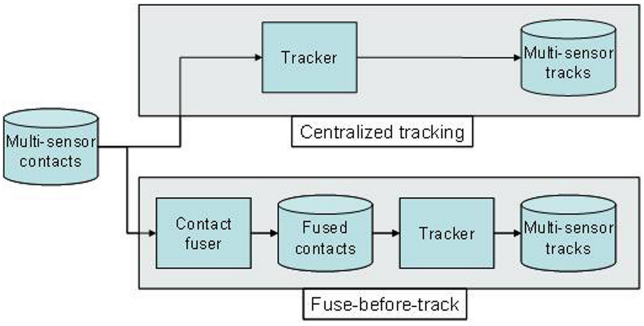


Fig. 7. Candidate fusion and tracking architectures.

TABLE II
Adaptive Track Management for the Results in Fig. 2

Number of Sensors (Z)	Track Initiation (N)
1	3
10	4
100	5
1,000	5
10,000	6

We find that adaptive track management does have an impact on centralized tracking performance. In particular, note that the 100-sensor performance curve is better than the 10-sensor curve, which in turn outperforms the one-sensor curve. Nonetheless, the qualitative findings noted above still hold: significant reduction in false objects with automatic tracking, non-monotonicity in tracker performance curves, and saturation in performance benefits for a large-enough number of sensors. Correspondingly, the optimal choice for the number of sensors still depends on the false track requirement.

3. THE FUSE-BEFORE-TRACK ARCHITECTURE

Is it possible to exceed centralized tracking performance? At a conceptual level, the answer would seem to be no: every algorithmic step that is possible in multi-stage or distributed processing can be achieved in a centralized processing configuration.

A practical question of interest is whether a particular (sub-optimal) tracking module is better employed in a single-stage processing architecture or in a two-stage architecture. For the latter approach, we are interested to explore a *fuse-before-track* (FbT) processing scheme, whereby contact fusion across sensors precedes tracking over time. The two approaches are illustrated in Fig. 7.

It is important to note that the FbT architecture supports real-time processing just as the centralized architecture does: as fused contacts are produced, they provide input to the second-stage scan-based tracker.

For the purposes of this discussion, the particular choice of tracking module (here, a track-oriented multi-

hypothesis tracker [12–13]) is not critical; however, what *is* critical is that our tracker is representative of a real-time, scan-based algorithm that necessarily discards data-association hypotheses (in the case of non-Bayesian tracking) or that combines hypotheses (in the case of Bayesian tracking). Indeed, while amenable to real-time surveillance requirements, scan-based tracking approaches lack the performance potential of batch processing schemes such as [3].

For simplicity, we assume that the (active) sensors are (nearly) synchronized; that is, we assume that the scans of contact-level data are acquired for the same sequence of times, for all sensors. An alternative time-series representation of the two architectures in Fig. 7 is illustrated in Fig. 8.

The motivation for investigating the FbT architecture is as follows. The static fusion stage is not hindered by the requirement for scan-based processing, since all the sensors scan the surveillance region simultaneously. Thus, for large sensor networks, the two-stage architecture leverages the strength of batch processing in the fusion stage, while maintaining the real-time surveillance requirement with scan-based tracking.

Let us return for a moment to the argument that the same processing results obtained with FbT are in principle achievable with single-stage or centralized processing. After all, as we saw in the model-based results documented in Section 2, there is an advantage to appropriately scaling the track-management parameters with the sensor data rate. However, key data association parameters *do not* scale well with increasing data rate. For instance, for computational reasons, the *n-scan* track hypothesis depth parameter that is common in multi-hypothesis tracking (MHT) approaches cannot be scaled with the data rate. A similar consideration holds for multiple-model filters. Thus, as we will see in Section 5, an adjustment to track-management parameters for centralized processing is insufficient to match the promising performance results exhibited by the FbT architecture.

4. STATIC FUSION AND THE CONTACT SIFTING APPROACH

We focus now on the static fusion problem, which represents the first stage in the *fuse-before-track* (FbT) architecture. We will introduce the concept of *probability of localization* and use this concept in studying the (simple) contact sifting approach and its performance characteristics; subsequently, we will briefly discuss an alternative approach to static fusion.

4.1. Probability of Localization

Assume we have a surveillance region A composed of N_{cell} detection cells of equal size. Assume further that detection statistics in both the target-present and target-absent cases follow Rayleigh statistics, and that the expected target SNRs are given by d. Given a detection threshold D, it can be shown that targets have the following detection probability (same as equation (1)):

$$P_{\text{D}} = \exp\left(-\frac{D}{1+d}\right). \tag{21}$$

Further, the probability of false contact in any cell is given by:

$$P_{\text{FA}} = \exp(-D). \tag{22}$$

Accordingly, the number of false contacts is Poisson with parameter $\lambda_A = P_{\text{FA}} N_{\text{cell}}$. Further, the false contacts are uniformly distributed in the surveillance region.

The well-known classical ROC curve in this case is given by varying D over a range of values, with the following relationship between P_{D} and P_{FA}:

$$P_{\text{D}} = (P_{\text{FA}})^{1/(1+d)}. \tag{23}$$

A slightly modified ROC curve provides P_{D} as a function of the expected number of false contacts (same as λ_A):

$$N_{\text{FA}} = P_{\text{FA}} N_{\text{cell}}. \tag{24}$$

From an operational perspective, we are interested in detecting and localizing targets: target detections that are distant from the true target location are indistinguishable from false contacts; similarly, false contacts that fortuitously are close to the true target location are indistinguishable from target detections. Thus, we introduce the notion of *probability of localization*: the probability that a contact exists close enough to the target. This notion couples detection and localization metrics into a single quantity of interest; related work on the coupling between detection and localization objectives is found in [21].

Assume that our sensor provides two-dimensional Cartesian positional measurements of target location, with uncorrelated and identically distributed Gaussian errors in x and y, and variance σ_x^2. Let ε be the maximum distance for acceptable target localization. This defines a circular validation region around the target of area $B = \pi\varepsilon^2$. False contacts are uniformly distributed in this region with parameter $\lambda_B = \lambda_A B/A$. Define $\xi = \varepsilon/\sigma_x$.

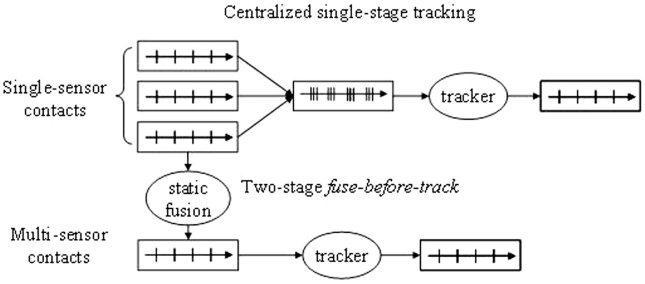


Fig. 8. FbT includes static fusion and scan-based tracking.

The probability of localization P_L is the following:

$$P_L = P_D\left(1-\exp\left(-\frac{\xi^2}{2}\right)\right)+(1-\exp(-\lambda_B)) - P_D\left(1-\exp\left(-\frac{\xi^2}{2}\right)\right)(1-\exp(-\lambda_B)). \tag{25}$$

The first term is the probability that a target is detected, with the detection lying in the validation region. The second term is the probability that at least one false contact is in the validation region; it can be derived by denoting $C = (1 - B/A)$ and observing that the probability that no false contact is in the validation region is given by:

$$\sum_{k=0}^{\infty}\frac{\lambda_A^k}{k!}\exp(-\lambda_A)C^k = \exp(-\lambda_A(1-C)) \times\sum_{k=0}^{\infty}\frac{(\lambda_A C)^k}{k!}\exp(-\lambda_A C) = \exp(-\lambda_A B/A) = \exp(-\lambda_B).$$

The third term in equation (25) is the probability that both target and non-target contacts are present in the validation region.

Contact sifting performance is defined analytically through equations (21–25). Note that these equations are based on some simplifying assumptions:

- Targets are not closely spaced: the model does not account for the presence of contacts from one target in the validation region of another.
- Targets are not near the edge of the surveillance region: we consistently use B as the size of the validation region.
- The impact of imperfect sensor resolution (grid cell size) is neglected: false contacts are assumed to be uniformly distributed in the surveillance area, and target detections are Gaussian distributed and centered on true target location.
- False contact statistics are based on the contact-absent case. That is, we do not include target contacts outside the validation region in the model for the number of false contact: the impact is minimal for non-trivial false contact rates.

Note that in the limit $\varepsilon \to 0$ and $\xi \to 0$ (i.e., $\sigma_x \to 0$ faster than $\varepsilon \to 0$), we see from equation (25) that $P_L \to P_D$. (One can think of P_L as a generalization of P_D in the presence of localization error.)

It is important to use the *probability of localization* as defined here (i.e., detection *and* localization) as a performance measure, rather than simply the *probability of detection*. Indeed, we are interested in comparing the performance at the input and output of the static fusion process. Accordingly, it is important to measure performance consistently: at the output of static fusion processing, we define target-induced and false contacts based on a localization threshold.

4.2. Contact Sifting Performance

Assume we have N independent identical sensors with performance as described in the previous section. Note that N in this section should not be confused with the track-initiation window size in Section 2: the two may be different in general.

Indeed, note that in Section 2 we reasoned over both Z sensors and the temporal window of N scans for track initiation. Here, we study the static fusion problem, where we can alternatively think of having a set of synchronous sensors, or a set of scans from the *same* sensor. That is, we only have Z or N to consider. We have chosen to denote this buffer size by N, as this buffer size relates directly to the previous track-initiation discussion.

Surveillance performance will depend critically on the specific fusion algorithm that we employ to combine N sets of contacts into a single set. In this section, we consider the simple batch-processing approach.

Even with the assumption of like-performing sensors, one might ask whether optimal multi-sensor performance will be achieved by requiring that all sensors use the same detection threshold D. It is known from the distributed detection literature that the assumption of equal local thresholds may lead to sub-optimal performance; nonetheless, in certain cases optimality is indeed achieved with identical local thresholds [25]. Further, under this assumption the form of the optimal fusion rule is known to be of the form K-of-N, though the optimal choice of K will depend on the local threshold, i.e., a fixed K-of-N fusion rule is not optimal in general. ROC performance of the K-of-N fusion rule is the following [22]:

$$P_D(K,N) = \sum_{j=K}^{N}\binom{N}{j}P_D^j(1-P_D)^{N-j} \tag{26}$$

$$P_{FA}(K,N) = \sum_{j=K}^{N}\binom{N}{j}P_{FA}^j(1-P_{FA})^{N-j}. \tag{27}$$

Note that K in this section should not be confused with the track-termination threshold defined in Section 2.

As in the single-sensor case, we are interested in a slightly modified definition of the classical ROC curve, where we replace $P_{FA}(K,N)$ with $N_{FA}(K,N)$; these are related as follows:

$$N_{FA}(K,N) = P_{FA}(K,N)N_{cell}. \tag{28}$$

For simplicity, we proceed with an assumption of equal local sensor statistics and local detection thresholds. The *contact sifting* approach relies on a sifting grid in measurement space. In each sifting cell, we sum the number of contacts over all N sensors. Sifting cells in which the number of contacts exceeds K lead to a fused contact, with localization based on appropriate averaging over the location of the single-sensor contacts (accounting for contact uncertainties).

In order to develop a simple model for contact sifting performance, we introduce a number of simplifying approximations, in addition to those previously introduced. The impact of these simplifying approximations will be evaluated experimentally.

- Assume targets are located at the center of sifting cells.
- Assume sifting cell size and shape is same as the validation region B introduced previously.
- Neglect overlaps among circular grid cells, and assume the cells fully cover the surveillance region. This simplifies the computation of false contact statistics, and ensures that successful localization corresponds precisely to fused contacts in target-present cells.

Let the random variables $N_{\mathrm{T}}^{\mathrm{cell}}$ and $N_{\mathrm{FA}}^{\mathrm{cell}}$ denote the number of target and non-target contacts in a particular sifting cell, respectively. Then, the probability of localization $P_{\mathrm{L}}(K,N)$ can be expressed by leveraging the following decomposition:

$$\begin{aligned} P_{\mathrm{L}}(K,N) &= P(N_{\mathrm{T}}^{\mathrm{cell}} + N_{\mathrm{FA}}^{\mathrm{cell}} \geq K) \\ &= P(N_{\mathrm{T}}^{\mathrm{cell}} \geq K) + \sum_{j=0}^{K-1} P(N_{\mathrm{T}}^{\mathrm{cell}} = j) P(N_{\mathrm{FA}}^{\mathrm{cell}} \geq K-j). \end{aligned}$$

Accordingly, letting $\tilde{P}_{\mathrm{D}} = P_{\mathrm{D}}(1-\exp(-\xi^2/2))$, we have

$$\begin{aligned} P_{\mathrm{L}}(K,N) = &\sum_{j=K}^{N} \binom{N}{j} \tilde{P}_{\mathrm{D}}^{j} (1-\tilde{P}_{\mathrm{D}})^{N-j} \\ &+ \sum_{j=0}^{K-1} \left[\binom{N}{j} \tilde{P}_{\mathrm{D}}^{j} (1-\tilde{P}_{\mathrm{D}})^{N-j} \sum_{i=K-j}^{\infty} \frac{\lambda_B^i}{i!} \exp(-\lambda_B) \right]. \end{aligned} \tag{29}$$

The expected number of false contacts $\tilde{N}_{\mathrm{FA}}(K,N)$ is given by the following, where $N_{\mathrm{cell}}^{\mathrm{sift}}$ is the number of sifting cells that is not to be confused with the number of sensor grid cells N_{cell}:

$$\tilde{N}_{\mathrm{FA}}(K,N) = N_{\mathrm{cell}}^{\mathrm{sift}} \sum_{j=K}^{\infty} \frac{(N\lambda_B)^j}{j!} \exp(-N\lambda_B). \tag{30}$$

More compact analytical expressions for (29–30) based on the incomplete gamma function [1] are given as follows:

$$\begin{aligned} P_{\mathrm{L}}(K,N) = &\sum_{j=K}^{N} \binom{N}{j} \tilde{P}_{\mathrm{D}}^{j} (1-\tilde{P}_{\mathrm{D}})^{N-j} \\ &+ \sum_{j=0}^{K-1} \left[\binom{N}{j} \tilde{P}_{\mathrm{D}}^{j} (1-\tilde{P}_{\mathrm{D}})^{N-j} \Gamma(K-j, \lambda_B) \right] \end{aligned} \tag{31}$$

$$\tilde{N}_{\mathrm{FA}}(K,N) = N_{\mathrm{cell}}^{\mathrm{sift}} \Gamma(K, N\lambda_B). \tag{32}$$

Further, when N is large enough and λ_B small, $P_{\mathrm{L}}(K,N)$ given by equation (29) can be calculated by using the Laplace-De Moivre approximation [20]:

$$P_{\mathrm{L}}(K,N) = Q\left(\frac{K - N\tilde{P}_{\mathrm{D}}}{\sqrt{N\tilde{P}_{\mathrm{D}}(1-\tilde{P}_{\mathrm{D}})}} \right). \tag{33}$$

$Q(\cdot)$ is the complementary distribution function of the standard normal random variable:

$$Q(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt. \tag{34}$$

It is of interest to compare ROC curves based on equations (28–30) with performance curves based on equations (31–32). Letting $N_{\mathrm{cell}}^{\mathrm{sift}} = N_{\mathrm{cell}}$, we have $P_{\mathrm{FA}} = \lambda_B$. For large N, we have exact equivalence of equations (28) and (30); we have reasonable agreement for modest values of N. Next, we examine the limit $\varepsilon \to 0$ and $\xi \to 0$ (i.e., $\sigma_x \to 0$ faster than $\varepsilon \to 0$) in equation (29). Note that this limit impacts both the validation region and the sifting cell size, as we have fixed these to be the same. We have both $\tilde{P}_{\mathrm{D}} \to P_{\mathrm{D}}$ and $\lambda_B \to 0$. It follows immediately that $P_{\mathrm{L}}(K,N) \to P_{\mathrm{D}}(K,N)$.

Thus, contact sifting with $N_{\mathrm{cell}}^{\mathrm{sift}} = N_{\mathrm{cell}}$, extremely small localization errors ($\sigma_x \to 0$), and extremely large number of sensors ($N \to \infty$) corresponds precisely to the distributed-detection problem characterized by equations (26–28). Even for finite N, there is close agreement with the analytical performance curve given by equations (29–30).

The choice of contact sifting cell size has inherent tradeoffs. Large cells will reduce the contrast between target-absent and target-present statistics. Likewise, a small cell containing a target is less likely to contain those target-originated contacts that incur significant localization errors. In addition to cell size, the choice of threshold parameter includes non-trivial tradeoffs. As in distributed detection theory (and as noted earlier), in general the optimal contact-sifting fusion rule will dependent on the single-sensor (local) detection threshold.

Experimental validation of contact sifting performance (equations 29–30) is documented in [14].

4.3. Other Approaches to Contact Fusion

The *contact sifting* algorithm is not effective in close-target scenarios. Another approach to the problem is

to consider a *multi-sensor probabilistic data association* algorithm. Further details of this approach may be found in [16].

Under this approach, we take a *generalized likelihood ratio test* (GLRT) approach for both the detection and estimation problems. For each hypothesized target, we find its location estimate that maximizes the likelihood function, and choose the hypothesis that has the largest likelihood. This results in a procedure that maximizes the likelihood function with respect to the number of targets and their respective locations.

A constraint is imposed on the maximum allowable number of targets present in the surveillance region. A sequential search over the number of targets is used for a computationally feasible solution. The technique provides location estimates as part of the detection process. The location estimates can always be further refined by an estimation process. This approach in a different context is developed in [6–7]. In [7], comparisons are made between the proposed method and the unstructured and structured techniques based on Akaike information theoretic criteria (AIC) [2], minimum description length (MDL) [23], and Bayesian predictive density [11].

A known limitation of the GLRT approach is the *runaway* degree-of-freedom phenomenon. In [16], by not considering a penalty factor as prescribed by the AIC or *Bayesian Information Criterion* (BIC) approaches, we did face the problem of overestimating the number of targets.

Compared to the more complex approaches to contact fusion, the contact-sifting algorithm is simple and handles situations where we have large number of targets, albeit not closely spaced.

5. FUSE-BEFORE-TRACK PERFORMANCE STUDY

We have argued in Section 3 that the *fuse-before-track* architecture holds potential for target tracking in large sensor networks. Here, we provide results of Monte Carlo experimental validation. For both the centralized and FbT architectures, we use a track-oriented multi-hypothesis tracker [12–13]. Simulation and algorithmic parameters are in Table III.

We have stochastically-generated target ground truth based on a nearly constant velocity motion model, for which positional measurements are obtained from a number of like-performing sensors that are synchronized in time and with a fixed sensor revisit time. All target trajectories are initiated at scenario initiation and target death results if a target exits the scenario area. Initial target location is uniformly distributed in the surveillance region. The tracker is assumed to have knowledge of target motion and sensor parameters; by this we mean that the statistical characteristics of target motion and of the sensor measurement error are known.

Performance evaluation relies on track classification, whereby those tracks with sufficiently large average localization error from all target trajectories are classified as false. Otherwise, the closest trajectory is identified. For tracks that extend in time beyond a given target death, the last target location is used for positional comparison.

TABLE III
Parameters for Single-Sensor Tracking, Multi-Sensor Tracking, and FbT Simulation-Based Performance Evaluation

Parameter	Setting
Monte Carlo realizations	500
Number of targets	10
Target SNR	13 dB
Target maneuverability index	0.01 m^2s^{-3}
Initial velocity std. dev.	1 ms^{-1}
Sensor threshold	10.5 dB
Contact measurement error std. dev. (in both x and y dimensions)	10 m
Number of sensors	10
Sensor revisit time	10 sec
Scenario duration	3 min
Surveillance region	$(1.5\ km)^2$
Detection cell size	$(1\ m)^2$
Sifting cell size	$(30\ m)^2$
Sifting threshold (number of contacts)	3
Track initiation (FbT)	4-of-4
Track initiation (centralized)	12-of-40
FbT track termination (allowed misses)	3
Centralized track termination (allowed misses)	20
Hypothesis tree depth (n-scan)	2
Track classification distance threshold	14.14 m

Note that the track-initiation setting for the multi-sensor (centralized) configuration is different than in FbT, so as to have a comparable track rate at the processing output. Indeed, a concatenation of M-of-N track initiation criteria is roughly comparable to a rule where the Ms are multiplied together, and likewise for the Ns. Similarly, it can be verified that a track that is kept alive with a concatenation of M-of-N rules (with $M = 1$ in the second stage) has maintenance statistics comparable to a single (centralized) 1-of-N with an appropriate choice of N (in our case, 20 allowed missed detections).

Similarly, track termination is based on the maximum time since the last track update, rather than on the number of missed updates.

The sensor threshold and target SNR settings above lead to a target probability of detection (P_D) of 0.62. The sensor threshold, detection cell size, and surveillance region sizes lead to a contact false alarm rate (λ_{FAR}) of 30 contacts per scan. Given the scenario revisit time and scenario duration, for each Monte Carlo realization there are 18 contact files for each sensor, and 180 in total, leading to 18 fused-contact files.

Given the sifting cell size and sifting thresholds above, the first stage of FbT processing generally leads to approximately 20 fused contacts per scan. The fused-contact location is given by the mean of the contacts in the sifting cell and, correspondingly, the fused-contact measurement covariance is smaller than that of single-sensor contacts. These statistics follow directly from

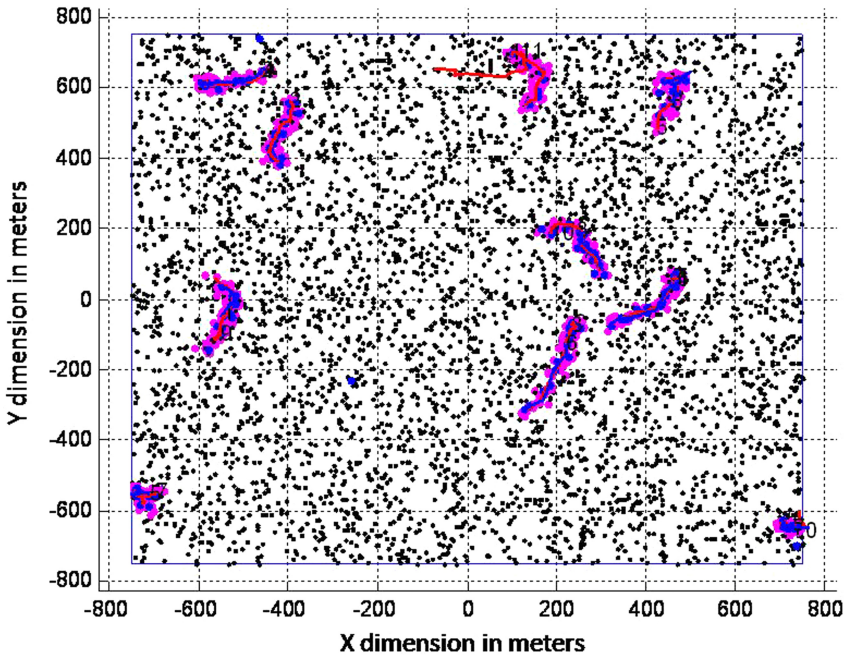


Fig. 9. One realization in the simulation-based analysis of centralized and fuse-before-track processing. False contacts are black dots, target-originated contacts are magenta dots, target trajectories are magenta, fused contacts in FbT processing are blue dots, FbT tracks are blue, and centralized tracks are red.

TABLE IV
Tracker Performance Results

Metric	Centralized	FbT
P_T	0.90	0.60
λ_{FTR}	186	28
Execution time ratio	0.78	0.013

least-squares estimation, and may alternatively be interpreted as the result of Kalman filtering of a sequence of contacts at the same measurement time.

Performance results are given in Table IV. Note first that the overall input false contact rate is roughly 108,000 contacts per hour. Thus, in both centralized and FbT configurations, target tracking provides a significant data reduction.

The key track detection statistics are *track hold* (P_T), the average fraction of time during which an active track exists on a target, and the *false track rate* (λ_{FTR}), the number of false tracks produced per unit time. The centralized tracking and FbT results are comparable: the former has better track hold, while the latter has a lower false track rate. Both achieve a dramatic reduction in false objects, for a track hold that is comparable or better than the target probability of detection.

The results documented here are encouraging, since the first stage of FbT processing (static fusion) could be improved further, as discussed in Section 4.3. Thus, the FbT architecture is promising and deserves further investigation.

Perhaps a more compelling motivation for the FbT architectures is the dramatic improvement in processing time. Indeed, contact sifting provides a dramatic reduction in the contact data rate: fused contact files include approximately 20 contacts. This, combined with the ten-fold reduction in the number of contact files, leads to considerable computational savings in MHT processing. The savings in execution time are a combination of reduced computational load as well as saving in reading and writing a much small number of input and output files, respectively.

The *execution time ratio* in Table IV is the ratio of average tracker processing time and scenario duration. We see that centralized tracking achieves slightly faster than real-time processing, while FbT requires only a small fraction of the processing time. (Results are generated on a DELL OPTIPLEX GX620 with Intel Pentium D processor.)

Fig. 9 provides an illustration of a realization of contact-level and track-level data, while Figs. 10–12 illustrate some examples in detail. As indicated in Table III, generally we find that the FbT approach exhibits good track stability, at the cost of a longer track initiation time that in turn induces a lower track hold than in centralized tracking. Centralized tracking exhibits poorer tracking stability and, correspondingly, a higher false track rate.

The example in Fig. 12 shows two target trajectories that start in close proximity. While it takes longer to initiate tracks with the FbT approach, this is achieved without false track formation.

It should be noted that this study has been limited to random target tracks in a fairly wide surveillance region, which rarely leads to dense multi-target instantiations. These would challenge the FbT architecture as implemented here, as the first-stage contact-sifting

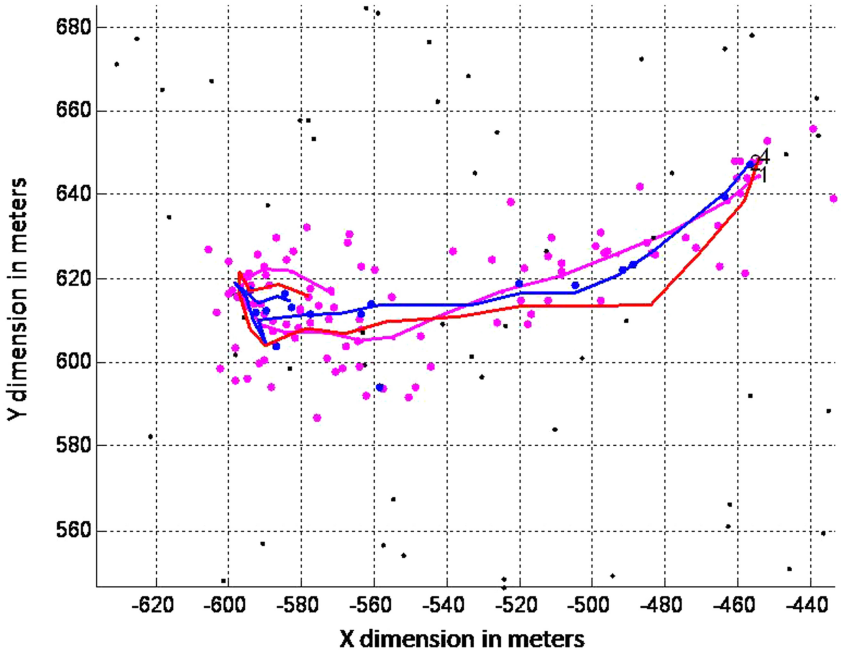


Fig. 10. An instance of comparable performance of centralized (red) and FbT (blue) tracks.

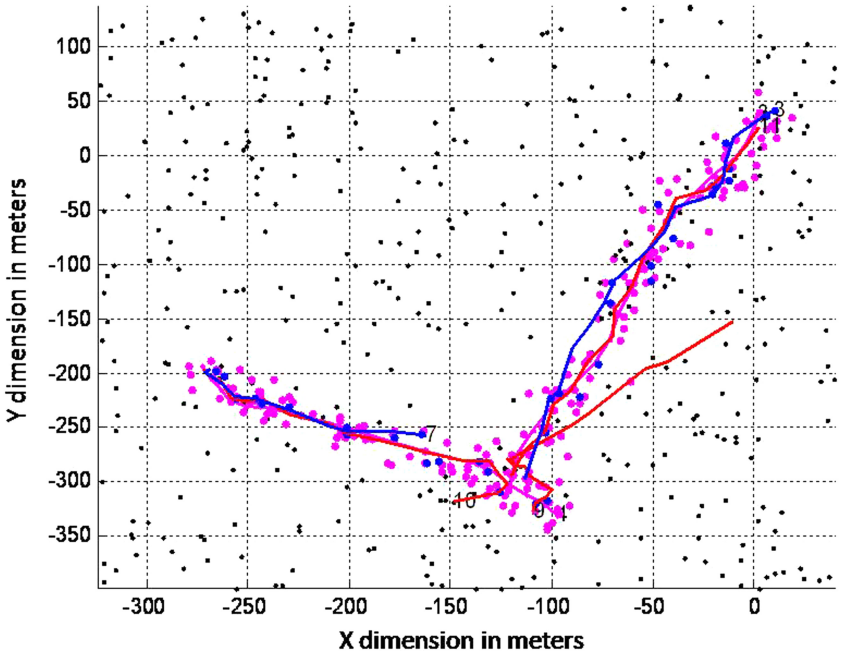


Fig. 11. An illustration of lower track hold but with more stable tracks in the FbT (blue) approach.

algorithm is known to be inappropriate for close-target cases. Likewise, in a computational sense, dense-target scenarios would challenge the centralized tracker more severely than it would in the FbT approach.

6. CONCLUSIONS

While large sensor networks hold great potential for surveillance performance, current scan-based target tracking technology by itself may not offer the best processing paradigm. Conversely, existing batch processing approaches do not provide real-time surveillance outputs. Thus, we believe a two-stage architecture that leverages the strengths of both batch and scan-based processing holds great potential for effective surveillance performance. In particular, contact fusion for a large number of nearly simultaneous sensor scans may be followed quite effectively by scan-based tracking.

This paper has addressed these contributions. First, in Section 2 we introduced an analytical performance model for scan-based tracking, and studied the performance limitations that the model suggests for increasing date rates (or number of sensors). Next, in Section 3 we introduced the *fuse-before-track* (FbT) architecture

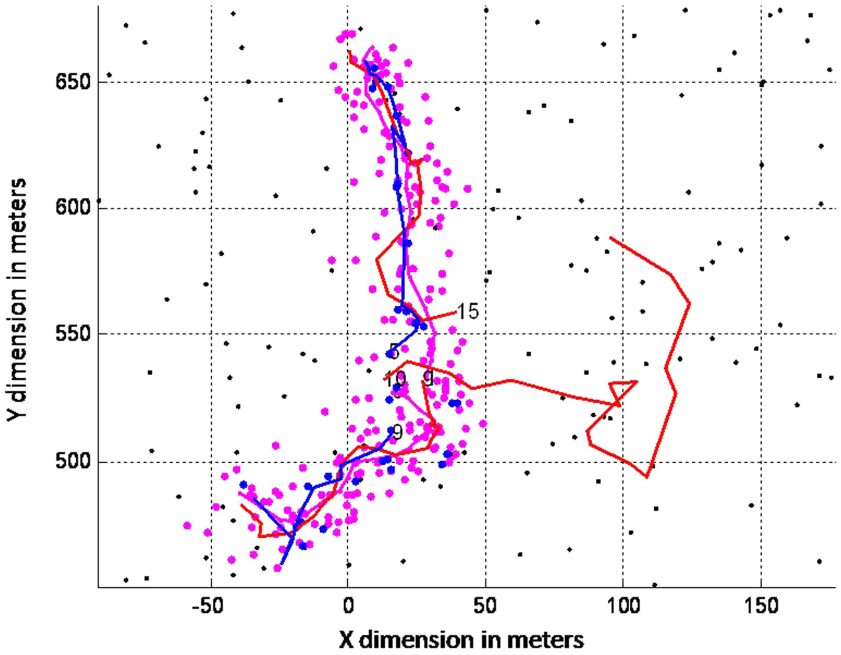


Fig. 12. Another illustration of reduced FbT (blue) track hold, but without false track confirmation as observed in the centralized case (red).

for automatic tracking in large sensor networks, which includes contact fusing followed by scan-based tracking; the specific FbT instantiation that we have implemented utilizes the *contact sifting* algorithm followed by an MHT tracker. After validating an analytical performance model for contact sifting and briefly describing another approach to contact fusion in Section 4, in Section 5 we describe simulation results that compare centralized and FbT processing results. We found that both approaches hold some merit and indeed both provide a dramatic reduction in false object rates. The FbB provides considerable computational savings, good track stability, and a lower false track rate, at the cost of reduced track hold.

A number of directions for future work exist. Principally, and in addition to a more effective first stage in the FbT architecture to handle closely-spaced target scenarios, the future direction includes an analysis of the impact of synchronized vs. staggered sensor sampling times [18, 26], for which analysis in the large-sensor case is lacking.

ACKNOWLEDGMENTS

Peter Willett was supported under ONR Contract N00014-07-1-0055. Marco Guerriero was partially supported through a six-month NURC visiting research appointment in 2007.

REFERENCES

[1] M. Abramowitz and I. Stegun
Handbook of Mathematical Functions.
Dover, New York, 1970.

[2] H. Akaike
A new look at the statistical model identification.
IEEE Transactions on Automatic Control, **19** (Dec. 1974).

[3] D. Avitzour
A maximum likelihood approach to data association.
IEEE Transactions on Aerospace and Electronic Systems, **28**, 2 (Apr. 1992).

[4] Y. Bar-Shalom and X. Li
Multitarget-Multisensor Tracking.
YBS Publishing, 1995.

[5] Y. Bar-Shalom, L. Campo and P. Luh
From receiver operating characteristic to system operating characteristic: Evaluation of a track formation system.
IEEE Transactions on Automatic Control, **35**, 2 (Feb. 1990).

[6] R. Bethel and K. Bell
A multi-hypothesis GLRT approach to the combined source detection and direction of arrival estimation problem.
In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP '01), Salt Lake City, UT, May 2001.

[7] R. Bethel and K. Bell
Maximum likelihood approach to joint array detection/estimation.
IEEE Transactions on Aerospace and Electronic Systems, **40**, 3 (July 2004).

[8] S. Blackman
Multi-Target Tracking with Radar Application.
Artech House, 1986.

[9] S. Blackman and R. Popoli
Design and Analysis of Modern Tracking Systems.
Artech House, 1999.

[10] W. Blanding, P. Willett, Y. Bar-Shalom and S. Coraluppi
Multisensor track termination for targets with fluctuating SNR.
In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP), Honolulu, HI, Apr. 2007.

[11] C. Cho and P. Djuric
Detection and estimation of DOAs of signals via Bayesian predictive densities.
IEEE Transactions on Signal Processing, **42**, 11 (Nov. 1994).

[12] S. Coraluppi and C. Carthel
Distributed tracking in multistatic sonar.
IEEE Transactions on Aerospace and Electronic Systems, **41**, 3 (July 2005).

[13] S. Coraluppi, C. Carthel, D. Grimmett and O. Gerard
Multistatic sonar tracking: Algorithmic advances and sea trial evaluation.
U.S. Navy Journal of Underwater Acoustics, **56**, 3 (July 2006).

[14] S. Coraluppi, M. Guerriero and P. Willett
Contact fusion in large sensor networks: Operational performance analysis.
In *Proceedings of the NATO RTO SET Panel Symposium on Sensors and Technology for Defence Against Terrorism*, Mannheim, Germany, Apr. 2008.

[15] S. Coraluppi, M. Guerriero and P. Willett
Optimal fusion performance modeling in sensor networks.
In *Proceedings of the 11th International Conference on Information Fusion*, Cologne, Germany, July 2008.

[16] M. Guerriero, S. Coraluppi and P. Willett
Analysis of scan and batch processing approaches to static fusion in sensor networks.
In *Proceedings of the SPIE Conference on Signal and Data Processing of Small Targets*, Orlando FL, Mar. 2008.

[17] P. Kumar and P. Varaiya
Stochastic Systems.
Prentice-Hall, 1986.

[18] R. Niu, P. Varshney, K. Mehrotra and C. Mohan
Temporally staggered sensors in multi-sensor target tracking systems.
IEEE Transactions on Aerospace and Electronic Systems, **41**, 3 (July 2005).

[19] M. Prior and A. Baldacci
The physical causes of clutter and its suppression via subband processing.
In *Proceedings of OCEANS 2006*, Boston, MA, 2006.

[20] A. Papoulis
Probability, Random Variables, and Stochastic Processes.
McGraw-Hill, New York, 1984.

[21] C. Rago, P. Willett and M. Alford
Predetection fusion: Resolution cell grid effects.
IEEE Transactions on Aerospace and Electronic Systems, **35**, 3 (July 1999).

[22] A. Riebman and L. Nolte
Optimal detection and performance of distributed sensor systems.
IEEE Transactions on Aerospace and Electronic Systems, **23**, 1 (Jan. 1987).

[23] J. Rissanen
Modeling by shortest data description.
Automatica, **14** (1978).

[24] L. Stone, C. Barlow and T. Corwin
Bayesian Multiple Target Tracking.
Artech House, 1999.

[25] J. Tsitsiklis
Decentralized detection.
Advances in Signal Processing, JAI Press, 1993.

[26] X. Zhang, P. Willett and Y. Bar-Shalom
Uniform versus nonuniform sampling when tracking in clutter.
IEEE Transactions on Aerospace and Electronic Systems, **42**, 2 (Apr. 2006).

Stefano Coraluppi (SM'XX) received B.S. degrees in electrical engineering and in mathematics from Carnegie Mellon University, Pittsburgh, PA, in 1990, and M.S. and Ph.D. degrees in electrical engineering from the University of Maryland, College Park, MD, in 1992 and 1997.

He has held research staff positions at ALPHATECH Inc., Burlington, MA (1997–2002) and at the NATO Undersea Research Centre in La Spezia, Italy (since 2002). He has contributed to numerous programs in ground, undersea, and maritime surveillance for security and defense applications. His research interests include detection and estimation theory, target tracking, data fusion, and stochastic control.

Dr. Coraluppi served as General Cochair of the 9th ISIF/IEEE International Conference on Information Fusion (FUSION) in Florence, Italy, in 2006 (with Peter Willett) and Program Chair of the NATO Workshop on Data Fusion and Anomaly Detection for Maritime Situational Awareness in La Spezia, Italy, in 2009. He has served as member of the NATO Research and Technology Organization (RTO) Sensors and Electronics Technology (SET) Panel, associate editor for both the *IEEE Journal of Oceanic Engineering* and the *ISIF Journal of Advances in Information Fusion* (JAIF), and lecturer for the NATO RTO, the Italian Naval Academy, and the University of Pisa. Currently, he is associate editor for Target Tracking and Multisensor Systems for the *IEEE Transactions on Aerospace and Electronic Systems*, area editor for Tracking for the ISIF JAIF, and member of the Board of Directors of the International Society of Information Fusion (serving as president in 2010).

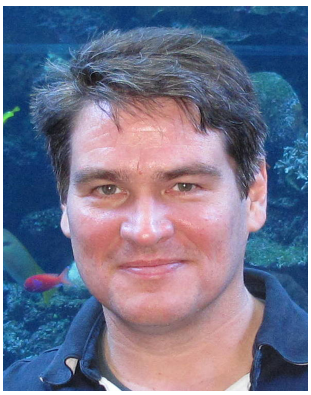
Marco Guerriero was born in Salerno, Italy, on June 18, 1981. He received his B.A.Sc. and M.Sc. (electrical engineering) from the University of Salerno, Italy, in 2002 and 2005, respectively and his Ph.D. degree from University of Connecticut in 2009.

In the fall of 2009 he was a visiting scientist at the NATO Undersea Research Centre (NURC) La Spezia, Italy. He is now a system analyst at the Research and Advanced System Design Department at ELETTRONICA S.p.A. in Rome. His research interests lie in the areas of signal processing, with particular focus on distributed detection and estimation in sensor networks, target tracking and data fusion and radar signal processing.

Peter Willett (F'03) received his B.A.Sc. (engineering science) from the University of Toronto in 1982, and his Ph.D. degree from Princeton University in 1986.

He has been a faculty member at the University of Connecticut since 1986, and since 1998 has been a professor. He has published 135 journal articles (13 more under review), 290 conference papers, and 9 book chapters. His primary areas of research have been statistical signal processing, detection, machine learning, data fusion and tracking. He has interests in and has published in the areas of change/abnormality detection, optical pattern recognition, communications and industrial/security condition monitoring.

He is editor-in-chief for *IEEE Transactions on Aerospace and Electronic Systems*, and until recently was associate editor for three active journals—*IEEE Transactions on Aerospace and Electronic Systems* (for Data Fusion and Target Tracking) and *IEEE Transactions on Systems, Man, and Cybernetics*, parts A and B. He is also associate editor for the IEEE AES Magazine, editor of the AES Magazine's periodic Tutorial issues, associate editor for ISIF's electronic *Journal of Advances in Information Fusion*, and is a member of the editorial board of IEEE's Signal Processing Magazine. He was a member of the IEEE AESS Board of Governors 2003–2009. He was general cochair (with Stefano Coraluppi) for the 2006 ISIF/IEEE Fusion Conference in Florence, Italy, Program Co-Chair (with Eugene Santos) for the 2003 IEEE Conference on Systems, Man & Cybernetics in Washington, D.C., and program cochair (with Pramod Varshney) for the 1999 Fusion Conference in Sunnyvale. He was coorganizer of the tracking subsession at the 1999 IEEE Aerospace Conference, and has been organizer of the Remote Sensing Track of that conference 2000–2003. Jointly with T. Kirubarajan he has coorganized the SPIE "System Diagnosis and Prognosis: Security and Condition Monitoring Issues" Conference in Orlando, 2001–2003. He has been a member of the IEEE Signal Processing Society's Sensor-Array & Multichannel (SAM) Technical Committee since 1997, and both serves on that TC's SAM Conferences' Program Committees and maintains the SAM website.



Craig Carthel received B.S. degrees in physics and mathematics in 1988, a M.S. in mathematics in 1992, and a Ph.D. in mathematics in 1995, all from the University of Houston, where he did research in numerical analysis and optimization theory.

He is a senior scientist in the Applied Research Department at the NATO Undersea Research Centre in La Spezia, Italy, where he works on military operations research, simulation, optimization, and data fusion problems associated with maritime environments. From 1995 to 1997, he worked at the Institute for Industrial Mathematics at Johannes Kepler University, in Linz, Austria on parameter identification and inverse problems. From 1998 to 2002, he was a senior mathematician at ALPHATECH Inc. in Burlington, MA, where he worked on image processing, multisensor data fusion and ground target tracking. In 2006, he served as the technical program chair for The 9th International Conference on Information Fusion.

A Pragmatic Approach for the use of Dempster-Shafer Theory in Fusing Realistic Sensor Data

PIERRE VALIN
PASCAL DJIKNAVORIAN
ÉLOI BOSSÉ

This article addresses the performance of Dempster-Shafer (DS) theory, when it is slightly modified to prevent it from becoming too certain of its decision upon accumulation of supporting evidence. Since this is done by requiring that the ignorance never becomes too small, one can refer to this variant of DS theory as Thresholded-DS. In doing so, one ensures that DS can respond quickly to a consistent change in the evidence that it fuses. Only realistic data is fused, where realism is discussed in terms of data certainty and data accuracy, thereby avoiding Zadeh's paradox. Performance measures of Thresholded-DS are provided for various thresholds in terms of sensor data certainty and fusion accuracy to help designers assess beforehand, by varying the threshold appropriately, the achievable performance in terms of the estimated certainty, and accuracy of the data that must be fused. The performance measures are twofold, first in terms of stability when fused data are consistent, and second in terms of the latency in the response time when an abrupt change occurs in the data to be fused. These two performance measures must be traded off against each other, which is the reason why the performance curves will be very helpful for designers of multi-source information fusion systems using Thresholded-DS.

Manuscript received August 4, 2009; revised December 2, 2009; released for publication December 15, 2009.

Refereeing of this contribution was handled by Jean Dezert.

Authors' address: P. Valin and El. Bossé, Decision Support Systems section, Defence R&D Canada Valcartier, 2459 Blvd Pie XI Nord, Québec, G3J 1X5, Canada; P. Djiknavorian, Département de Génie Électrique et de Génie Informatique, Université Laval, Québec, G1K 7P4, Canada.

1. INTRODUCTION

Potential users of Dempster-Shafer (DS) theory [5, 10] are often faced at the outset with a list of its pitfalls, which they must somehow solve or at least live with:

1. When confronted with Bayesian reasoning over N identities, DS theory seems at a disadvantage. Indeed, since DS theory reasons over the power set, which has $2^N - 1$ elements, excluding the null set, the storage of all of the intermediate fusion results and the processing of them quickly can become overwhelming, when compared to Bayesian reasoning. However, many solutions were developed from 1993 until 1997, such as those of Simard et al. [4, 11], Tessem [14], and Bauer [1]. They all involve approximation (or truncation) schemes with 3 tunable parameters, and some have been researched extensively [2, 3] as to which values are appropriate for a given situation. One therefore takes the view that this problem can be solved, and we will then focus on cases with small values of N.
2. When the evidence to be fused is too consistent, DS theory will become certain of it after a sufficient number of steps, and will have an extremely hard time to react to a sudden real change in the evidence to be fused. This was solved by Simard et al. [4, 11] by preventing the ignorance from falling below a certain threshold, hereafter called $I_{\min}$, after each fusion step, one of the three tunable parameters mentioned previously. After setting the ignorance to $I_{\min}$, all the other masses are rescaled proportionately, so that these rescaled masses now sum up to $(1 - I_{\min})$. This is the approach we will follow here.
3. When evidence is too conflicting, the normalization step in DS theory can cause wild behaviours from one extreme to another. This is partially a problem in modeling the uncertainty of the data to be fused. We take the approach that the data must correctly be modeled by specifying its accuracy and certainty in a reasonable and realistic manner.

At this point, one should make more precise what is meant by data certainty and accuracy:

1. Certainty is a feature of the sensor that declares that a certain proposition is true with a given mass value m. With little loss of generality, one can assume for simplicity that the sensor declares only one proposition with mass m, and that the rest is assigned to the ignorance. This is likely the case, when the time allowed for decisions is critical, since it provides at each time step only one likely candidate for the declaration. In the example scenario described later, an Electronic Support Measures (ESM) sensor is likely to provide such a behaviour. In order to stress this point, the article will always mention in the text "sensor certainty."

2. Accuracy refers here to how often the data is likely to be wrong. For example, the association mechanism that is necessary to select which sensor data is to be associated to which track can sometimes be erroneous, particularly if it is single scan in nature. Accuracy is therefore a characteristic of the fusion process, not the sensor itself. In the case of the ESM sensor, miss-associations can occur for the bearing-only reports when the targets are densely found in that bearing angle. In order to stress this point, the article will always mention in the text "declaration accuracy."

One should point out at this time that any sensor will have a value for the uncertainty (or certainty) of its declaration(s), and that, however complex the association mechanism, the association mechanism will occasionally err in its contact-to-track (or track-to-track) correlations, which will provide an inaccuracy in the fusion results. In this sense, the performance characteristics that will be provided later below for Thresholded-DS can be applied to a wide range of sensors and positional fusion algorithms, with only very minor modifications.

2. STATEMENT OF THE PROBLEM AND SCENARIO

The selected problem was already used in publications [6–8] that addressed the use of Dezert-Smarandache (DSm) theory [12, 13] and compared it to Thresholded-DS. When the two approaches were compared in these publications, the focus was on DSm performance, while neglecting Thresholded-DS performance. It became quickly clear that, if one did not insist on conformance to STANAG 1241 [9] (which only DSm can provide), Thresholded-DS theory performed quite well. This article aims to fill this gap by exploring at much greater length the stability and response time of the theory for various threshold levels $I_{\min}$ in terms of sensor data accuracy and declaration certainty.

A possible illustration of the problem chosen is through the fusion of three types of ESM reports: Friend ($*gq_1$), Neutral (θ_2), or Hostile (θ_1). Since $N = 3$, the first pitfall of DS theory mentioned in the introduction is avoided, and no approximation schemes are necessary.

The approach followed in this article will be to study the ESM problem using a Modeling and Simulation (M&S) approach, first on specific representative scenarios, followed by a thousand Monte-Carlo runs to confirm the conclusions that can be reached.

The list of the prerequisites that any scenario must address are:

- Should have a clearly defined ground truth, which is sufficiently complex to test stability and latency in the response time.
- Should contain sufficient miss-associations, leading to values of average fusion accuracy that are in a realistic range.
- Should only provide partial knowledge about the ESM sensor declaration and to varying degrees, which therefore leads to sensor uncertainty (or sensor certainty) values that are in a realistic range.

The following scenario parameters have therefore been chosen accordingly:

1. The known ground truth is Friend (θ_1) for the first 50 time stamps of the scenario, and Hostile (θ_3) for the last 50 time stamps.
2. The percentage of correct associations is approximately Acc%, corresponding to countermeasures appearing $(100 - \text{Acc})\%$ of the time. Acc% will be explored over a realistic range between 60% and 90%. If the accurate allegiance is Friend (as is the case for the first 50 time stamps), then the declarations which correspond to miss-associations are equally distributed between Neutral and Hostile. Similarly, for the last 50 time stamps when Hostile is the correct allegiance, the miss-associations are distributed evenly between Friend and Neutral.
3. The ESM declaration has a mass of m, with the rest $(1 - m)$ being assigned to the ignorance, reflecting a certainty percentage Cer% in the declaration. Cer% will be explored over a realistic range between 60% and 90%.

This section will show a representative example of such a scenario, but the rest of the paper addresses the general trends that can be established from 1000 Monte-Carlo runs, where a different random seed is chosen for each member of the sequence in each Monte-Carlo run.

Thresholded-DS should be able to adequately represent the main features of the ground truth (which is known in an M&S approach), namely

1. Show stability under occasional miss-associations, namely show stability when fused data are generally consistent, specifically for the first 50 time stamps (after a short ramp-up time) and the last 50 time stamps (after the ramp-up time, or latency, due to the allegiance change).
2. Switch allegiance when the ground truth does so, namely have a reasonable measured latency in the response time (or delay, hereafter denoted Δ) when an abrupt change occurs in the data to be fused.

A typical scenario, with the random number generator set to produce *on average* (for a set of Monte-Carlo runs) an Acc% = 80%, is shown in Fig. 1, with the x-axis representing the time index.

For this scenario, Thresholded-DS achieves the results shown in Fig. 2, given a typical value of $I_{\min} = 0.02$. In Fig. 2, the x-axis represents the time index, and the y-axis represents the value of basic belief assignment (or mass) associated with the given hypothesis. Note that Thresholded-DS therefore never becomes more than 98% sure of its fused result (as mentioned in the introduction).

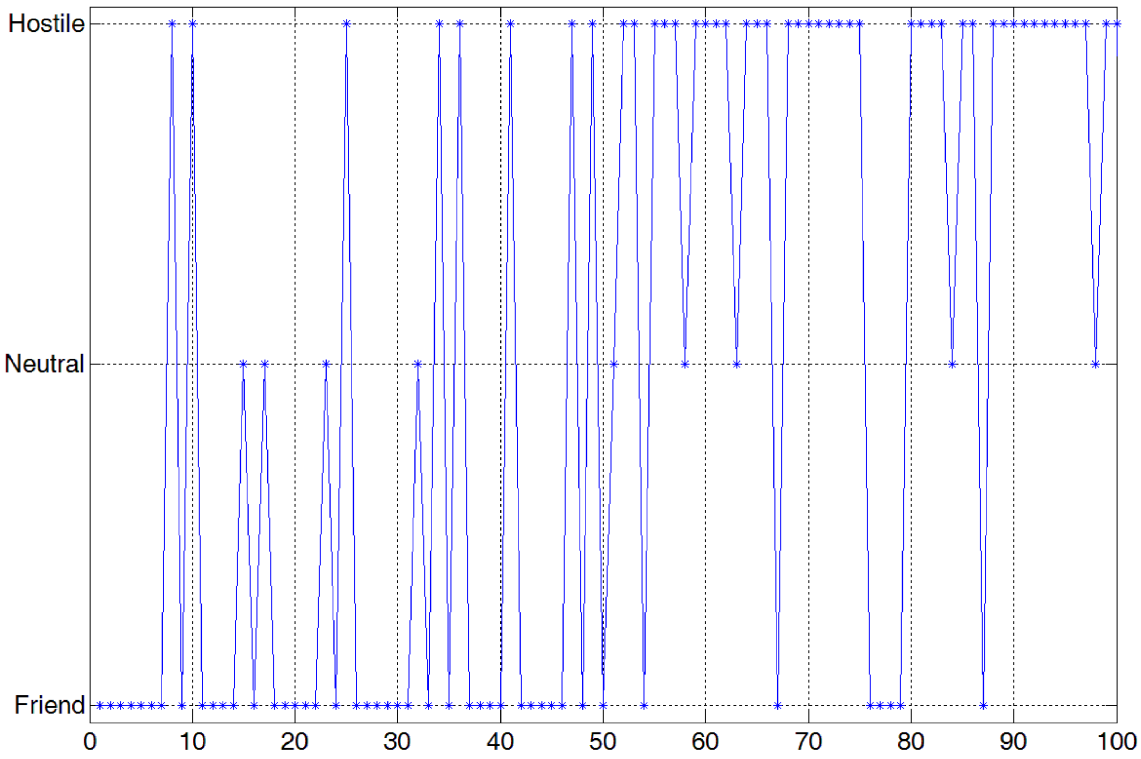


Fig. 1. Typical scenario with Acc% = 80%.

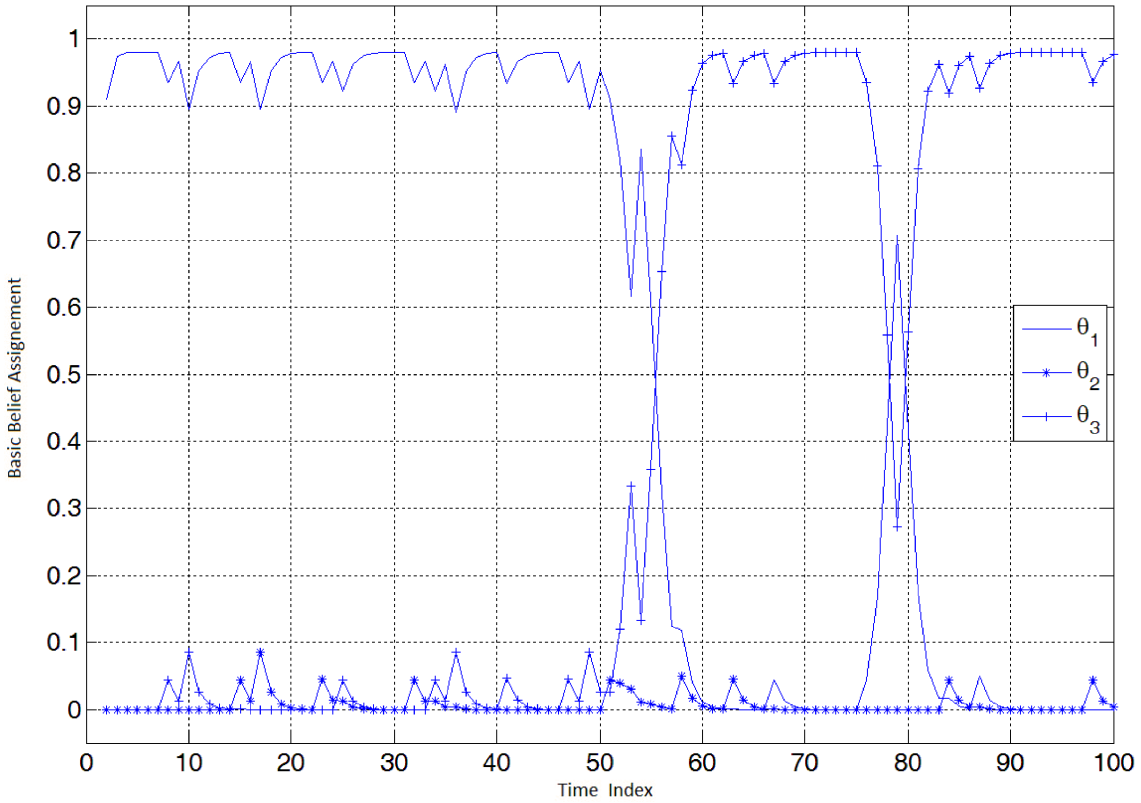


Fig. 2. Thresholded-DS for the typical scenario with Acc% = 80% and Cer% = 70%.

DS never becomes confused, shows good stability when miss-associations arrive randomly spaced out, which is the case until iteration 50. It then reacts reasonably quickly and takes about 8–10 reports before switching allegiance as it should. Furthermore, after being confused for an iteration around the sequence of

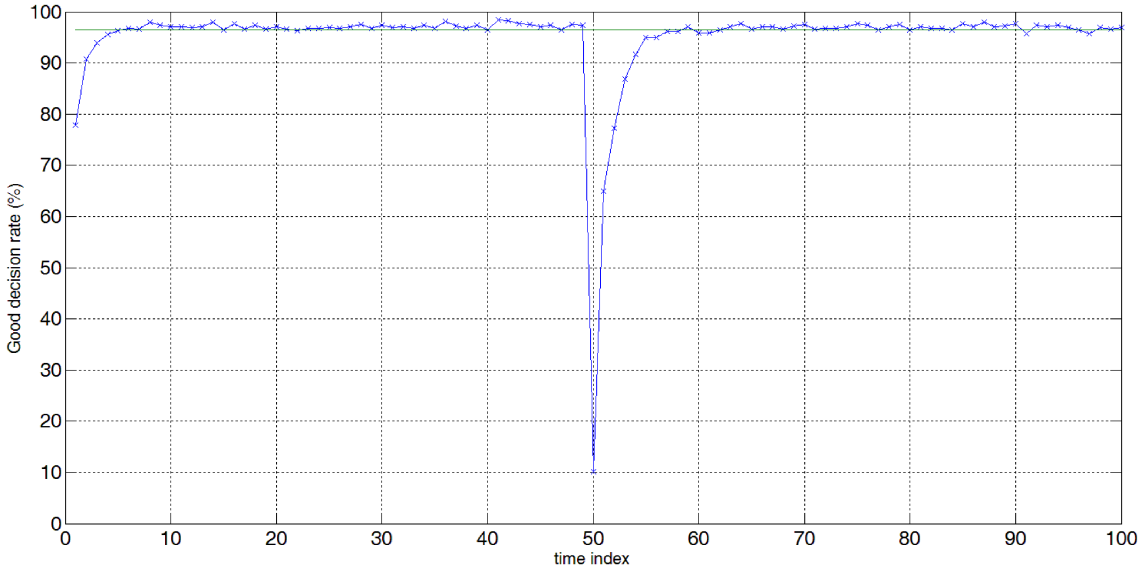


Fig. 3. Good decision rate for the scenario with Acc% = Cer% = 80% and 1000 runs.

four Friend reports starting at iteration 76, it quickly reverts to the correct Hostile status.

Fig. 3 shows a sample of a good decision rate of the target identification for Thresholded-DS using an input case such as the one from Fig. 1 generated randomly 1000 times. More specifically, it is the result of a Monte-Carlo simulation run of 1000 with an ESM sensor having values of accuracy and certainty both at 80% with the DS threshold at $I_{\min} = 0.05$ at every fusion step.

In order to evaluate the latency in the reaction time around iteration 50, we first determine the empirical mean averaged over time index 15 to 45 and 65 to 95, and then we subtract three times the value of the empirical standard deviation (3σ) averaged over the same interval. This interval has been chosen arbitrarily to exclude most of the instability that is mostly due to the initialization instability and the change of allegiance instability. So it will only include the instability of the decision system and the input data. The measure of latency then starts at time index 50, and ends at the time index at which the good decision rate reaches the threshold for reaction time performance shown as a horizontal line in Fig. 3. This horizontal line corresponds to the mean determined by the method above minus three standard deviations σ, which indicate the stability in the above mentioned time periods, according to the formulae for σ:

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2, \qquad \mu = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

The standard deviation σ tends to a fixed value as a function of increasing n, as shown in Figs. 4(a) for 100 Monte-Carlo iterations and 4(b) for 1000 Monte-Carlo iterations (0.16% in this case on the y-axis, with the x-axis being again the time index), but show less noise as n increases. This shows that σ is a dynamical feature of

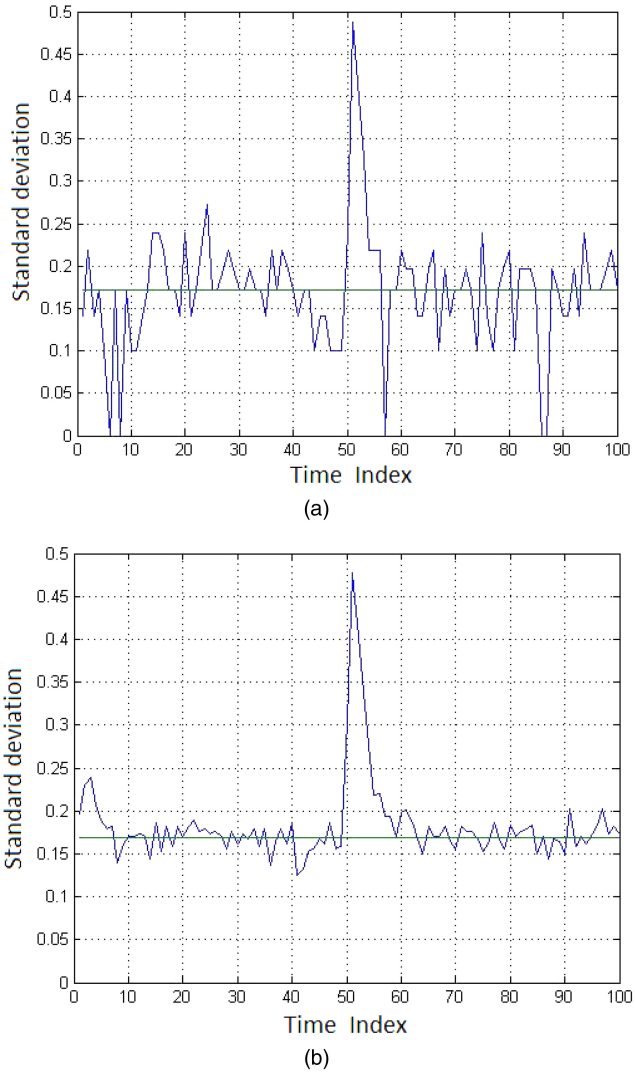


Fig. 4. (a) (top) and (b) (bottom). Standard deviations σ for stability (in %).

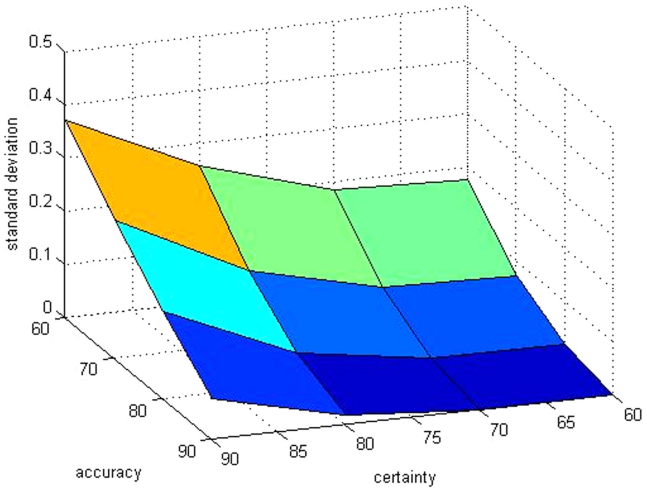


Fig. 5. Measure of stability σ for $I_{\min} = 0.01$.

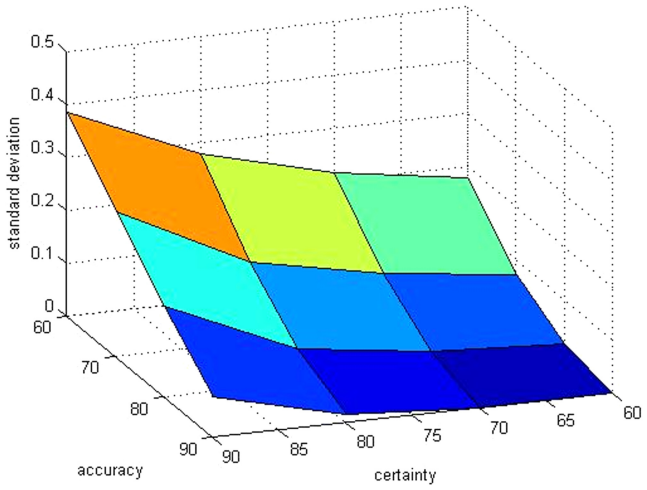


Fig. 6. Measure of stability σ for $I_{\min} = 0.02$.

the process, rather than being dependent on the number of Monte-Carlo runs.

Please note that this is just a practical definition of the latency, in order to show the trends in latency, when the parameters are varied, particularly $I_{\min}$. Other definitions may be more appropriate for other applications.

3. NUMERICAL GRAPHICAL RESULTS FROM MONTE-CARLO RUNS

This section shows the graphs for stability in the first subsection and reaction time latency (or delay) in the second subsection, for 1000 Monte-Carlo runs, for various values of the threshold in Thresholded-DS. Since one has three parameters to vary (certainty, accuracy, and $I_{\min}$), the presentation in this section focuses on showing the stability (in Subsection 3.1), and the reaction time latency (in Subsection 3.2) as a function of certainty and accuracy, with different figures corresponding to different choices for values of $I_{\min}$.

3.1. Stability

For an increase in the threshold of the minimum ignorance of 0.01 for each different figure, we have the

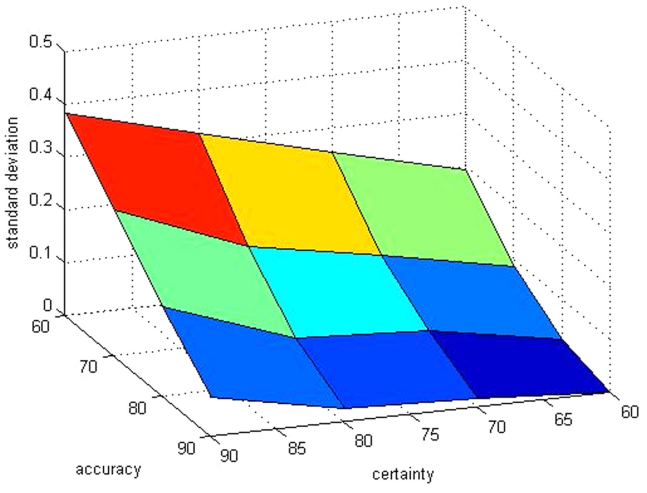


Fig. 7. Measure of stability σ for $I_{\min} = 0.03$.

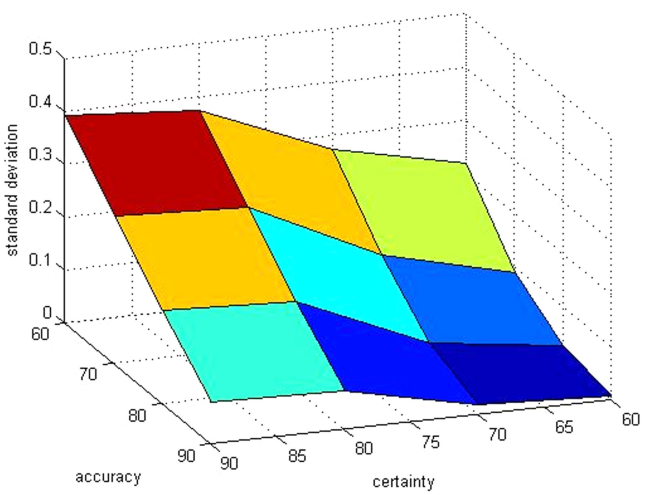


Fig. 8. Measure of stability σ for $I_{\min} = 0.04$.

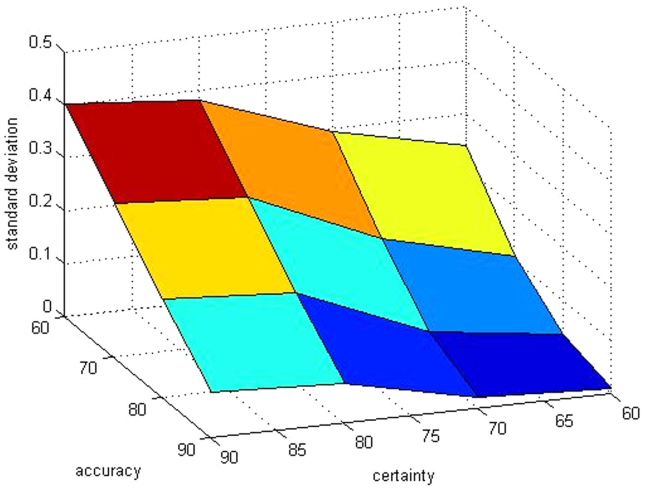


Fig. 9. Measure of stability σ for $I_{\min} = 0.05$.

following results for the standard deviation σ indicative of stability, for $I_{\min} = 0.01$ (Fig. 5), $I_{\min} = 0.02$ (Fig. 6), $I_{\min} = 0.03$ (Fig. 7), $I_{\min} = 0.04$ (Fig. 8), and $I_{\min} = 0.05$ (Fig. 9).

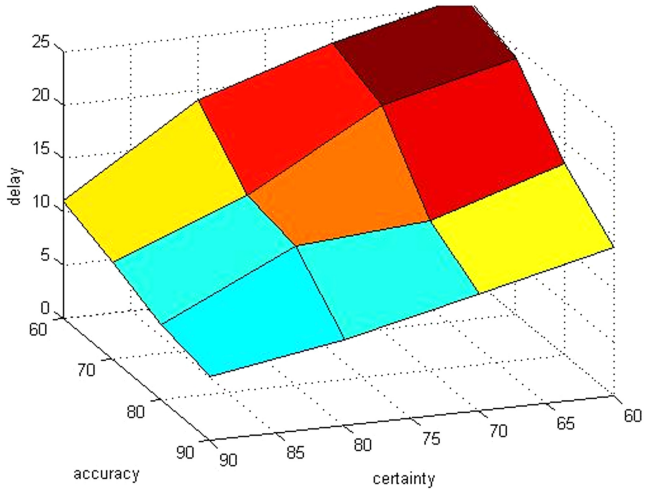


Fig. 10. Reaction time latency or delay for $I_{\min} = 0.01$.

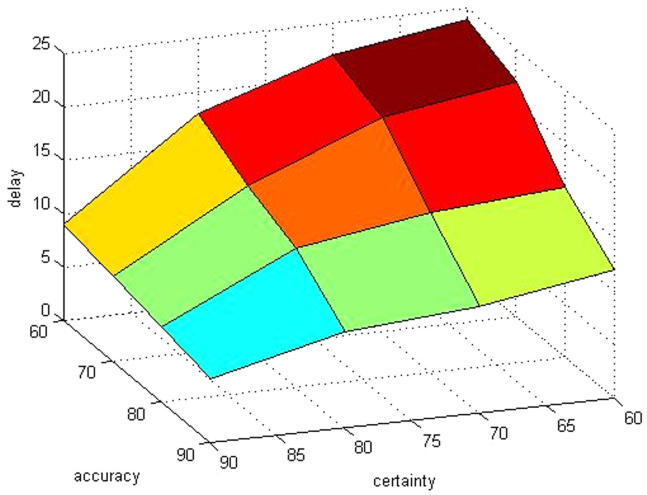


Fig. 11. Reaction time latency or delay for $I_{\min} = 0.02$.

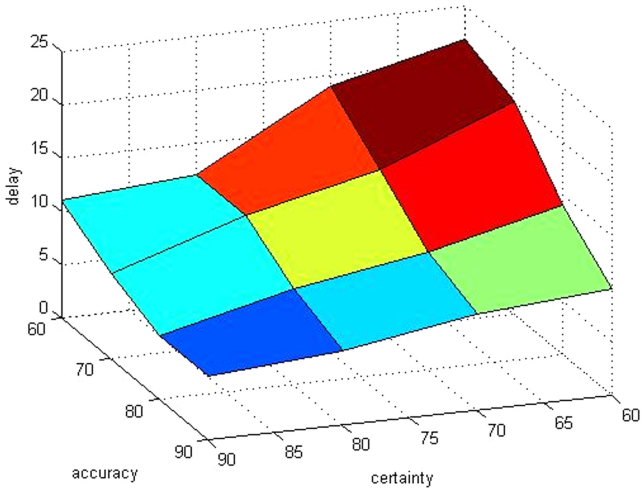


Fig. 12. Reaction time latency or delay for $I_{\min} = 0.03$.

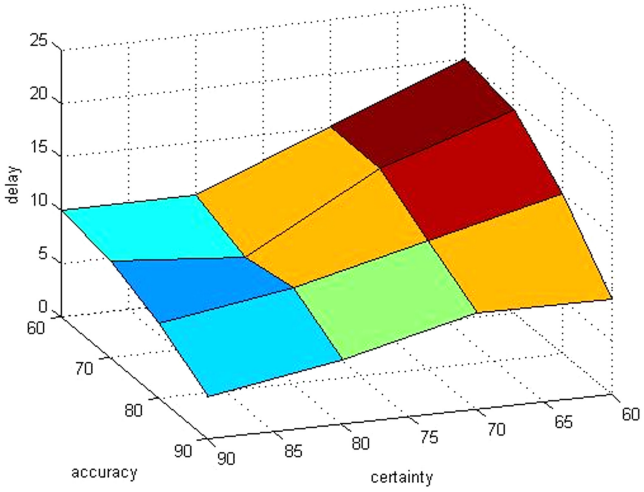


Fig. 13. Reaction time latency or delay for $I_{\min} = 0.04$.

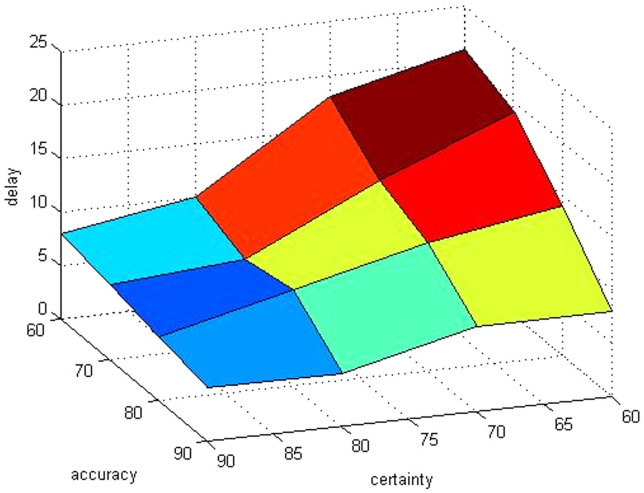


Fig. 14. Reaction time latency or delay for $I_{\min} = 0.05$.

Any much smaller value than 0.01 would result in too much rigidity when an allegiance changes, resulting in longer reaction time latency or delay (as will be shown in the next subsection). These figures show that any much larger result than 0.05 adversely affects stability, as can be seen when comparing Fig. 9, which becomes concave and has higher σ over all of the values of certainty and accuracy, with Fig. 5, which is convex and has lower σ over all of the values of certainty and accuracy. The intermediate figures show the slow deterioration in stability as $I_{\min}$ increases.

3.2. Reaction time latency

For an increase in the threshold of the minimum ignorance of 0.01 for each different figure, we have the following results for the reaction time latency (or delay Δ) in time units of the simulation scenario, with $I_{\min} = 0.01$ (Fig. 10), $I_{\min} = 0.02$ (Fig. 11), $I_{\min} = 0.03$ (Fig. 12), $I_{\min} = 0.04$ (Fig. 13), and $I_{\min} = 0.05$ (Fig. 14). Again this corresponds to 1000 Monte-Carlo runs.

These figures show that much smaller values of $I_{\min}$ than 0.01 result in too much rigidity when an allegiance changes, resulting in longer reaction time latency or

delay. This is clearly seen by the much higher values for the delays in the surface of Fig. 10 when compared to Fig. 14, over all of the values of certainty and accuracy.

This is particularly notable for low values of certainty and accuracy: the delay exceeds 25 time units (or more than half the total time to recover from an allegiance change) when compared to Fig. 14, where it is about 20 time intervals. The effect is also very noticeable for high accuracy values (towards the reader). The intermediate figures show the slow improvement in the reaction time latency as $I_{\min}$ increases.

4. ANALYSIS OF THE GRAPHICAL RESULTS IN ORDER TO IDENTIFY TRENDS

The large amount of graphical data shown in the previous section can be interpreted rather simply for the instability (in Subsection 4.1 for Figs. 5–9) and reaction time latency Δ (in Subsection 4.2 for Figs. 10–14). Although the trends discussed in the following subsections can be phrased rather straight-forwardly, the trends themselves are non-linear, as can be seen by close inspection of the figures in the previous section.

4.1. Instability

Analysis of the performance measure of stability (or instability) of the Thresholded-DS system can identify the following trends from our various simulations shown in the last section.

1. For a fixed value of certainty, the value of instability increases when the accuracy decreases.
2. For a fixed value of accuracy, the value of instability increases when the certainty increases.
3. For fixed values of certainty and accuracy, the value of the **instability increases** when the value of the total ignorance threshold $I_{\min}$ **is increased**.
4. A change in accuracy affects more the instability than the certainty does.
5. Lower values of instability (good) are achieved with higher accuracy and lower certainty, and vice versa.

4.2. Reaction time latency

Analysis of the performance measure of reaction time latency (or delay Δ) of the Thresholded-DS system can identify the following trends from our various simulations shown in the last section.

1. For a fixed value of certainty, the value of the delay increases when the accuracy decreases.
2. For a fixed value of accuracy, the value of the delay increases when the certainty decreases.
3. For fixed values of certainty and accuracy, the value of the **delay increases** when the value of the total ignorance threshold $I_{\min}$ **is decreased**.

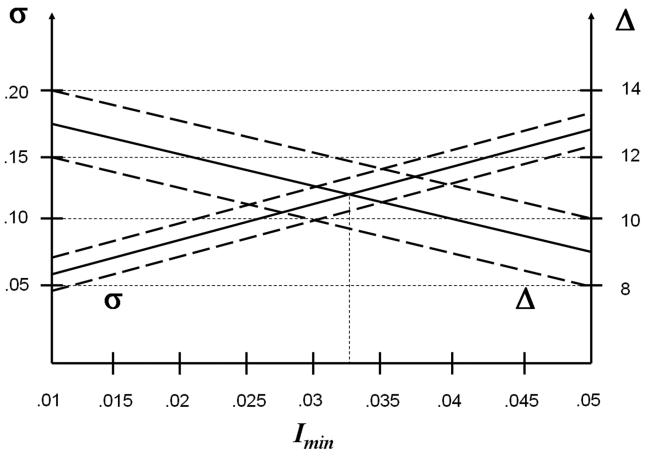


Fig. 15. Reaching a compromise for low σ and low Δ.

4. A change in accuracy affects more the delay than the certainty does.
5. Lower values of delay (good) are achieved with higher accuracy and higher certainty, and vice versa.

Points 3 in the above two lists clearly show that a compromise must be achieved when using Thresholded-DS between being responsive to any real change in the data, yet not being too responsive to fluctuations in the data, due to either poor sensor data certainty or fusion accuracy. In general, a high value for $I_{\min}$ will tend to respond to a stream of false reports rather quickly (bad) but will be very responsive to a real change in the data (good). A low value for $I_{\min}$ will provide excellent stability (good), but will react slowly to a real change in the data (bad).

The trends shown above are correct over the vast majority of the 16 points shown in the preceding Figs. 5–14. Only the exact values are shown in those figures, without the estimated errors from the Monte-Carlo runs.

The following Fig. 15 shows such a compromise as a function of $I_{\min}$, for a value of %Acc = %Cer = 80% with an estimate of errors, which cannot easily be portrayed in Figs. 5–14. The vertical axes represent σ (in % on the left) and Δ (in time units of the simulation) on the right, with the dashed lines showing approximate error bars given the limited number of Monte-Carlo runs (about 1000 runs). The figure shows that the interval $I_{\min} \in [0.025, 0.04]$ with a best value around 0.0325 can be selected.

5. CONCLUSIONS

This paper has provided performance measures of Thresholded-DS for various thresholds in terms of sensor data certainty and fusion accuracy to help designers assess beforehand, by varying the threshold appropriately, the achievable performance in terms of the estimated certainty and accuracy of the data that must be fused, i.e., an operating point for the application.

The threshold that the designers can choose according to figures similar to Fig. 15 depends on appropriate

definitions for sensor certainty and latency (or delay) for their given application. Reasonable values were chosen here for an ESM application. In real applications, one should have an independent way of assessing the sensor certainty and the fusion accuracy in real-time. The Monte-Carlo runs provide the operating points, but it has to be assumed that the user can assess these operating points by monitoring the performance of the sensor as the mission develops (for example on well-isolated targets), and has calibrated the performance of the association mechanism in various conditions, which any manufacturer of such software should have done.

The performance measures are twofold, first in terms of stability when fused data are consistent, and second in terms of the latency in the response time when an abrupt change occurs in the data to be fused. These two performance measures must be traded off against each other, which is the reason why the performance curves will be very helpful for designers of identification fusion using Thresholded-DS.

REFERENCES

[1] M. Bauer
Approximation algorithms and decision making in the Dempster-Shafer theory of evidence—An empirical study.
International Journal of Approximate Reasoning, **17**, 2–3 (1997), 217–237.

[2] D. Boily and P. Valin
Truncated Dempster-Shafer optimization and benchmarking.
In *Sensor Fusion: Architectures, Algorithms, and Applications IV*, SPIE Aerosense 2000, vol. 4051, 237–246.

[3] D. Boily and P. Valin
Optimization and benchmarking of truncated Dempster-Shafer for airborne surveillance.
In NATO Advanced Study Institute on Multisensor and Sensor Data Fusion, Kluwer Academic Publishers, NATO Science Series, II, Mathematics Physics and Chemistry, vol. 70, 617–624.

[4] É. Bossé and M-A. Simard
Managing evidential reasoning for identity information fusion.
Optical Engineering, special section on sensor fusion, **37**, 2 (Feb. 1998), 391–400.

[5] A. Dempster
Upper and lower probabilities induced by multivalued mapping.
Ann. Math. Statist., **38** (1967), 325–339.

[6] P. Djiknavorian, D. Grenier and P. Valin
Analysis of information fusion combining rules under the DSm theory using ESM inputs.
In 10th International Conference on Information Fusion, 2007.

[7] P. Djiknavorian, P. Valin and D. Grenier
Dezert-Smarandache theory applied to highly conflicting reports for identification and recognition—Illustrative example of ESM associations in dense Environments.
DRDC Valcartier TR 2008-537, 34 p.

[8] P. Djiknavorian, P. Valin and D. Grenier
Fusion of ESM allegiance reports using DSmT.
In Advances and Applications of DSmT for Information Fusion, Collected Works edited by F. Smarandache and J. Dezert, vol. 3, American Research Press, 2008, 503–518.

[9] NATO Standardization Agreement (STANAG)
NATO standard identity description structure for tactical use.
STANAG 1241 (ed. 4), October 16, 1996.

[10] G. Shafer
A Mathematical Theory of Evidence.
Princeton University Press, 1976.

[11] M-A. Simard, P. Valin and E. Shahbazian
Fusion of ESM, radar, IFF and other attribute information for target identity estimation and a potential application to the Canadian Patrol Frigate.
In AGARD 66th Symposium on Challenge of Future EW System Design, AGARD-CP-546, May 1994, 14.1–14.18.

[12] F. Smarandache and J. Dezert (Eds.)
Advances and applications of DSmT for information fusion.
vol. 1, American Research Press, 2004.

[13] F. Smarandache and J. Dezert (Eds.)
Advances and Applications of DSmT for Information Fusion.
vol. 2, American Research Press, 2006.

[14] B. Tessem
Approximations for efficient computation in the theory of evidence.
Artificial Intelligence, **61** (June 1993), 315–329.

Pierre Valin received a B.Sc. honours physics (1972) and a M.Sc. degree (1974) from McGill University, then a Ph.D. in theoretical high energy physics from Harvard University (1980), under the supervision of the 1979 Nobel Laureate Dr. Sheldon Glashow.

He was faculty lecturer at McGill and Associate Professor of Physics in New Brunswick at Moncton and Fredericton before joining Lockheed Martin Canada in 1993 (then called Paramax), as a Principal Member of R&D. In 2004, he became a defence scientist at Defence R&D Canada (DRDC) at Valcartier, where he currently leads a research group in Future C2 Concepts & Structures. He is thrust leader for Air Command at DRDC since 2007.

He has been particularly active in the International Society of Information Fusion (ISIF) though the organization of FUSION 2001 and 2007. He has been ISIF board member since 2003, VP-membership since 2004, and was president in 2006. He is also an associate editor for the *Journal of Advances in Information Fusion* (JAIF).

Dr. Valin's interests focus mainly on the following topics: Multi-Sensor Data Fusion (MSDF) requirements and design, C2 systems, algorithmic benchmarking, use of a priori information databases, imagery classifiers (EO/IR and SAR) and their fusion, neural networks, fuzzy logic, information processing and uncertainty representation, reasoning techniques for recognition and identification (Bayes, Dempster-Shafer and Dezert-Smarandache), SAR image processing, Network Centric Warfare, distributed information fusion, dynamic resource management, as well as theoretical and mathematical physics.

Pascal Djiknavorian obtained a B.Eng. in computer engineering and a certificate in business administration in 2005 from Laval University. He also completed in 2008 an M.Sc. in electrical engineering on information fusion within the Dezert-Samarandache theory framework applied to ESM reports under STANAG 1241.

He is currently a Ph.D. student in information fusion at Laval University and is supervised by Professor Dominic Grenier and Professor Pierre Valin. He has ten publications in books, journals, and conference proceedings. His research interests include evidential theory, DSm theory, approximations algorithms, optimization methods.

Éloi Bossé received the B.A.Sc., M.Sc. from Laval University in 1981 in electrical engineering and Ph.D. degrees in 1990 from Ottawa/Carleton and Laval Universities.

In 1981 he joined the Communications Research Centre, Ottawa, Canada, where he worked on high resolution spectral analysis and radar tracking in multipath. In 1988 he was transferred to the Defence Research Establishment Ottawa (DREO) and in 1992 to the Defence Research and Development Canada Valcartier (DRDC Valcartier) where he worked on information fusion and decision support. He is an adjunct professor at Laval University (Québec), University of Calgary and McMaster University (Hamilton). He now heads the C2 Decision Support Systems Section at DRDC Valcartier. He was the executive chair of the 10th International Conference on Information Fusion (FUSION 2007), held in July 2007 in Québec City, Canada.

Dr. Bossé has published over 160 papers in journals, conference proceedings and technical reports. He is coauthor of the following book on information fusion: *Concepts, Models, and Tools for Information Fusion*, Artech House, Norwood, Dec. 2006, 352 p. (NAMRAD Principals 2006 achievement AWARD).

Exact Association Probability for Data with Bias and Features

JAMES P. FERRY

A crucial prerequisite to data fusion is data association: i.e., the specification of which data arise from the same source. The Bayesian approach to association pioneered by Mori and Chong is based on principled probability formulas, which thus provide reliable confidence estimates for association hypotheses, in contrast to approaches that rely on costs which can only be heuristically transformed into probabilities. This paper extends the Bayesian approach in several ways. It presents a general derivation of association probability between any number of sensors for arbitrary data types, then derives specific results for kinematic and non-kinematic cases. The kinematic case includes bias and is novel in three ways. First, it is a proper Bayesian approach to bias which integrates over all bias hypotheses rather than selecting one. Second, it handles bias on an arbitrary number of sensors. Third, the formula is exact: previous treatments of even the unbiased case involve an integral approximation which is not needed here. The treatment of features allows for several complex phenomena, including feature behavior which depends on object type, and noisy and/or missing feature data. A rigorous verification procedure is used to demonstrate that the implementation of these formulas produces correct probabilities.

Manuscript received December 04, 2008; revised August 20, 2009; released for publication December, 13, 2009.

Refereeing of this contribution was handled by Shozo Mori.

Authors' address: Metron, Inc., Reston, VA, E-mail: ferry@metsci.com.

1. INTRODUCTION

The data association problem arises whenever multiple sensors are trained on a common region containing multiple objects, which, in turn produce multiple measurements on each sensor. Having multiple views provides more information about the state of the objects in the region, provided the sensor data can be fused correctly. In order to fuse the data, however, it is necessary to know which measurements on different sensors arose from the same object. This is the data association problem, and it is roughly forty years old [24, 25]. In many applications, these "measurements" are not raw sensor measurements, but the posterior state estimates given by a single-sensor tracker such as a Kalman filter, and one speaks of "measurement-to-track" or "track-to-track" associations. Solving the association problem is necessary in Multiple Hypothesis Tracking [23], and much effort has gone into the development of algorithms to find the best association given track estimates with Gaussian error covariances [4]. The key difference between raw measurement data and track posteriors, however, is that the former lack inter-sensor correlation (for given object states). Such correlations are important in the track case, however, particularly if the posterior distributions have been influenced by previous inter-sensor data fusion. Indeed, even the posteriors of single-sensor trackers are correlated due to common process noise [3]. The scope of this paper is limited to situations in which such inter-sensor correlations are absent or have been compensated for.

The association problem was originally formulated in terms of *costs* with statistically motivated definitions —one found or devised a credible cost function and used it to seek low-cost associations. In 1990 Chong *et al.* introduced a more rigorous framework for assessing the quality of associations [7]. Mori and Chong extended this work in a series of papers in the early 2000s [16, 17, 18, 19, 20]. For the important case of two sensors with Gaussian kinematic data, they replaced the heuristic framework of costs with a rigorous, Bayesian reformulation of the association problem. In doing so, they gave a meaningful definition of the *probability* of an association, and argued that rather than seeking the association with minimal cost, one should seek the MAP association—i.e., the one with Maximal *A posteriori* Probability. In practice, the MAP method looks similar to the older method. The probabilities can be converted to costs, and one ends up computing the same quantities as before with one subtle difference: Mori and Chong showed that the correct cost threshold is not constant, but depends on the covariance matrices of the two measurements involved. Hence it came to be known as an *adaptive* threshold.

In 2002, Stone *et al.* generalized the work of Mori and Chong to non-kinematic data with the XMAP (eXtended Maximal *A posteriori* Probability) method [27]. In this and later work [6, 9, 11], association probabil-

ity formulas have been derived that take into account continuous and non-continuous data types beyond the purely kinematic, primarily for the case of two sensors.

This paper encompasses and extends the previous XMAP work, beginning with a general derivation for association probability in an abstract setting in Section 2. Like [18], this derivation includes an arbitrary prior on the number of objects and (in Appendix A) false alarms. Its novel aspects include a dependence on systematic errors (such as bias or covariance inflation) and a correlated prior on object state. In Section 3, this abstract derivation is applied to cases comprising kinematic and non-kinematic data types, and the association probability formula is decomposed. The kinematic component is dealt with in Section 4. The key contribution of this section is its treatment of bias, which improves on previous work in three ways: (1) rather than removing "the bias" (i.e., some particular bias hypothesis), it performs the proper Bayesian operation of integrating over all bias hypotheses; (2) it holds for an arbitrary number of sensors; and (3) it is exact—a certain integral approximation typically made even in the non-bias case is circumvented here. Section 5 demonstrates how to handle non-kinematic data in fairly complex cases, such as when the feature distributions and detection probabilities vary with object type. It also shows how to deal with missing data, and provides a robust and general method for handling noisy features. Finally, Section 6 works through an example in detail and demonstrates that the meaningful, exact probabilities produced by XMAP can be used to verify the formulas and their implementation to high precision.

2. ASSOCIATION PROBABILITY DERIVATION

This section derives a general formula for association probability in an abstract space. Working in an abstract space allows us to handle arbitrary types of measurement data in a consistent manner, whether it be traditional kinematic data, or, say, the messy output of a feature extractor which combines real-valued data with object classification calls and status flags. Thus, the bulk of this paper may be viewed as applying the general Theorem 2.4 below to special cases. A technical detail to bear in mind with this abstract treatment is that integrating over an abstract space requires one to specify a *measure* over the space. There will be no need, in this paper, to use measures other than Lebesgue measure for continuous data and counting measure for discrete data (which converts integrals to sums), so we will assume that the measure is clear from context.

We begin with a simple result for the probability density of getting a specific array of measurements on a single sensor s given the states of the objects that produced them. This result depends on the measurement likelihood function $L^s(z \mid x, \beta)$, which specifies the probability density of the measurement z arising given that the object which produced it was detected and was in state x, and that a systematic error β is acting on all measurements on sensor s. It depends also on the detection probability $P_{\mathrm{D}}^s(x)$ for an object in state x, and we use the notation $Q_{\mathrm{D}}^s(x) = 1 - P_{\mathrm{D}}^s(x)$ to denote the non-detection probability. This systematic error β may represent any measurement error process that acts on all measurements on a sensor at once, such as a translational bias or covariance inflation.

Let $\mathbf{z}^s = (z_i^s)_{i=1}^{n^s}$ denote an array of the n^s measurements on sensor s at some fixed time, and $\mathbf{x} = (x_j)_{j=1}^{n}$ denote the array of states of the n objects in scene. We let $J = \{1, 2, \ldots, n\}$ denote the set of all objects, and J_{D}^s denote the subset of objects detected on sensor s. We use the mapping $a^s : J_{\mathrm{D}}^s \to \{1, 2, \ldots, n^s\}$ to specify which object produced which measurement. We assume there are no false alarms (the false-alarm case is addressed in Appendix A), that there are no split or merged measurements, and that all permutations of measurement labels are equally likely. With these assumptions, we obtain the following preliminary result.

LEMMA 2.1 *The probability density of the measurement array $\mathbf{z}^s$ arising according to the mapping a^s given the object state array $\mathbf{x}$ and the systematic error β^s is*

$$\Pr(\mathbf{z}^s, a^s \mid \mathbf{x}, \beta^s) = \frac{1}{n^s!} \prod_{j \in J_{\mathrm{D}}^s} P_{\mathrm{D}}^s(x_j) L^s(z_{a^s(j)}^s \mid x_j, \beta^s) \times \prod_{j \in J \setminus J_{\mathrm{D}}^s} Q_{\mathrm{D}}^s(x_j). \tag{2.1}$$

PROOF Given the object state array $\mathbf{x}$, the probability $\Pr(J_{\mathrm{D}}^s \mid \mathbf{x})$ of the subset of detected objects being precisely J_{D}^s is the product of $P_{\mathrm{D}}^s(x_j)$ over $j \in J_{\mathrm{D}}^s$ times the product of $Q_{\mathrm{D}}^s(x_j)$ over $j \in J \setminus J_{\mathrm{D}}^s$. Given J_{D}^s (and $\mathbf{x}$), each of the possible $n^s!$ mappings a^s are equally likely, so $\Pr(a^s, J_{\mathrm{D}}^s \mid \mathbf{x})$, which is identical to $\Pr(a^s \mid \mathbf{x})$ because a^s determines J_{D}^s, can be expressed $\Pr(a^s \mid \mathbf{x}) = \Pr(a^s \mid J_{\mathrm{D}}^s, \mathbf{x}) \Pr(J_{\mathrm{D}}^s \mid \mathbf{x}) = \Pr(J_{\mathrm{D}}^s \mid \mathbf{x}) / n^s!$. The probability density of the measurement array $\mathbf{z}^s$ given a^s, $\mathbf{x}$, and β^s is the product of the individual likelihood functions $L^s(z_i^s \mid x_j, \beta^s)$, where $i = a^s(j)$, over all $j \in J_{\mathrm{D}}^s$. Equation (2.1) now follows from $\Pr(\mathbf{z}^s, a^s \mid \mathbf{x}, \beta^s) = \Pr(\mathbf{z}^s \mid a^s, \mathbf{x}, \beta^s) \Pr(a^s \mid \mathbf{x})$.

It is straightforward to generalize Lemma 2.1 from a single sensor to a set of sensors S. We let $\mathbf{z} = (\mathbf{z}^s)_{s \in S}$ denote the array of all individual measurement arrays $\mathbf{z}^s$, and $\boldsymbol{\beta} = (\beta^s)_{s \in S}$ denote the array of all systematic errors. We assume that the measurement process is independent between sensors, bearing in mind that this limits the applicability to tracking unless the inter-sensor dependence can be compensated for. With multiple sensors it is convenient to express the information contained in the mappings $\{a^s\}_{s \in S}$ in terms of

$$a(j) = \{(s, i) : a^s(j) = i\} \qquad \text{and} \qquad \bar{a}(j) = \{s : j \in J_{\mathrm{D}}^s\}. \tag{2.2}$$

The function $\bar{a}$ gives the set of sensors which detect each object j, whereas a gives the measurement indices of each detection too. With this notation, we have the following corollary of Lemma 2.1.

COROLLARY 2.2 *The probability density of the measurement arrays* $\mathbf{z}$ *arising according to the function* a *given the object state array* $\mathbf{x}$ *and the systematic errors* $\boldsymbol{\beta}$ *is*

$$\Pr(\mathbf{z}, a \mid \mathbf{x}, \boldsymbol{\beta}) = \left(\prod_{s \in S} \frac{1}{n^s!} \right) \prod_{j \in J} \times \left(\prod_{(s,i) \in a(j)} P_{\mathrm{D}}^s(x_j) L^s(z_i^s \mid x_j, \beta^s) \prod_{s \notin \bar{a}(j)} Q_{\mathrm{D}}^s(x_j) \right). \tag{2.3}$$

PROOF The assumption of independence between the measurement processes implies that $\Pr(\mathbf{z}, a \mid \mathbf{x}, \boldsymbol{\beta})$ may be obtained as the product of (2.1) over $s \in S$. The result (2.3) is merely a rearrangement of this.

We now seek to eliminate the dependence on the state array $\mathbf{x}$. To do so, we need an expression for the prior distribution of the state. The expression introduced below is novel, and requires some discussion of (a) the shortcomings of the usual approach, (b) how the problem should be solved in principle, and (c) the compromise used here. Although the derivation in this section applies to an arbitrary state space, the problematic case is the traditional, kinematic one, so we shall think of $\mathbf{x}$ as a kinematic quantity for this discussion.

The usual assumption about the prior distribution of $\mathbf{x}$ when bias is absent is that its components x_j are i.i.d., each distributed according to some known distribution p^0. The distribution $p^0(x)$ of a single object state x is then taken to be uniform over some finite region of space X^0 [17]. An appropriate volume V for X^0 may be estimated from the data, and $p^0(x)$ modeled as $I_{X^0}(x)/V$, where $I_{X^0}(x)$ is the indicator function for the region X^0. The precise location of the region X^0 does not matter because $p^0(x)$ is later approximated by the constant $1/V$ in the integrals where it appears. This approximation is valid because the measurement errors are typically much smaller than X^0, and this formulation, which leads to the *adaptive threshold* works well in practice [26].

Bias errors may be larger than X^0, however, so the approximation $p^0(x) \approx 1/V$ fails in the bias case. Maintaining $p^0(x)$ as $I_{X^0}(x)/V$ yields intractable integrals, so it is natural to consider a Gaussian model for $p^0(x)$. In this case, however, the precise location of the Gaussian's peak in state space must be estimated from the measurements, while correcting for the (unknown) biases, and the resulting formulation becomes messy and *ad hoc*. Indeed, it must be *ad hoc* because it involves estimating the location of the Gaussian's peak *from the data*—a clear violation of Bayesian methodology.

One is led to such violations because of a faulty initial assumption: that the prior distribution of $\mathbf{x}$ is well modeled as the product of $p^0(x_j)$ for some *known* distribution p^0. The characteristic size V and nominal *center* ξ of this prior distribution are usually both unknown. However, it is reasonable to model the states x_j as being conditionally i.i.d. given V and ξ, and then to specify priors on V and ξ. This leads to additional integrals over V and ξ. However, one may argue, for example, that the weight in the integrand of the integral over V is concentrated in the region of V-space that is reinforced by the measurement data, so evaluating the integrand at a single point V^* (estimated from the data) amounts to a reasonable approximation of the integral. This provides a Bayesian justification for an otherwise *ad hoc* procedure. However, in the bias case there is no single value of ξ where the integrand is concentrated: this center location depends on the unknown biases β^s. Rather than replace ξ by this function of the biases, it is fairly easy to retain it and perform the integrals exactly. Doing the same for V, however, (which, in the Gaussian case is actually an entire covariance matrix) is too difficult, so we will estimate it from the data, relying on the argument above for justification.

We therefore assume the following prior distribution on the state array $\mathbf{x}$ (which we now resume treating as abstract rather than kinematic):

$$p^0(\mathbf{x} \mid n, \xi) = \prod_{j \in J} p^0(x_j \mid \xi). \tag{2.4}$$

Multiplying (2.3) by $p^0(\mathbf{x} \mid n, \xi)$ and integrating over $\mathbf{x}$ results in a product of integrals over x_j for each object $j \in J$. These have the form

$$P^\alpha(\mathbf{z} \mid \boldsymbol{\beta}, \xi) = \int p^0(x \mid \xi) \prod_{(s,i) \in \alpha} P_{\mathrm{D}}^s(x) L^s(z_i^s \mid x, \beta^s) \times \prod_{s \notin \bar{\alpha}} Q_{\mathrm{D}}^s(x) dx \tag{2.5}$$

for $\alpha = a(j)$. For the special case of α containing only the single measurement (s,i) we will use the notation $P^s(z_i^s \mid \boldsymbol{\beta}, \xi)$ in lieu of $P^\alpha(\mathbf{z} \mid \boldsymbol{\beta}, \xi)$. Let J_{D} denote the subset of objects in J detected on some sensor, and $n_{\mathrm{D}} = |J_{\mathrm{D}}|$ be the number of detected objects. For $j \notin J_{\mathrm{D}}$, (2.5) takes a particularly simple form:

$$q = \int p^0(x \mid \xi) \prod_{s \in S} Q_{\mathrm{D}}^s(x) dx. \tag{2.6}$$

Here we are imposing the condition that q is independent of ξ. This will happen automatically later when we stipulate in (3.18) that $P_{\mathrm{D}}^s(x)$ be independent of the kinematic component of x. This stipulation is not realistic: $P_{\mathrm{D}}^s(x)$ can vary greatly with aspect angle and range. However, the case in which $P_{\mathrm{D}}^s(x)$ has kinematic dependence makes the calculations in Section 4 too complicated, though it would be a suitable topic for future work.

COROLLARY 2.3 *The probability density of the measurement arrays* $\mathbf{z}$ *arising according to the function* a *given*

the number of objects n, the systematic errors $\boldsymbol{\beta}$, and the center ξ of the prior region of state space is

$$\Pr(\mathbf{z}, a \mid n, \boldsymbol{\beta}, \xi) = \frac{q^{n-n_{\mathrm{D}}}}{\prod_{s\in S} n^s!} \prod_{j\in J_{\mathrm{D}}} P^{a(j)}(\mathbf{z} \mid \boldsymbol{\beta}, \xi). \quad (2.7)$$

PROOF This is obtained by multiplying (2.3) by (2.4) and integrating over $\mathbf{x}$.

The function a defined in (2.2) maps each object to the set of measurements it produces. An association $[a]$ is defined to be the collection of these sets, $[a] = \{a(j) : j \in J_{\mathrm{D}}\}$. There are exactly $n!/(n-n_{\mathrm{D}})!$ functions a' for which $[a'] = [a]$, all of which are equally probable, so the association probability $\Pr(\mathbf{z}, [a] \mid n, \boldsymbol{\beta}, \xi)$ is $n!/(n-n_{\mathrm{D}})!$ times $\Pr(\mathbf{z}, a \mid n, \boldsymbol{\beta}, \xi)$. We may now eliminate the dependence on the total number of objects n. The prior probability for the number of objects being n is denoted $\rho^0(n)$, so

$$\Pr(\mathbf{z}, [a] \mid \boldsymbol{\beta}, \xi) = \frac{\gamma^0(n_{\mathrm{D}})}{\prod_{s\in S} n^s!} \prod_{\alpha\in[a]} P^{\alpha}(\mathbf{z} \mid \boldsymbol{\beta}, \xi), \quad (2.8)$$

where

$$\gamma^0(n_{\mathrm{D}}) = \sum_{n=n_{\mathrm{D}}}^{\infty} \rho^0(n) \frac{n!}{(n-n_{\mathrm{D}})!} q^{n-n_{\mathrm{D}}}. \quad (2.9)$$

We denote the prior on the systematic errors $P^0(\boldsymbol{\beta})$, and the prior on the center, $P^0_{\Xi}(\xi)$. When integrating (2.8), the key quantity to compute is

$$F(\mathbf{z}, [a]) = \int\int \prod_{\alpha\in[a]} P^{\alpha}(\mathbf{z} \mid \boldsymbol{\beta}, \xi) P^0(\boldsymbol{\beta}) P^0_{\Xi}(\xi) d\boldsymbol{\beta}\, d\xi. \quad (2.10)$$

Finally, we introduce the following ratios of γ^0 and F to their values for the null association a_0, which assigns each of the n_T measurements in $\mathbf{z}$ to a distinct object:

$$g([a]) = \frac{\gamma^0(n_{\mathrm{D}})}{\gamma^0(n_T)} \qquad \text{and} \qquad G(\mathbf{z}, [a]) = \frac{F(\mathbf{z}, [a])}{F(\mathbf{z}, [a_0])}. \quad (2.11)$$

THEOREM 2.4 *The probability of the association $[a]$ given the measurements $\mathbf{z}$ is*

$$\Pr([a] \mid \mathbf{z}) = \Pr([a_0] \mid \mathbf{z}) g([a]) G(\mathbf{z}, [a]). \quad (2.12)$$

PROOF From (2.8) we observe that joint probability density $\Pr(\mathbf{z}, [a])$ is $F(\mathbf{z}, [a])$ times $\gamma^0(n_{\mathrm{D}})$ divided by the product of the $n^s!$. The conditional probability $\Pr([a] \mid \mathbf{z})$ of an association given the measurement data is $\Pr(\mathbf{z}, [a])/\Pr(\mathbf{z})$. Dividing $\Pr([a] \mid \mathbf{z})$ by the normalization constant $\Pr([a_0] \mid \mathbf{z})$ yields (2.12).

The key to computing the association probability is evaluating $F(\mathbf{z}, [a])$ (or its normalization $G(\mathbf{z}, [a])$). This is the topic of Sections 3–5. The combinatorial factor $g([a])$ encapsulates the effect of the prior distribution of the number of objects. Formulas for it are given in Appendix B. Although the derivation in this section assumed there are no false alarms, Appendix A demonstrates that the effect of false alarms may be included by modifying the factor $g([a])$—no change to the definition of $G(\mathbf{z}, [a])$ is necessary.

3. SIMPLIFICATION FOR SPECIAL CASES

Theorem 2.4 gives a general formula for association probability, but requires the evaluation of the integrals in (2.5) and (2.10). With complicated data types, evaluating these integrals is not as simple as it may appear. Therefore we demonstrate how the problem simplifies in various special cases. Section 3.1 gives a formula much simpler than (2.12) which holds when there is no dependence on $\boldsymbol{\beta}$ or ξ. Section 3.1.1 specializes this further to the two-sensor, kinematic case, connecting the general XMAP formulation presented here to the original MAP formulation of Mori and Chong [16, 17]. Finally, Section 3.2 demonstrates how to decompose the problem into kinematic and non-kinematic components when both data types are present. The kinematic component is then evaluated explicitly in Section 4, and the non-kinematic component in Section 5.

3.1. Simplification in the Absence of Systematic Error

When the systematic errors $\boldsymbol{\beta}$ and the center ξ are known, we may assume each to be zero (by suitably transforming the data $\mathbf{z}$): i.e., $P^0(\boldsymbol{\beta}) = \delta(\boldsymbol{\beta})$, and $P^0_{\Xi}(\xi) = \delta(\xi)$. Letting $P^{\alpha}(\mathbf{z})$ and $P^s(z^s_i)$ denote $P^{\alpha}(\mathbf{z} \mid \mathbf{0}, 0)$ and $P^s(z^s_i \mid 0, 0)$, respectively, we define

$$R^{\alpha}(\mathbf{z}) = \frac{P^{\alpha}(\mathbf{z})}{\prod_{(s,i)\in\alpha} P^s(z^s_i)}, \quad (3.1)$$

which is the ratio of the probability density of the measurements in α arising from a single object to the probability density that each arises from a different object (aside from a factor which accounts for the different number of detected objects in the two cases—this is embedded in $g([a])$).

THEOREM 3.1 *When the systematic errors $\boldsymbol{\beta}$ and the center ξ are known, the probability of the association $[a]$ given the measurements $\mathbf{z}$ is*

$$\Pr([a] \mid \mathbf{z}) = \Pr([a_0] \mid \mathbf{z}) g([a]) \prod_{\alpha\in[a]^+} R^{\alpha}(\mathbf{z}), \quad (3.2)$$

where $[a]^+$ denotes the subset of those $\alpha \in [a]$ with at least two measurements.

PROOF Because $R^{\alpha}(\mathbf{z}) = 1$ when $|\alpha| = 1$, the product over $\alpha \in [a]^+$ in (3.2) may be extended to $\alpha \in [a]$. This product equals $F(\mathbf{z}, [a])/F(\mathbf{z}, [a_0])$ when we set $P^0(\boldsymbol{\beta}) = \delta(\boldsymbol{\beta})$ and $P^0_{\Xi}(\xi) = \delta(\xi)$ in (2.10).

3.1.1. Two-sensor kinematic case

To recover the original MAP result [16, 17], we begin with Theorem 3.1 and make four further simplify-

ing assumptions: first, that $\rho^0(n)$ is Poisson distributed; second, that there are only two sensors; third, that the detection probabilities on each sensor are constant; and fourth, that the data is purely kinematic, with Gaussian measurement error distributions. The two-sensor kinematic case without bias is important because it admits a computationally efficient solution. One first constructs a cost matrix whose entries c_{ij} are the Mahalanobis distances between measurement i on sensor 1 and measurement j on sensor 2. Then a *cost threshold* is subtracted from each c_{ij}. Finally, one finds the association with minimal total cost using, for example, the JVC algorithm [8, 13], and, if desired, iterates this process using Murty's algorithm [21] to get the k best associations. (When there are more than two sensors, however, finding the association with least cost is known to be NP-hard. Approximate methods have been employed based on Lagrangian relaxation [22] or on stitching together solutions to pairwise problems. These approaches are compared in [2].)

The effect of letting $\rho^0(n)$ be Poisson distributed is discussed in Appendix B. It allows us to replace (3.2) with the purely multiplicative form (B.4). Specializing to two sensors simplifies matters further because in this case every $\alpha \in [a]^+$ has the form $\alpha = \{(1,i),(2,j)\}$. We may re-write this more compactly as $\alpha = (i,j)$. Thus the two-sensor version of (B.4) may be written

$$\Pr([a] \mid \mathbf{z}) = \Pr([a_0] \mid \mathbf{z}) \prod_{(i,j)\in[a]^+} \tilde{R}_{ij}(\mathbf{z}), \tag{3.3}$$

where

$$\tilde{R}_{ij}(\mathbf{z}) = \frac{\tilde{P}^{12}(z_i^1, z_j^2)}{\tilde{P}^1(z_i^1)\tilde{P}^2(z_j^2)}. \tag{3.4}$$

The functions $\tilde{P}^s$ and $\tilde{P}^{12}$ are special cases of (2.5), with β and ξ eliminated, P_{D}^s set to a constant, and factors of ν in (B.5) included:

$$\tilde{P}^s(z_i^s) = \nu P_{\mathrm{D}}^s Q_{\mathrm{D}}^{3-s} \int p^0(x) L^s(z_i^s \mid x) dx \qquad \text{and} \tag{3.5}$$

$$\tilde{P}^{12}(z_i^1, z_j^2) = \nu P_{\mathrm{D}}^1 P_{\mathrm{D}}^2 \int p^0(x) L^1(z_i^1 \mid x) L^2(z_j^2 \mid x) dx. \tag{3.6}$$

(To incorporate false alarms, replace the factor $\nu P_{\mathrm{D}}^s Q_{\mathrm{D}}^{3-s}$ in (3.5) with $\nu P_{\mathrm{D}}^s Q_{\mathrm{D}}^{3-s} + \nu_{\mathrm{FA}}^s$, where ν_{FA}^s is the expected number of false alarms on sensor s: see (B.18).)

The integrals in (3.5) and (3.6) are simple to evaluate when L^s has a Gaussian distribution. We use the notation

$$\mathcal{N}(x;\mu,V) = \frac{1}{\sqrt{|2\pi V|}} \exp\left(-\frac{1}{2}(x-\mu)^T V^{-1}(x-\mu)\right) \tag{3.7}$$

for a Gaussian in x with mean μ and covariance matrix V. Specializing to the standard kinematic case, we let the value z_i^s of measurement i on sensor s have the form $z_i^s = (y_i^s, V_i^s)$, where y_i^s the state estimate, and V_i^s is the error estimate on y_i^s. We then stipulate that

$$L^s(z_i^s \mid x) = \mathcal{N}(y_i^s; x, V_i^s) P_V(V_i^s). \tag{3.8}$$

Essentially this means that y_i^s has a Gaussian distribution centered at the true state x, with covariance matrix given by V_i^s. There is an additional complication, however: the estimated covariance matrix V_i^s is part of the data, so its distribution must be modeled as well. The simplest assumption is that it is independent of x (and of y_i^s). In this case the precise form of the distribution P_V does not matter: it drops out of the calculation. (More sophisticated treatments are certainly possible: for example, there may be systematic over- or under-reporting of covariance, or the size of the covariance matrix itself may yield object-type information, in which case P_V would be modeled to have a dependence on the object type component of the state.)

Following [7], we let the prior distribution on an object's state x be constant over some region X^0:

$$p^0(x) = \frac{1}{\mathrm{Vol}(X^0)} I_{X^0}(x), \tag{3.9}$$

where $I_{X^0}(x)$ is equal to 1 for $x \in X^0$, and 0 otherwise. Given the above assumptions, (3.4) may be written

$$\tilde{R}_{ij}(\mathbf{z}) = \mathcal{N}(y_i^1; y_j^2, V_i^1 + V_j^2) \frac{\mathrm{Vol}(X^0)}{\nu q} \times \frac{\int_{X^0} \mathcal{N}(x;\mu_{ij},W_{ij}) dx}{\int_{X^0} \mathcal{N}(x;y_i^1,V_i^1) dx \int_{X^0} \mathcal{N}(x;y_j^2,V_j^2) dx}, \tag{3.10}$$

where $q = Q_{\mathrm{D}}^1 Q_{\mathrm{D}}^2$, and μ_{ij} and W_{ij} are given by

$$\begin{aligned} \mu_{ij} &= y_i^1 + V_i^1 (V_i^1 + V_j^2)^{-1} (y_j^2 - y_i^1) \qquad \text{and} \\ W_{ij} &= V_i^1 (V_i^1 + V_j^2)^{-1} V_j^2. \end{aligned} \tag{3.11}$$

Assuming the Gaussians in the integrals in (3.10) have most of their weight within X^0, each integral is approximately 1. This yields the following *cost* of associating i and j:

$$\begin{aligned} c_{ij} &= -2 \log \tilde{R}_{ij}(\mathbf{z}) \\ &= (y_i^1 - y_j^2)^T (V_i^1 + V_j^2)^{-1} (y_i^1 - y_j^2) - A_{ij}, \end{aligned} \tag{3.12}$$

where

$$A_{ij} = 2 \log \mathrm{Vol}(X^0) - \log |2\pi (V_i^1 + V_j^2)| - 2 \log(\nu q). \tag{3.13}$$

The cost c_{ij} is thus seen to be the Mahalanobis distance $(y_i^1 - y_j^2)^T (V_i^1 + V_j^2)^{-1} (y_i^1 - y_j^2)$ between the measurement pair (i,j) minus Mori and Chong's adaptive threshold A_{ij} [16, 17, 18]. Traditionally, a variety of methods had been used to set this threshold [4, 5]. Although the main historical significance of Mori and Chong's work is the introduction of a rigorous Bayesian

approach to association, the more immediate impact was the introduction of a threshold A_{ij} which has been shown to be superior to the previous, fixed thresholds [26].

There is a minor flaw with (3.13), however: it fails when some V_i^s are large relative to the region X^0 because this violates the assumption that allowed the integrals in (3.10) to be approximated as 1. Such measurements must be preprocessed out as unassociatable when using (3.13). Additional, non-kinematic data may render such measurements associatable, however, so it is preferable to modify (3.13) to be robust to any input V_i^s. A complicated method for doing this is given in [9], but here a much simpler method is given. When the covariance of a Gaussian being integrated is large compared to X^0, the integral may be approximated as the Gaussian's peak value times $\mathrm{Vol}(X^0)$. This leads to the following robust modification of (3.13), which has the heuristic interpretation of limiting the uncertainty of an object's location to X^0 even if $V_i^s \to \infty$:

$$A_{ij} = \max(2\log \mathrm{Vol}(X^0) - \log|2\pi(V_i^1 + V_j^2)|, 0) - 2\log(\nu q). \tag{3.14}$$

To use the adaptive threshold (3.14) one needs values for q, ν, and $\mathrm{Vol}(X^0)$. Section 2 notes that when such parameters are unknown, the proper Bayesian procedure is to give them a prior distribution and integrate them out of the problem. In practice, however, setting values of P_{D}^s that are even approximately correct produces better results than those obtained using the traditional, fixed threshold [26]. These values are used to compute $q = (1 - P_{\mathrm{D}}^1)(1 - P_{\mathrm{D}}^2)$ and, using (B.6), $\nu = (n^1 + n^2)/(P_{\mathrm{D}}^1 + P_{\mathrm{D}}^2)$.

It remains to estimate $\mathrm{Vol}(X^0)$. Following [9], we do this by first estimating the covariance of the location data y_i^s for all measurements. In the absence of sensor bias, the unbiased covariance estimator $\hat{V}$ may be computed as follows. Collect all n_T measurement positions y_i^s on all sensors into a single array with elements y_i for $i = 1, 2, \ldots, n_T$. Then

$$\hat{V} = \frac{1}{n_T - 1}\sum_{i=1}^{n_T}(y_i - \hat{y})(y_i - \hat{y})^T \qquad \text{where} \quad \hat{y} = \frac{1}{n_T}\sum_{i=1}^{n_T} y_i. \tag{3.15}$$

In Section 4 we will consider the case with r sensors and bias. In this case, $\hat{V}$ is given by

$$\hat{V} = \frac{1}{n_T - r}\sum_{s\in S}\sum_{i=1}^{n^s}(y_i^s - \hat{y}^s)(y_i^s - \hat{y}^s)^T \qquad \text{where} \quad \hat{y}^s = \frac{1}{n^s}\sum_{i=1}^{n^s} y_i^s. \tag{3.16}$$

To compute the volume of X^0 from $\hat{V}$, we assume that X^0 is a Cartesian product of ellipsoidal regions with covariance matrix $\hat{V}$. If X^0 is a product of m-dimensional ellipsoids (e.g., x could be 6-dimensional, with $m = 3$ being the physical dimension of position- and of velocity-space), then

$$\mathrm{Vol}(X^0) = \sqrt{|2\pi\theta\hat{V}|} \qquad \text{where} \quad \theta = (1 + m/2)(m/2)!^{-2/m}. \tag{3.17}$$

The values of θ for $m = 1$, 2, and 3 are $6/\pi \approx 1.91$, 2, and $(5/3)\sqrt[3]{6/\pi} \approx 2.07$, respectively. Because these values are so close, setting $\theta = 2$ for all problems is an acceptable approximation.

This volume estimate is not ideal. It is sensitive to outliers and to measurements being close to co-planar. Fortunately, its effect is limited to the threshold—i.e., the decision of whether to associate two measurements at all. Some authors dispense with the volume estimation entirely, using a diffuse spatial prior [14, 15], which is perfectly valid, but limits the power of the resulting method to hypothesis tests between associations representing the same number of detected objects.

3.2. Splitting into Components

When $\mathbf{z}$ comprises various data types, with some degree of independence in how each type is generated, (2.12) can be split into components for each data type. Here we will make a major simplification by splitting the kinematic data from any non-kinematic data types present. The kinematic data retain the complications due to β and ξ, and Section 4 demonstrates how to handle this. The non-kinematic data is modeled to be without the complications due to β and ξ: therefore the non-kinematic component of association probability simplifies into a product over $\alpha \in [a]^+$, as in Theorem 3.1. Examples of how to model various non-kinematic data types are given in Section 5.

To split the problem into components, we split both the state x and each measurement z_i^s into a kinematic component (K) and another component (J, for all non-kinematic variables *jointly*): let $x = (x^K, x^J)$ and $z_i^s = (z_i^{Ks}, z_i^{Js})$. We make the following assumptions about how the prior distribution, detection probability, and measurement likelihood functions split:

$$P_{\mathrm{D}}^s(x) = P_{\mathrm{D}}^{Js}(x^J), \tag{3.18}$$

$$p^0(x \mid \xi) = p^{K0}(x^K - \xi)p^{J0}(x^J) \qquad \text{and} \tag{3.19}$$

$$L^s(z_i^s \mid x, \beta^s) = L^{Ks}(z_i^{Ks} \mid x^K, \beta^s)L^{Js}(z_i^{Js} \mid x^J). \tag{3.20}$$

Equation (3.18) stipulates that the detection probability is independent of the kinematic state. As discussed in Section 2, this is an unfortunate but necessary oversimplification. It is allowed to depend on non-kinematic variables, however: for example, it is plausible that one could have a reasonable model of detection probability

as a function of object type (cf. Section 5.1). Equations (3.19) and (3.20) stipulate that the complications due to the center ξ of the object region and the systematic error β^s are solely kinematic phenomena. The assumptions (3.18)–(3.20) permit a relatively simple treatment of the non-kinematic variables, while addressing the effects of kinematic bias.

The definition of q in (2.6) may be simplified to

$$q = \int \prod_{s \in S} Q_{\mathrm{D}}^{Js}(x^J) dx^J, \tag{3.21}$$

because for any ξ the integral of $p^{K0}(x^K - \xi)$ over all x^K is 1. Note that q is independent of ξ, as required in the text following (2.6). The key probability density $P^\alpha(\mathbf{z} \mid \boldsymbol{\beta}, \xi)$ may be split as follows:

$$P^\alpha(\mathbf{z} \mid \boldsymbol{\beta}, \xi) = P^{K\alpha}(\mathbf{z}^K \mid \boldsymbol{\beta}, \xi) P^{J\alpha}(\mathbf{z}^J), \tag{3.22}$$

using (2.5) to give us these formulas for the kinematic and non-kinematic components of P^α:

$$P^{K\alpha}(\mathbf{z}^K \mid \boldsymbol{\beta}, \xi) = \int p^{K0}(x^K - \xi) \prod_{(s,i) \in \alpha} L^{Ks}(z_i^{Ks} \mid x^K, \beta^s) dx^K \quad \text{and} \tag{3.23}$$

$$P^{J\alpha}(\mathbf{z}^J) = \int p^{J0}(x^J) \prod_{(s,i) \in \alpha} P_{\mathrm{D}}^{Js}(x^J) L^{Js}(z_i^{Js} \mid x^J) \times \prod_{s \notin \bar{\alpha}} Q_{\mathrm{D}}^{Js}(x^J) dx^J. \tag{3.24}$$

THEOREM 3.2 *Given the assumptions* (3.18)–(3.20)*, the probability of the association* $[a]$ *given the measurements* $\mathbf{z}$ *is*

$$\Pr([a] \mid \mathbf{z}) = \Pr([a_0] \mid \mathbf{z}) g([a]) G^K(\mathbf{z}^K, [a]) G^J(\mathbf{z}^J, [a]), \tag{3.25}$$

with

$$G^K(\mathbf{z}^K, [a]) = \frac{F^K(\mathbf{z}^K, [a])}{F^K(\mathbf{z}^K, [a_0])}, \tag{3.26}$$

where

$$F^K(\mathbf{z}^K, [a]) = \int \int \prod_{\alpha \in [a]} P^{K\alpha}(\mathbf{z}^K \mid \boldsymbol{\beta}, \xi) P^0(\boldsymbol{\beta}) P_\Xi^0(\xi) d\boldsymbol{\beta}\, d\xi, \tag{3.27}$$

and

$$G^J(\mathbf{z}^J, [a]) = \prod_{\alpha \in [a]^+} R^{J\alpha}(\mathbf{z}^J), \tag{3.28}$$

where

$$R^{J\alpha}(\mathbf{z}^J) = \frac{P^{J\alpha}(\mathbf{z}^J)}{\prod_{(s,i) \in \alpha} P^{Js}(z_i^{Js})}. \tag{3.29}$$

PROOF This result follows directly from the definitions of the quantities involved.

4. THE KINEMATIC COMPONENT

In this section we derive an exact formula for the kinematic component $G^K(\mathbf{z}^K, [a])$ of the association probability in the case of an arbitrary number of sensors with bias effects included. We will drop the superscript K throughout this section. Equation (3.26) shows that evaluating $G(\mathbf{z}, [a])$ involves integrating over the sensor biases $\boldsymbol{\beta}$. This differs from the more typical approaches which identify and remove a bias hypothesis (either a distinct hypothesis for each association, or, more crudely, one hypothesis for all associations). Such approaches can fail even in quite simple scenarios, such as the one discussed in [10].

To obtain a formula for the kinematic component $G(\mathbf{z}, [a])$ of the association probability, we must evaluate the integrals in (3.23) and (3.27). To do so, we need appropriate models for the quantities which appear in them. For the bias prior we assume

$$P^0(\boldsymbol{\beta}) = \prod_{s \in S} P^{0s}(\beta^s) = \prod_{s \in S} \mathcal{N}(\beta^s; \beta_0^s, B^s). \tag{4.1}$$

Here β_0^s is the mean bias on sensor s, and B^s is the bias covariance matrix. In practice, one would typically set β_0^s to zero because one could simply add it to each measurement on sensor s in a pre-processing step. The bias covariance matrix B^s should be part of the performance specifications for sensor s. If it is not, however, one may set B^s to be diffuse. We let the prior on ξ be diffuse:

$$P_\Xi^0(\xi) = \mathcal{N}(\xi; \xi_0, V_\Xi) \qquad \text{where} \quad V_\Xi \to \infty. \tag{4.2}$$

The irrelevant value of ξ_0 will be retained until the step that takes $V_\Xi \to \infty$ eliminates it.

We let the measurement likelihood function be Gaussian, writing the measurement as $z_i^s = (y_i^s, V_i^s)$, as in Section 3.1.1. Generalizing (3.8) to include bias, we have

$$L^s(z_i^s \mid x, \beta^s) = \mathcal{N}(y_i^s; x - \beta^s, V_i^s). \tag{4.3}$$

Here the factor $P_V(V_i^s)$ which appeared in (3.8) has been set to 1 (because the value does not matter, and it would, in fact, be 1 in a suitably chosen measure space). Finally, we assume that the prior distribution on x is Gaussian with mean ξ and known variance V_0,

$$p^0(x - \xi) = \mathcal{N}(x; \xi, V_0), \tag{4.4}$$

for the reasons discussed before Equation (2.4). Equation (3.16) may be used to produce a value for V_0 in practice.

LEMMA 4.1

$$P^\alpha(\mathbf{z} \mid \boldsymbol{\beta}, \xi) = \sqrt{|2\pi W_\alpha|}\, \mathcal{N}(\xi, \mu_\alpha, V_0) \prod_{(s,i) \in \alpha} \mathcal{N}(y_i^s + \beta^s; \mu_\alpha, V_i^s), \tag{4.5}$$

where

$$W_\alpha = \left(V_0^{-1} + \sum_{(s,i) \in \alpha} (V_i^s)^{-1} \right)^{-1} \qquad \text{and}$$
$$\mu_\alpha = W_\alpha \left(V_0^{-1} \xi + \sum_{(s,i) \in \alpha} (V_i^s)^{-1} (y_i^s + \beta^s) \right). \tag{4.6}$$

PROOF With (4.1)–(4.4), the integrand in (3.23) becomes a product of Gaussians, which may be integrated using the standard formula (D.3).

Substituting (4.5) into (3.27) produces the following key integral to evaluate:

$$F(\mathbf{z},[a]) = \left(\prod_{\alpha\in[a]}\sqrt{|2\pi W_\alpha|}\right)\int\int\cdots \times\int\prod_{\alpha\in[a]}\left(\mathcal{N}(\xi,\mu_\alpha,V_0)\prod_{(s,i)\in\alpha}\mathcal{N}(y_i^s+\beta^s;\mu_\alpha,V_i^s)\right) \times\prod_{s\in S}\mathcal{N}(\beta^s;\beta_0^s,B^s)\mathcal{N}(\xi;\xi_0,V_\Xi)d\beta^1 d\beta^2\cdots d\beta^r d\xi. \tag{4.7}$$

Evaluating (4.7) yields the kinematic component $G(\mathbf{z},[a])$ of the association probability $\Pr([a]\,|\,\mathbf{z})$ in (3.25). Two formulas for $G(\mathbf{z},[a])$ will be given. First, Theorem 4.2 provides a formula based on the direct evaluation of (4.7). Theorem 4.3 then gives a more computationally efficient formula achieved by applying certain transformations to the first result.

THEOREM 4.2

$$G(\mathbf{z},[a]) = C\exp\left(-\frac{1}{2}\left(\kappa([a])+\sum_{\alpha\in[a]}\kappa_\alpha\right)\right), \tag{4.8}$$

where C is chosen so that $G(\mathbf{z},[a_0])=1$, and the costs $\kappa([a])$ and κ_α are defined by

$$\kappa([a]) = \log|\mathbf{U}| - \mathbf{b}^T\mathbf{U}^{-1}\mathbf{b}, \qquad \text{and} \tag{4.9}$$

$$\kappa_\alpha = \log|V_0W_\alpha^{-1}| - m_\alpha^T W_\alpha^{-1} m_\alpha. \tag{4.10}$$

The matrix $\mathbf{U}$ *and vector* $\mathbf{b}$ *have the following block structure:*

$$\mathbf{U} = \begin{pmatrix} U_{1,1} & U_{1,2} & \cdots & U_{1,r} & U_{1,\emptyset} \\ U_{2,1} & U_{2,2} & \cdots & U_{2,r} & U_{2,\emptyset} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ U_{r,1} & U_{r,2} & \cdots & U_{r,r} & U_{r,\emptyset} \\ U_{\emptyset,1} & U_{\emptyset,2} & \cdots & U_{\emptyset,r} & U_{\emptyset,\emptyset} \end{pmatrix} \quad \text{and} \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_r \\ b_\emptyset \end{pmatrix}. \tag{4.11}$$

The entries in $\mathbf{U}$ *are matrices defined as follows. For* $s_1, s_2 \in S$,

$$U_{s_1,s_2} = \delta_{s_1 s_2}U^{s_1} - \sum_{\alpha\in[a]} I_\alpha^{s_1} I_\alpha^{s_2} (V_{\iota_\alpha^{s_1}}^{s_1})^{-1} W_\alpha (V_{\iota_\alpha^{s_2}}^{s_2})^{-1}, \tag{4.12}$$

$$U_{s_1,\emptyset} = -\sum_{\alpha\in[a]} I_\alpha^{s_1}(V_{\iota_\alpha^{s_1}}^{s_1})^{-1} W_\alpha V_0^{-1}, \tag{4.13}$$

$$U_{\emptyset,s_2} = -\sum_{\alpha\in[a]} I_\alpha^{s_2} V_0^{-1} W_\alpha (V_{\iota_\alpha^{s_2}}^{s_2})^{-1},$$

$$U_{\emptyset,\emptyset} = n_D V_0^{-1} - \sum_{\alpha\in[a]} V_0^{-1} W_\alpha V_0^{-1}, \tag{4.14}$$

where $I_\alpha^s = 1$ if $s\in\bar{\alpha}$ (and ι_α^s denotes the (unique) i for which $(s,i)\in\alpha$ in this case), and $I_\alpha^s = 0$ when $s\notin\bar{\alpha}$. Similarly, for $s\in S$, the entries in $\mathbf{b}$ *are vectors defined by*

$$b_s = (B^s)^{-1}\beta_0^s - \sum_{\alpha\in[a]} I_\alpha^s (V_{\iota_\alpha^s}^s)^{-1}(y_{\iota_\alpha^s}^s - m_\alpha) \qquad \text{and}$$

$$b_\emptyset = V_0^{-1}\sum_{\alpha\in[a]} m_\alpha. \tag{4.16}$$

The components of $\mathbf{U}$ *and* $\mathbf{b}$ *depend on the following quantities:*

$$W_\alpha = \left(V_0^{-1} + \sum_{(s,i)\in\alpha}(V_i^s)^{-1}\right)^{-1}, \tag{4.17}$$

$$m_\alpha = W_\alpha \sum_{(s,i)\in\alpha}(V_i^s)^{-1} y_i^s \qquad \text{and}$$

$$U^s = (B^s)^{-1} + \sum_{i=1}^{n^s}(V_i^s)^{-1}. \tag{4.18}$$

PROOF Equation (4.7) is more complicated than it appears: the notation μ_α, defined in (4.6), conceals a dependence on each of the integration variables in many of the Gaussian factors. Nevertheless, (4.7) is just an integral of products of Gaussians, where the arguments of the Gaussians are linear combinations of the integration variables. Equation (D.10) provides a formula for integrals of this form. To use this formula, we first must explicitly cast (4.7) in the form of (D.4). The Gaussians in (4.7) are of four types: type 1 is $\mathcal{N}(\xi;\mu_\alpha,V_0)$, indexed by $\alpha\in[a]$; type 2 is $\mathcal{N}(y_i^s+\beta^s;\mu_\alpha,V_i^s)$, indexed by $(s,i)\in\alpha$ and $\alpha\in[a]$; type 3 is $\mathcal{N}(\beta^s;\beta_0^s,B^s)$, indexed by $s\in S$, and type 4 is the single Gaussian $\mathcal{N}(\xi;\xi_0,V_\Xi)$. The integration variables in (4.7) are of two types: type a is β^{s_1}, indexed by $s_1\in S$; and type b is the single variable ξ. Thus, there are eight cases for the quantity A_{ij} in (D.4). In case 1a, for example, A_{α,s_1} is the coefficient of β^{s_1} in the exponent of $\mathcal{N}(\xi;\mu_\alpha,V_0)$, i.e., $A_{\alpha,s_1} = -I_\alpha^{s_1} W_\alpha (V_{\iota_\alpha^{s_1}}^{s_1})^{-1}$. The quantity m_i in (D.4) is simpler: there are only four cases to consider. For example, in case 2, $m_{(s,i,\alpha)}$ is the constant term in the exponent of $\mathcal{N}(y_i^s+\beta^s;\mu_\alpha,V_i^s)$, i.e., $m_{(s,i,\alpha)} = m_\alpha - y_i^s$. After A_{ij} and m_i have been expressed explicitly for each case, one may use (D.5)–(D.7) to obtain formulas for for $\mathbf{U}$, $\mathbf{b}$, and c, letting $V_\Xi\to\infty$ in each. The results for $\mathbf{U}$ and $\mathbf{b}$ are given in (4.11)–(4.16). The expression for c simplifies to

$$c = -\sum_{\alpha\in[a]} m_\alpha^T W_\alpha^{-1} m_\alpha + \sum_{s\in S}\sum_{i=1}^{n^s}(y_i^s)^T(V_i^s)^{-1}y_i^s + \sum_{s\in S}(\beta_0^s)^T(B^s)^{-1}\beta_0^s. \tag{4.19}$$

Invoking (D.10) to evaluate (4.7) now yields the result (4.8).

The need to invert the $(r+1)\times(r+1)$ block matrix $\mathbf{U}$ may seem rather excessive, and indeed it is. The next theorem reduces this requirement to one of inverting an $(r-1)\times(r-1)$ block matrix. Thus in the two-sensor case there is no need to form oversize matrices at all. Another problem with inverting $\mathbf{U}$ is that it is singular in the important case of all the bias prior covariances B^s being diffuse. Although this can be dealt with easily as a special case (by deleting the final row and column blocks of $\mathbf{U}$ and the final block of $\mathbf{b}$), the formula is ill-conditioned for large B^s. The next theorem provides a well conditioned formula.

THEOREM 4.3 *If the measurements* $\mathbf{z}$ *have been preprocessed so that* $\beta_0^s = 0$ *for all* $s \in S$*, then the definition of* $\kappa([a])$ *in* (4.9) *may be replaced with*

$$\kappa([a]) = \log(|\mathbf{W}_-^*|\,|H|) - (\tilde{\mathbf{b}}_-^{*T}\mathbf{W}_-^{*-1}\tilde{\mathbf{b}}_-^* + h^T H^{-1} h), \tag{4.20}$$

where the quantities involved are defined as follows. First,

$$\mathbf{W}_-^* = \tilde{\mathbf{U}}_{0-}^* + \mathbf{D}_-^*, \tag{4.21}$$

where $\mathbf{D}_-^*$ *encapsulates the dependence on the bias prior information:*

$$\begin{aligned} \mathbf{D}_-^* &= (D_{ss'}^*)_{s,s'=1}^{r-1} \qquad \text{with} \\ D_{ss'}^* &= \delta_{ss'}(B^s)^{-1} - (B^s)^{-1} D^{-1} (B^{s'})^{-1}, \\ D &= \sum_{s\in S} (B^s)^{-1}. \end{aligned} \tag{4.22}$$

The other quantities are

$$\begin{aligned} \tilde{\mathbf{U}}_{0-}^* &= (F_{ss'} - G_s H^{-1} G_{s'}^T)_{s,s'=1}^{r-1} \qquad \text{and} \\ \tilde{\mathbf{b}}_-^* &= (g_s - G_s H^{-1} h)_{s=1}^{r-1}, \end{aligned} \tag{4.23}$$

which are defined in terms of

$$F_{ss'} = \delta_{ss'} \sum_{i=1}^{n^s} (V_i^s)^{-1} - \sum_{\alpha\in[a]} I_\alpha^s I_\alpha^{s'} (V_{\iota_\alpha^s}^s)^{-1} W_\alpha (V_{\iota_\alpha^{s'}}^{s'})^{-1}, \tag{4.24}$$

$$G_s = -\sum_{\alpha\in[a]} I_\alpha^s (V_{\iota_\alpha^s}^s)^{-1} W_\alpha, \quad g_s = \sum_{\alpha\in[a]} I_\alpha^s (V_{\iota_\alpha^s}^s)^{-1} (m_\alpha - y_{\iota_\alpha^s}^s), \tag{4.25}$$

$$H = n_D V_0 - \sum_{\alpha\in[a]} W_\alpha \qquad \text{and} \qquad h = \sum_{\alpha\in[a]} m_\alpha. \tag{4.26}$$

PROOF Equation (4.20) is based on two reductions. The first eliminates the $\emptyset$ (or ξ) component from the block matrix $\mathbf{U}$. The second transforms the coordinate system of the r absolute biases to that of the $r-1$ relative biases and the sum of the biases, then eliminates the sum-of-biases component as well.

We begin the simplification with the following block decompositions of $\mathbf{U}$ and $\mathbf{b}$ into their (r-dimensional) $\boldsymbol{\beta}$ part and their (1-dimensional) ξ part:

$$\mathbf{U} = \begin{pmatrix} \tilde{\mathbf{U}}_0 + \mathbf{B}^{-1} & \tilde{\mathbf{U}}_\emptyset \\ \tilde{\mathbf{U}}_\emptyset^T & U_{\emptyset,\emptyset} \end{pmatrix} \qquad \text{and} \qquad \mathbf{b} = \begin{pmatrix} \tilde{\mathbf{b}} \\ b_\emptyset \end{pmatrix}, \tag{4.27}$$

where $\mathbf{B}$ denotes the $r\times r$ block diagonal matrix of bias covariance matrices B^s. From (4.12)–(4.16) we find that $\tilde{\mathbf{U}}_\emptyset = -\tilde{\mathbf{U}}_0\mathbf{1}$, $U_{\emptyset,\emptyset} = \mathbf{1}^T\tilde{\mathbf{U}}_0\mathbf{1}$, and $b_\emptyset = -\mathbf{1}^T\tilde{\mathbf{b}}$, where $\mathbf{1}$ denotes the $r\times 1$ block matrix of identity matrices. We may now eliminate the ξ component by applying (D.19) with the $\emptyset$ component in the role of the integrated variable I:

$$\mathbf{b}^T\mathbf{U}^{-1}\mathbf{b} = b_\emptyset^T U_{\emptyset,\emptyset}^{-1} b_\emptyset + \tilde{\mathbf{b}}^{*T}\tilde{\mathbf{U}}^{*-1}\tilde{\mathbf{b}}^* \qquad \text{and} \qquad |\mathbf{U}| = |U_{\emptyset,\emptyset}|\,|\tilde{\mathbf{U}}^*|. \tag{4.28}$$

In these equations, $\tilde{\mathbf{U}}^*$ and $\tilde{\mathbf{b}}^*$ are given by (D.14) and (D.15). To keep the bias prior information separated out, we write $\tilde{\mathbf{U}}^* = \tilde{\mathbf{U}}_0^* + \mathbf{B}^{-1}$, with

$$\tilde{\mathbf{U}}_0^* = \tilde{\mathbf{U}}_0 - \tilde{\mathbf{U}}_\emptyset U_{\emptyset,\emptyset}^{-1} \tilde{\mathbf{U}}_\emptyset^T \qquad \text{and} \qquad \tilde{\mathbf{b}}^* = \tilde{\mathbf{b}} - \tilde{\mathbf{U}}_\emptyset U_{\emptyset,\emptyset}^{-1} b_\emptyset. \tag{4.29}$$

For the second reduction, we simplify the expressions $\tilde{\mathbf{b}}^{*T}\tilde{\mathbf{U}}^{*-1}\tilde{\mathbf{b}}^*$ and $|\tilde{\mathbf{U}}^*|$ in (4.28). To do this, we transform the problem from the coordinates of the bias vector $\boldsymbol{\beta}$ to those of a vector $\mathbf{C}\boldsymbol{\beta}$ comprising the $r-1$ relative biases $\beta^s - \beta^r$, and the sum of the biases $\beta^1 + \beta^2 + \cdots + \beta^r$. The $r\times r$ block matrix $\mathbf{C}$ which accomplishes this, and its inverse, are

$$\mathbf{C} = \begin{pmatrix} \mathbf{I}_- & -\mathbf{1}_- \\ \mathbf{1}_-^T & I \end{pmatrix} \qquad \text{and} \qquad \mathbf{C}^{-1} = \frac{1}{r}\begin{pmatrix} r\mathbf{I}_- - \mathbf{1}_-\mathbf{1}_-^T & \mathbf{1}_- \\ -\mathbf{1}_-^T & I \end{pmatrix}, \tag{4.30}$$

where $\mathbf{I}_-$ is the $(r-1)\times(r-1)$ block identity matrix, and $\mathbf{1}_-$ is the $(r-1)\times 1$ block matrix of identity matrices. To simplify $\tilde{\mathbf{b}}^{*T}\tilde{\mathbf{U}}^{*-1}\tilde{\mathbf{b}}^*$ and $|\tilde{\mathbf{U}}^*|$, we work instead with the transformed quantities $\mathbf{W} = \mathbf{C}^{-T}\tilde{\mathbf{U}}^*\mathbf{C}^{-1}$ and $\mathbf{d} = \mathbf{C}^{-T}\tilde{\mathbf{b}}^*$, breaking them into the same blocks as $\mathbf{C}$:

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_- & \mathbf{W}_r \\ \mathbf{W}_r^T & W_{rr} \end{pmatrix}, \qquad \mathbf{d} = \begin{pmatrix} \mathbf{d}_- \\ 0 \end{pmatrix}. \tag{4.31}$$

After some manipulation, we find

$$\tilde{\mathbf{b}}^{*T}\tilde{\mathbf{U}}^{*-1}\tilde{\mathbf{b}}^* = \mathbf{d}^T\mathbf{W}^{-1}\mathbf{d} = \mathbf{d}_-^T\mathbf{W}_-^{*-1}\mathbf{d}_- \qquad \text{and} \tag{4.32}$$

$$|\tilde{\mathbf{U}}^*| = |\mathbf{C}|^2|\mathbf{W}| = |\mathbf{C}|^2|W_{rr}|\,|\mathbf{W}_-^*|, \tag{4.33}$$

where (D.19) gives $\mathbf{W}_-^* = \mathbf{W}_- - \mathbf{W}_r W_{rr}^{-1}\mathbf{W}_r^T$. The original $\mathbf{U}$ and $\mathbf{b}$ quantities may now be written in terms of much simpler quantities:

$$\mathbf{b}^T\mathbf{U}^{-1}\mathbf{b} = \mathbf{d}_-^T\mathbf{W}_-^{*-1}\mathbf{d}_- + b_\emptyset^T U_{\emptyset,\emptyset}^{-1} b_\emptyset \qquad \text{and} \tag{4.34}$$

$$|\mathbf{U}| = |\mathbf{W}_-^*|\,|U_{\emptyset,\emptyset}|. \tag{4.35}$$

The result follows directly. The value of $\kappa([a])$ in (4.20) differs from that in (4.9) by $\log|D| - 2\log|V_0|$, but this can be absorbed into the normalizing constant C.

The expression (4.22) simplifies in certain special cases. When all the bias covariances B^s are diffuse, $\mathbf{D}^*_- = 0$, and in the two-sensor case, $\mathbf{D}^*_- = (B^1 + B^2)^{-1}$.

5. NON-KINEMATIC COMPONENTS

Equation (3.29) gives a simple formula for $R^{J\alpha}(\mathbf{z}^J)$, the contribution to association probability due to all non-kinematic data. (We will henceforth drop the superscript J.) Whereas the results of Section 4 are theorems that can be used "off-the-shelf," the category of non-kinematic data is too diverse to allow results that are this explicit yet broadly applicable. The explicit evaluation of $R^\alpha(\mathbf{z})$ using (3.29) depends on the statistical characteristics of the data, so one must be prepared to derive an appropriate formula for $R^\alpha(\mathbf{z})$ for one's particular problem. In this section we will present examples for various illustrative cases. These cases cover a range of possible types of non-kinematic data. They may be used directly for certain applications, modified for others, or referred to for guidance in developing appropriate formulas for applications further afield. Section 5.1 discusses how to handle object classification information in conjunction with feature data which may depend on object type. Sections 5.2 and 5.3 provide general methods for handling noisy and missing data, respectively.

5.1. Object-Type-Dependent Features

We will use the term "object type" to refer to a specific kind of feature: a discrete one, representing a finite number of classes to which the object could belong. It is typically measured by a classifier. The possible classification calls c are often the same as the possible object types t, but for the purposes of data association there is no requirement that the two be related. The quality of a classifier is determined by its confusion matrix. We use $L^s(c \mid t)$ to denote the confusion matrix entries for sensor s: i.e., the probability that an object of type t will be classified as c by sensor s.

Now consider a joint feature whose state space is parameterized by a state $x = (t, y)$, where t is the object type and y is the state of some other feature. Similarly, we decompose a measurement z into its classification c and the measurement w provided by a feature extractor attempting to measure y: i.e., $z = (c, w)$.

Equation (3.29) may be re-written

$$R^\alpha(\mathbf{c},\mathbf{w}) = \frac{P^\alpha(\mathbf{c},\mathbf{w})}{\prod_{(s,i)\in\alpha} P^s(c_i^s, w_i^s)}, \tag{5.1}$$

where (3.24) is now

$$P^\alpha(\mathbf{c},\mathbf{w}) = \sum_t \int p^0(t,y) \prod_{(s,i)\in\alpha} P_{\mathrm{D}}^s(t,y) L^s(c_i^s, w_i^s \mid t,y) \times \prod_{s\notin\bar{\alpha}} Q_{\mathrm{D}}^s(t,y)\,dy. \tag{5.2}$$

and $P^s(c_i^s, w_i^s)$ denotes $P^\alpha(\mathbf{c},\mathbf{w})$ for the special case $\alpha = \{(s,i)\}$. To evaluate (5.2) we need to specify what assumptions we are making about p^0, L^s, and P_{D}^s. Section 5.1.1 describes the simplest case in which R^α splits into $\mathbf{c}$ and $\mathbf{w}$ components. Section 5.1.2 describes a more interesting case in which the distribution of a measurement w depends not only on y, but on the object type t as well.

5.1.1. Independent Case

Suppose the following independence assumptions for p^0 and L^s are applicable,

$$p^0(t,y) = p^0(t)p^0(y), \tag{5.3}$$

$$L^s(c_i^s, w_i^s \mid t,y) = L^s(c_i^s \mid t) L^s(w_i^s \mid y). \tag{5.4}$$

(We are being somewhat cavalier here in the overloading of the notation p^0 and L^s, but trust that the meaning is clear in context because of the symbols used in their arguments.) Also suppose that P_{D}^s depends only on the object type t:

$$P_{\mathrm{D}}^s(t,y) = P_{\mathrm{D}}^s(t). \tag{5.5}$$

(Generalizing this to, say, $P_{\mathrm{D}}^s(t,y) = P_{\mathrm{D}}^s(t)P_{\mathrm{D}}^s(y)$ would cause complications because $Q_{\mathrm{D}}^s(t,y)$ would not enjoy the same simple multiplicative form.) With these assumptions, we find

$$R^\alpha(\mathbf{c},\mathbf{w}) = R^{C\alpha}(\mathbf{c}) R^{W\alpha}(\mathbf{w}), \tag{5.6}$$

where

$$R^{C\alpha}(\mathbf{c}) = \frac{P^{C\alpha}(\mathbf{c})}{\prod_{(s,i)\in\alpha} P^{Cs}(c_i^s)} \quad \text{and} \qquad R^{W\alpha}(\mathbf{w}) = \frac{P^{W\alpha}(\mathbf{w})}{\prod_{(s,i)\in\alpha} P^{Ws}(w_i^s)}, \tag{5.7}$$

with

$$P^{C\alpha}(\mathbf{c}) = \sum_t p^0(t) \prod_{(s,i)\in\alpha} P_{\mathrm{D}}^s(t) L^s(c_i^s \mid t) \prod_{s\notin\bar{\alpha}} Q_{\mathrm{D}}^s(t) \quad \text{and} \tag{5.8}$$

$$P^{W\alpha}(\mathbf{w}) = \int p^0(y) \prod_{(s,i)\in\alpha} L^s(w_i^s \mid y)\,dy, \tag{5.9}$$

and with $P^{Cs}(c_i^s)$ and $P^{Ws}(w_i^s)$ being the usual shorthand for the $\alpha = \{(s,i)\}$ case. Also note that the object-type–dependent detection probability leads to the following form for the non-detection parameter q:

$$q = \sum_t p^0(t) \prod_{s\in S} Q_{\mathrm{D}}^s(t). \tag{5.10}$$

In the absence of object-type data $\mathbf{c}$, $R^{C\alpha}(\mathbf{c})$ simplifies because $L^s(c_i^s \mid t)$ may be replaced by 1 in (5.8). When there is, furthermore, no object-type component of state t, $R^{C\alpha}(\mathbf{c})$ simplifies further:

$$R^{C\alpha} = \frac{\prod_{s\in\bar{\alpha}} P_{\mathrm{D}}^s \prod_{s\notin\bar{\alpha}} Q_{\mathrm{D}}^s}{\prod_{s\in\bar{\alpha}} \left(P_{\mathrm{D}}^s \prod_{s'\neq s} Q_{\mathrm{D}}^{s'}\right)} = q^{1-|\alpha|}, \qquad (5.11)$$

where q is simply the product of the Q_{D}^s over all $s \in S$, rather than (5.10).

The form (5.6) is quite convenient, as it allows one to separate the computations dealing with the object-type C data from that of the other feature W. However, it is often the case that objects of different types will have substantially different feature statistics. For example, objects of one type t_1 may have a certain prior distribution $p^0(y \mid t_1)$ on feature values as well has a certain measurement likelihood function $L^s(w \mid y,t_1)$, while objects of another type t_2 may not even have the feature in question. In cases like this, we need a more accommodating approach.

5.1.2. Dependent Case

Instead of making assumptions (5.3) and (5.4), we write this general decomposition of p^0 and L^s, which makes no assumptions—just manipulations of conditional probabilities:

$$p^0(t,y) = p^0(t)p^0(y \mid t), \qquad (5.12)$$

$$L^s(c_i^s, w_i^s \mid t,y) = L^s(c_i^s \mid t,y)L^s(w_i^s \mid c_i^s,t,y). \qquad (5.13)$$

We now make two mild assumptions about (5.13) in place of the radical assumption made in (5.4). First, we assume that $L^s(c_i^s \mid t,y)$ does not depend on y. Although the classifier behavior may in fact depend on the feature state y, we assume that this is a minor effect. This is convenient because although one expects a confusion matrix $L^s(c \mid t)$ to be provided with a classifier for a given sensor configuration, one is unlikely to be provided with the dependence of the classifier on y. Second, we assume that $L^s(w_i^s \mid c_i^s,t,y)$ does not depend on c_i^s. In other words, we assume that the true object type suffices to determine how the measurement w_i^s depends on the feature state y, and that the called type yields little additional information. With these assumptions, (5.13) simplifies to

$$L^s(c_i^s, w_i^s \mid t,y) = L^s(c_i^s \mid t)L^s(w_i^s \mid t,y). \qquad (5.14)$$

Finally, as in (5.5) of Section 5.1.1 we assume that the detection probability depends only on the object type: $P_{\mathrm{D}}^s(t,y) = P_{\mathrm{D}}^s(t)$.

Using (5.12), (5.14), and (5.5) we may simplify (5.2) to a relatively simple sum over t,

$$P^\alpha(\mathbf{c},\mathbf{w}) = \sum_t p^0(t) \prod_{(s,i)\in\alpha} P_{\mathrm{D}}^s(t)L^s(c_i^s \mid t) \prod_{s\notin\bar{\alpha}} Q_{\mathrm{D}}^s(t)P^\alpha(\mathbf{w} \mid t), \qquad (5.15)$$

where

$$P^\alpha(\mathbf{w} \mid t) = \int p^0(y \mid t) \prod_{(s,i)\in\alpha} L^s(w_i^s \mid t,y)dy. \qquad (5.16)$$

The formula (5.10) for q also holds in this case. Thus this object-type–dependent case requires one to evaluate an analog of (5.9) for each object type t—namely (5.16). The result of this is coupled with the classification data, via (5.15), resulting in a calculation which is somewhat more complicated than (5.8), but is nevertheless rather straightforward, especially considering its much greater generality and realism.

It remains to evaluate the integral in (5.16). In doing so, we will first consider two general phenomena that affect its evaluation. Section 5.2 will address the issue that feature extraction is often a noisy procedure, and propose a robust, general principle for coping with this. Finally, Section 5.3 will describe how to cope with the fact that features might be missing and/or assessed to be missing.

5.2. Non-Informative Noise

Let us consider the object type t in (5.16) fixed in this section, and suppress it in the notation. This is equivalent to considering (5.9) instead (suppressing the W from the notation). Either interpretation yields

$$P^\alpha(\mathbf{w}) = \int p^0(y) \prod_{(s,i)\in\alpha} L^s(w_i^s \mid y)dy. \qquad (5.17)$$

A typical feature model one might use for $L^s(w \mid y)$ is a Gaussian. Such a model carries the risk of returning incredibly tiny assessments of probability density for a measurement w arising (e.g., 10^{-100}, 10^{-1000}, or smaller) when it doesn't match y well. In a realistic situation, the probability density of a measurement w could never be that small because there is always the possibility of some glitch in the feature extraction routine. By allowing such tiny probability densities to occur in the model, one runs the risk of the feature extractor completely preventing a pair of tracks being associated even when the kinematic information is extremely favorable to association. Because one of the chief fears in incorporating a feature extractor into an association algorithm is that it might ruin kinematic-only performance that is already fairly good, it is prudent to account for the possibility of noise in the feature measurement model. (Note: one could make the same argument to point out that the rapid decay of Gaussians allows anomalous kinematic data to override perfect feature matches, so one might include a noise term in the kinematic association terms as well. This would be a point worth considering when feature extraction technology reaches the maturity of kinematic tracking.)

We regard a function as representing pure noise when the distribution of measurements w it yields is independent of the actual feature state y. One option for a noise model is a uniform distribution of w over a certain range. Although this seems simple, it introduces an additional parameter (the width of the distribution) and complicates the required integrals, while not necessarily being a good model of noise. Instead, we promote the use of a *non-informative* model for noise. This model sets the distribution of measurements due to noise equal to the overall distribution of valid measurements. It is called non-informative because a measurement w provides no information as to whether it arose from a valid measurement of some object or merely from noise. Were one to know how noise differs statistically from valid measurements, one could use this information to flag certain measurements w as more likely to have arisen from noise than others, and perhaps squeeze even more performance out of an association algorithm, but at the risk of algorithm robustness should the noise behave differently than expected. In contrast, the non-informative assumption provides a conservative, robust baseline model for noise.

We use the non-informative assumption as follows. Rather than developing a model for the measurement likelihood function $L^s(w \mid y)$ directly, we develop a model $L^s_\mu(w \mid y)$ for how we expect the measurement w to be distributed given that (a) the true feature value is y, and (b) the measurement is actually behaving according to the model. We let a^s_μ denote the probability that the feature is obeying the model and express the overall measurement likelihood function $L^s(w \mid y)$ as

$$L^s(w \mid y) = a^s_\mu L^s_\mu(w \mid y) + (1 - a^s_\mu) P^s_\nu(w), \quad (5.18)$$

where $P^s_\nu(w)$ is the distribution of w given that it arises from noise. Note that there is no dependence on the true state y in this case. We now define the following analog of (5.17) for L^s_μ:

$$P^\alpha_\mu(\mathbf{w}) = \int p^0(y) \prod_{(s,i)\in\alpha} L^s_\mu(w^s_i \mid y) dy. \quad (5.19)$$

The special case $\alpha = \{(s,i)\}$ is particularly important here. Thinking of w as representing w^s_i here, we define

$$P^s_\mu(w) = \int p^0(y) L^s_\mu(w \mid y) dy. \quad (5.20)$$

Equation (5.20) expresses the overall distribution $P^s_\mu(w)$ of w values arising on sensor s when the features are "obeying the model." The non-informative noise assumption is that the distribution in the noise case $P^s_\nu(w)$ is identical to $P^s_\mu(w)$:

$$P^s_\nu(w) = P^s_\mu(w). \quad (5.21)$$

Using this, and substituting the expression in (5.18) for $L^s(w \mid y)$ into (5.17) yields

$$\begin{aligned} P^\alpha(\mathbf{w}) &= \sum_{\alpha'\subseteq\alpha} A^{\bar\alpha,\bar\alpha'}_\mu P^{\alpha'}_\mu(\mathbf{w}) \prod_{(s,i)\in\alpha\setminus\alpha'} P^s_\mu(w^s_i) \\ &= \sum_{\alpha'\subseteq\alpha \mid \alpha'\mid\geq 2} A^{\bar\alpha,\bar\alpha'}_\mu P^{\alpha'}_\mu(\mathbf{w}) \prod_{(s,i)\in\alpha\setminus\alpha'} P^s_\mu(w^s_i) \\ &\quad + C^{\bar\alpha}_\mu \prod_{(s,i)\in\alpha} P^s_\mu(w^s_i), \end{aligned} \quad (5.22)$$

where

$$A^{\bar\alpha,\bar\alpha'}_\mu = \prod_{s\in\bar\alpha'} a^s_\mu \prod_{s\in\bar\alpha\setminus\bar\alpha'} (1 - a^s_\mu) \qquad \text{and} \qquad C^{\bar\alpha}_\mu = \sum_{\bar\alpha'\subseteq\bar\alpha \mid \bar\alpha'\mid\leq 1} A^{\bar\alpha,\bar\alpha'}_\mu. \quad (5.23)$$

Equation (5.22) simplifies significantly in the two-sensor case. Letting $\alpha = \{(1,i),(2,j)\}$, it reduces to

$$P^\alpha(\mathbf{w}) = a^1_\mu a^2_\mu P^\alpha_\mu(\mathbf{w}) + (1 - a^1_\mu a^2_\mu) P^1_\mu(w^1_i) P^2_\mu(w^2_j). \quad (5.24)$$

The formula (5.22) (or (5.24)) applies both to the relatively simple case of Section 5.1.1, and the more complicated case of Section 5.1.2. In the latter case, it should be interpreted as a formula for $P^\alpha(\mathbf{w} \mid t)$, the dependence on object type t having been suppressed. In this case, one would use (5.19) to compute $P^\alpha_\mu(\mathbf{w} \mid t)$ from $p^0(y \mid t)$ and $L^s_\mu(w \mid t, y)$, and the parameters a^s_μ may (or may not) depend on t as well. The resulting formulas for $P^\alpha(\mathbf{w} \mid t)$ for each t would then be used in (5.15) in place of (5.16). In the two-sensor case, this amounts to a fairly trivial modification.

Section 5.1.1 treats the case where the feature is independent of object type. In this case, the result may be expressed in terms of a quantity $R^\alpha_\mu(\mathbf{w})$ which represents what we would use if we ignored noise modeling:

$$R^\alpha_\mu(\mathbf{w}) = \frac{P^\alpha_\mu(\mathbf{w})}{\prod_{(s,i)\in\alpha} P^s_\mu(w^s_i)}. \quad (5.25)$$

The inclusion of noise modeling yields the following forms for $R^\alpha(\mathbf{w})$,

$$\begin{aligned} R^\alpha(\mathbf{w}) &= \sum_{\alpha'\subseteq\alpha \mid \alpha'\mid\geq 2} A^{\bar\alpha,\bar\alpha'}_\mu R^{\alpha'}_\mu(\mathbf{w}) + C^{\bar\alpha}_\mu, \qquad \text{or} \\ R^\alpha(\mathbf{w}) &= a^1_\mu a^2_\mu R^\alpha_\mu(\mathbf{w}) + (1 - a^1_\mu a^2_\mu), \end{aligned} \quad (5.26)$$

in the general and two-sensor cases, respectively. The two-sensor formula has a pleasing interpretation as a convex combination of the model value $R^\alpha_\mu(\mathbf{w})$ and the neutral value 1.

To use the non-informative noise model, then, one takes the formula one has developed for $P^\alpha(\mathbf{w} \mid t)$ or $R^\alpha(\mathbf{w})$, renames it $P^\alpha_\mu(\mathbf{w} \mid t)$ or $R^\alpha_\mu(\mathbf{w})$, respectively, then uses the appropriate equation above ((5.22), (5.24), or (5.26)) to incorporate the possibility that real-world measurements may not obey one's model. This is a nice

form of modularity, which allows us to develop formulas without explicitly considering the effect of noise: it can be incorporated with any model one develops. Similarly, one may develop models for various object types t and combine them using (5.15). Next we develop a similarly modular capability for handling the possibility of missing features.

5.3. Missing Features

In this subsection we consider the complication that a feature may not be present on every object. If the feature extractor cannot handle this situation, but returns some meaningless value when the feature is absent, one can use the non-informative noise assumption of Section 5.2. However, when the feature extractor exhibits a distinctive behavior for objects that lack the feature, then it is advantageous to exploit this information in a way that the non-informative noise model does not. Here we assume that either the designer of the feature model has created a flag that indicates a belief that the feature is absent, or that some post-processing of the output produces such a flag. In either case, the performance would undoubtedly be imperfect, and may be characterized by a detection probability P_{d}^s (the probability of the feature being declared present on sensor s when it is indeed present) and a false alarm probability P_{fa}^s (the probability of the feature being declared present when it is in fact missing). We use the lower case d to distinguish the missing feature detection probability P_{d}^s here from the object detection probability P_{D}^s.

We extend the state space for the feature to account for the possibility of it being missing. If y denotes the feature state, then we distinguish between two types of values it can take: y_e, which denotes a value for a feature that exists, and $\emptyset$, which denotes that the feature is missing. The prior $p^0(y)$ on the feature state may be expressed as

$$p^0(y) = \begin{cases} P_{\mathrm{E}} p_{\mathrm{e}}^0(y_e) & \text{for} \quad y = y_e, \\ Q_{\mathrm{E}} & \text{for} \quad y = \emptyset, \end{cases} \tag{5.27}$$

where P_{E} is the prior probability of the feature existing, $Q_{\mathrm{E}} = 1 - P_{\mathrm{E}}$, and $p_{\mathrm{e}}^0(y_e)$ is the prior distribution on y_e given that the feature exists.

We let w_i^s denote the feature measurement on sensor s. Like the feature state, the measurement can take two types of values: a proper value w_{pi}^s, or a call from the feature extractor on sensor s that it is missing, $\emptyset^s$. To express the measurement likelihood function $L^s(w_i^s \mid y)$ now requires four cases:

$$\begin{aligned} L^s(w_{pi}^s \mid y_e) &= P_{\mathrm{d}}^s L_{\mathrm{d}}^s(w_{pi}^s \mid y_e), \\ L^s(w_{pi}^s \mid \emptyset) &= P_{\mathrm{fa}}^s L_{\mathrm{fa}}^s(w_{pi}^s), \end{aligned} \tag{5.28}$$

$$\begin{aligned} L^s(\emptyset^s \mid y_e) &= Q_d^s, \\ L^s(\emptyset^s \mid \emptyset) &= Q_{\mathrm{fa}}^s, \end{aligned} \tag{5.29}$$

where $Q_{\mathrm{d}}^s = 1 - P_{\mathrm{d}}^s$, $Q_{\mathrm{fa}}^s = 1 - P_{\mathrm{fa}}^s$, $L_{\mathrm{d}}^s(w_{pi}^s \mid y_e)$ is the measurement likelihood function given that the feature exists and measurement is proper, and $L_{\mathrm{fa}}^s(w_{pi}^s)$ is the measurement likelihood function given that the feature does not exist but the measurement is proper. These formulas are somewhat different from those in [11], where it is assumed that a "false alarm" can occur even in the case where the feature exists.

We now evaluate the integral in (5.9), suppressing the superscript W:

$$P^\alpha(\mathbf{w}) = \int p^0(y) \prod_{(s,i)\in\alpha} L^s(w_i^s \mid y) dy. \tag{5.30}$$

Let α_p denote the subset of $(s,i) \in \alpha$ for which w_i^s is a proper measurement, and $\alpha_\emptyset$ denote the subset for which $w_i^s = \emptyset^s$. We make a number of definitions similar to those of Section 5.2 for non-informative noise. Analogously to (5.19) we define

$$P_{\mathrm{d}}^{\alpha_p}(\mathbf{w}) = \int p_{\mathrm{e}}^0(y_e) \prod_{(s,i)\in\alpha_p} L_{\mathrm{d}}^s(w_{pi}^s \mid y_e) dy_e \tag{5.31}$$

to be the version of $P^\alpha(\mathbf{w})$ one would use when not considering the possibility of missing features. Thinking of w as representing w_{pi}^s we write the special case $\alpha = \{(s,i)\}$ of (5.31) as

$$P_{\mathrm{d}}^s(w) = \int p_{\mathrm{e}}^0(y_e) L_{\mathrm{d}}^s(w \mid y_e) dy_e. \tag{5.32}$$

We employ the non-informative noise assumption to model $L_{\mathrm{fa}}^s(w_{pi}^s)$:

$$L_{\mathrm{fa}}^s(w_{pi}^s) = P_{\mathrm{d}}^s(w_{pi}^s). \tag{5.33}$$

With the above definitions, we may re-write (5.30) as

$$\begin{aligned} P^\alpha(\mathbf{w}) = P_{\mathrm{E}} \left(\prod_{(s,i)\in\alpha_p} P_{\mathrm{d}}^s \prod_{(s,i)\in\alpha_\emptyset} Q_{\mathrm{d}}^s \right) P_{\mathrm{d}}^{\alpha_p}(\mathbf{w}) \\ + Q_{\mathrm{E}} \left(\prod_{(s,i)\in\alpha_p} P_{\mathrm{fa}}^s \prod_{(s,i)\in\alpha_\emptyset} Q_{\mathrm{fa}}^s \right) \prod_{(s,i)\in\alpha_p} P_{\mathrm{d}}^s(w_{pi}^s). \end{aligned} \tag{5.34}$$

This section has described how to incorporate object-type-dependence, noise, and missing features into the evaluation of the feature component of association probability. Ultimately, regardless of what effects are included, one must evaluate an integral of the form (5.30) where the prior $p^0(y)$ and the measurement likelihood function $L^s(w_i^s \mid y)$ are appropriate to some specific feature. A Gaussian model for $L^s(w_i^s \mid y)$ may be appropriate, in which case one can treat the feature in the same way as one-dimensional kinematic data. A more complex example is presented in Section 6, where an angle-valued measurement has a von Mises error distribution coupled with the possibility of being off by 180°. The example shows explicitly how to evaluate (5.30) in

this case, and then combine it with all three phenomena in this section to produce association probabilities.

6. EXAMPLE

6.1. Sample Calculation

This section provides an example of the computation of association probabilities in a complex scenario. It has three sensors with unknown biases, object-type classification data and object-type-dependent detection probabilities, and finally, noisy and possibly missing features whose distributions vary between object types. We perform this sample calculation for the following data set:

- Measurements:

$$s=1:\quad y_1^1=(1.5,3.7),\qquad y_2^1=(9.3,1.9),\qquad y_3^1=(13.2,-11.2),$$

$$s=2:\quad y_1^2=(-6.2,14.9),\qquad y_2^2=(-3.6,6.9),\qquad y_3^2=(3.0,4.5),$$

$$s=3:\quad y_1^3=(-16.8,-4.8),\qquad y_2^3=(-7.3,-11.1).$$

- Error covariances:

$$s=1:\quad V_1^1=\begin{pmatrix}1.58 & -0.12\\ -0.12 & 2.27\end{pmatrix},\quad V_2^1=\begin{pmatrix}8.0 & -2.0\\ -2.0 & 10.2\end{pmatrix},\quad V_3^1=\begin{pmatrix}8.7 & -5.8\\ -5.8 & 12.8\end{pmatrix},$$

$$s=2:\quad V_1^2=\begin{pmatrix}13.4 & -0.6\\ -0.6 & 4.0\end{pmatrix},\quad V_2^2=\begin{pmatrix}57. & -31.\\ -31. & 30.\end{pmatrix},\quad V_3^2=\begin{pmatrix}3.22 & -1.15\\ -1.15 & 1.76\end{pmatrix},$$

$$s=3:\quad V_1^3=\begin{pmatrix}1.32 & 0.52\\ 0.52 & 0.94\end{pmatrix},\quad V_2^3=\begin{pmatrix}13.6 & 10.7\\ 10.7 & 9.3\end{pmatrix}.$$

- Called types:

$$s=1:\quad c_1^1=\blacktriangle,\qquad c_2^1=\blacklozenge,\qquad c_3^1=\bullet,$$

$$s=2:\quad c_1^2=\blacklozenge,\qquad c_2^2=\blacklozenge,\qquad c_3^2=\blacksquare,$$

$$s=3:\quad c_1^3=\blacklozenge,\qquad c_2^3=\blacktriangleright.$$

- Feature measurements:

$$s=1:\quad w_1^1=\emptyset^1,\qquad w_2^1=5.5771,\qquad w_3^1=6.2067,$$

$$s=2:\quad w_1^2=4.9773,\qquad w_2^2=4.9101,\qquad w_3^2=6.2011,$$

$$s=3:\quad w_1^3=5.1253,\qquad w_2^3=3.0885.$$

The data set is visualized in Fig. 1. The measurements are labeled from left to right on each sensor. The kinematic error covariances are represented by 90% containment ellipses. The called object types for the classifiers

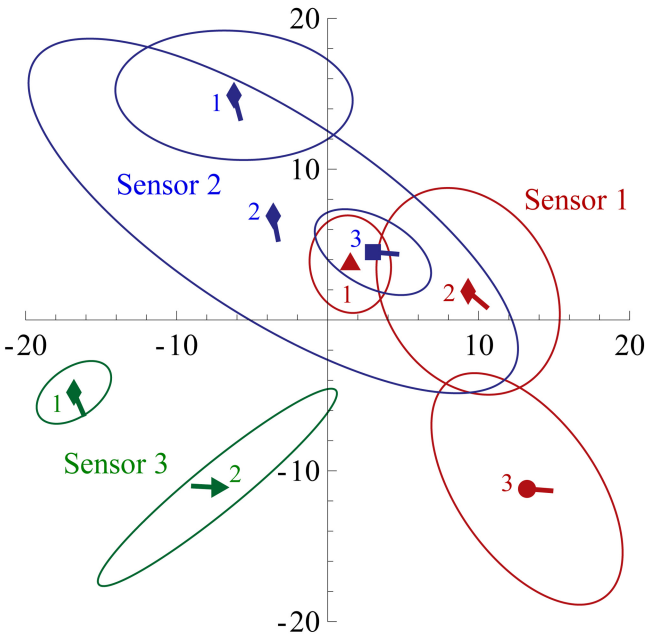


Fig. 1. Measurements and error covariances, with ellipses denoting 90% containment regions, symbols denoting object-type calls, and line segments denoting measured feature values (angles).

are the symbols used for plotting them. The feature measurements are angles ranging from 0 to 2π, represented by a small line segment, or the special value $\emptyset^s$ denoting a missing feature call by sensor s.

There are 778 possible associations for $\mathbf{n}=(3,3,2)$. To speak meaningfully of the probabilities of these associations given the data, one must make assumptions about the data generation process. Here, we make the following assumptions:

- Log-diffuse prior on number of objects (cf. Section B.1)
- Covariance of prior kinematic distribution:

$$V^0=\begin{pmatrix}8 & -9\\ -9 & 11\end{pmatrix}$$

- Diffuse bias priors for all sensors
- Four object types (■, ♦, ▲, and ▼) with statistics given in Tables I and II
- An object-type–dependent angular feature with statistics described below

The four ground-truth object types listed above may be thought of as the possible shapes of an object. The Bayesian methodology requires a prior distribution $p^0(t)$ on these types, which is given in Table I. Some object types may be easier to detect on a given sensor than others. Table I also gives the detection probability for each type on each sensor.

Although the prior probability is much larger for $t=\blacktriangledown$ than for other object types ($p^0(\blacktriangledown)=0.55$), (C.7) implies that the expected number $n_D(t)$ of each object type t detected on at least one sensor is roughly equal

TABLE I
Prior and Detection Probabilities for Object Types

	True type t			
	■	♦	▲	▼
$p^0(t)$	0.1	0.15	0.2	0.55

		True type t			
$P_D^s(t)$		■	♦	▲	▼
Sensor s	1	0.9	0.8	0.95	0.3
	2	0.8	0.99	0.7	0.9
	3	0.4	0.8	0.9	0.99

TABLE II
Confusion Matrices for Each Sensor

$L^1(c\|t)$		Called type c				
		■	♦	▲	▼	●
True type t	■	0.75	0.01	0.03	0.01	0.2
	♦	0.04	0.62	0.02	0.07	0.25
	▲	0.03	0.01	0.74	0.02	0.2
	▼	0.07	0.08	0.05	0.5	0.3

$L^2(c\|t)$		Called type c			
		■	♦	▲	▼
True type t	■	0.8	0.1	0.05	0.05
	♦	0.08	0.75	0.12	0.05
	▲	0.02	0.15	0.6	0.23
	▼	0.01	0.01	0.03	0.95

$L^3(c\|t)$		Called type c		
		■	♦	►
True type t	■	0.95	0.02	0.03
	♦	0.01	0.96	0.03
	▲	0.02	0.01	0.97
	▼	0.04	0.01	0.95

across types:

$$\mathbb{E}[n_D(\blacksquare) \mid \mathbf{n}] = 0.765169, \qquad \mathbb{E}[n_D(\blacktriangle) \mid \mathbf{n}] = 0.795727,$$
$$\mathbb{E}[n_D(\blacklozenge) \mid \mathbf{n}] = 0.763363, \qquad \mathbb{E}[n_D(\blacktriangledown) \mid \mathbf{n}] = 0.849989.$$

The measurement likelihood functions $L^s(c \mid t)$ for each sensor are given by the confusion matrices in Table II. In a real application, such values would be the result of training a classifier. Although the set of possible called types for a classifier is often identical to the set of ground truth types, only the sensor 2 classifier operates this way in the example. Sensor 1 has an additional called type ● representing "unknown," whereas Sensor 3 cannot distinguish between ▲ and ▼, so it issues a call ► which represents either one.

We now specify phenomenologically rich feature distributions for each of the object types above. First, we assume that the feature may be missing, and let $P_{\mathrm{E}}(t)$ be the prior probability that a proper, ground-truth feature value exists for an object of type t. The values of $P_{\mathrm{E}}(t)$ are given in Table III. Assuming a proper value y_e does exist, we assume it to be a uniformly distributed angular quantity: i.e., $p_{\mathrm{e}}^0(y_e \mid t) = 1/2\pi$.

The measurement likelihood function for object type t on sensor s is governed by five quantities: $a_\mu^s(t)$, $P_{\mathrm{d}}^s(t)$, $P_{\mathrm{fa}}^s(t)$, $p_{\mathrm{jump}}^s(t)$, $\sigma^s(t)$. These are given in Table III. As in Section 5.2, $a_\mu^s(t)$ represents the probability that the measurement w obeys the model described by the rest of the measurement likelihood function. The probabilities $P_{\mathrm{d}}^s(t)$ and $P_{\mathrm{fa}}^s(t)$ are the detection and false alarm probabilities for the decision problem of declaring a proper feature value or missing, as described in Section 5.3. Finally, we specify an underlying model for the non-noise, detection case based on the parameters $p_{\mathrm{jump}}^s(t)$ and $\sigma^s(t)$. The jump probability $p_{\mathrm{jump}}^s(t)$ is the probability that an additional π has been erroneously added to the feature measurement. The pseudo-standard–deviation $\sigma^s(t)$ is an analog to the standard deviation of a Gaussian in the case of a feature whose domain is the entire real line. The angular analog of a Gaussian is the Von Mises distribution [1]:

$$\mathcal{M}(\theta; \mu, \kappa) = \frac{e^{\kappa \cos(\theta - \mu)}}{2\pi I_0(\kappa)}, \tag{6.1}$$

where I_0 is the modified Bessel function of order 0. The κ in (6.1) is analogous to an inverse variance. Thus the distribution of a feature value w (before introducing the

TABLE III
Prior and Detection Probabilities for Object Types

	True type t			
	■	♦	▲	▼
$P_E(t)$	0.9	0.95	0.7	0.99

$a^s_\mu(t)$		True type t			
		■	♦	▲	▼
Sensor s	1	0.99	0.998	0.8	0.99
	2	0.8	0.999	0.9	$1-10^{-5}$
	3	0.99	0.99	0.95	$1-10^{-10}$

$P^s_d(t)$		True type t			
		■	♦	▲	▼
Sensor s	1	0.94	0.94	0.96	0.83
	2	0.99	0.99	0.98	0.97
	3	0.6	0.92	0.93	0.997

$P^s_{fa}(t)$		True type t			
		■	♦	▲	▼
Sensor s	1	0.02	0.04	0.03	0.002
	2	0.05	0.1	0.06	0.02
	3	0.001	0.01	0.02	0.3

$p^s_{\text{jump}}(t)$		True type t			
		■	♦	▲	▼
Sensor s	1	0.01	0.08	0	0.5
	2	0.02	0.15	0	0.5
	3	0.06	0.05	0.03	0.5

$\sigma^s(t)$		True type t			
		■	♦	▲	▼
Sensor s	1	0.1	0.23	0.05	0.02
	2	0.07	0.2	0.15	0.05
	3	0.05	0.3	0.08	0.03

additional effects of jump, missing features, and noise), is $\mathcal{M}(w; y, \sigma^s(t)^{-2})$.

The probability distribution is now fully specified, giving sufficient information to compute the probability of each possible association. In order to do so, we will use the following identity for the Von Mises distribution:

$$\frac{1}{2\pi}\int_0^{2\pi}\prod_{s=1}^{r}\mathcal{M}(\theta;\mu^s,\kappa^s)d\theta = I_0\left(\left|\sum_{s=1}^{r}\kappa^s\exp(i\mu^s)\right|\right)\Bigg/\prod_{s=1}^{r}2\pi I_0(\kappa^s). \tag{6.2}$$

We now compute the probability of one of the most plausible of the 778 possible associations, namely:

$$[a] = \{(1,1,1),(2,2,\emptyset),(3,3,2)\}$$

(which is shorthand for $[a] = \{\{(1,1),(2,1),(3,1)\}, \{(1,2),(2,2)\},\{(1,3),(2,3),(3,2)\}\}$). We begin by evaluating the quantities in (4.17):

$$W_{(1,1,1)} = \begin{pmatrix} 0.32753 & -0.08142 \\ -0.08142 & 0.33302 \end{pmatrix},$$

$$W_{(2,2,\emptyset)} = \begin{pmatrix} 2.78935 & -2.85439 \\ -2.85439 & 3.67734 \end{pmatrix},$$

$$W_{(3,3,2)} = \begin{pmatrix} 0.36425 & -0.08525 \\ -0.08525 & 0.33942 \end{pmatrix},$$

and, writing column vectors in rows for convenience,

$$m_{(1,1,1)} = (-4.87067, 3.66306),$$

$$m_{(2,2,\emptyset)} = (1.62393, -1.04784),$$

$$m_{(3,3,2)} = (3.09849, -1.43294).$$

These are then used to compute κ_α from (4.10) for each $\alpha \in [a]$:

$$\kappa_{(1,1,1)} = -87.4344,$$

$$\kappa_{(2,2,\emptyset)} = -0.2445,$$

$$\kappa_{(3,3,2)} = -23.8285.$$

The values of W_α and m_α above are also used to compute H, $\mathbf{W}^*_-$, h, and $\mathbf{b}^*_-$:

$$H = \begin{pmatrix} 20.519 & -23.979 \\ -23.979 & 28.650 \end{pmatrix},$$

$$\mathbf{W}^*_- = \begin{pmatrix} 0.56727 & 0.05332 & -0.18105 & -0.20300 \\ 0.05332 & 0.48860 & -0.10474 & -0.20623 \\ -0.18105 & -0.10474 & 0.34568 & 0.08847 \\ -0.20300 & -0.20623 & 0.08847 & 0.63862 \end{pmatrix},$$

$$h = (-0.14826, 1.18228),$$

$$\tilde{\mathbf{b}}^*_- = (-5.5339, -0.04816, -1.2648, -6.9713).$$

From these we have $|\mathbf{W}^*_-||H| = 0.473943$, and $\tilde{\mathbf{b}}^{*T}_- \cdot \mathbf{W}^{*-1}_- \tilde{\mathbf{b}}^*_- + h^T H^{-1} h = 247.8660$, so (4.20) yields $\kappa([a]) = -248.613$. Combining this in (4.8) with the values of κ_α we have

$$G^K(\mathbf{z}^K, [a]) = C \exp\left(-\frac{1}{2}\left(\kappa([a]) + \sum_{\alpha \in [a]} \kappa_\alpha\right)\right)$$
$$= 1.58147 \times 10^{78} C.$$

In addition to $G^K(\mathbf{z}^K, [a])$ there are two other contributions to the association probability in (3.25). We compute $g([a])$ from (B.10). This requires the probability q that an object is undetected on all sensors, which is given by (5.10):

$$q = \sum_t p^0(t) \prod_{s \in S} Q^s_{\mathrm{D}}(t) = 0.001945 \quad \text{(exactly)}.$$

Therefore $g([a]) = (2!/7!)0.998055^5 = 0.000392981$.

Finally, we compute the joint object-type–feature component $G^J(\mathbf{z}^J, [a])$, which is somewhat involved. Fortunately, (3.28) and (3.29) reduce this calculation to the computation of $P^{J\alpha}(\mathbf{z}^J)$ for each α, and there are only 47 possible sets α (compared to 778 associations). Because the feature distributions and detection probabilities are object-type–dependent, for any α, $P^{J\alpha}(\mathbf{z}^J)$ is given by $P^\alpha(\mathbf{c}, \mathbf{w})$ in (5.15) (the J being suppressed, and $\mathbf{z}^J$ expanded into $(\mathbf{c}, \mathbf{w})$). This, in turn, requires the computation of $P^\alpha(\mathbf{w} \mid t)$ in (5.16) for each object type t. We will compute $P^\alpha(\mathbf{w} \mid t)$ explicitly for one of the sets α in our example $[a]$, namely $\alpha = (3,3,2) = \{(1,3),(2,3),(3,2)\}$. To do so, we successively break down the computation in order to account for various phenomena—in each case expressing $P^\alpha(\mathbf{w} \mid t)$ in terms of some simpler version of $P^\alpha(\mathbf{w} \mid t)$ which does not include the phenomenon. Specifically, we do this for noise, then missing features, then the jump by π.

The first phenomenon to account for is noise, so we evaluate $P^\alpha(\mathbf{w})$ using (5.22), suppressing the dependence on t in the notation for brevity. This requires that we compute $P^{\alpha'}_\mu(\mathbf{w})$ for all eight subsets $\alpha' \subseteq \alpha$, where α' represents a hypothesis about which measurements in α arise from the model (as opposed to noise). To compute each $P^{\alpha'}_\mu(\mathbf{w})$, we turn to (5.34) to handle missing features. In this case, $\alpha_p = \alpha'$ for any $\alpha' \subseteq \alpha$ because w^s_i is proper (i.e., non-missing) for each $(s,i) \in \alpha$: $w^1_3 = 6.2067$, $w^2_3 = 6.2011$, and $w^3_2 = 3.0885$. This, in turn, requires us to evaluate $P^{\alpha'}_{\mathrm{d}}(\mathbf{w})$, accounting the the jump phenomenon. Unlike noise and missing features, we do not have a general equation to account for the possible jump by π, as it is a rather specific phenomenon. Therefore we adapt (5.17) to this specific case, providing a general formula for $P^\alpha(\mathbf{w})$ (which we then use to evaluate $P^{\alpha'}_{\mathrm{d}}(\mathbf{w})$):

$$P^\alpha(\mathbf{w}) = \int p^0(y) \prod_{(s,i) \in \alpha} (q^s_{\mathrm{jump}} L^s(w^s_i \mid y) + p^s_{\mathrm{jump}} L^s(w^s_i - \pi \mid y)) dy$$
$$= \sum_{\alpha' \subseteq \alpha} \prod_{(s,i) \in \alpha'} p^s_{\mathrm{jump}} \prod_{(s,i) \in \alpha \setminus \alpha'} q^s_{\mathrm{jump}} P^\alpha_M(\mathbf{w}^{\alpha'}). \tag{6.3}$$

Here α' represents a hypothesis about which measurements in α have experienced a jump by π, and $\mathbf{w}^{\alpha'}$ is the same as $\mathbf{w}$, but with π subtracted from each w^s_i for which $(s,i) \in \alpha'$. Finally, we must compute the $P^\alpha_M(\mathbf{w})$ in (6.3). This is given by (6.2).

$$P^\alpha_M(\mathbf{w}) = I_0\left(\left|\sum_{(s,i) \in \alpha} (\sigma^s)^{-2} \exp(i w^s_i)\right|\right) \Big/ \prod_{(s,i) \in \alpha} 2\pi I_0((\sigma^s)^{-2}). \tag{6.4}$$

Now, we may begin the numerical computation of the original $P^\alpha(\mathbf{w} \mid t)$ (including all phenomena) for $\alpha = (3,3,2) = \{(1,3),(2,3),(3,2)\}$.

$$P^\alpha_M(\mathbf{w} \mid \blacksquare) = \frac{I_0(|0.1^{-2} e^{6.2067i} + 0.07^{-2} e^{6.2011i} + 0.05^{-2} e^{3.0885i}|)}{(2\pi)^3 I_0(0.1^{-2}) I_0(0.07^{-2}) I_0(0.05^{-2})}$$
$$= 8.87511 \times 10^{-264}.$$

Naturally, this value is tiny because $w^3_2 = 3.0885$ is many sigmas away from w^1_3 and w^2_3. However, when we evaluate (6.3), we sum over cases where π is subtracted from w^s_i for some subset α' of α. This will produce a large value for the subset $\alpha' = \{(3,2)\}$:

$$P^\alpha_M(\mathbf{w}^{\alpha'} \mid \blacksquare)$$
$$= \frac{I_0(|0.1^{-2} e^{6.2067i} + 0.07^{-2} e^{6.2011i} + 0.05^{-2} e^{(3.0885-\pi)i}|)}{(2\pi)^3 I_0(0.1^{-2}) I_0(0.07^{-2}) I_0(0.05^{-2})}$$
$$= 2.55152.$$

Naturally, the same value is produced for $\alpha' = \{(1,3),(2,3)\}$, but all other $\alpha' \subseteq \alpha$ produce negligible values. Therefore (6.3) may be evaluated as

$$P^\alpha_{\mathrm{d}}(\mathbf{w} \mid \blacksquare) = 0.99 \times 0.98 \times 0.06 \times 2.55152$$
$$+ 0.01 \times 0.02 \times 0.94 \times 2.55152 + (\text{tiny})$$
$$= 0.149009.$$

Here we use the subscript d in anticipation of the next step: incorporating the effects of missing features via (5.34). This will require the value of $P^\alpha_{\mathrm{d}}(\mathbf{w} \mid \blacksquare)$ for one-element sets α—this value is always equal to the prior value $1/(2\pi)$ for this specific feature. Referring to Table III for the values of P_{E}, P^s_{d} and P^s_{fa}, we obtain the following probability density which accounts for the possibility of the object lacking the feature (and all three measurements in α arising as false alarms):

$$P^\alpha_\mu(\mathbf{w} \mid \blacksquare) = 0.9 \times 0.94 \times 0.99 \times 0.6 \times 0.149009$$
$$+ 0.1 \times 0.02 \times 0.05 \times 0.001 \times (2\pi)^{-3}$$
$$= 0.0748806.$$

Here we use the subscript μ in anticipation of the final step: incorporating noise. For this we need values of $P_\mu^{\alpha'}$ for all subsets $\alpha' \subseteq \alpha$. Repeating the above steps for each subset yields

$$P_\mu^{(3,3,2)}(\mathbf{w} \mid \blacksquare) = 0.0748806, \qquad P_\mu^{(\emptyset,\emptyset,\emptyset)}(\mathbf{w} \mid \blacksquare) = 1,$$
$$P_\mu^{(\emptyset,3,2)}(\mathbf{w} \mid \blacksquare) = 0.0289101, \qquad P_\mu^{(3,\emptyset,\emptyset)}(\mathbf{w} \mid \blacksquare) = 0.134963,$$
$$P_\mu^{(3,\emptyset,2)}(\mathbf{w} \mid \blacksquare) = 0.0193784, \qquad P_\mu^{(\emptyset,3,\emptyset)}(\mathbf{w} \mid \blacksquare) = 0.142603,$$
$$P_\mu^{(3,3,\emptyset)}(\mathbf{w} \mid \blacksquare) = 0.421703, \qquad P_\mu^{(\emptyset,\emptyset,2)}(\mathbf{w} \mid \blacksquare) = 0.0859596.$$

Therefore

$$\begin{aligned} P^\alpha(\mathbf{w} \mid \blacksquare) &= 0.99 \times 0.8 \times 0.99 \times 0.0748806 \\ &\quad + 0.8 \times 0.99 \times 0.01 \times 0.0289101 \times 0.134963 + \cdots \\ &\quad + 0.01 \times 0.2 \times 0.01 \times 1 \times 0.134963 \times 0.142603 \\ &\quad \times 0.0859596 = 0.0595787. \end{aligned}$$

Carrying out the above computation for each object type t, we get

$$P^\alpha(\mathbf{w} \mid \blacksquare) = 0.0595787, \qquad P^\alpha(\mathbf{w} \mid \blacktriangle) = 0.0221195,$$
$$P^\alpha(\mathbf{w} \mid \blacklozenge) = 0.00999947, \qquad P^\alpha(\mathbf{w} \mid \blacktriangledown) = 2.06241.$$

These are the quantities needed in (5.15) to compute $P^\alpha(\mathbf{c},\mathbf{w})$ for $\alpha = (3,3,2)$. Using the parameters in Tables I and II with the called types $c_3^1 = \bullet$, $c_3^2 = \blacksquare$, and $c_2^3 = \blacktriangleright$ we have

$$\begin{aligned} P^\alpha(\mathbf{c},\mathbf{w}) &= 0.1 \times 0.9 \times 0.8 \times 0.4 \times 0.2 \times 0.8 \times 0.03 \\ &\quad \times 0.0595787 + 0.15 \times 0.8 \times 0.99 \times 0.8 \times 0.25 \\ &\quad \times 0.08 \times 0.03 \times 0.00999947 + 0.2 \times 0.95 \times 0.7 \\ &\quad \times 0.9 \times 0.2 \times 0.02 \times 0.97 \times 0.0221195 + 0.55 \\ &\quad \times 0.3 \times 0.9 \times 0.99 \times 0.3 \times 0.01 \times 0.95 \times 2.06241 \\ &= 0.000883213. \end{aligned}$$

This value is the largest among three-measurement sets α, and the eighth-largest overall. The three largest values are $P^{(\emptyset,\emptyset,2)}(\mathbf{c},\mathbf{w}) = 0.00598235$, $P^{(3,\emptyset,2)}(\mathbf{c},\mathbf{w}) = 0.00281482$, and $P^{(3,3,\emptyset)}(\mathbf{c},\mathbf{w}) = 0.00241721$. After computing $P^\alpha(\mathbf{c},\mathbf{w})$ for all 47 sets α, we may use (3.29) to get $R^{J\alpha}(\mathbf{z}^J)$ for each α, and then multiply them in (3.28) to get $G^J(\mathbf{z}^J,[a])$. For $[a] = \{(1,1,1),(2,2,\emptyset),(3,3,2)\}$ this yields

$$\begin{aligned} G^J(\mathbf{z}^J,[a]) &= 107662. \times 4673.65 \times 548368. \\ &= 2.75925 \times 10^{14}. \end{aligned}$$

Therefore (3.25) reduces to

$$\begin{aligned} \Pr([a] \mid \mathbf{z}) &= 0.000392981 \times 1.58147 \times 10^{78} C \\ &\quad \times 2.75925 \times 10^{14} \Pr([a_0] \mid \mathbf{z}) \\ &= 1.71484 \times 10^{89} C_0, \end{aligned}$$

where $C_0 = C \Pr([a_0] \mid \mathbf{z})$.

TABLE IV
The Five Most Probable Associations

Association	Probability
$\{(1,1,\emptyset),(2,2,1),(3,3,2)\}$	0.452373
$\{(1,2,\emptyset),(2,1,1),(3,3,2)\}$	0.226248
$\{(1,1,1),(2,2,\emptyset),(3,3,2)\}$	0.137136
$\{(\emptyset,2,\emptyset),(1,\emptyset,\emptyset),(2,1,1),(3,3,2)\}$	0.0765394
$\{(\emptyset,1,1),(1,\emptyset,\emptyset),(2,2,\emptyset),(3,3,2)\}$	0.0375843

After performing the above computation for all associations, we may determine the value of the normalization coefficient C_0 to find that $\Pr([a] \mid \mathbf{z}) = 0.137136$, which makes it the third most probable association of the 778. The most probable associations are shown in Table IV.

6.2. Simulation

The example above indicates that calculating association probability can be rather involved when one is incorporating many phenomena. The presence of bias in the kinematic component of the data introduces a term $\kappa([a])$ in the kinematic cost that depends on the association $[a]$ as a whole, preventing integer programming algorithms from finding optimal associations easily. The costs for the non-kinematic component of the data may be decomposed into the costs of individual sets $\alpha \in [a]$. In this case, however, there is a different kind of complication. The measurement likelihood function for the feature must be designed for specific feature of interest. In the example above, the feature exhibits a number of complicating phenomena (dependence on object type, missing values, noise, and the possibility of jumping by 180°). There is a valid concern that errors may exist in the derivation, transcription, and/or coding of such complicated formulas.

However, the probabilities produced, such as those in Table IV, have a precise meaning and are calculated exactly. As such, they enable a powerful check. For example, if we were able to collect independent events with probabilities of exactly 23%, they must obey precisely the same statistical law as flipping a coin whose probability of heads is 23%. To test the probabilities in the example of Section 6.1, we make 100,000 runs in which the data is generated according to the procedure specified by the given parameters, then compute the association probabilities and note, in each run, which association was correct. To do this, we require a method for generating error covariance matrices V_i^s. This is done as follows. First, a matrix is drawn from the Wishart distribution $\mathcal{W}(V_{\text{gen}}^s/m_{\text{gen}}^s, m_{\text{gen}}^s)$, then the result is scaled by a factor e^ξ where $\xi \sim \mathcal{N}(0, \log \gamma_{\text{gen}}^s)$. This process is carried out independently for each V_i^s. The parameters used in this procedure are

- Wishart parameters for error covariances: $m_{\text{gen}}^1 = 10$, $m_{\text{gen}}^2 = 10$, $m_{\text{gen}}^3 = 10$

- Baseline error covariances:

$$V^1_{\text{gen}} = \begin{pmatrix} 5 & 1 \\ 1 & 7 \end{pmatrix},$$

$$V^2_{\text{gen}} = \begin{pmatrix} 6 & -3 \\ -3 & 4 \end{pmatrix},$$

$$V^3_{\text{gen}} = \begin{pmatrix} 7 & 5 \\ 5 & 4 \end{pmatrix}$$

- Scale factors for error covariances: $\gamma^1_{\text{gen}} = 2$, $\gamma^2_{\text{gen}} = 3$, $\gamma^3_{\text{gen}} = 4$

There are some subtleties to the simulation process due to the diffuse nature of the priors employed, and to the requirement of not leaving extraneous signatures in the data. There are three places where diffuse priors occur: in the number of objects n; in ξ, the center of the region of objects; and in the bias β^s of some sensors. Appendix C shows how to simulate data efficiently with a specified number of tracks n^s on each sensor, even when the prior ρ^0 is diffuse. To circumvent directly sampling from diffuse priors for ξ and β^s, we impose the convention that scenes from sensors with diffuse bias priors are pre-processed to have a centroid of zero, and that all the other scenes are pre-processed jointly to have a centroid of zero. This renders the choices of ξ and of β^s for those sensors s with diffuse priors irrelevant because they leave no trace in the data.

One hundred thousand runs were made for this scenario, each run producing probabilities for each of the 778 possible associations. These associations were grouped into bins: those with probability less than 0.0001, those with probability between 0.0001 and 0.001, etc. For example, there were 452,911 associations with probability between 0.001 and 0.01. Of these, 1453 were correct associations, i.e., 0.32081%. This is certainly consistent—it lies between 0.1% and 1%—but we will demonstrate a far more stringent validation.

One may consider the 77,800,000 association events as independent flips of biased coins with various probabilities p_i of coming up heads (meaning "correct association"). (Technically the events are not completely independent but only very nearly so: two events picked at random have a 1-in-a-100,000 chance of coming from the same run, and hence exhibiting dependence.) We may consider either the entire data set of flips, or any subsets we like provided they are chosen without reference to ground truth. The law of large numbers says that the number of heads in any such data set will asymptotically approach a Gaussian distribution whose mean equals the sum of p_i over the events i, and whose variance equals the sum of $p_i(1 - p_i)$. Therefore, letting "tot" denote the total number of associations in a data set and "#" denote the number of these which are correct, we have

$$\begin{aligned} \mathbb{E}(\#/\text{tot}) &= \frac{1}{\text{tot}} \sum_{i=1}^{\text{tot}} p_i \qquad \text{and} \\ \sigma(\#/\text{tot}) &= \frac{1}{\text{tot}} \sqrt{\sum_{i=1}^{\text{tot}} p_i(1 - p_i)}. \end{aligned} \tag{6.5}$$

We may now make a much more incisive observation about 0.32081% than the fact that it lies between 0.1% and 1%. We find that $\mathbb{E}(\#/\text{tot}) = 0.32728\%$, and $\sigma(\#/\text{tot}) = 0.00848\%$. Thus, not only is 0.32081% between 0.1% and 1%, and not only is it close to the expected value 0.32728%, but it is precisely 0.76 standard deviations below the expected value. Furthermore, the large number of trials ensures that the distribution is approximately Gaussian, so we can convert the score $\zeta = -0.76$ into the statement that the number of correct associations lies at the 22nd percentile: i.e., it is a perfectly typical event.

Table V gives similar results for all probability bins. The first two columns specify the range of computed association probabilities in the bin. The third column gives the total number of associations (out of 77.8 million) with computed probabilities in the range of the bin. Column four gives the number of associations in the probability bin which are correct (and therefore sums to 100,000), and column five gives the fraction which are correct (column four divided by column three). The sixth and seventh columns give the expected value and standard deviation of the value in column five based on (6.5). Column eight gives the number of standard deviations which column five is above or below its expected value (i.e., $\zeta = (\#/\text{tot} - \mathbb{E}(\#/\text{tot}))/\sigma(\#/\text{tot})$). Finally, column nine converts column eight into the corresponding percentile of a Gaussian distribution.

The results in Table V verify the association probability formulas given in this paper in the scenario of Section 6.1. The most anomalous result is that only 99.42% of the associations with probability in the range 99% to 99.9% are correct, compared to an expected value of 99.53%, but this is only a 2.2 sigma event: still quite typical. Similar results hold for events other than complete associations. An event of great interest, for example, is whether the association deemed most probable is correct. As shown in Table VI, this occurs 82.8% of the time.

It is notable that both small and large probabilities are reliable—there is no failure in any probability range. Similar results are presented in [10] for a simpler, but denser, scenario (two sensors, kinematic-only data). There, a comparison is made between the exact bias integration formula presented in Section 4 and a method, currently considered state-of-the-art, in which the single most likely bias is selected and removed for each association hypothesis. Although the latter method performed reasonably accurately (17.6%, compared to 23.1% for the exact method) the probabilities it produced were off

TABLE V
Truth Versus Correctly Computed Probabilities for Various Probability Ranges

$p_{\min}$	$p_{\max}$	tot	#	#/tot	$\mathbb{E}$(#/tot)	σ(#/tot)	ζ	%
0	0.0001	75,986,700	85	0.0000011	0.0000010	0.0000001	0.65	0.74
0.0001	0.001	1,043,478	350	0.0003354	0.0003407	0.0000181	−0.29	0.39
0.001	0.01	452,911	1,453	0.0032081	0.0032728	0.0000848	−0.76	0.22
0.01	0.02	70,545	991	0.0140478	0.0141537	0.0004446	−0.24	0.41
0.02	0.05	64,167	2,020	0.0314804	0.0318053	0.0006919	−0.47	0.32
0.05	0.1	34,618	2,383	0.068837	0.0708085	0.0013765	−1.43	0.08
0.1	0.2	27,279	3,907	0.143224	0.143045	0.002113	0.08	0.53
0.2	0.3	14,298	3,579	0.250315	0.245706	0.003592	1.28	0.90
0.3	0.4	9,497	3,249	0.342108	0.347989	0.004879	−1.21	0.11
0.4	0.5	7,774	3,522	0.453049	0.448765	0.005631	0.76	0.78
0.5	0.6	6,804	3,753	0.551587	0.549095	0.006022	0.41	0.66
0.6	0.7	6,557	4,234	0.645722	0.650153	0.005879	−0.75	0.23
0.7	0.8	7,618	5,747	0.754397	0.751855	0.004937	0.51	0.70
0.8	0.9	11,162	9,602	0.860240	0.855228	0.003319	1.51	0.93
0.9	0.99	35,018	33,669	0.961477	0.960444	0.001033	1.00	0.84
0.99	0.999	20,013	19,897	0.994204	0.995258	0.000485	−2.17	0.01
0.999	1	1,561	1,559	0.998719	0.999301	0.000669	−0.87	0.19

TABLE VI
MAP Performance for Correctly Computed Probabilities

tot	#	#/tot	$\mathbb{E}$(#/*tot*)	σ(#/*tot*)	ζ	%
100000	82805	0.828050	0.826473	0.000996	1.58	0.94

by up to several hundred standard deviations, and were particularly inaccurate for events at the high- and low-probability extremes.

7. CONCLUSION

This paper espouses a Bayesian methodology for developing association probability formulas to use as a basis for developing high-quality, robust association algorithms. Section 2 derives a general formula for association probability given data from multiple sensors with uncorrelated errors at some fixed time. Sections 4 and 5 provide specific formulas for biased kinematic data and various kinds of non-kinematic data, respectively. A hallmark of the Bayesian approach is that the probabilities produced can be rigorously verified by the method introduced in Section 6.2.

One may argue that although the probabilities produced by a Bayesian method may be quite accurate in the scenario for which they are designed, real data will surely fail to conform to the underlying model and input parameters, so the assumption that Bayesian algorithms are superior to simpler non-Bayesian ones is unjustified for real data. This is not actually an argument for or against Bayesian algorithms, however—it merely makes the valid point that the true measure of the quality of an algorithm or a formula is its performance on real data. An ideal testbed would subject algorithms to the same range of phenomena that domain experts believe to occur in real data, presenting such a variety of scenarios that over-fitting is impossible. If an algorithm is able to produce association probabilities that are rigorously verifiable (in the sense of Section 6.2) within such a testbed, then when it outputs a result such as "the probability that these two measurements arose from the same object is 75%," this has a clear and useful meaning: of all possible states of the world which could give rise to the data we have observed, the measurements arise from the same object in three out of four of them.

To achieve such a reliable algorithm it is necessary to develop an accurate simulation capability, then to develop an association probability formula which exploits the domain expertise captured in the simulation to achieve accurate probability calculations. Only though a methodology for developing formulas which produces verifiably meaningful probabilities in simpler cases can one hope to achieve the goal of producing meaningful probabilities in the more realistic and complex cases. This paper supplies a general approach for developing such formulas, as well as results for certain specific cases.

ACKNOWLEDGMENTS

The author thanks Larry Stone for critiquing earlier, less elegant versions of Section 2, Roy Streit for suggesting the proper method for evaluating (D.4), and the reviewers for their insightful recommendations.

APPENDIX A. INCORPORATING FALSE ALARMS

The derivation in Section 2 assumes that none of the measurements are false alarms. The incorporation of false alarms affects only the combinatorial factor $g([a])$ in (2.12), however. Therefore the simpler, non-false-alarm case is used in the body of this paper. All results may be extended to include false alarms by using the modified value of $g([a])$ presented in this appendix.

When false alarms are present, the size n_{D}^s of J_{D}^s (the set of objects detected on sensor s) may fall short

of n^s (the number of measurements on sensor s). Let I^s_{FA} denote the set of false alarm measurements on sensor s and $n^s_{\mathrm{FA}} = |I^s_{\mathrm{FA}}|$, so that $n^s_{\mathrm{D}} + n^s_{\mathrm{FA}} = n^s$. The prior probability that the number of false alarms on sensor s is exactly n^s_{FA} is denoted $\rho^s_{\mathrm{FA}}(n^s_{\mathrm{FA}})$: this is assumed to be a known property of sensor s. If a measurement arises as a false alarm, the distribution of its (spurious) value z^s_i given the systematic error β^s and center ξ is denoted $L^s_{\mathrm{FA}}(z^s_i \mid \beta^s, \xi)$. The analog of Lemma 2.1 in the false-alarm case is the following.

LEMMA A.1 *The probability density of the measurement array* $\mathbf{z}^s$ *arising according to the mapping* a^s *given the object state array* $\mathbf{x}$*, the systematic error* β^s*, and the center* ξ *of the prior region of state space is*

$$\begin{aligned}\Pr(\mathbf{z}^s, a^s \mid \mathbf{x}, \beta^s, \xi) &= n^s_{\mathrm{FA}}! \rho^s_{\mathrm{FA}}(n^s_{\mathrm{FA}}) \prod_{i \in I^s_{\mathrm{FA}}} L^s_{\mathrm{FA}}(z^s_i \mid \beta^s, \xi) \\ &\quad \times \frac{1}{n^s!} \prod_{j \in J^s_{\mathrm{D}}} P^s_{\mathrm{D}}(x_j) L^s(z^s_{a^s(j)} \mid x_j, \beta^s) \prod_{j \notin J^s_{\mathrm{D}}} Q^s_{\mathrm{D}}(x_j).\end{aligned} \tag{A.1}$$

PROOF Given the object state array $\mathbf{x}$, the probability $\Pr(n^s, J^s_{\mathrm{D}} \mid \mathbf{x})$ that the subset of detected objects is J^s_{D} and the total number of measurements n^s reflects an additional $n^s_{\mathrm{FA}} = n^s - n^s_{\mathrm{D}}$ false alarms is $\rho^s_{\mathrm{FA}}(n^s_{\mathrm{FA}})$ times the product of $P^s_{\mathrm{D}}(x_j)$ and $Q^s_{\mathrm{D}}(x_j)$ over detected and undetected objects j, respectively. Given J^s_{D} and n^s, there are $n^s!/n^s_{\mathrm{FA}}!$ equally likely mappings a^s, so $\Pr(a^s \mid \mathbf{x}) = \Pr(n^s, J^s_{\mathrm{D}} \mid \mathbf{x}) n^s_{\mathrm{FA}}!/n^s!$. The probability density of the measurement array $\mathbf{z}^s$ given a^s, $\mathbf{x}$, β^s, and ξ is the product of the individual likelihood functions $L^s(z^s_i \mid x_j, \beta^s)$ for the detections $i = a^s(j)$ times the product of $L^s_{\mathrm{FA}}(z^s_i \mid \beta^s, \xi)$ for the false alarms. Equation (A.1) now follows from $\Pr(\mathbf{z}^s, a^s \mid \mathbf{x}, \beta^s, \xi) = \Pr(\mathbf{z}^s \mid a^s, \mathbf{x}, \beta^s, \xi) \Pr(a^s \mid \mathbf{x})$.

Equation (A.1) is identical to (2.1) except that it has a dependence on ξ and an additional false-alarm–related pre-factor. This pre-factor remains intact throughout the derivation of association probability in Section 2, until we arrive at this analog of (2.8):

$$\begin{aligned}\Pr(\mathbf{z}, [a]' \mid \boldsymbol{\beta}, \xi) &= n^s_{\mathrm{FA}}! \rho^s_{\mathrm{FA}}(n^s_{\mathrm{FA}}) \prod_{i \in I^s_{\mathrm{FA}}} L^s_{\mathrm{FA}}(z^s_i \mid \beta^s, \xi) \\ &\quad \times \frac{\gamma^0(n_{\mathrm{D}})}{\prod_{s \in S} n^s!} \prod_{\alpha \in [a]'} P^\alpha(\mathbf{z} \mid \boldsymbol{\beta}, \xi),\end{aligned} \tag{A.2}$$

where the notation $[a]'$ denotes an incomplete association—one for which not all measurements are included in some cluster $\alpha \in [a]'$ (the remaining measurements being false alarms). We may simplify (A.2) by prescribing a natural form for the false alarm likelihood function $L^s_{\mathrm{FA}}(z^s_i \mid \beta^s, \xi)$. To do this, we first define the probability p^s that an object is detected on sensor s and on no others (without regard to what value it produced there):

$$p^s = \int p^0(x \mid \xi) P^s_{\mathrm{D}}(x) \prod_{s' \neq s} Q^{s'}_{\mathrm{D}}(x) dx. \tag{A.3}$$

(Like q, p^s is independent of ξ when detection probability is independent of kinematic state.) We write $L^s_{\mathrm{D}}(z^s_i \mid \beta^s, \xi) = P^s(z^s_i \mid \beta^s, \xi)/p^s$ to denote the probability density of an object producing the value z^s_i on sensor s, given that it was detected by sensor s and no others, where $P^s(z^s_i \mid \beta^s, \xi)$ is given by (2.5) for the special case $\alpha = \{(s,i)\}$.

The likelihood ratio

$$\Lambda^s_{\mathrm{FA}}(z^s_i \mid \beta^s, \xi) = \frac{L^s_{\mathrm{FA}}(z^s_i \mid \beta^s, \xi)}{L^s_{\mathrm{D}}(z^s_i \mid \beta^s, \xi)} \tag{A.4}$$

gives the relative likelihood of the value z^s_i to arise from a false alarm versus a single-sensor object detection. Because the integrals of both its numerator and denominator (over z^s_i) are 1, if $\Lambda^s_{\mathrm{FA}} > 1$ for some value of z^s_i (meaning z^s_i is more likely to be a false alarm), then it must be less than 1 somewhere else. If one has a very clear notion of the characteristics of measurements that have arisen due to false alarms, it is appropriate to model L^s_{FA} directly. Otherwise, it is prudent to model Λ^s_{FA} instead, perhaps assigning it values larger than 1 for wild values of z^s_i, and somewhat less than 1 for more credible values. The simplest, most robust model for Λ^s_{FA} is that it is identically 1. We call this the *non-informative noise* assumption (cf. Section 5.2), because it stipulates that we have no *a priori* knowledge of the false alarm behavior that allows us to distinguish between false alarms and detections, even when we know the true values of the systematic errors β^s and the center ξ of the set of objects. The non-informative noise assumption has the additional benefit of simplifying the association equations.

If we make the non-informative noise assumption, then the effect of false alarms can be encapsulated in a factor $\gamma^0_{\mathrm{FA}}(n^s_{\mathrm{FA}})$, defined by

$$\gamma^0_{\mathrm{FA}}(n^s_{\mathrm{FA}}) = n^s_{\mathrm{FA}}! \rho^s_{\mathrm{FA}}(n^s_{\mathrm{FA}}) (p^s)^{-n^s_{\mathrm{FA}}}. \tag{A.5}$$

Given this definition, Equation (A.2) may be expressed more simply.

LEMMA A.2 *Using the non-informative noise assumption for false alarms, the probability density of the measurement arrays* $\mathbf{z}$ *arising according to the incomplete association* $[a]'$ *given the systematic error* β^s*, and the center* ξ *of the prior region of state space is*

$$\Pr(\mathbf{z}, [a]' \mid \boldsymbol{\beta}, \xi) = \prod_{s \in S} \gamma^0_{\mathrm{FA}}(n^s_{\mathrm{FA}}) \frac{\gamma^0(n_{\mathrm{D}})}{\prod_{s \in S} n^s!} \prod_{\alpha \in [a]} P^\alpha(\mathbf{z} \mid \boldsymbol{\beta}, \xi), \tag{A.6}$$

where $[a]$ *represents the completion of the incomplete association* $[a]'$*: i.e.,* $[a]'$ *with clusters* $\{(s,i)\}$ *adjoined for each false alarm measurement.*

PROOF Follows directly from (A.2)–(A.5).

The incomplete association $[a]'$ distinguishes between objects detected on only one sensor and false alarms. Its completion $[a]$ does not, but this can be quite useful in compressing the hypothesis space when one cares only about which measurements to associate. To obtain the probability of a completion $[a]$, we sum (A.6) over all $[a]'$ for which $[a]$ is the completion. This sum acts only on the combinatorial factors $\gamma^0_{\mathrm{FA}}(n^s_{\mathrm{FA}})$ and $\gamma^0(n_{\mathrm{D}})$. To express this sum we need additional notation. Let n^+_{D} be the number of sets $\alpha \in [a]$ with at least two elements, and n^s_1 be the number of measurements on sensor s which occur by themselves in $[a]$. We may write n_{D} in terms of n^+_{D}, by noting that the number of measurements on sensor s which arise from objects detected once only is $n^s_1 - n^s_{\mathrm{FA}}$, so

$$n_{\mathrm{D}} = |[a]'| = n^+_{\mathrm{D}} + \sum_{s\in S}(n^s_1 - n^s_{\mathrm{FA}}). \tag{A.7}$$

Summing (A.6) over all possible associations $[a]'$ yielding the same $[a]$ amounts to summing over all possible number of false alarms on each sensor (i.e., $n^s_{\mathrm{FA}} = 0$ to n^s_1) and accounting for all ways to choose the n^s_{FA} false alarms from n^s_1 measurements. Thus we introduce the quantity

$$\gamma(n^+_{\mathrm{D}}, \mathbf{n}_1) = \sum_{n^1_{\mathrm{FA}}=0}^{n^1_1} \cdots \sum_{n^r_{\mathrm{FA}}=0}^{n^r_1} \gamma^0\left(n^+_{\mathrm{D}} + \sum_{s\in S}(n^s_1 - n^s_{\mathrm{FA}})\right) \times \prod_{s\in S}\left(\binom{n^s_1}{n^s_{\mathrm{FA}}}\gamma^0_{\mathrm{FA}}(n^s_{\mathrm{FA}})\right), \tag{A.8}$$

where $\mathbf{n}_1 = (n^1_1, n^2_1, \ldots, n^r_1)$. The probability density $\Pr(\mathbf{z}, [a] \mid \boldsymbol{\beta}, \xi)$ may now be given by a formula identical to (2.8), but with $\gamma^0(n_{\mathrm{D}})$ replaced by $\gamma(n^+_{\mathrm{D}}, \mathbf{n}_1)$. Thus we arrive at this analog of Theorem 2.4:

THEOREM A.3 *When false alarms may occur, the probability of the association* $[a]$ *given the measurements* $\mathbf{z}$ *is*

$$\Pr([a] \mid \mathbf{z}) = g([a])G(\mathbf{z}, [a])\Pr([a_0] \mid \mathbf{z}), \tag{A.9}$$

where

$$g([a]) = \frac{\gamma(n^+_{\mathrm{D}}, \mathbf{n}_1)}{\gamma(0, \mathbf{n})}, \tag{A.10}$$

and $\mathbf{n}$ *denotes* $(n^1, n^2, \ldots, n^r)$.

PROOF This follows from (A.6), and the definition (A.8).

The significance of Theorem A.3 is that the effect of false alarms is entirely encapsulated in the combinatorial factor $g([a])$, which is independent of the measurement data $\mathbf{z}$. Although (A.10) is considerably more complicated than (2.11), it can be evaluated explicitly and stored, if desired, for any priors ρ^0 and ρ^s_{FA}. More typically, one simply assumes these priors to be Poisson: Section B.3 gives a simple formula for $g([a])$ in this case.

APPENDIX B. PRIOR ON THE NUMBER OF OBJECTS

The discussion preceding Equation (2.4) addresses the violation of Bayesian methodology one makes when estimating prior information from the data. This violation occurs when one makes the popular assumption that the prior distribution of the number of objects $\rho^0(n)$ is Poisson with the mean number of objects estimated from the data. This issue is addressed here. Section B.1 computes the combinatorial factor $g([a])$ for several choices of prior $\rho^0(n)$. Section B.2 demonstrates why the Poisson prior can work well in practice despite its questionable validity. Finally, Section B.3 computes $g([a])$ explicitly for the false alarm case, assuming Poisson priors.

B.1. The Combinatorial Factor

Poisson prior

For the Poisson prior

$$\rho^0(n) = e^{-\nu}\frac{\nu^n}{n!}, \tag{B.1}$$

(2.9) simplifies to

$$\gamma^0(n_D) = e^{\nu(q-1)}\nu^{n_D}, \tag{B.2}$$

and therefore (2.11) reduces to

$$g([a]) = \nu^{n_D - n_T}. \tag{B.3}$$

For this prior, the association probability in the absence of systematic errors (3.2) takes the following, purely multiplicative form

$$\Pr([a] \mid \mathbf{z}) = \Pr([a_0] \mid \mathbf{z}) \prod_{\alpha\in[a]^+} \tilde{R}^\alpha(\mathbf{z}), \tag{B.4}$$

where

$$\tilde{R}^\alpha(\mathbf{z}) = \frac{\nu P^\alpha(\mathbf{z})}{\prod_{(s,i)\in\alpha}\nu P^s(z^s_i)}. \tag{B.5}$$

This form is required for the traditional, efficient solution of the two-sensor association problem, because it allows the cost of an association to be represented as a sum of costs over each $\alpha \in [a]^+$. In the two-sensor case, one can organize these costs into a cost matrix, and find the association with Maximal A Posteriori Probability (the MAP association) [7] using efficient integer programming techniques such as the JVC algorithm [13]. The Poisson prior is the only prior on n which yields this purely multiplicative form [17, 18, 19], which makes it quite convenient to use.

To use the Poisson prior, however, requires that one estimate ν from the data. For constant detection probabilities P^s_{D}, if one considers the probability of the number of tracks on each sensor given ν, then one gets the following Maximum Likelihood Estimator of ν:

$$\hat{\nu} = \frac{n_T}{\sum_{s\in S} P^s_D}. \tag{B.6}$$

This, then, is a reasonable value of ν to use with the Poisson prior in practice.

A useful class of priors

For any nonnegative integer k and any $\lambda \in [0,1)$, the following prior is well defined for $n \in \mathbb{Z}$:

$$\rho^0(n) = CI[n \geq k]\frac{(n-k)!}{n!}\lambda^n, \tag{B.7}$$

where the indicator function $I[n \geq k]$ is 1 when $n \geq k$ and 0 otherwise, and C is a normalizing constant. For this prior, (2.11) reduces to

$$g([a]) = \frac{(n_D-k)!}{(n_T-k)!}(1-q)^{n_T-n_D}, \tag{B.8}$$

in the limit $\lambda \to 1^-$, provided $n_D \geq k$. This prior has a useful interpretation in the cases $k = 0$ and $k = 1$.

Diffuse prior

The diffuse prior is obtained when $k = 0$. This prior weights all possible number of objects equally, and one might think this would be a reasonable, canonical choice for a prior in the absence of information. In this case, (B.8) is

$$g([a]) = \frac{n_D!}{n_T!}(1-q)^{n_T-n_D}. \tag{B.9}$$

This diffuse prior does not have the same claim to impartiality as the diffuse spatial prior, however. The spatial case has a compelling invariance with respect to Euclidean transformations which has no analog here.

Log-diffuse prior

If one had to guess an integer in the absence of any information, the guess 5 would be better than 74,936 because the latter is so specific. Rather than assuming a uniform distribution for n, we assume a uniform distribution for $\log n$, in accordance with Benford's law. This yields the $k = 1$ case of (B.8):

$$g([a]) = \frac{(n_D-1)!}{(n_T-1)!}(1-q)^{n_T-n_D}. \tag{B.10}$$

This is the form of $g([a])$ recommended for general use when the computational issues favoring the Poisson prior are not important.

B.2. Justification of the Poisson prior

If we use the Poisson prior, then $\log(g([a]))$ varies linearly with n_D (aside from a constant offset): $\log(g([a])) = \log(\nu)(n_D - n_T)$. This is the behavior that produces the purely multiplicative form (B.4) for $\Pr([a] \mid \mathbf{z})$. An alternative, then, to using the Poisson prior is to linearize $\log(g([a])$ about some point n_D^* for a prior one believes is appropriate.

The slope of $\log(g([a])$ for the Poisson prior is constant,

$$\frac{d}{dn_D}\log(g([a]) = \log(\nu), \tag{B.11}$$

whereas this slope for the general prior (B.7) is

$$\begin{aligned}\frac{d}{dn_D}\log(g([a]) &= \psi(n_D+1-k) - \log(1-q)\\ &\approx \log\left(\frac{n_D+1/2-k}{1-q}\right),\end{aligned} \tag{B.12}$$

where ψ denotes the digamma function. Therefore the Poisson prior may, in fact, be regarded as a linearization of the general prior (B.7) about the point

$$n_D^* = (1-q)\nu + k - 1/2. \tag{B.13}$$

This is a quite reasonable place to linearize. If ν is a good estimate of the number of objects (such as the MLE value ν given by (B.6)), then $(1-q)\nu$ is a good estimate of the number of detected objects n_D. The offset by $-1/2$ or $1/2$ in the cases $k = 0$ or $k = 1$, respectively, is negligible.

Therefore, although using the Poisson prior with some estimate of ν based on the data violates Bayesian methodology, it results in a formula that is fairly accurate because it is a linearization of a justifiable diffuse or log-diffuse prior on the number of objects. Because it is an artificially tight distribution, however, it will underestimate the probabilities of associations with a very large or a very small numbers of detected objects n_D.

B.3. False Alarm Case

In contrast to $\rho^0(n)$, it is quite natural to assume the number of false alarms on a sensor is Poisson distributed. We let

$$\rho^s_{\mathrm{FA}}(n^s_{\mathrm{FA}}) = e^{-\nu^s_{\mathrm{FA}}}\frac{(\nu^s_{\mathrm{FA}})^{n^s_{\mathrm{FA}}}}{n^s_{\mathrm{FA}}!}. \tag{B.14}$$

For this prior, (A.5) simplifies to

$$\gamma^0_{\mathrm{FA}}(n^s_{\mathrm{FA}}) = e^{-\nu^s_{\mathrm{FA}}}\left(\frac{\nu^s_{\mathrm{FA}}}{p^s}\right)^{n^s_{\mathrm{FA}}}. \tag{B.15}$$

Even with this Poisson assumption, the value of $g([a])$ does not simplify much unless we also assume that $\rho^0(n)$ is Poisson distributed. If we accept (B.1) and (B.14), however, then the complicated quantity $\gamma(n_D^+, \mathbf{n}_1)$ in (A.8) simplifies to

$$\begin{aligned}\gamma(n_D^+, \mathbf{n}_1) = \exp&\left(-\nu(1-q) - \sum_{s\in S}\nu^s_{\mathrm{FA}}\right)\\ &\times \nu^{n_D^+}\prod_{s\in S}\left(\nu + \frac{\nu^s_{\mathrm{FA}}}{p^s}\right)^{n^s_1}.\end{aligned} \tag{B.16}$$

Using this, we may write (A.10) as

$$g([a]) = \nu^{n_D^+}\prod_{s\in S}\left(\nu + \frac{\nu^s_{\mathrm{FA}}}{p^s}\right)^{n^s_1 - n^s}. \tag{B.17}$$

As in the case without false alarms, the Poisson assumption results in the multiplicative form (B.4) for association probability, but with (B.5) replaced by

$$\tilde{R}^{\alpha}(\mathbf{z}) = \frac{\nu P^{\alpha}(\mathbf{z})}{\prod_{(s,i)\in\alpha}\left(\nu + \frac{\nu^s_{\mathrm{FA}}}{p^s}\right)P^s(z^s_i)}. \qquad \text{(B.18)}$$

APPENDIX C. SIMULATING A SPECIFIED NUMBER OF MEASUREMENTS

It would seem that the first step in the simulation of an association scenario would be to sample the total number of objects n from the prior $\rho^0(n)$. In the case of the diffuse and log-diffuse priors described in Appendix B, this is problematic: these priors have infinite mean. This difficulty may be overcome by fixing $\mathbf{n}$, the number of tracks observed on each sensor. Indeed, it is more convenient to gather statistics from runs with a constant value of $\mathbf{n}$ anyway, regardless of whether the prior ρ^0 forces one's hand. One could imagine using a rejection method to sample in this way: i.e., one could sample n from ρ^0, then generate tracks on each sensor in the usual way, but reject any scenario that does not end up with $\mathbf{n}$ tracks on the sensors. This is highly inefficient however. This appendix describes an efficient procedure which is statistically equivalent to the inefficient rejection method.

We begin with the probability of an object being detected on some subset $\bar{\alpha}$ of S and not on any other sensors:

$$P^{\bar{\alpha}} = \int p^0(x \mid \xi)\prod_{s\in\bar{\alpha}} P^s_{\mathrm{D}}(x)\prod_{s\notin\bar{\alpha}} Q^s_{\mathrm{D}}(x)dx. \qquad \text{(C.1)}$$

It is assumed that this result is independent of ξ. In particular, we are interested in two cases. First, when $P^s_{\mathrm{D}}(x)$ is independent of x we have

$$P^{\bar{\alpha}} = \prod_{s\in\bar{\alpha}} P^s_{\mathrm{D}}\prod_{s\notin\bar{\alpha}} Q^s_{\mathrm{D}}. \qquad \text{(C.2)}$$

On the other hand, if P^s_{D} exhibits as dependence on object type, as in Section 5.1.1 or 5.1.2, then

$$P^{\bar{\alpha}} = \sum_t p^0(t)\prod_{s\in\bar{\alpha}} P^s_{\mathrm{D}}(t)\prod_{s\notin\bar{\alpha}} Q^s_{\mathrm{D}}(t), \qquad \text{(C.3)}$$

which generalizes the formula for q given in (5.10).

We define $[\bar{a}]$ to be the *multiset* of values of $\bar{\alpha}$ for the $\alpha \in [a]$. That is, if $\alpha = \{(1,5),(3,2)\}$ (i.e., according to $[a]$ some object produced measurement 5 on sensor 1, measurement 2 on sensor 3, and no others), then $\bar{\alpha} = \{1,3\}$. If there are exactly four $\alpha \in [a]$ for which $\bar{\alpha} = \{1,3\}$, then we write $m(\{1,3\}) = 4$. This multiplicity function m is part of the multiset $[\bar{a}]$. After some manipulation, we find that

$$\Pr([\bar{a}]) = \gamma^0(n_{\mathrm{D}})\prod_{\bar{\alpha}\in[\bar{a}]}\frac{(P^{\bar{\alpha}})^{m(\bar{\alpha})}}{m(\bar{\alpha})!}. \qquad \text{(C.4)}$$

Each $[\bar{a}]$ determines the number of tracks $\mathbf{n}$ on the sensors, so we define $I_{\mathbf{n}}([\bar{a}])$ to be 1 if $[\bar{a}]$ is consistent with $\mathbf{n}$ and 0 otherwise. The conditional probability $\Pr([\bar{a}] \mid \mathbf{n})$ is given by

$$\Pr([\bar{a}] \mid \mathbf{n}) = \frac{I_{\mathbf{n}}([\bar{a}])\Pr([\bar{a}])}{\sum_{[\bar{a}]'} I_{\mathbf{n}}([\bar{a}]')\Pr([\bar{a}]')}. \qquad \text{(C.5)}$$

This formula may be used to randomly generate $[\bar{a}]$ given $\mathbf{n}$. Then, given $[\bar{a}]$, one may choose a value of $[a]$ from the equally likely possibilities.

Having obtained a random association $[a]$, one must then sample a state for each $\alpha \in [a]$, bearing in mind the statistical influence of the fact that the object is detected on the sensors $\bar{\alpha}$ and no others. In the case (C.2) of object-independent detection probabilities P^s_{D}, there is no effect on the state: it may be sampled in the usual fashion. In the case (C.3), where P^s_{D} depends on the object type t, the object type must be sampled more carefully, however. This requires accounting for its dependence on $\bar{\alpha}$. To sample the object type t for an object detected on sensors $s \in \bar{\alpha}$ but no others, one must draw from $\Pr(t \mid \bar{\alpha})$ rather than $p^0(t)$, where

$$\begin{aligned}\Pr(t \mid \bar{\alpha}) &= \frac{p^0(t)\prod_{s\in\bar{\alpha}} P^s_{\mathrm{D}}(t)\prod_{s\notin\bar{\alpha}} Q^s_{\mathrm{D}}(t)}{P^{\bar{\alpha}}} \\ &= \frac{p^0(t)\prod_{s\in\bar{\alpha}} P^s_{\mathrm{D}}(t)\prod_{s\notin\bar{\alpha}} Q^s_{\mathrm{D}}(t)}{\sum_{t'} p^0(t')\prod_{s\in\bar{\alpha}} P^s_{\mathrm{D}}(t')\prod_{s\notin\bar{\alpha}} Q^s_{\mathrm{D}}(t')}.\end{aligned} \qquad \text{(C.6)}$$

When detection probability is a function of object type t, the distribution of object types on detected objects may differ greatly from $p^0(t)$. When formulating an association scenario, it is helpful to know how many objects of each type to expect. The expected value of $n_D(t)$, the number of objects of type t detected, for a given value of $\mathbf{n}$ is

$$\mathbb{E}[n_D(t) \mid \mathbf{n}] = \sum_{[\bar{a}]}\Pr([\bar{a}] \mid \mathbf{n})\sum_{\bar{\alpha}\in[\bar{a}]}\Pr(t \mid \bar{\alpha}). \qquad \text{(C.7)}$$

When P^s_{D} is independent of t, $\Pr(t \mid \bar{\alpha}) = p^0(t)$, so $\mathbb{E}[n_D(t) \mid \mathbf{n}]$ reduces to $\mathbb{E}[n_D \mid \mathbf{n}]p^0(t)$.

APPENDIX D. A CORRELATED GAUSSIAN PRODUCT FORMULA

A standard formula for the product of k Gaussians is

$$\prod_{i=1}^{k}\mathcal{N}(x;m_i,V_i) = \sqrt{|2\pi W|}\,\mathcal{N}(x;\mu,W)\prod_{i=1}^{k}\mathcal{N}(m_i;\mu,V_i), \qquad \text{(D.1)}$$

where

$$W = \left(\sum_{i=1}^{k} V_i^{-1}\right)^{-1} \qquad \text{and} \qquad \mu = W\sum_{i=1}^{k} V_i^{-1}m_i, \qquad \text{(D.2)}$$

and therefore

$$\int \prod_{i=1}^{k} \mathcal{N}(x; m_i, V_i) dx = \sqrt{|2\pi W|} \prod_{i=1}^{k} \mathcal{N}(m_i; \mu, V_i). \tag{D.3}$$

Theorem 4.2 requires the evaluation of a product of k Gaussians which is much more complicated than (D.3). Let

$$v = \int \int \cdots \int \prod_{i=1}^{k} \mathcal{N}\left(\sum_{j=1}^{p} A_{ij} x_j; m_i, V_i \right) dx_1 dx_2 \cdots dx_p, \tag{D.4}$$

where each $x_j \in \mathbb{R}^n$, each $m_i \in \mathbb{R}^m$, each V_i is a symmetric positive definite $m \times m$ matrix, and each A_{ij} is an $m \times n$ matrix. In evaluating (D.4) the product over i becomes a sum over i within the argument of the Gaussian, which leads to the requirement for computing the following critical quantities:

$$U_{jj'} = \sum_{i=1}^{k} A_{ij}^T V_i^{-1} A_{ij'} \qquad \text{for} \quad 1 \le j, j' \le p, \tag{D.5}$$

$$b_j = \sum_{i=1}^{k} A_{ij}^T V_i^{-1} m_i \qquad \text{for } 1 \le j \le p \qquad \text{and} \tag{D.6}$$

$$c = \sum_{i=1}^{k} m_i^T V_i^{-1} m_i. \tag{D.7}$$

These may be used to express the integrand in (D.4) as a quadratic form over $\mathbb{R}^{np}$. We define

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix}, \qquad \mathbf{U} = \begin{pmatrix} U_{11} & U_{12} & \cdots & U_{1p} \\ U_{21} & U_{22} & \cdots & U_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ U_{p1} & U_{p2} & \cdots & U_{pp} \end{pmatrix} \qquad \text{and} \tag{D.8}$$

$$\mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix},$$

so $\mathbf{x}$ and $\mathbf{b}$ are vectors in $\mathbb{R}^{np}$, and $\mathbf{U}$ is an $np \times np$ symmetric positive semi-definite matrix. For a broad class of matrices A_{ij}, $\mathbf{U}$ is invertible and we may therefore define $\boldsymbol{\mu} = \mathbf{U}^{-1}\mathbf{b}$. The integrand of (D.4) may now be written

$$\prod_{i=1}^{k} \mathcal{N}\left(\sum_{j=1}^{p} A_{ij} x_j; m_i, V_i \right) = v \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{U}^{-1}), \tag{D.9}$$

provided $|\mathbf{U}| > 0$, where

$$v = \left(\exp(c - \mathbf{b}^T \mathbf{U}^{-1} \mathbf{b}) (2\pi)^{mk-np} |\mathbf{U}| \prod_{i=1}^{k} |V_i| \right)^{-1/2}. \tag{D.10}$$

D.1 Gaussian Transformations

Theorem 4.3 requires certain formulas for manipulating Gaussians. First, for any invertible $np \times np$ matrix $\mathbf{C}$, a simple algebraic manipulation shows that

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{U}^{-1}) = \|\mathbf{C}\| \mathcal{N}(\mathbf{C}\mathbf{x}; \mathbf{C}\boldsymbol{\mu}, (\mathbf{C}^{-T} \mathbf{U} \mathbf{C}^{-1})^{-1}), \tag{D.11}$$

where $\|\mathbf{C}\|$ denotes the absolute value of the determinant of $\mathbf{C}$.

We now give a formula for integrating (D.9) over x_{t+1} through x_p only for any $0 \le t \le p$. We let $\mathbf{x}_F = (x_1, x_2, \ldots, x_t)$ denote the free variables, and $\mathbf{x}_I = (x_{t+1}, x_{t+2}, \ldots, x_p)$ denote the variables of integration. We may then decompose $\mathbf{U}$ and $\mathbf{b}$ into corresponding blocks:

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_F \\ \mathbf{x}_I \end{pmatrix}, \qquad \mathbf{U} = \begin{pmatrix} \mathbf{U}_{FF} & \mathbf{U}_{FI} \\ \mathbf{U}_{IF} & \mathbf{U}_{II} \end{pmatrix} \qquad \text{and} \qquad \mathbf{b} = \begin{pmatrix} \mathbf{b}_F \\ \mathbf{b}_I \end{pmatrix}, \tag{D.12}$$

where $\mathbf{U}_{FI}^T = \mathbf{U}_{IF}$. We define the following function of $\mathbf{x}_F$,

$$\boldsymbol{\mu}_I' = \mathbf{U}_{II}^{-1}(\mathbf{b}_I - \mathbf{U}_{IF}\mathbf{x}_F), \tag{D.13}$$

as well as the Schur complement [12] of $\mathbf{U}_{II}$ in $\mathbf{U}$ (denoted $\mathbf{U}_{FF}^*$), and related quantities:

$$\mathbf{U}_{FF}^* = \mathbf{U}_{FF} - \mathbf{U}_{FI}\mathbf{U}_{II}^{-1}\mathbf{U}_{IF}, \tag{D.14}$$

$$\mathbf{b}_F^* = \mathbf{b}_F - \mathbf{U}_{FI}\mathbf{U}_{II}^{-1}\mathbf{b}_I \qquad \text{and} \tag{D.15}$$

$$\boldsymbol{\mu}_F^* = \mathbf{U}_{FF}^{*-1}\mathbf{b}_F^*. \tag{D.16}$$

The Gaussian in (D.9) may now be written as

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{U}^{-1}) = \mathcal{N}(\mathbf{x}_F; \boldsymbol{\mu}_F^*, \mathbf{U}_{FF}^{*-1}) \mathcal{N}(\mathbf{x}_I; \boldsymbol{\mu}_I', \mathbf{U}_{II}^{-1}), \tag{D.17}$$

whence

$$\int \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{U}^{-1}) d\mathbf{x}_I = \mathcal{N}(\mathbf{x}_F; \boldsymbol{\mu}_F^*, \mathbf{U}_{FF}^{*-1}). \tag{D.18}$$

The decomposition (D.12) gives an alternative formula for v in which the following quantities in (D.10) are expressed in an alternative fashion:

$$\begin{aligned} \mathbf{b}^T \mathbf{U}^{-1} \mathbf{b} &= \mathbf{b}_I^T \mathbf{U}_{II}^{-1} \mathbf{b}_I + \mathbf{b}_F^{*T} \mathbf{U}_{FF}^{*-1} \mathbf{b}_F^* \qquad \text{and} \\ |\mathbf{U}| &= |\mathbf{U}_{II}| \, |\mathbf{U}_{FF}^*|. \end{aligned} \tag{D.19}$$

These matrix identities may also be obtained directly through the block matrix inversion formulas [12].

REFERENCES

[1] M. Abramowitz and I. A. Stegun
Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables.
Dover, New York, 1964.

[2] J. Areta, Y. Bar-Shalom and M. Levedhal
A hierarchical benchmark association problem in missile defense.
Signal and Data Processing of Small Targets, 2005, 382–393.

[3] Y. Bar-Shalom and H. Chen
Multisensor track-to-track association for tracks with dependent errors.
43rd IEEE Conference on Decision and Control, 2004.

[4] Y. Bar-Shalom and T. E. Fortmann
Tracking and Data Association.
Academic Press, San Diego, 1988.

[5] S. Blackman and R. Popoli
Design and Analysis of Modern Tracking Systems.
Artech, Boston, 1999.

[6] C-Y. Chong and S. Mori
Metrics for feature-aided track association.
9th International Conference on Information Fusion, July 2006.

[7] C. Y. Chong, S. Mori and K-C. Chang
Distributed multitarget multisensor tracking.
In Y. Bar-Shalom (Ed.), *Multitarget-Multisensor Tracking: Advanced Applications*, Artech, Boston, 1990.

[8] O. E. Drummond, D. A. Castañon and M. S. Bellovin
Comparison of 2-D assignment algorithms for sparse, rectangular, floating point, cost matrices.
In Proceedings of the SDI Panels on Tracking, vol. 4, 1990.

[9] J. Ferry
XMAP: Track-to-track association with metric, feature, and target-type data.
9th International Conference on Information Fusion, July 2006.

[10] J. Ferry
Exact bias removal for the track-to-track association problem.
12th International Conference on Information Fusion, July 2009.

[11] J. P. Ferry and L. D. Stone
Track-to-track association with missing features.
In Proceedings MSS Sensor and Data Fusion, May 2005.

[12] Roger A. Horn and Charles R. Johnson
Matrix analysis.
Cambridge University Press, 1985.

[13] R. Jonker and A. Volgenant
A shortest augmenting path algorithm for dense and sparse linear assignment problems.
Computing, **38** (1987), 325–34.

[14] L. M. Kaplan and W. D. Blair
Assignment costs for multiple sensor track-to-track association.
7th International Conference on Information Fusion, June 2004.

[15] L. M. Kaplan, W. D. Blair and Y. Bar-Shalom
Simulation studies of multisensor track association and fusion methods.
In Proceedings of the 2006 IEEE Aerospace Conference, Mar. 2006.

[16] S. Mori and C-Y. Chong
Effects of unpaired objects and sensor biases on track-to-track association: problems and solutions.
In Proceedings MSS Sensor and Data Fusion, vol. 1, 2000, 137–151.

[17] S. Mori and C-Y. Chong
Track-to-track association metric—I.I.D.–Non-Poisson cases.
6th International Conference on Information Fusion, July 2003.

[18] S. Mori and C-Y. Chong
Data association hypothesis evaluation for i.i.d. but non-Poisson multiple target tracking.
APIE Defense & Security 2004 Symposium on Signal and Data Processing for Small Targets, 2004.

[19] S. Mori and C-Y. Chong
Evaluation of data association hypotheses: Non-Poisson I.I.D. cases.
7th International Conference on Information Fusion, 2004.

[20] S. Mori and C-Y. Chong
Hybrid multiple hypothesis data fusion for missile defense system object tracking and discrimination.
In Proceedings MSS Sensor and Data Fusion, May 2005.

[21] K. G. Murty
An algorithm for ranking all the assignments in order of increasing cost.
Operations Research, **16**, 3 (1968), 682–687.

[22] R. Popp, K. Pattipati and Y. Bar-Shalom
m-best *S*-D assignment algorithm with application to multitarget tracking.
IEEE Transactions on Aerospace and Electronic Systems, **37**, 1 (2001), 22–39.

[23] D. B. Reid
An algorithm for tracking multiple targets.
IEEE Transactions on Automatic Control, **AC-24**, 6 (1979), 843–854.

[24] R. A. Singer and A. J. Kanyuck
Computer control of multiple site correlation.
Automatica, **7** (1971), 455–463.

[25] R. W. Sittler
An optimal data association problem in surveillance theory.
IEEE Transactions on Military Electronics, (1964), 125–139.

[26] L. Stone, T. M. Tran and M. L. Williams
Improvement in track-to-track association from using an adaptive threshold.
12th International Conference on Information Fusion, July 2009.

[27] L. D. Stone, M. L. Williams and T. M. Tran
Track-to-track association and bias removal.
SPIE AeroSense International Conference, Apr. 2002.

Jim Ferry received his S.B. in mathematics from M.I.T., and his Sc.M. and Ph.D. in applied mathematics from Brown, where he employed group representation theory to analyze the global energetics of thermal convection.

He worked for six years at the Center for Simulation of Advanced Rockets at the University of Illinois developing algorithms for simulating multiphase flows within solid rocket motors. Since coming to Metron in 2004, he has been the principal investigator of projects exploring such topics as cargo analysis, data association, network detection theory, and group tracking on dynamic networks.

INTERNATIONAL SOCIETY OF INFORMATION FUSION

ISIF Website: http://www.isif.org

2010 BOARD OF DIRECTORS*

2008–2010	2009–2011	2010–2012
Steve Andler	Éloi Bossé	Simon Maskell
Joachim Biermann	Uwe D. Hanebeck	Peter Willett
Mahendra Mallick	Roy Streit	Wolfgang Koch

*Board of Directors are elected by the members of ISIF for a three year term.

PAST PRESIDENTS

Elisa Shahbazian, 2009	W. Dale Blair, 2005	Pramod Varshney, 2001
Darko Musicki, 2008	Chee Chong, 2004	Yaakov Bar-Shalom, 2000
Erik Blasch, 2007	Xiao-Rong Li, 2003	Jim Llinas, 1999
Pierre Valin, 2006	Yaakov Bar-Shalom, 2002	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 of papers in one of the technical areas listed on the inside cover of JAIF are encouraged to submit their papers for peer review at http://jaif.msubmit.net.

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 5 Number 1 June 2010 ISSN 1557-6418