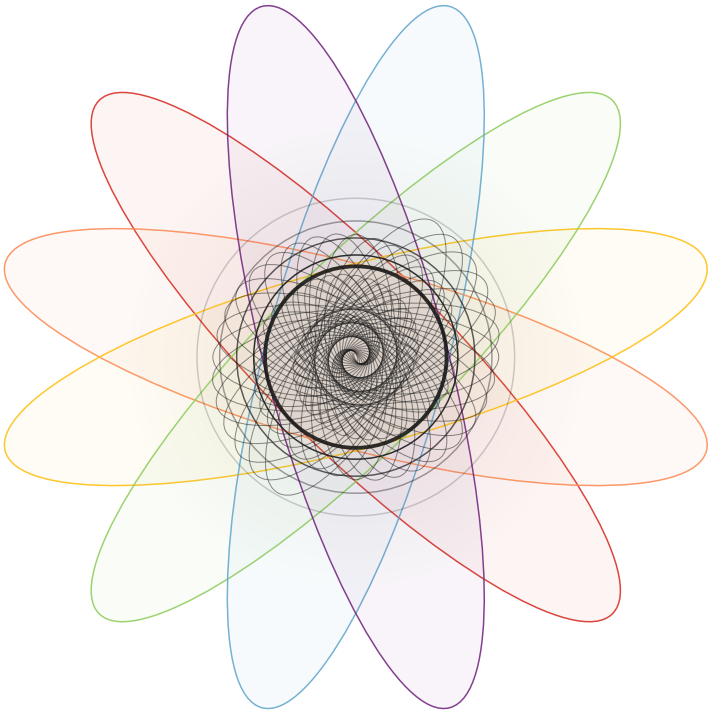
The Bright Side of Decentralized Target Tracking

A Thesis Summary Dedicated to the Practitioner

Robin Forsling



Preface

This technical report is a summary, mainly dedicated to system engineers and other practitioners, of the thesis [1].

> R. Forsling, "The Dark Side of Decentralized Target Tracking: Unknown Correlations and Communication Constraints", Dissertations. No. 2359, Linköping University, Linköping, Sweden, Nov. 2023.

The thesis [1] is the result of my five years as an industrial PhD student, starting in 2018 and ending in 2023, at the Division of Automatic Control at Linköping University. During this time, I have been employed at Saab Aeronautics in Linköping, where I have been working with decision support systems and tactical autonomy.

We deal with optimal dimension reduction. I sometimes get the question *which of the theory and algorithms in the thesis have been, or are to be, implemented in specific products?*. In my opinion, this point of view misses the main motivation for an industrial PhD project. As somebody wise once told me, "An industrial PhD student does not return to the industry with a thesis, but rather in a pair of shoes". The point is that it is not essential that the PhD student develops solutions to specific problems that must be addressed in the design of a certain product. What is instead important is that the PhD student returns with knowledge and expertise in a domain crucial to the company. Of course, it is beneficial if the PhD student can use a problem formulation derived from the industry during the PhD studies. However, the most important part is the academic work, where the PhD student is educated to become a standalone researcher rather than a top system developer. The latter can be done when the PhD studies are finished.

One main goal of the report is to provide the practitioner with intuition and guidelines on how to reason during the design and evaluation of target tracking systems deployed in network-centric operations. Network-centric operations in this context primarily refer to data fusion and target tracking in distributed and decentralized sensor networks. Two aspects of network-centric target tracking are addressed: (i) fusion of estimates that might be correlated to an unknown degree; and (ii) managing a constrained communication resource. I will try to be brief, and hence not all the contents of the thesis are considered. I will also try to be light in the theoretical and mathematical sense. However, some basic mathematics is required for this report to be useful.

Contents

1 Introduction **1**
- 1.1 Target Tracking in Sensor Networks . . . 2
  - 1.1.1 Sensor Network Design . . . 2
  - 1.1.2 Decentralized Target Tracking . . . 3
  - 1.1.3 Correlations . . . 4
  - 1.1.4 Communication . . . 4
- 1.2 Problem Statement . . . 4
- 1.3 Scope . . . 5
- 1.4 Report Outline . . . 5
- 1.5 Source Code Accessibility . . . 5

2 Track Fusion: Design and Evaluation **6**
- 2.1 A Decentralized Single-Target Tracking System . . . 6
  - 2.1.1 State Estimation . . . 7
  - 2.1.2 Track Fusion . . . 8
- 2.2 Evaluation Measures . . . 10
  - 2.2.1 Cramér-Rao Lower Bound . . . 11
  - 2.2.2 Root Mean Squared Error . . . 11
  - 2.2.3 Root Mean Trace . . . 11
  - 2.2.4 Average Normalized Estimation Error Squared . . . 11
  - 2.2.5 Conservativeness Index . . . 11
- 2.3 Track Fusion Evaluation . . . 12
  - 2.3.1 Simulation Specifications . . . 12
  - 2.3.2 Simulation Results and Discussion . . . 13
- 2.4 Summary . . . 14

3 Communication Management: Design and Implementation **15**
- 3.1 Communicating Diagonal Covariance Matrices . . . 15
  - 3.1.1 The Diagonal Covariance Approximation . . . 15
  - 3.1.2 Methods for Preserving Conservativeness . . . 16
- 3.2 Dimension-Reduction Using Eigenvalue Optimization . . . 18
  - 3.2.1 Reducing Dimensionality . . . 18
  - 3.2.2 Communication Considerations . . . 19
  - 3.2.3 Fusion Optimal Dimension Reduction . . . 19
  - 3.2.4 The Common Information Estimate . . . 19
- 3.3 Summary . . . 22

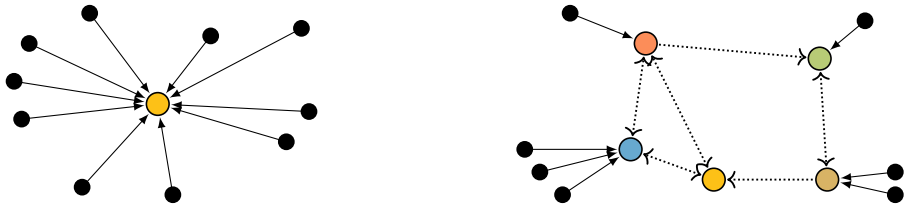
4 Final Remarks **23**

Chapter 1

Introduction

Network-centric operations involve multiple agents that cooperate to achieve common goals. This can also be thought of as a *system of systems* problem. A network-centric battle scene is depicted in Figure 1.1. We see multiple agents, e.g., manned and unmanned aircraft and vessels, ground vehicles, and command & control. These platforms use local *sensors*, e.g., radar and imaging systems, to extract information about targets. The sensor measurements are processed locally in *processing units*, e.g., Kalman filters, where *target estimates* are computed. In target tracking these estimates are often referred to as *tracks*. The agents are also connected using a *datalink* which provides means for sharing local information, e.g., the local tracks.

Figure 1.1. A network-centric battle scene. Multiple agents, both manned and unmanned, of different classes, e.g., ground vehicles, ground stations, aircraft, and ships, use onboard sensor to extract information. The information is then shared among the agents over a datalink for improved situation awareness and decision-making.



(a) Centralized sensor network **(b)** Decentralized sensor network

Figure 1.2. Sensor networks. Black and colored circles resemble sensor nodes and processing units, respectively. The solid and dotted arrows illustrate sensor-to-processing unit and inter-processing unit communication, respectively.

1.1 Target Tracking in Sensor Networks

One crucial benefit of network-centric *target tracking* is the information sharing which allows for improved tracks to be obtained. By utilizing *data fusion* techniques local tracks can be fused with tracks received over the datalink. However, for the data fusion to work properly care must be taken regarding which method is being used for track fusion. In particular, the uncertainty assessment is important. Some methods might compute a too small uncertainty which makes the fused track not trustworthy. Other methods might compute a too large uncertainty such that the fused track becomes useless. Which method to use is essentially a design choice but depends on factors such as the network design, communication pattern, and processing units being used.

The networks considered here are *sensor networks*. A sensor network consists of sensor nodes and processing unit nodes with edges resembling the datalink. In the next we will discuss some crucial aspects of target tracking in sensor networks.

1.1.1 Sensor Network Design

The price paid when using multiple sensor and processing units is increased complexity in terms of computations, communication, and system design. How and to what extent the complexity is increased depends on the network design. In a *centralized sensor network*, see Figure 1.2a, all sensors communicate their measurements to a central processing unit. This arrangement allows for optimal target tracking performance but suffers from high communication demands and single points of failure. Hence, a centralized sensor network is not suitable for critical applications such as the one depicted in Figure 1.1.

The focus in the thesis is target tracking in *decentralized sensor networks* (DSNs). A DSN is illustrated in Figure 1.2b. This type of network is, e.g., realized by wireless ad hoc networks such as wireless sensor networks [2]. According to [3], a DSN is characterized by the following properties:

C1 There is no single point of failure. Failure of one node of the network does not cause a complete network breakdown.

C2 There is no central communication management system. Nodes can, in general, only communicate on a node-to-node basis.

C3 There is no global knowledge about the network topology locally available at the nodes.

In a DSN, measurements are passed to a local processing unit, where local tracks are computed. The local tracks are then exchanged between the processing units to enhance

the tracking quality by fusion. A decentralized design yields the following advantages over a centralized counterpart [4, 5]:

- *Robustness.* Since there is no single point of failure, a decentralized design is inherently fault-tolerant.
- *Modularity.* The network is decomposed into smaller, self-contained subsystems, which reduces the overall complexity and makes the network scalable. This also makes the system design process easier since these components can be developed and maintained separately.
- *Flexibility.* It is possible to connect, update, and disconnect the self-contained subsystems on-the-fly.

These properties are vital for the scene illustrated in Figure 1.1.

Remark 1.1. A DSN is an example of distributed processing. However, the terms decentralized and distributed should not be used interchangeably. While distributed sensor networks have multiple interconnected processing units, there is typically a central coordinator or some global processing result is obtained, e.g., globally available tracks [6]. This work is limited to methods that are able to cope with the constraints imposed by C1–C3, thereby excluding many distributed processing techniques. On the contrary, decentralized methods are typically applicable in distributed processing problems.

1.1.2 Decentralized Target Tracking

Data association and state estimation are fundamental components of the processing unit within a target tracking system. The data association involves the assignment of measurements to tracks, which often requires solving an optimization problem in order to identify the optimal pairings between measurements and tracks based on a specified cost function. In state estimation, the tracks are updated by incorporating the assigned measurements. The existing body of literature pertaining to target tracking, particularly in a centralized setting, is extensive, with numerous publications dedicated to this topic. However, in the context of DSNs, there remains a considerable amount of work to be undertaken to properly address the specific issues that arise in *decentralized target tracking* (DTT).

Figure 1.3 provides a schematic view of the processing unit in a DTT system which contains the following functional components:

1. *Measurement-to-track association.* Assigns measurements to the local tracks.
2. *State estimation.* Updates local tracks with the assigned measurements.
3. *Track-to-track association.* Assigns local tracks to the received tracks.
4. *Track fusion.* Fuses local tracks with the assigned tracks.
5. *Communication management.* Decides which track information to exchange.

These functions are run recursively. The dashed line in Figure 1.3 illustrates that local tracks are predicted and used in the next time step. Roughly speaking, a centralized target tracking system involves steps 1–2 but not steps 3–5.

Remark 1.2. There are other ways to model a DTT system than the functional breakdown of Figure 1.3. For instance, basically all target tracking systems contain some track management logic, which is excluded here. It should, however, be noted that there is no distinct generic model used to illustrate every target tracking system. In the current scope, the model in Figure 1.3 is sufficiently rich.

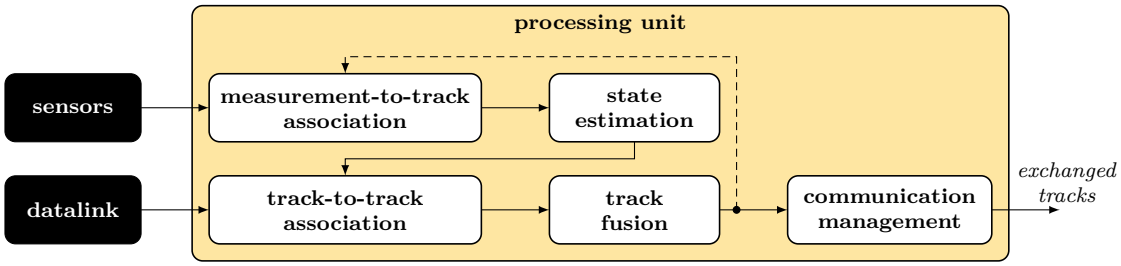


Figure 1.3. A decentralized target tracking system.

Remark 1.3. The operation governing the track fusion is often also referred to as data fusion or sensor fusion.

The DSNs considered here are modeled as a network of agents. Each agent typically has one or several sensors, but in some cases none. Moreover, each agent comprises a self-contained target tracking system and a communication system. The agent can operate on its own under constraints C1–C3 and independently of the other agents.

1.1.3 Correlations

The essence of Figure 1.1 is the huge amount of data that is transferred. Tracks and other estimates are exchanged between agents and further processed. This results in dependencies, i.e., the tracks become correlated with each other. For small, simple networks, the correlations might be tractable. However, in general, it is impossible to maintain explicit knowledge about these correlations among all agents. The issue of having only partial knowledge about correlations is a fundamental aspect of DTT.

In the scope of this thesis, a track is specified by a vector and a covariance matrix. These are the quantities that are being exchanged between agents for track fusion. Simply neglecting the correlations generally leads to undesirable results where the actual covariance matrix of a fused track is underestimated. In the worst case, tracks start to diverge due to incorrect uncertainty assessment [4]. To avoid this, the notion of *conservativeness* has been introduced. A conservative estimator is able to guarantee that the computed covariance matrix is no smaller than the covariance matrix of the actual error.

1.1.4 Communication

Another crucial aspect of DTT problems is communication constraints [7]. State estimates and covariance matrices need to be communicated for a legion of targets and to and from many agents. At some point, the communication link will become a bottleneck, and hence the finite size of the communication resource must be taken into account. There are also situations where the communication needs to be reduced for other reasons, e.g., to be able to operate with a low electromagnetic signature. Hence, there is a need to study DTT under communication constraints [8].

1.2 Problem Statement

The main goal of the thesis is to increase the performance and usability of DTT systems. The research focus is related to the *track fusion* and *communication management* components of Figure 1.3. In particular, the following two subproblems are addressed:

P1 Robust track fusion under unknown correlations. To be useful, the track fusion needs to consider both conservativeness and tracking performance.

P2 Efficient usage of the communication resource. The main design criteria for the communication management are the amount of communicated data, conservativeness, and tracking performance.

1.3 Scope

The scope of this report slightly differs from the thesis. The report takes a more practical approach. In particular, we will focus on: (i) the evaluation of track fusion design; and (ii) how to implement the communication management. (i) and (ii) are related to P1 and P2, respectively.

Many of the theoretical considerations of the thesis are excluded here. We also assume a single-target tracking environment and that the data association process can be ignored.

1.4 Report Outline

This report is organized as follows:

Chapter 2 deals with the track fusion design and evaluation in the single-target tracking case. Three commonly adopted track fusion methods are presented. Several important evaluation measures are defined. The track fusion methods are analyzed using the considered evaluation measures.

Chapter 3 deals with DTT under communication constraints. Methods for preserving conservativeness under certain communication constraints are proposed. We define and discuss a framework for reducing dimensionality for optimal track fusion performance. A resolution is proposed to the problem of only having access to local information when reducing dimensionality.

Chapter 4 concludes this report and specifies which parts of the thesis that are not included here.

1.5 Source Code Accessibility

MATLAB® source code related to the thesis is publicly accessible at `https://github.com/robinforsling/dtt`. This includes the source code for all numerical evaluations and all examples where it is applicable. The repository also contains a DTT simulation environment that can be adapted for testing and evaluating new theory and algorithms.

Chapter 2

Track Fusion: Design and Evaluation

Track fusion is a crucial component of a DTT system. In this chapter we will discuss how the design and evaluation of the track fusion component as part of a DTT system in a single-target configuration. One such configuration is illustrated in Figure 2.1, where multiple agents measure and track a common target, and communicate the local tracks with the other agents for track fusion. In the single-target context, the DTT system reduces to state estimation, track fusion, and communication management. The choice of track fusion method is a compromise between *tracking performance*, which is related to the computed covariance of the track, and *uncertainty assessment*, which is related to conservativeness. It should be stressed that there is no track fusion method that is best in general problems. The most suitable track fusion for a certain problem essentially depends solely on the user needs.

This chapter starts by describing state estimation and track fusion. Several important evaluation measures are then described. We finally discuss how to design and evaluate a track fusion component.

2.1 A Decentralized Single-Target Tracking System

In the single-target tracking case, the measurement-to-track association and track-to-track association are often trivial or already solved and therefore can be disregarded. This case is assumed here. Hence, the DTT system comprises state estimation, track fusion, and communication management. A decentralized single-target tracking system is illustrated in Figure 2.2.

The focus here is the design and evaluation of track fusion, but we need to also briefly

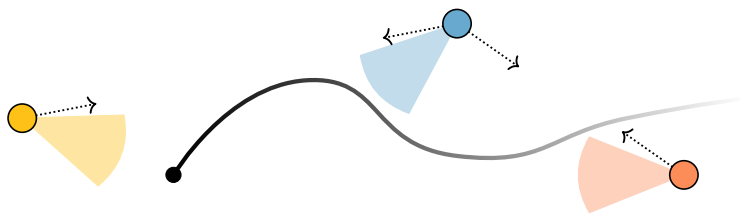


Figure 2.1. A single-target tracking scenario. Multiple agents (colored circles) track a dynamic target (black circle) using sensors (colored cones) and internal processing units. The agents exchange local track estimates over a datalink (dotted arrows) for track fusion.

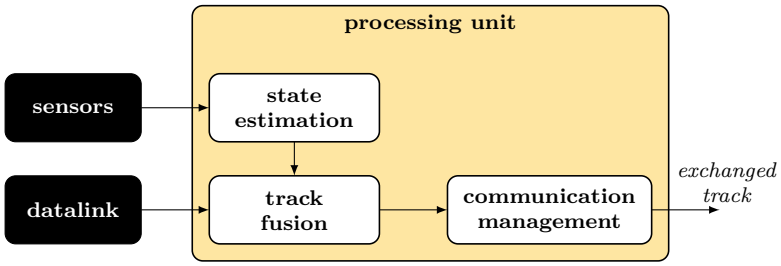


Figure 2.2. A decentralized single-target tracking system.

discuss state estimation as this is an important part as well. However, to be able to analyze track fusion we want the state estimation to be well-tuned in the case of no fusion. If the state estimation is not sufficiently tuned, then it will be difficult to evaluate the track fusion as it becomes very hard to see if it the assessed performance and conservativeness is due to the state estimation or the track fusion. The communication management is discussed in the subsequent chapter.

2.1.1 State Estimation

Let $x_k \in \mathbb{R}^{n_x}$, where n_x is the dimensionality, be the target state at time k. In the thesis there is an explicit differentiation between a random variable and a realization thereof. Random variables are bold face, e.g., $\boldsymbol{a}$, and realizations are normal face, e.g., a. This notation is kept here for consistency. The expectation value of $\boldsymbol{a}$ is denoted $\mathrm{E}(\boldsymbol{a})$. The covariance of $\boldsymbol{a}$ is denoted $\mathrm{cov}(\boldsymbol{a})$ and $\mathrm{cov}(\boldsymbol{a}, \boldsymbol{b})$ is the cross-covariance between $\boldsymbol{a}$ and $\boldsymbol{b}$.

A *state-space model* (SSM) is used to describe the state and measurements of the target. The SSM comprises a *process model* for the target dynamics and a *measurement model* that relates measurements to the target state. A family of discrete time SSMs is given by

$$x_{k+1} = F_k x_k + w_k, \qquad \boldsymbol{w}_k \sim \mathcal{N}(0, Q_k), \tag{2.1a}$$

$$z_k = h(x_k) + e_k, \qquad \boldsymbol{e}_k \sim \mathcal{N}(0, C_k), \tag{2.1b}$$

where F_k is the state transition model, Q_k is the covariance of the process noise $\boldsymbol{w}_k$, z_k is a measurement vector, h is a nonlinear measurement function, and C_k is the covariance of the measurement noise $\boldsymbol{e}_k$. It is assumed that $\mathrm{cov}(\tilde{\boldsymbol{x}}_{k|l}, \boldsymbol{w}_k) = 0$ for all $k \geq l$, where $\tilde{\boldsymbol{x}}_{k|l} = \hat{\boldsymbol{x}}_{k|l} - x_k$. Moreover, it is assumed that $\mathrm{cov}(\boldsymbol{w}_k, \boldsymbol{e}_l) = 0$ for all k, l. Subscript $k|l$ denotes filtered quantities evaluated at time k using measurements up to and including time l.

The SSM in (2.1) is assumed throughout the thesis. The process model is linear, but the measurement model in general involves a nonlinear function. The EKF [9] in Algorithm 1 is used to recursively compute a state estimate of x_k using a time update and a measurement update in each filter recursion. If $h(x_k) = H_k x_k$, then the EKF in Algorithm 1 reduced to the linear Kalman filter (KF, [10]). In this scope $(\hat{x}_{k|k}, P_{k|k})$ is referred to as a *track*. See [1] for more information about relevant process and measurement models.

Filter Tuning

All local filters need to be tuned for their particular applications. For well-designed sensors, this essentially boils down to designing the process noise covariance Q_k. A systematic

Algorithm 1 Extended Kalman Filter

Input: State-space model (2.1), initial values $\hat{x}_{0|0} = \hat{x}_0$ and $P_{0|0} = P_0$

Time update:

$$\hat{x}_{k+1|k} = F_k \hat{x}_{k|k}, \qquad P_{k+1|k} = F_k P_{k|k} F_k^{\mathsf{T}} + Q_k. \tag{2.2}$$

Measurement update:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \left(z_k - h(\hat{x}_{k|k-1}) \right), \qquad P_{k|k} = (I - K_k H_k) P_{k|k-1}, \tag{2.3}$$

where $K_k = P_{k|k-1} H_k^{\mathsf{T}} (H_k P_{k|k-1} H_k^{\mathsf{T}} + C_k)^{-1}$ and $H_k = \left. \frac{\partial h(x')}{\partial x'} \right|_{x'=\hat{x}_{k|k-1}}$.

Output: $(\hat{x}_{k|k}, P_{k|k})$

approach to filter tuning is described in [11]. The authors of [12] propose a methodology for auto-tuning of filters. In this scope, we just assume that the filter has already been tuned.

2.1.2 Track Fusion

In DTT, tracks and covariances are exchanged between agents. The performance of the *track fusion* depends on the fusion rule used to combine tracks. In essence, this fusion rule is an estimator. The following notation is used when describing track fusion:

- The local tracks to be fused are represented by $(y_1, R_1), \ldots, (y_N, R_N)$, where (y_i, R_i) is the local track in Agent i and N is the number of agents. For instance, assume Agent 1 and Agent 2 compute local tracks, e.g., using EKFs, and that these local tracks are fused in Agent 2. Then it is said that Agent 1 transmits (y_1, R_1) to Agent 2 who fuses (y_1, R_1) with its local track (y_2, R_2).
- The estimate computed in the track fusion is denoted $(\hat{x}, P)$.

Track fusion is an instantaneous operation, where two or more tracks evaluated at the same time are merged. This means, in contrast to state estimation, that time indices in general can be disregarded when describing track fusion.

An important special case in the thesis is the fusion of $N = 2$ local estimates given as

$$y_1 = x + v_1, \qquad R_1 = \operatorname{cov}(\boldsymbol{v}_1), \tag{2.4a}$$

$$y_2 = H_2 x + v_2, \qquad R_2 = \operatorname{cov}(\boldsymbol{v}_2), \tag{2.4b}$$

where x is the target state (with time index omitted). The cross-covariance between $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ is denoted $R_{12} = \operatorname{cov}(\boldsymbol{y}_1, \boldsymbol{y}_2)$. Note, in many fusion problems $H_2 = I$, where I is the identity matrix.

Conservative Estimation

An estimator, or fusion method, that computes $(\hat{x}, P)$, where $\hat{x}$ is an estimate of x and P the computed covariance of $\hat{x}$, is *conservative* i.f.f.

$$P - \mathrm{E}(\tilde{\boldsymbol{x}} \tilde{\boldsymbol{x}}^{\mathsf{T}}) \succeq 0, \tag{2.5}$$

where $\tilde{\boldsymbol{x}} = \hat{\boldsymbol{x}} - x$ and $A - B \succeq 0$ means that the difference $A - B$ is positive semidefinite. The notion of conservative has been introduced in distributed/decentralized estimation

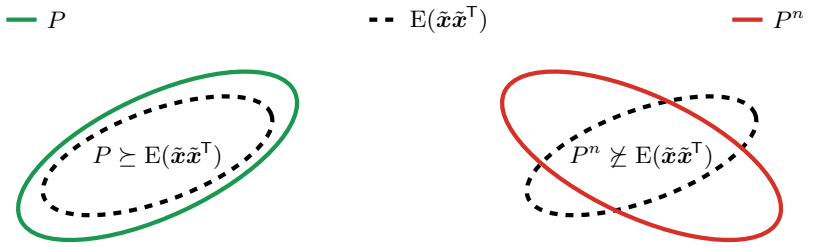


Figure 2.3. A conservative estimate and a non-conservative estimate.

Algorithm 2 Kalman Fuser

Input: (y_1, R_1), (y_2, R_2), and H_2

The estimates are fused according to:

$$\hat{x} = P(R_1^{-1}y_1 + H_2^{\mathsf{T}}R_2^{-1}y_2), \qquad P = (R_1^{-1} + H_2^{\mathsf{T}}R_2^{-1}H_2\ )^{-1}. \tag{2.6}$$

Output: $(\hat{x}, P)$

due to the difficulty of correct uncertainty assessment. This difficulty typically arises because the cross-covariance, e.g., R_{12}, is unknown. This is also what we mean when we say that the "correlations are unknown". If nonzero R_{12} is ignored, then (2.5) typically does not hold and the true covariance $\mathrm{E}(\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^{\mathsf{T}})$ is underestimated.

Figure 2.3 illustrates the ellipses[1] of the covariances of a conservative estimate $(\hat{x}, P)$ and a non-conservative estimate $(\hat{x}^n, P^n)$.

Track Fusion Methods

Three relevant track fusion methods are now stated. These methods will be used to illustrate the track fusion design and evaluation. There are many other methods that might be consider, see the thesis [1] for additional track fusion methods. The thesis also describe a general optimization based framework for track fusion. However, for the sake of simplicity, this framework has been excluded here.

The measurement update of a Kalman filter can be used to fuse two (or more) tracks. This type of fusion is henceforth denoted the *Kalman fuser* (KF), and is defined in Algorithm 2 for the model in (2.4). In fact, if $R_{12} = 0$, then the KF is optimal. If KF is used when $R_{12} \neq 0$, then the fused result is not conservative[2]. However, it is still useful to include KF in the track fusion design—not at least as a baseline method. In particular, KF is often the preferred choice when some decorrelation step has been implemented[3].

Covariance intersection (CI, [13]) is one of the most popular methods for fusing estimates under unknown correlations. CI is provided in Algorithm 3 for the fusion of two estimates defined as in (2.4). It has been shown that CI always provide a conservative estimate given that the estimates to be fused are conservative.

The *largest ellipsoid* (LE, [14]) method[4] is a less conservative alternative to CI. A generalization of the LE method is provided in Algorithm 4.

[1]The ellipsoid of an $n \times n$ symmetric positive definite matrix S is given by the set of points $\mathcal{E}(S) = \{a \in \mathbb{R}^n \,|\, a^{\mathsf{T}}S^{-1}a \leq 1\}$.

[2]The case when KF is used despite $R_{12} \neq 0$ is often referred to as *naïve fusion* or *naïve KF*.

[3]Decorrelation here refers to the process of subtracting an estimate from another such that the new estimate is (approximately) uncorrelated with the estimate(s) to be fused with.

[4]The LE method has also been named *safe fusion* [15] and *ellipsoidal intersection* [16].

Algorithm 3 Covariance Intersection

Input: (y_1, R_1), (y_2, R_2), and H_2

The estimates are fused according to:

$$\hat{x} = P(\omega R_1^{-1} y_1 + (1-\omega) H_2^{\mathsf{T}} R_2^{-1} y_2), \qquad P = (\omega R_1^{-1} + (1-\omega) H_2^{\mathsf{T}} R_2^{-1} H_2)^{-1}, \tag{2.7}$$

with ω given by

$$\underset{\omega}{\text{minimize}} \quad J(P), \tag{2.8}$$

for a matrix increasing function $J(P)$, e.g., $J(P) = \operatorname{tr}(P)$.

Output: $(\hat{x}, P)$

Algorithm 4 The Largest Ellipsoid Method

Input: (y_1, R_1), (y_2, R_2), and H_2

The estimates are fused according to:

1. Transform to the information domain

$$\iota_1 = R_1^{-1} y_1, \qquad \mathcal{I}_1 = R_1^{-1}, \qquad \iota_2 = H_2^{\mathsf{T}} R_2^{-1} y_2, \qquad \mathcal{I}_2 = H_2^{\mathsf{T}} R_2^{-1} H_2.$$

2. Factorize $\mathcal{I}_1 = U_1 \Sigma_1 U_1^{\mathsf{T}}$ and let $T_1 = \Sigma_1^{-\frac{1}{2}} U_1^{\mathsf{T}}$. Factorize $T_1 \mathcal{I}_2 T_1^{\mathsf{T}} = U_2 \Sigma_2 U_2^{\mathsf{T}}$ and let $T_2 = U_2^{\mathsf{T}}$. Transform using $T = T_2 T_1$ according to

$$\iota_1' = T\iota_1, \qquad \mathcal{I}_1' = T\mathcal{I}_1 T^{\mathsf{T}} = I, \qquad \iota_2' = T\iota_2, \qquad \mathcal{I}_2' = T\mathcal{I}_2 T^{\mathsf{T}}.$$

3. For each $i = 1, \ldots, n_x$, compute

$$([\iota']_i, [\mathcal{I}']_{ii}) = \begin{cases} ([\iota_1']_i, 1), & \text{if } 1 \geq [\mathcal{I}_2']_{ii}, \\ ([\iota_2']_i, [\mathcal{I}_2']_{ii}), & \text{if } 1 < [\mathcal{I}_2']_{ii}. \end{cases}$$

Output: $\hat{x} = PT^{-1}\iota'$ and $P = (T^{-1}\mathcal{I}'T^{-\mathsf{T}})^{-1}$

2.2 Evaluation Measures

The track fusion component is evaluated with respect to (w.r.t.) tracking performance and conservativeness. Tracking performance is typically evaluated w.r.t. the *root mean squared error* (RMSE). However, the RMSE requires the true error to be known which is not realistic in an online application. Hence, it is more relevant to evaluate the tracking performance using a measure of the actually computed covariance. For this reason we introduce the *root mean trace* (RMT). It is of interest to analyze how close to the estimation performance is to a theoretical reference or fundamental reference of the estimation performance. Hence, we will also use the *Cramér-Rao Lower Bound* (CRLB). In the literature, conservativeness is often evaluated using the *average normalized estimation error squared* (ANEES). Here we propose the more precise measure *conservativeness index* (COIN) and it is shown that ANEES is the average eigenvalue of the *normalized estimation error squared* (NEES) matrix.

The numerical evaluations in the thesis are based on *Monte Carlo* (MC) simulations. We denote by $\hat{x}_k^i$ the estimate of x_k in the ith MC simulation and P_k^i is the associated covariance, at time k. The number of MC runs is M.

2.2.1 Cramér-Rao Lower Bound

Assume that the true dynamics and measurements of the state x_k are according to the SSM in (2.1). The parametric *Cramér-Rao lower bound* (CRLB, [17]) P^0, of an unbiased estimator of x_k, is computed recursively as [18]

$$P^0_{k+1|k} = F_k P_{k|k} F_k^\mathsf{T} + Q_k, \tag{2.9a}$$

$$P^0_{k|k} = \left((P^0_{k|k-1})^{-1} + (H^0_k)^\mathsf{T} C_k^{-1} H^0_k\right)^{-1}, \tag{2.9b}$$

where $H^0_k = \frac{\partial h(x')}{\partial x'}\Big|_{x'=x_k}$. Often only the position components of P^0 are of interest. This quantity is denoted P^0_{pos} and is given by the upper left block of P^0.

2.2.2 Root Mean Squared Error

The RMSE evaluated at time k is given by

$$\text{RMSE}_k = \sqrt{\frac{1}{M}\sum_{i=1}^{M} (\tilde{x}^i_k)^\mathsf{T} \tilde{x}^i_k} = \sqrt{\text{tr}\left(\hat{P}_k\right)}, \tag{2.10}$$

where $\tilde{x}^i_k = \hat{x}^i_k - x_k$ and

$$\hat{P}_k = \frac{1}{M}\sum_{i=1}^{M} \tilde{x}^i_k (\tilde{x}^i_k)^\mathsf{T}. \tag{2.11}$$

2.2.3 Root Mean Trace

The RMT of the computed covariance is given by

$$\text{RMT}_k = \sqrt{\text{tr}\left(\frac{1}{M}\sum_{i=1}^{M} P^i_k\right)} = \sqrt{\frac{1}{M}\sum_{i=1}^{M} \text{tr}\left(P^i_k\right)}. \tag{2.12}$$

2.2.4 Average Normalized Estimation Error Squared

The *average normalized estimation error squared* (ANEES, [19]) evaluated at time k is given by

$$\text{ANEES}_k = \frac{1}{n_x M}\sum_{i=1}^{M} (\tilde{x}^i_k)^\mathsf{T} (P^i_k)^{-1} \tilde{x}^i_k. \tag{2.13}$$

2.2.5 Conservativeness Index

Since P^i_k is positive definite, it has a Cholesky factorization $P^i_k = L^i_k (L^i_k)^\mathsf{T}$, where L^i_k is invertible. Hence, using the cyclic property of $\text{tr}(\,\cdot\,)$, ANEES can be rewritten as

$$\begin{aligned}
\text{ANEES}_k &= \frac{1}{n_x M}\sum_{i=1}^{M} (\tilde{x}^i_k)^\mathsf{T} (P^i_k)^{-1} \tilde{x}^i_k = \frac{1}{n_x M}\sum_{i=1}^{M} \text{tr}\left((L^i_k)^{-1} \tilde{x}^i_k (\tilde{x}^i_k)^\mathsf{T} (L^i_k)^{-\mathsf{T}}\right) \\
&= \frac{1}{n_x}\,\text{tr}\left(\underbrace{\frac{1}{M}\sum_{i=1}^{M} (L^i_k)^{-1} \tilde{x}^i_k (\tilde{x}^i_k)^\mathsf{T} (L^i_k)^{-\mathsf{T}}}_{C_k}\right),
\end{aligned}$$

where $\mathcal{C}_k$ is here defined as the *NEES matrix*. Since the trace of a matrix equals the sum of the matrix' eigenvalues we have that

$$\text{ANEES}_k = \frac{1}{n_x} \sum_{j=1}^{n_x} \lambda_j(\mathcal{C}_k),$$

where $\lambda_j(\mathcal{C}_k)$ is the jth eigenvalue of $\mathcal{C}_k$. We define COIN as[5]

$$\text{COIN}_k = \lambda_{\max}\left(\mathcal{C}_k\right), \tag{2.14}$$

where $\lambda_{\max}(\,\cdot\,)$ denotes the largest eigenvalue. The eigenvalues are often ordered in descending order. If so, $\text{COIN}_k = \lambda_1(\mathcal{C}_k)$.

From the definition of a conservative estimate in (2.5) it follows that we want $\text{COIN}_k \leq 1$ for the estimator to be conservative.

2.3 Track Fusion Evaluation

We now consider a numerical DTT example and discuss how to reason when choosing the track fusion method. The numerical evaluation is based on RMT and COIN. MATLAB® source code for the evaluation is available at `https://github.com/robinforsling/dtt`.

2.3.1 Simulation Specifications

The scenario design is of course important when evaluating the track fusion. The scenario must be representative in the sense that it excites the intended usage of the developed system. This includes the simulated target which must be simulated with realistic dynamics. In addition, during the MC simulations it is important that the the state estimation is realistic and correctly tuned, and that the communication is according to the deployed datalink protocol. We assume all these criteria are met. The proposed methodology for track fusion design and evaluation is summarized in the textbox below [20].

Track Fusion Design and Evaluation

1. Specify local sensors, local state estimation filters, and communication pattern.
2. Specify the considered track fusion methods.
3. Define a metrics for tracking performance and conservativeness.
4. Define characteristic target trajectories.
5. Tune the local filters for the characteristic trajectories.
6. Using MC simulations, evaluate each fusion method with respect to performance and conservativeness.

The considered scenario is defined in two spatial dimensions and is illustrated in Figure 2.4. Two agents estimate the same target using local sensors and communicate their local tracks with each other. A range-bearing sensor is used in each agent and both agents assume a constant acceleration model for the target dynamics, i.e., $n_x = 6$. Relevant simulation parameters are summarized in Table 2.1. The following track fusion methods are compared:

[5] This is a generalization of the definition used in [1]

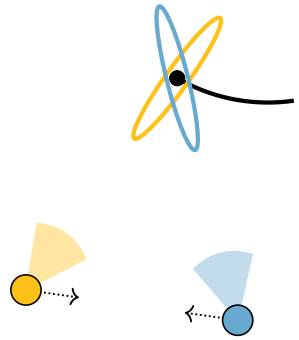


Figure 2.4. Scenario used in the numerical evaluation. The two agents are placed at fixed locations $(-2\,000, 1\,000)$ m and $(5\,000, 0)$ m. The target is initially located at $(3000, 8000)$ m, represented by the black circle, and moves along the black trajectory. The ellipses represent the measurement error covariance of each sensor.

TABLE 2.1
PARAMETERS USED IN THE SIMULATIONS

Parameter	Comment
$d = 2$	spatial dimensionality
$n_x = 6$	state dimensionality
$T_s = 1$	sampling time [s]
$\sigma_w = 4$	standard deviation of process noise [ms$^{-\frac{5}{2}}$]
$\sigma_r = 1\,000$	standard deviation of radial measurement noise [m]
$\sigma_\theta = 1$	standard deviation of azimuth measurement noise [°]
$(-2000, 1000)$	Agent 1 location [m]
$(5000, 0)$	Agent 2 location [m]
$(3000, 8000)$	target initial position [m]
$n_k = 18$	number of time steps
$M = 10\,000$	number of MC runs

- CI. Covariance intersection as defined in Algorithm 3 with $H_2 = I$.
- LE. The largest ellipsoid method as defined in Algorithm 4 with $H_2 = I$.
- NKF. A naïve KF as defined in Algorithm 2 with $H_2 = I$. The NKF neglects any nonzero correlations.

As a reference, a local KF (LKF) is included. The LKF only uses local information, i.e., no track fusion is considered in this case.

2.3.2 Simulation Results and Discussion

The RMT and COIN for one agent, evaluated at time instants where fusion occurs, are given in Figure 2.5. The results for the other agent are similar. RMT is computed for the spatial components and normalized using the CRLB. This means that RMT around 1 is approximately optimal. COIN is computed for the full state x.

Covariance intersection: CI is always conservative w.r.t. COIN. However, it achieves the poorest tracking performance among CI, LE, and NKF. This is expected—CI must sacrifice performance to be able to cope with any degree of correlations. Meanwhile, we see that CI performs much better than the LKF.

The largest ellipsoid method: LE is not always conservative w.r.t., e.g., at $k = 10$, COIN $\approx 1.1 > 1$. However, LE yields a tracking performance that approaches the CRLB.

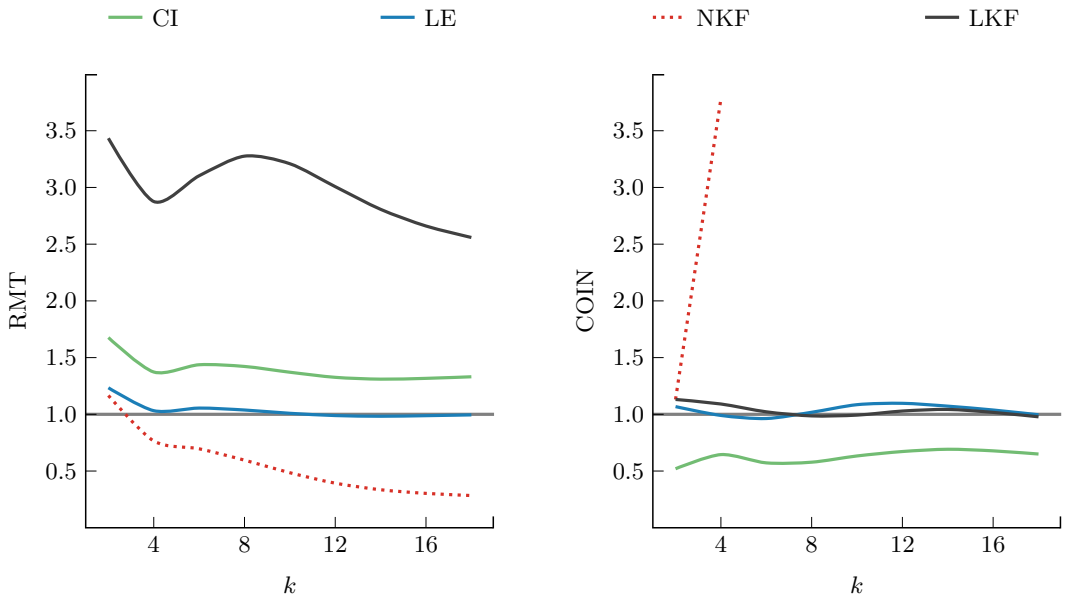


Figure 2.5. Results from the track fusion evaluation.

Hence, depending on the application, it might be acceptable to have COIN slightly above 1 in order to allow for better accuracy in the estimates.

The naïve Kalman fuser: As indicated by the COIN plot, the estimates computed by the NKF quickly diverges and does towards infinity. This is also indicated by the RMT where the NKF attains an RMT smaller than the theoretical lower bound given by the CRLB. The underestimated covariance computed by the NKF essentially originates from the neglected correlations—by ignoring dependencies the NKF reuses previously used information (double counting of information) during track fusion.

2.4 Summary

The purpose of this chapter was to give some basic guidelines on how to design and evaluate the track fusion of a DTT system. Using a numerical evaluation it was demonstrated how different methods can be analyzed when designing the track fusion component in a DTT system. In particular, it was illustrated that the choice of track fusion method essentially boils down to a compromise between tracking performance and conservativeness. At the same time, it should be stressed that a method is not automatically useless just because COIN is above 1. For instance, it might be relevant to use LE since it provides sufficiently reliable estimates w.r.t. the given DTT system requirements. Ultimately, it is up to the system engineers to compromise between different aspects, e.g., performance and conservativeness, to come up with a system design that is acceptable for the end user.

Chapter 3

Communication Management: Design and Implementation

A main advantage of DTT is the sharing of information about targets. However, it is not always possible to communicate all target data. For instance, it might be that only specific parts of tracks are allowed to be exchanged or that we want to minimize the transmitted data to be able to lower the electromagnetic signature. In this chapter, we first deal with the case where only the diagonal entries of covariance matrices are exchanged. We then consider the case where the communication is reduced by transforming the communicated tracks into a lower-dimensional subspace. At the end, we propose a resolution to the problem of only having access to local information when reducing dimensionality.

3.1 Communicating Diagonal Covariance Matrices

Let $y_i = x + v_i$, such that $y_i \in \mathbb{R}^{n_x}$ and $R_i = \text{cov}(\boldsymbol{v}_i) \succ 0$. Assume that Agent i wants to exchange (y_i, R_i). Then n_x parameters for y_i and $n_x(n_x+1)/2$ parameters for R_i must be transmitted, yielding a total of $n_x(n_x+3)/2$ parameters to be transmitted. One way to reduce the communication load is to only communicate the diagonal elements of R_i, hence reducing the total number of transmitted parameters to $2n_x$.

In this section, the communication is constrained such that the complete y_i but only a diagonal representation of R_i are allowed to be exchanged. The problem is referred to as the *diagonal covariance approximation* (DCA).

3.1.1 The Diagonal Covariance Approximation

Let $d_{i,j}$ be the jth diagonal entry of R_i. Let $s_j \geq 1$ be a real-valued scalar. Then D_i and D_i^s are defined as

$$D_i = \text{diag}(d_{i,1}, \ldots, d_{i,n_x}), \qquad D_i^s = \text{diag}(s_1 d_{i,1}, \ldots, s_{n_x} d_{i,n_x}). \tag{3.1}$$

Provided next is an example of the implication of the DCA. Thereafter, the specific problem studied in this section is defined.

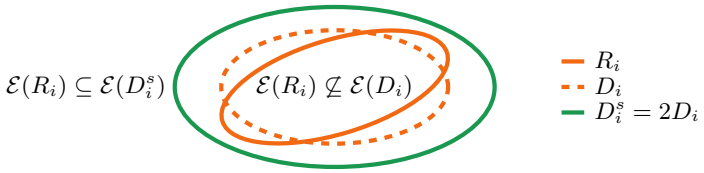


Figure 3.1. Motivating example for the DCA problem. Let (y_i, R_i) be the original local estimate. If R_i is replaced by the diagonal matrix D_i, then the resulting estimate (y_i, D_i) is not conservative. A conservative estimate (y_i, D_i^s) is obtained by replacing R_i by $D_i^s = 2D_i$.

Motivating Example

Let $R_i = \left[\begin{smallmatrix} 4 & 1 \\ 1 & 1 \end{smallmatrix}\right]$, such that $D_i = \operatorname{diag}(4, 1)$. Since $R_i = \operatorname{cov}(\boldsymbol{v}_i) = \mathrm{E}(\boldsymbol{v}_i \boldsymbol{v}_i^\mathsf{T})$, (y_i, R_i) is a conservative estimate. However, since

$$D_i - \mathrm{E}(\boldsymbol{v}_i \boldsymbol{v}_i^\mathsf{T}) = D_i - R_i = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \not\succeq 0,$$

the estimate (y_i, D_i) is *not* conservative. Consider now $s_1 = s_2 = 2$ such that $D_i^s = 2D_i$. In this case,

$$D_i^s - \mathrm{E}(\boldsymbol{v}_i \boldsymbol{v}_i^\mathsf{T}) = \begin{bmatrix} 4 & -1 \\ -1 & 1 \end{bmatrix} \succeq 0,$$

and hence (y_i, D_i^s) is a conservative estimate. This example is illustrated in Figure 3.1.

Problem Formulation

The previous example demonstrates that it is possible to maintain conservativeness under the DCA provided that the $D_i - R_i \neq 0$ is handled in some way, e.g., by using an inflated diagonal approximation D_i^s of R_i. Assume without loss of generality (w.l.o.g.) that Agent 2 transmits a local estimate to Agent 1 under the DCA. The goal is for Agent 1 to fuse the received estimate with its local estimate, where the fused estimate is conservative. How conservativeness is preserved under the DCA depends on what data is being exchanged. The following two options are considered:

D1 Agent 2 transmits (y_2, D_2^s) to Agent 1, where $D_2^s \succeq R_2$. In this case, Agent 2 has already preserved conservativeness, and hence, Agent 1 can use the received estimate directly without any additional action.

D2 Agent 2 transmits (y_2, D_2) to Agent 1. In this case, Agent 1 must explicitly handle that $D_2 \not\succeq R_2$ to ensure conservativeness after track fusion.

If $\lambda_{\max}(R_2) < \infty$, then it is always possible to scale D_2 using finite valued scaling factors $s_1, \ldots, s_{n_x}$ such that $D_2^s \succeq R_2$. However, since information is the inverse of covariance, inflating D_2 implies information is lost. Hence, it is desirable to scale D_2 sufficiently to ensure that $D_2^s \succeq R_2$, but no more than that.

3.1.2 Methods for Preserving Conservativeness

The thesis proposes four methods for preserving conservativeness. The methods that use option D1 are:

- *Eigenvalue based scaling* (DCA-EIG). Agent 2 exchanges (y_2, D_2^s), where $D_2^s = s^\star D_2$ and $s^\star$ is given in [1, Theorem 3.10].

- *Optimization based scaling* (DCA-OPT). Agent 2 exchanges (y_2, D_2^s), where D_2^s is according to (3.1) and $s_1, \ldots, s_{n_x}$ are the solutions to [1, (3.36)].
- *Diagonal-dominance based scaling* (DCA-DOM). Agent 2 exchanges (y_2, D_2^s), where D_2^s is computed according to [1, (3.38)].

The method that uses option D2 is:

- *Hyperrectangle enclosing* (DCA-HYP). Agent 1 receives (y_2, D_2) and computes a conservative estimate using, e.g., (3.6) or (3.7).

Only DCA-EIG and DCA-HYP are considered in this report.

Eigenvalue Based Scaling

Assume that the scaling factors are restricted as $s_1 = \cdots = s_{n_x} = s$ such that $D_2^s = sD_2$ is obtained by uniform scaling of D_2. Under this restriction, the optimal scaling factor $s^\star$ is given by

$$\begin{aligned} \underset{s}{\text{minimize}} \quad & s \\ \text{subject to} \quad & D_2^s = sD_2 \succeq R_2. \end{aligned} \tag{3.2}$$

The solution to the problem in (3.2) is provided by [1, Theorem 3.10]. Essentially, the theorem states that $s^\star$ is given by the largest eigenvalue of the correlation matrix $C_2 = D_2^{-\frac{1}{2}} R_2 D_2^{-\frac{1}{2}}$.

If DCA-EIG is used, then (y_1, R_1) and (y_2, D_2^s) can be fused directly by the preferred track fusion method.

Hyperrectangle Enclosing

If Agent 1 receives (y_2, D_2) from Agent 2, then Agent 1 needs to inflate D_2 in order to be able to reach a conservatively fused result. However, from the point-of-view of Agent 1, R_2 can be any element in the set

$$\mathcal{A} = \left\{ B \in \mathbb{S}_+^{n_x} \mid [B]_{ii} = d_{2,i} \right\}, \tag{3.3}$$

where $d_{2,i} = [D_2]_{ii}$. The set $\mathcal{A}$ is interpreted as the union $\mathcal{R} = \bigcup_{B \in \mathcal{A}} \mathcal{E}(B)$. In essence, $\mathcal{R}$ is an axis-aligned *hyperrectangle*, where the length of the ith side is $2\sqrt{[R_2]_{ii}}$. The hyperrectangle is illustrated in Figure 3.2 for $n_x = 2$ and $R_2 = \left[\begin{smallmatrix} 4 & 1 \\ 1 & 1 \end{smallmatrix}\right]$.

Agent 1 derives (y_2, D_2^ω) from (y_2, D_2), where D_2^ω needs to satisfy $D_2^\omega \succeq R_2$ in order to preserve conservativeness. However, since Agent 1 does not know which element $B \in \mathcal{A}$ which is equal to R_2, Agent 1 must ensure $D_2^\omega \succeq B, \forall B \in \mathcal{A}$ to be sure that $D_2^\omega \succeq R_2$. In particular, Agent 1 wants to solve

$$\begin{aligned} \underset{s_1,\ldots,s_{n_x} \geq 1}{\text{minimize}} \quad & J(D_2^s) \\ \text{subject to} \quad & D_2^s = \operatorname{diag}\left(s_1 d_{2,1}, \ldots, s_{n_x} d_{2,n_x}\right) \succeq B, \forall B \in \mathcal{A}. \end{aligned} \tag{3.4}$$

The solution to (3.4) is given by the following [1, Theorem 3.12], which provides a specific parametrization D_2^ω of D_2^s, i.e.,

$$D_2^\omega = \operatorname{diag}\left(\frac{d_{2,1}}{\omega_1}, \ldots, \frac{d_{2,n_x}}{\omega_{n_x}}\right), \qquad \omega_i \in (0,1), \qquad \sum_{i=1} \omega_i = 1, \tag{3.5}$$

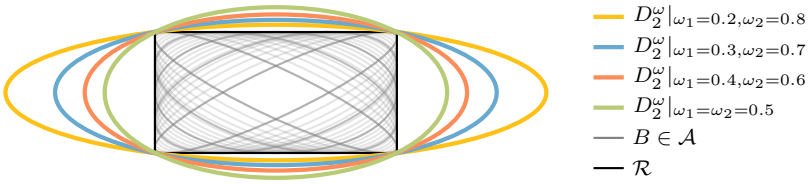


Figure 3.2. Illustration of $\mathcal{A}$. The set $\mathcal{A}$ can be geometrically interpreted as a rectangle $\mathcal{R} = \bigcup_{B\in\mathcal{A}} \mathcal{E}(B)$ if $n_x = 2$. The ellipse $\mathcal{E}(D_2^\omega)$, with D_2^ω given according to (3.5), tightly encloses $\mathcal{R}$ as the boundary of $\mathcal{E}(D_2^\omega)$ intersects all corners of $\mathcal{R}$.

where $d_{2,i} = [D_2]_{ii}$ and $\omega_i = 1/s_i$. Crucial here are the following two properties: (i) $D_2^\omega \succeq B, \forall B \in \mathcal{A}$; and (ii) $\mathcal{E}(D_2^\omega)$ bounds $\mathcal{R}$ tightly. Several D_2^ω, for different values of $\omega_1 = 1-\omega_2$, are illustrated in Figure 3.2, where also the tightness concept is demonstrated.

As noted in [21], D_2^ω can be integrated directly into the CI algorithm. The estimates (y_1, R_1) and (y_2, D_2) are fused conservatively using CI as

$$\hat{x} = P\left(\omega_1 R_1^{-1} y_1 + \sum_{i=1}^{n_x} \omega_{2,i} H_{2,i}^\mathsf{T} [D_2^{-1}]_{ii} [y_2]_i\right), \tag{3.6a}$$

$$P = \left(\omega_1 R_1^{-1} + \sum_{i=1}^{n_x} \omega_{2,i} H_{2,i}^\mathsf{T} [D_2^{-1}]_{ii} H_{2,i}\right)^{-1}, \tag{3.6b}$$

where $\omega_1, \omega_{2,i} \in [0,1]$, $\omega_1 + \sum_{i=1}^{n_x} \omega_{2,i} = 1$, and $H_{2,i} = \begin{bmatrix}\delta_{1i} & \dots & \delta_{n_x i}\end{bmatrix}$.

The same technique can be applied to KF, which, for instance, is relevant if y_1 and y_2 are uncorrelated and Agent 1 receives (y_2, D_2) from Agent 2. In that case

$$\hat{x} = P\left(R_1^{-1} y_1 + \sum_{i=1}^{n_x} \omega_{2,i} H_{2,i}^\mathsf{T} [D_2^{-1}]_{ii} [y_2]_i\right), \tag{3.7a}$$

$$P = \left(R_1^{-1} + \sum_{i=1}^{n_x} \omega_{2,i} H_{2,i}^\mathsf{T} [D_2^{-1}]_{ii} H_{2,i}\right)^{-1}, \tag{3.7b}$$

where $\omega_{2,i} \in [0,1]$ and $\sum_{i=1}^{n_x} \omega_{2,i} = 1$.

3.2 Dimension-Reduction Using Eigenvalue Optimization

Another way to reduce the communicated data is to transmit *dimension-reduced* (DR) estimates. This approach is used in this section and is based on [1, Chapter 5].

3.2.1 Reducing Dimensionality

Assume (y_1, R_1) and (y_2, R_2) are according to

$$y_1 = x + v_1, \qquad R_1 = \operatorname{cov}(\boldsymbol{v}_1), \qquad y_2 = H_2 x + v_2, \qquad R_2 = \operatorname{cov}(\boldsymbol{v}_2), \tag{3.8}$$

with $y_2 \in \mathbb{R}^{n_2}$ and $R_{12} = \operatorname{cov}(\boldsymbol{v}_1, \boldsymbol{v}_2)$. Assume w.l.o.g. that Agent 2 transmits its local estimate to Agent 1. If Agent 2 transmits (y_2, R_2) to Agent 1, then $n_2(n_2+3)/2$ parameters must be exchanged. To reduce the communication load, Agent 2 can instead exchange (y_Ψ, R_Ψ) defined as

$$y_\Psi = \Psi y_2, \qquad R_\Psi = \Psi R_2 \Psi^\mathsf{T}, \tag{3.9}$$

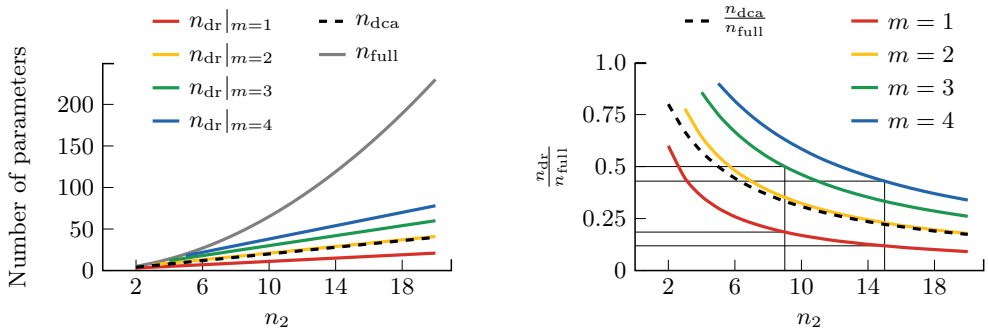


Figure 3.3. Illustration of the communication reduction as a function of n_2 when using DR for different values of m. DCA is included for comparison.

where $\Psi \in \mathbb{R}^{m \times n_2}$, $m < n_2$, and $\operatorname{rank}(\Psi) = m$. The cross-covariance $R_{1\Psi} = \operatorname{cov}(\boldsymbol{v}_1, \Psi \boldsymbol{v}_2) = R_{12}\Psi^{\mathsf{T}}$. The operation in (3.9) is essentially a linear transformation of (y_2, R_2) from a n_2-dimensional space to a m-dimensional subspace.

3.2.2 Communication Considerations

This transformation requires that also Ψ is exchanged. However, by coding[1] (y_Ψ, R_Ψ, Ψ) as described in [1, Section 5.1.4], the number of exchanged parameters can be reduced to $(2mn_2 - m^2 + 3m)/2$. The communication reduction is illustrated in Figure 3.3 as a function of n_2 for different m. DCA is included for comparison.

3.2.3 Fusion Optimal Dimension Reduction

Reducing dimensionality using Ψ can be done in an infinite number of ways. Here, we are interested in reducing dimensionality with a minimum performance loss. Let $(\hat{x}, P)$ be the result of fusing (y_1, R_1) and (y_Ψ, R_Ψ). In this context a *fusion optimal* $\Psi \in \mathbb{R}^{m \times n_2}$, denoted $\Psi^\star$, solves the problem

$$\underset{\Psi}{\text{minimize}} \quad \operatorname{tr}(P). \tag{3.10}$$

The covariance P is specified by the particular track fusion method used. Hence, the optimal Ψ depends on the track fusion method. Algorithms for dimension reduction tailored to specific fusion methods are provided below.

The Generalized Eigenvalue Optimization Method

The framework for fusion optimal dimension reduction is denoted the *generalized eigenvalue optimization* (GEVO) method since it boils down to a generalized eigenvalue problem. GEVO for KF, CI, and LE are given in Algorithm 5, Algorithm 6, and Algorithm 7, respectively.

3.2.4 The Common Information Estimate

The GEVO framework requires that Agent 2 has access to R_1 when computing Ψ. This is not realistic in practice—Agent 2 will only have access to its local information but not to the local information of Agent 1. We will now see how we can replace R_1 by

[1]The message coding involves several technicalities which have been excluded from this report.

Algorithm 5 GEVO-KF

Input: $R_1 \in \mathbb{S}_{++}^{n_x}$, $R_2 \in \mathbb{S}_{++}^{n_2}$, $H_2 \in \mathbb{R}^{n_2 \times n_x}$, and m

1. Let $Q = H_2 R_1^2 H_2^\mathsf{T}$ and $S = H_2 R_1 H_2^\mathsf{T} + R_2$.
2. Compute $\lambda_1, \ldots, \lambda_{n_2}$ and $u_1, \ldots, u_{n_2}$ by solving $Qu = \lambda S u$.
3. Define $\Phi = \mathrm{col}(u_1^\mathsf{T}, \ldots, u_m^\mathsf{T})$, and compute $\Omega = \mathrm{col}(v_1^\mathsf{T}, \ldots, v_m^\mathsf{T})$ such that $v_i^\mathsf{T} v_j = \delta_{ij}$ and $\mathrm{rowspan}(\Omega) = \mathrm{rowspan}(\Phi)$.
4. Compute $\Omega R_2 \Omega^\mathsf{T} = U \Sigma U^\mathsf{T}$ and let $\Psi^\star = U^\mathsf{T} \Omega$.

Output: $\Psi^\star$

Algorithm 6 GEVO-CI

Require: ω_0, $R_1 \in \mathbb{S}_{++}^{n_x}$, $R_2 \in \mathbb{S}_{++}^{n_2}$, $H_2 \in \mathbb{R}^{n_2 \times n_x}$, m, $k = 0$, and ϵ

1. Let $k \leftarrow k + 1$. Compute $\lambda_1, \ldots, \lambda_{n_2}$ and $u_1, \ldots, u_p$ by solving $Qu = \lambda S u$, where $Q = H_2 R_1^2 H_2^\mathsf{T} / \omega_{k-1}^2$ and $S = H_2 R_1 H_2^\mathsf{T} / \omega_{k-1} + R_2/(1 - \omega_{k-1})$. Let $\Phi_k = \mathrm{col}(u_1^\mathsf{T}, \ldots, u_m^\mathsf{T})$, where u_i is a generalized eigenvector associated with λ_i.
2. Let $R_\Phi = \Phi_k R_2 \Phi_k^\mathsf{T}$. Compute ω_k by solving
$$\underset{\omega}{\text{minimize}} \quad \mathrm{tr}\left(\left(\omega R_1^{-1} + (1-\omega) H_2^\mathsf{T} \Phi_k^\mathsf{T} R_\Phi^{-1} \Phi_k H_2 \right)^{-1} \right).$$
3. Let J_k be according to [1, (5.39)]. If $(J_{k-1} - J_k)/J_k > \epsilon$, then go back to step 1. Otherwise continue to step 4.
4. Define $\Phi = \mathrm{col}(u_1^\mathsf{T}, \ldots, u_m^\mathsf{T})$, and compute $\Omega = \mathrm{col}(v_1^\mathsf{T}, \ldots, v_m^\mathsf{T})$ such that $v_i^\mathsf{T} v_j = \delta_{ij}$ and $\mathrm{rowspan}(\Omega) = \mathrm{rowspan}(\Phi)$.
5. Compute $\Omega R_2 \Omega^\mathsf{T} = U \Sigma U^\mathsf{T}$ and let $\Psi = U^\mathsf{T} \Omega$.

Ensure: Ψ

another quantity that can be computed locally at Agent 2. This quantity is referred to as the *common information estimate* (CIE). It should be emphasized that CIE has the additional feature that it can be used to decorrelate estimates. This can be very useful in practice since it allows for simplified logic when it comes to dimension reduction and track fusion.

Computing The Common Information Estimate

CIE is denoted by $(\hat{\gamma}, \Gamma)$ and is filtered in an EKF setting analogously to the local estimate $(\hat{x}, P)$. The local estimate is for now denoted by $(\hat{x}, P)$ instead of (y_2, R_2). These estimates are computed as follows:

- $(\hat{x}, P)$ is predicted at each time step and filtered with local measurements (z, C), and fused with datalink estimates $(y^{\mathrm{dl}}, R^{\mathrm{dl}})$ received from other agents.
- $(\hat{\gamma}, \Gamma)$ is predicted at each time step, fused with (i) datalink estimates $(y^{\mathrm{dl}}, R^{\mathrm{dl}})$ received from others, and (ii) locally computed DR estimates (y_Ψ, R_Ψ) transmitted to other agents.

The same process model is assumed for both $(\hat{x}, P)$ and $(\hat{\gamma}, \Gamma)$. The process noise covariance Q acts as a forgetting factor that ages previously exchanged information [22]. The larger Q is, the faster previously exchanged information is forgotten.

Schematics of the CIE is provided in Figure 3.4, where only the computation of covariances is illustrated. It is suggested that $(\hat{\gamma}, \Gamma)$ is initialized at the same time as $(\hat{x}, P)$

Algorithm 7 GEVO-LE

Require: $R_1 \in \mathbb{S}^{n_x}_{++}$, $R_2 \in \mathbb{S}^{n_2}_{++}$, $H_2 \in \mathbb{R}^{n_2 \times n_x}$, and m

1. Transform into the information domain

$$\mathcal{I}_1 = R_1^{-1}, \qquad \mathcal{I}_2 = H_2^{\mathsf{T}} R_2^{-1} H_2.$$

2. Factorize $\mathcal{I}_1 = U_1 \Sigma_1 U_1^{\mathsf{T}}$ and let $T_1 = \Sigma_1^{-\frac{1}{2}} U_1^{\mathsf{T}}$. Factorize $T_1 \mathcal{I}_2 T_1^{\mathsf{T}} = U_2 \Sigma_2 U_2^{\mathsf{T}}$ and let $T_2 = U_2^{\mathsf{T}}$. Transform using $T = T_2 T_1$ according to

$$\mathcal{I}_1' = T \mathcal{I}_1 T^{\mathsf{T}} = I, \qquad \mathcal{I}_2' = T \mathcal{I}_2 T^{\mathsf{T}}.$$

3. Let D be diagonal. For each $i = 1, \ldots, n_x$ compute

$$[D]_{ii} = \min\left(1, [\mathcal{I}_2']_{ii}\right).$$

4. Let $\mathcal{I}_\gamma = T^{-1} D T^{-\mathsf{T}}$ and $R_{12} = R_1 \mathcal{I}_\gamma H_2^{\mathsf{T}} R_2$. Compute Ψ using Algorithm **??** with inputs R_1, R_2, R_{12}, H_2, and m.

Ensure: Ψ

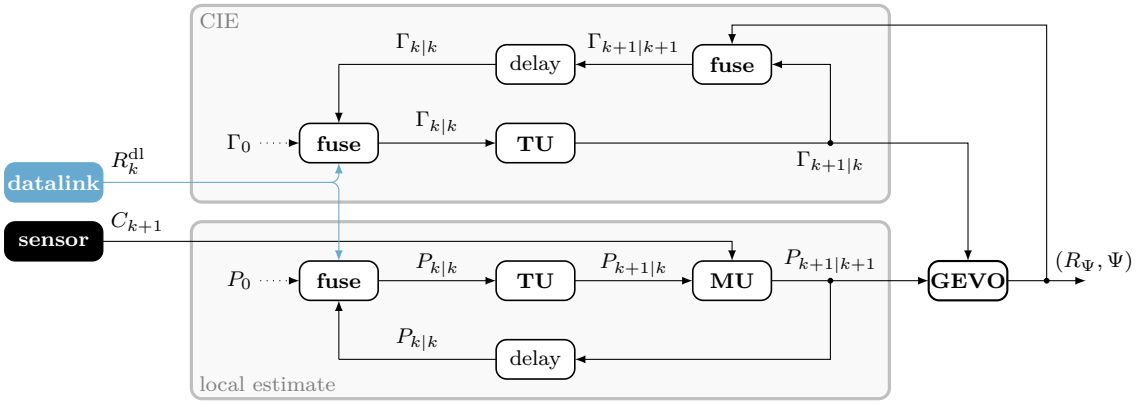


Figure 3.4. Schematics of the CIE methodology. Subscript 0 refers to initial values. Sensor information is used only in the measurement update (MU) of the local estimate.

using $\hat{\gamma}_0 = \hat{x}_0$ while $\Gamma_0 \succ P_0$ is chosen sufficiently large to be consistent with the fact that initially Γ^{-1} is negligible. Initialization of $(\hat{x}, P)$ is done by any standard procedure from target tracking [23].

Using The Common Information Estimate With GEVO

Utilizing $(\hat{\gamma}, \Gamma)$ in GEVO-CI and GEVO-LE is straighforward: R_1 is replaced by Γ and R_2 by P. Then GEVO-CI or GEVO-LE is used. To be able to apply GEVO-KF, $(\hat{x}, P)$ and $(\hat{\gamma}, \Gamma)$ must first be decorrelated. This is accomplished by subtracting $(\hat{\gamma}, \Gamma)$ from $(\hat{x}, P)$. In particular, R_1 is replaced by Γ and R_2 by $(P^{-1} - \Gamma^{-1})^{-1}$, and then GEVO-KF is run. This decorrelation procedure is in practice very useful and is further analyzed in [1, Section 5.4.1]. Table 3.1 summarizes the mapping between the quantities used in this section and the input variables of the different GEVO algorithms.

TABLE 3.1
GEVO INPUT MAPPING

Method	R_1	R_2	R_Ψ
GEVO-KF	Γ	$(P^{-1} - \Gamma^{-1})^{-1}$	$\Psi(P^{-1} - \Gamma^{-1})^{-1}\Psi^\mathsf{T}$
GEVO-CI	Γ	P	$\Psi P \Psi^\mathsf{T}$
GEVO-LE	Γ	P	$\Psi P \Psi^\mathsf{T}$

3.3 Summary

In this chapter we have dealt with communication constraints that might arise in network-centric operations. First we looked at the DCA framework which essentially is about preserving conservativeness under specific communication constraints where only the diagonal entries of covariance matrices are exchanged. We also discussed using DR estimates for communication reduction. To this end, the GEVO framework was proposed. Finally, the CIE was proposed as a resolution to the problem of only having access to local information when using GEVO. No numerical evaluations were included in this chapter, for this, see [1, Chapter 5].

Chapter 4

Final Remarks

The main purpose with this report was to summarize the thesis [1] with a particular focus on practical aspects. We have discussed: (i) the evaluation of track fusion design; and (ii) how to design the communication management in case of communication constraints.

Not Included Contents

The majority of the contents presented in the thesis have been excluded from this report. A list of the excluded material is provided below. The chapter numbering refer to the thesis chapters.

Chapter 2

- mathematical preliminaries
- linear estimation, the best linear unbiased estimation, and conservative estimation
- review of prior work

Chapter 3

- Bar-Shalom–Campo (BSC) formulas for track fusion
- sources of correlations: common process noice and common information
- example of why it in general is impossible to keep track of correlations in DTT applications
- DCA
  - two methods for preserving conservativeness
  - numerical evaluation

Chapter 4

- the conservative linear unbiased estimation (CLUE) framework for optimal conservative estimation
  - problem formalization
  - theoretical bounds

  - theoretical and numerical evaluation
- general CLUE using robust optimization
- specialized CLUE
  - theoretical results related to conservativeness and optimality
  - inverse covariance intersection, and several other track fusion methods tailored to specific correlation structures

Chapter 5

- message coding solution for DR estimates
- the principal component optimization (PCO) method
- GEVO
  - derivation of the GEVO framework
  - GEVO tailored to BSC
  - change of basis and the singular case
  - choosing the parameter m
  - convergence analysis for GEVO-CI
  - theoretical properties of GEVO-LE
  - theoretical comparison to PCO
  - numerical evaluation
- CIE
  - motivation
  - analysis of the decorrelation procedure
  - convergence example
  - numerical evaluation and comparison with the case when global knowledge is available
- dimension reduction for association quality

Bibliography

[1] R. Forsling, "The dark side of decentralized target tracking: Unknown correlations and communication constraints," Dissertations. No. 2359, Linköping University, Linköping, Sweden, Nov. 2023.

[2] C. K. Toh, *Wireless ATM and Ad-Hoc Networks: Protocols and Architectures*. New York, NY, USA: Springer-Verlag, 1997.

[3] S. H. Grime and H. F. Durrant-Whyte, "Data fusion in decentralized sensor networks," *Control Engineering Practice*, vol. 2, no. 5, pp. 849–863, Oct. 1994.

[4] S. J. Julier and J. K. Uhlmann, "General decentralized data fusion with covariance intersection," in *Handbook of Multisensor Data Fusion: Theory and Practice*, M. Liggins, D. Hall, and J. Llinas, Eds. Boca Raton, FL, USA: CRC Press, 2009, ch. 14.

[5] J. K. Uhlmann, "Covariance consistency methods for fault-tolerant distributed data fusion," *Information Fusion*, vol. 4, no. 3, pp. 201–215, Sep. 2003.

[6] F. Castanedo, "A review of data fusion techniques," *The Scientific World Journal*, vol. 2013, pp. 1–19, 2013.

[7] M. A. Razzaque, C. Bleakley, and S. Dobson, "Compression in wireless sensor networks: A survey and comparative evaluation," *ACM Transactions on Sensor Networks*, vol. 10, no. 1, Dec. 2013.

[8] N. Kimura and S. Latifi, "A survey on data compression in wireless sensor networks," in *Proceedings of the IEEE International Conference on Information Technology: Coding and Computing*, vol. 2, Las Vegas, NV, USA, Apr. 2005, pp. 8–13.

[9] A. Jazwinski, *Stochastic processes and filtering theory*. New York, NY, USA: Academic Press, 1970.

[10] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82(Series D), pp. 35–45, 1960.

[11] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York, NY, USA: John Wiley & Sons, Ltd, 2001.

[12] Z. Chen, C. Heckman, S. J. Julier, and N. Ahmed, "Weak in the NEES?: Auto-tuning Kalman filters with Bayesian optimization," in *Proceedings of the 21st IEEE International Conference on Information Fusion*, Cambridge, UK, Jul. 2018, pp. 1072–1079.

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

[14] A. R. Benaskeur, "Consistent fusion of correlated data sources," in *Proceedings of the 28th Annual Conference of the IEEE Industrial Electronics Society*, Sevilla, Spain, Nov. 2002, pp. 2652–2656.

[15] F. Gustafsson, *Statistical Sensor Fusion*. Lund, Sweden: Studentlitteratur, 2018.

[16] J. Sijs, M. Lazar, and P. P. J. v. d. Bosch, "State fusion with unknown correlation: Ellipsoidal intersection," in *Proceedings of the 2010 American Control Conference*, Baltimore, MD, USA, Jun. 2010, pp. 3992–3997.

[17] J. H. Taylor, "The Cramér-Rao estimation error lower bound computation for deterministic nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 24, no. 2, pp. 343–344, Apr. 1979.

[18] C. Fritsche, U. Orguner, and F. Gustafsson, "On parametric lower bounds for discrete-time filtering," in *Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing*, Shanghai, China, Mar. 2016, pp. 4338–4342.

[19] X.-R. Li and Z. Zhao, "Measuring estimator's credibility: Noncredibility index," in *Proceedings of the 9th IEEE International Conference on Information Fusion*, Florence, Italy, Jul. 2006.

[20] R. Forsling, B. Noack, and G. Hendeby, "A quarter-century of covariance intersection: Correlations still unknown?" *IEEE Control Systems Magazine*, accepted for publication in Sep. 2023. Scheduled for the Apr. 2024 issue.

[21] R. Forsling, Z. Sjanic, F. Gustafsson, and G. Hendeby, "Consistent distributed track fusion under communication constraints," in *Proceedings of the 22nd IEEE International Conference on Information Fusion*, Ottawa, Canada, Jul. 2019.

[22] ——, "Communication efficient decentralized track fusion using selective information extraction," in *Proceedings of the 23rd IEEE International Conference on Information Fusion*, Rustenburg, South Africa, Jul. 2020.

[23] S. S. Blackman and R. Popoli, *Design and analysis of modern tracking systems*. Norwood, MA, USA: Artech House, 1999.