Stony Brook University

State Estimation and Fusion in Communication- and Computation-Constrained Long-Haul Sensor Networks

A Dissertation Presented

by

Qiang Liu

to

The Graduate School

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

August 2014

STONY BROOK UNIVERSITY
THE GRADUATE SCHOOL

Qiang Liu

We, the dissertation committee for the above candidate for the
Doctor of Philosophy degree,
hereby recommend acceptance of this dissertation.

Xin Wang, Advisor of Dissertation
Associate Professor, Department of Electrical and Computer Engineering

Thomas Robertazzi, Chairperson of Defense
Professor, Department of Electrical and Computer Engineering

Peter Milder, Assistant Professor
Department of Electrical and Computer Engineering

Nageswara S. V. Rao, Corporate Fellow in Computer Science
Oak Ridge National Laboratory

This dissertation is accepted by the Graduate School.

Charles Taber
Dean of the Graduate School

Abstract of the Dissertation

State Estimation and Fusion in Communication- and Computation-Constrained Long-Haul Sensor Networks

by

Qiang Liu

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

2014

In a long-haul sensor network, sensors are remotely deployed over a large geographical area to perform certain tasks, such as tracking the states of one or more dynamic targets. The communication loss and delay inherent over certain types of long-haul links, such as the satellite channels, pose severe challenges to the successful execution of such tasks. First, free and complete information flow over the network is constrained, as a direct result of the lost and delayed messages over the imperfect communication links. Next, the system is often error- and delay-tolerant to a certain extent, but still the underlying tracking tasks are to be finished within a tightly stipulated deadline with outputs, i.e., the state estimates, that satisfy the system requirement on maximum tolerable error. In addition, due mainly to the limited computational capabilities of the entities within the network, information processing may be inaccurate, inefficient, and time-consuming as well, which in turn negatively affects the final tracking performance. Sensor fusion is a powerful means to aggregate information from multiple sources so that the fused information is expected to possess a much higher

quality than that at any of the sensors. However, the above communication and computation constraints over long-haul sensor networks may severely limit the potential of sensor fusion, resulting in reduced fusion gain and degraded estimation performance. In this dissertation, various approaches are proposed to counteract the effects of imperfect communication and computation in long-haul sensor networks, so that the performance of the tracking tasks in the context of sensor fusion can, even under extremely adverse conditions, be guaranteed to meet the system requirements on both accuracy and timeliness. Some of the solutions are message retransmission and retrodiction, information-driven selective fusion, staggered and asynchronous sensing scheduling, information feedback, as well as learning-based fusion design. Extensive analytical and simulation studies specifically for different types of target tracking applications over satellite-based long-haul sensor networks are carried out to demonstrate the major benefits as well as limitations of adopting these approaches.

Contents

List of Figures

Acknowledgments

First and foremost, I want to express my sincerest gratitude to my advisor, Prof. Xin Wang, for the valuable time, ideas, and funding she contributed all these years to make this dissertation possible. Her enthusiasm and perseverance will undoubtedly continue to inspire me. Thanks are also due to the other dissertation committee members for their time and suggestions. I also thank Dr. Nageswara Rao of Oak Ridge National Laboratory for financially supporting the project, as well as Kathy Brigham and her advisor Prof. Vijayakumar Bhagavatula of Carnegie Mellon University for their collaborative efforts. In addition, I want to thank the past and present members of the WINS Lab for their friendship and help over the years. Last but not least, I would like to dedicate this dissertation to my beloved parents and sister, who have always stood by me and provided tremendous emotional support through all the ups and downs in my life.

Chapter 1

Introduction

The past decade has seen increasing research and development interests and efforts in networked sensing systems. This is due to ever expanding needs for assisting and extending the capabilities of human operators in monitoring large and complex spaces such as airports, train stations, parking lots, bridges, tunnels, coastal areas, and air space. In order to increase the monitoring accuracy and coverage, multiple sensors – each consisting of sensing, data processing, and communication components – are often deployed and networks interconnected. Besides dedicated sensors in conventional wireless sensor networks [4], more and more portable wireless devices are equipped with different types of sensors and can serve as opportunistic sensor nodes. All these entities can be considered components of a networked sensing system. The overarching application of networked sensing systems and the ubiquitous use of smart monitoring devices also contribute to the significant increase in the volumes of data generated, and more importantly, the urgent need for effective and efficient techniques to abstract useful information from such massive amounts of data.

In a typical tracking/monitoring application, a sensor needs to measure certain characteristics of its environment and report its sensing findings either in the form of raw measurements, or estimates that are processed from measurements and serve as an approximation of the system "states" at given time instants. Measurements or estimates provided by a

number of such sensors are often sent over a network to a data processing "fusion center", which collects, extracts, and aggregates the information so that it can generate a global output that can better describe the parameters or attributes of the system. If a networked sensing system is very large, to reduce the computational load and bandwidth consumption, data sent from a subset of sensors may also be pre-processed at local nodes, which could ease the computational burden and storage space requirement at a central location, before being sent to a global fusion node for final processing.

Depending on specific applications, features of different monitoring systems can vary tremendously in terms of coverage, sensor types and capabilities, and energy supply, among others. Behind such diversity, however, is a common theme for nearly all systems, that is, to obtain high quality measures about the underlying system via networked sensing and information processing. Perhaps no other category of applications exemplifies this theme better than target tracking. In order to automatically track and assess the ongoing movement of one or more dynamic targets in the monitored area, a set of networked sensors are used to provide their measurements or estimates of the target location(s) to a fusion center. Limited sensing capabilities of any sensor necessitate fusion of sensing data generated at multiple such sensors so that more accurate "snapshots" approximating the actual movement of the target(s) can be obtained by the fusion center in a timely manner. Real-time detection, tracking, recognition and activity interpretation of moving targets with multiple sensors represent the fundamental issues to be solved in order to develop tracking/monitoring/surveillance systems that are able to autonomously monitor large and complex environments. Accurate and timely detection and estimation of the targets is of critical importance for the reliable operation of such systems.

1.1 Motivation

We are especially interested in one subclass of the networked sensing systems, namely, the long-haul sensor networks. In a long-haul sensor network, sensors are deployed remotely to cover a vast geographical area, which could be a continent or even the entire globe depending on the specific application. Such long-haul sensing applications can be found in a multitude of real-world scenarios [1,2,6,7,15,28,32,39,61,64,67,71], ranging from civilian applications, such as air traffic or satellite orbit control, tracking of greenhouse gas emission and air pollution levels, monitoring of seismic, atmospheric, and oceanic activities and events, localization of autonomous vehicles in intelligent transportation systems, and monitoring of the health of power grids, to military applications, such as the operation of unmanned aerial vehicles (UAVs) and unmanned underwater vehicles (UUVs) for surveillance and reconnaissance, and more broadly battle management and national defense systems. There could be significant variations in the response time requirements of such long-haul sensor networks, from a few seconds in detecting cyber attacks on critical infrastructures to years in detecting global trends in greenhouse gas emissions. We focus on a particular class of long-haul sensor networks that are tasked to detect and track events and/or targets within a timescale of seconds.

The long-haul connections can be established by an infrastructure of extensive terrestrial and submarine fibers and/or satellite links. Under many circumstances, however, satellite links may be the only type of cost-effective medium for such long-range communications because of the prohibitive cost of extending existing submarine and terrestrial fiber connections to rough terrains and sparsely populated areas. Therefore, in this dissertation, we focus on tracking/monitoring applications where data are communicated over satellite links. Fig. 1.1 illustrates the structure of such a satellite-based long-haul sensor network. The measurement data of the remote sensors are used to generate the state estimates of the dynamic target(s). These estimates are then individually sent via long-haul satellite links to the remote fusion center, which, upon successful reception of these estimates, applies a certain fusion rule and

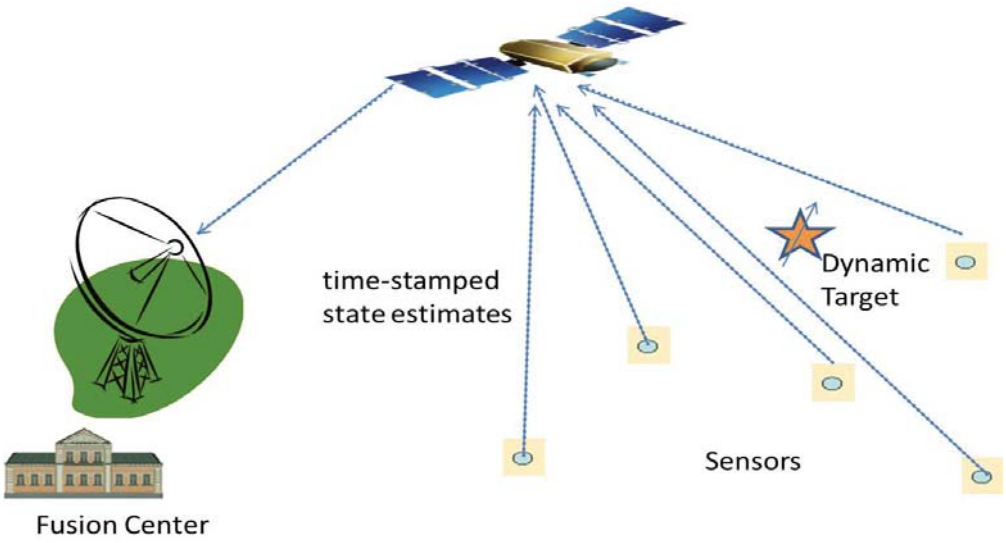


Figure 1.1: Target tracking over satellite links: Sensors generate their state estimates of the dynamic target(s), which are sent over long-haul satellite links to a remote fusion center.

obtains a final fused estimate to be reported.

It is well known that fusion of estimates from different sensors is a viable means to reduce the estimation error [37]. Had the communication links between the sensors and the fusion center been perfect, a fused estimate would normally reach an accuracy level far superior to those of the individual sensor estimates, which is often referred to in the literature as the fusion gain. However, many challenges exist in a long-haul sensor network that may limit the potential of the fusion center to achieve this gain. These challenges can be largely categorized into communication and computation constraints.

In satellite-based long-haul sensor networks, the connection distances to the fusion center are often in the range of several to tens of thousands of miles. Because of the long communication distance, the propagation time of signals is significant. For example, the round-trip time (RTT) for signal propagation with a geostationary earth orbit (GEO) satellite is more than a half second [73]. More importantly, communication over the satellite links is characterized by sporadic high bit-error rates (BERs) and burst losses. Losses either incurred during transmission or resulting from the high BERs could further effectively reduce the number of messages available at the fusion center. The long distance also subjects a high

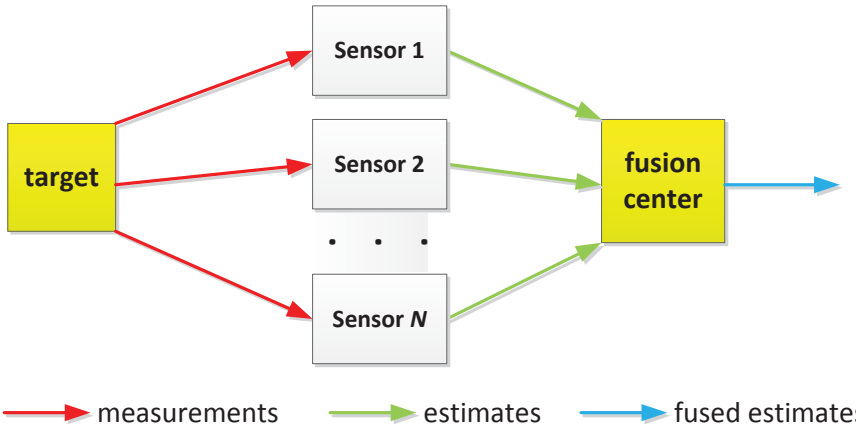


Figure 1.2: Fusion of state estimates generated by a total of N sensors

proportion of data in flight at any given time, thereby increasing the chance that a random delay is experienced due to variations in weather and/or obstacles. The collective effect of the above loss and delay is that the amount of information received by the fusion center is reduced; and as a result, only a portion of the potential fusion gain could be achieved and the quality of the finally fused estimate may be deemed unacceptable by the system operator.

A significant limiting factor in tracking performance is the uncertainty associated with the sensing data and the sensing process itself. More specifically, we consider the quality of the estimates generated by the sensors as well as the specific algorithms that the fusion center applies to fuse the available estimates. Due to the dynamic nature of the system and environment, variable levels of uncertainty exist in ambient noise, targets, and sensors, which could potentially compromise the existing data processing algorithms and lead to much degraded system performance.

Fig. 1.2 schematically shows the information flow in a typical estimate fusion application. Upon taking the measurements from the targets, the sensors generate state estimates individually. The raw measurement data usually come in larger volumes, and, for efficiency purposes, are not directly sent to the fusion center. When such an estimate is sent to the

fusion center, even with hypothetically perfect communications, the fusion center may not be able to generate a reasonably "good" estimate because of the potential errors of the sensors' estimates in the first place. Sensor measurement bias is one of the main factors that cause such performance degradation. In practice, a malfunctioning sensor after extended use, or operating under extreme conditions, or poorly calibrated against noise, may yield estimates that are highly biased. But even without these hardware conditions, a "healthy" sensor may still experience bias as part of its intrinsic behavior. For example, some nonlinear estimation techniques may lead to bias over time. In the context of multiple sensors, measurements from multiple sources often need to be transformed into a common spatial reference frame (coordinate system). Errors occurring during this process, if uncorrected, will result in large tracking errors or even multiple (ghost) tracks for the same target.

In addition, in multi-target tracking, prior to fusion, the estimates generated by the sensors must be associated with the correct targets, which is especially challenging when multiple targets move in close proximity to one another [41]. This is the so-called association problem [13]. The number of targets within the range of a sensor also varies with time, which means that a sensor should first detect the presence of a target before it generates the state estimates for the target. Due to weather, obstacles, and atmospheric disturbances, unwanted echoes may be detected by a sensor, leading to a false alarm, which is also known as a clutter [69]. The above erroneous association and detection results are likely to be exacerbated by the long-haul target-sensor link as well. The drawbacks of the satellite links could work against the very purpose of the underlying task – to promptly and accurately report state estimates – and may result in failure to comply with the requirement on the worst-case estimation error and maximum tolerable reporting delay.

At the other end the system, the fusion center may apply a variety of detection, association, and fusion algorithms to combine the available sensor data and generate the state estimates. However, the performance of these algorithms may differ significantly under variable network conditions, sensor data qualities, and resource constraints such as the available

time to execute these algorithms. The goal of this dissertation is to propose different approaches, mainly to be implemented at the fusion center, that can be used to counteract the effect of the above communication and computation constraints so that the quality of the fused estimates, and the overall tracking performance, can be improved.

1.2 Overview of the Dissertation

The remainder of the dissertation is organized as follows: In Chapter 2, an information metric is proposed as the criterion used by the fusion center to dynamically decide when to stop waiting for the randomly delayed sensor estimates so that both the accuracy and timeliness requirements are accounted for. In Chapter 3, along with active interpolation of the missing data by the fusion center via retrodiction, retransmission is also scheduled to help recover information that was previously lost en route to the fusion center. Chapter 4 explores staggered sensing scheduling so that the network communication delay is used opportunistically by the fusion center to reduce the tracking errors in the conventional scheduling practices. Chapter 5 investigates the major benefits and limitations of sending previously fused state estimates back to the individual sensors under various link conditions and sensor profiles. Having studied mostly closed-form information fusers in previous chapters, in Chapter 6 we propose an artificial neural network learning-based fuser and explore its various enhanced implementations that can potentially improve the tracking performance. Finally, we conclude this dissertation in Chapter 7 with a summary of the work as well as directions for future research.

Chapter 2

Selective Fusion Based on Projected Sensor Information Contribution

2.1 Introduction

In most sensing applications, there are two competing requirements for the fused estimates generated over long-haul sensor networks. First, the accuracy of the fused estimate must exceed that of any single sensor and the estimation error should also be below a predefined maximum tolerable threshold. This generally requires the incorporation of most, if not all, sensor data into the fused estimate. On the other hand, the fused estimate must be generated within a certain, often tight, deadline. For instance, in some military applications, the position of an aircraft or missile is required to be reported within a few seconds. Unfortunately, over long-haul links with severe loss and delay, data from sensors may even fail to arrive at the fusion center by the reporting deadline.

Faced with these challenges, the fusion center can take two types of "extreme" actions. If it simply combines only the data that have arrived, the quality of the fused state is often suboptimal. On the other hand, waiting for all the data to arrive could compromise the timeliness requirement. Accounting for the trade-offs between the estimation accuracy and

reporting timeliness, in this study, we design an *online selective fusion* scheme that enables the fusion center to opportunistically make decisions on when to fuse the estimates to achieve a balance between the two.

Some literature studies have considered state estimation in communication-constrained settings. Fixed or relatively stable arrival delay can be easily handled by the technique of *state augmentation* [73], where adjacent states in time are grouped together to form a "super-state" with a higher dimension. Nevertheless, this can inflate the computation overhead significantly with longer delay. More importantly, with highly random arrival delay, it is difficult to effectively apply this approach as the dimension of the augmented state would keep changing. On the other hand, some literature studies have considered independent packet losses (i.e., a packet either arrives on time or is permanently lost) for one sensor-estimator. For example, in [20], an upper bound of the packet loss rate is derived above which the estimation error will go unbounded.

In multi-sensor state estimation problems, fusion schemes have been proposed under the condition that all packets arrive on time; see [23] and the references therein. [60] attempted to address fusion by combining various sources of degradation (delay, loss, and packet drop) in a probabilistic manner. Their scheme calls for highly intensive computation to run on multiple sensors because of the high dimensionality of the augmented states on top of the already complicated one-sensor case. Besides, the underlying solution requires that the probabilities of encountering different types of degradation are known a priori, which apparently is a very unrealistic assumption.

There have been a series of studies addressing the out-of-sequence-measurement (OOSM) issues. In these problems, due to random latency, sensor data – often in the form of raw measurements – arrive out of order. One of their main focuses is on how to re-incorporate late arrivals. The initial one-step lag problem [10] has been extended to the multi-lag case [11], and the single-OOSM problem in [10, 11] has been extended to the multi-OOSM case [84, 85] as well. Whereas a similar concept of "selective" information processing based on threshold-

ing is proposed in [75], the focus is mainly on the re-incorporation of OOSMs (without loss) over the time domain, rather than information fusion from multiple sensors. Still, there are few multi-sensor studies under adverse link conditions, and the time-domain constraints – in the form of a reporting deadline – have not been accounted for in any of the above works.

Compared to the related studies, our design is not confined to a particular type of packet delay/loss. It is neither possible nor necessary that the fusion center can ascertain why a packet is missing. Instead, we focus on the impact of missing[1] packets on current estimation accuracy and timeliness, regardless of the delay/loss patterns. In this light, our approach is more of an online decision-making process under tight constraints on accuracy and delay.

Our study has a few salient features compared to the related works:

(1) This study is among the very first to address state estimation with severe data loss/delay and strict enforcement on estimation accuracy and timeliness. The delay performance has often been overlooked in previous studies. The fusion center must balance the pros and cons of early versus late reporting to meet the dual requirements on timeliness and accuracy.

(2) A fusion center in long-haul state estimation applications often possesses sufficient storage and computation capabilities to handle large amounts of data sent from the sensors that can be queried and retrieved easily. However, from our performance study results, the storage/computation requirements on the fusion center are fairly minimal because the waiting period is often short following our information-gain-based fusion algorithm.

(3) In our study, prior statistical knowledge about delay and loss is not required. Although such information can be learned over a long period of time by the fusion center, and, once obtained, will prove to be useful, it is not a prerequisite of our selective fusion scheme. Therefore, our scheme can be applied more universally.

(4) We highlight the "online" nature of our scheme as the fusion center determines the best strategy on the go, based on its current accuracy and delay performance of the fused

[1]We use "missing" here since an unavailable message could either be lost or delayed.

estimates. Combining this and the "selective" criteria whereby the fusion center determines the potential contribution from each missing estimate, our scheme works well even under a high degree of system uncertainty.

For the convenience of presentation, in this chapter we mainly consider tracking of a target with its dynamics being characterized by a linear system with zero-mean Gaussian measurement and process noise profiles. Nevertheless, the ideas introduced here, especially the information-based online selective fusion, can be extended to somewhat more complex target models as well, with higher implementation costs. In addition, the results can be easily extended to the scenario when multiple distributed fusion centers are deployed in a long-haul sensor network.

The major contributions of this chapter are listed as follows:

- We study the impact of communication delay and loss on the accuracy of the fused estimate and provide the motivation for selective fusion;
- We propose a novel information metric that the fusion center can utilize for online selective fusion decisions based on the projected differential information contribution, which effectively reduces the reporting delay while meeting the accuracy requirement;
- We apply retrodiction to the selective fusion process so that the recovery of the missing information can be *proactively* expedited and the estimation accuracy improved within the same amount of time under variable degrees of data loss and delay;
- Although the loss and noise profiles are not essential to the online decision making process, we carry out some analysis of the estimation performance from known or learned link loss and delay statistics;
- We perform extensive simulations of a tracking example to demonstrate the effectiveness of our online scheme.

The rest of the chapter is organized as follows. The optimal fusion rule with fully received

state estimates from the sensors is first reviewed in Section 2.2, before we discuss design considerations for a selective fusion algorithm in Section 2.3. We then propose our selective fusion scheme based on the projected differential information gain metric and backward retrodiction in Section 2.4 and provide a complete algorithmic description. In Section 2.5, as an extension to our scheme, the notion of expected information contribution is discussed and its impact on fusion performance is explored. Simulation results are presented and analyzed in Section 2.6 before we conclude the chapter in Section 2.7.

2.2 Optimal Fusion with Fully Received State Estimates from the Sensors

2.2.1 System Model and Kalman Filtering (KF) Basics

We consider the following multi-sensor discrete linear system (the subscript k and superscript i are time and sensor indices respectively):

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{w}_k, \qquad E[\mathbf{w}_k \mathbf{w}_l^T] = \mathbf{Q}\delta_{k-l}, \tag{2.1}$$

$$\mathbf{y}_k^i = \mathbf{H}^i \mathbf{x}_k + \mathbf{v}_k^i, \qquad E[\mathbf{v}_k^i (\mathbf{v}_l^i)^T] = \mathbf{R}^i \delta_{k-l}, \tag{2.2}$$

where $\mathbf{F}$ is the state transition matrix and $\mathbf{H}$ is the measurement matrix. These matrices are often known from the underlying system and time-invariant in tracking applications. The vector $\mathbf{x}$ denotes the state of the target and $\mathbf{y}$ the sensor measurement. The process noise $\mathbf{w}$ and measurement noise $\mathbf{v}$ are white and independent (δ is the Kronecker delta function), whose covariances are $\mathbf{Q}$ and $\mathbf{R}$ respectively. While $\mathbf{Q}$ measures the internal system uncertainty, $\mathbf{R}$ measures the imperfect performance of the sensors as exhibited by errors in their measurements. Note that here these noise statistics are considered time-invariant[2].

[2]An analysis on bounds of errors under varying noise statistics can be similarly derived as in [72].

The well-known Kalman filter (KF) operates recursively on streams of noisy input data to produce statistically optimal estimates of the underlying dynamic system states. At sensor i, the filter evolves recursively according to the following set of equations:

$$\hat{\mathbf{x}}^i_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}^i_{k-1|k-1} \tag{2.3}$$

$$\mathbf{P}^i_{k|k-1} = \mathbf{F}\mathbf{P}^i_{k-1|k-1}\mathbf{F}^T + \mathbf{Q} \tag{2.4}$$

$$\mathbf{K}^i_k = \mathbf{P}^i_{k|k-1}(\mathbf{H}^i)^T(\mathbf{H}^i\mathbf{P}^i_{k|k-1}(\mathbf{H}^i)^T + \mathbf{R}^i)^{-1} = \mathbf{P}^i_{k|k}(\mathbf{H}^i)^T(\mathbf{R}^i)^{-1} \tag{2.5}$$

$$\hat{\mathbf{x}}^i_{k|k} = \hat{\mathbf{x}}^i_{k|k-1} + \mathbf{K}^i_k(\mathbf{y}^i_k - \mathbf{H}^i\hat{\mathbf{x}}^i_{k|k-1}) \tag{2.6}$$

$$\mathbf{P}^i_{k|k} = (\mathbf{I} - \mathbf{K}^i_k\mathbf{H}^i)\mathbf{P}^i_{k|k-1} = ((\mathbf{P}^i_{k|k-1})^{-1} + (\mathbf{H}^i)^T(\mathbf{R}^i)^{-1}\mathbf{H}^i)^{-1} \tag{2.7}$$

In these equations, $\hat{\mathbf{x}}_{k|k-1}$ and $\hat{\mathbf{x}}_{k|k}$ denote respectively the a priori and a posteriori estimate at time k. These notations also apply to $\mathbf{P}$, the *error covariance matrix* of the estimate, defined as $\mathbf{P}^i_k = E[(\hat{\mathbf{x}}^i_k - \mathbf{x}_k)(\hat{\mathbf{x}}^i_k - \mathbf{x}_k)^T]$. Each of the diagonal element in $\mathbf{P}$ measures the mean-square-error (MSE) of the corresponding element state estimate. Eqs. (2.3) and (2.4) form the *prediction* step and Eqs. (2.5)–(2.7) the *correction* step. The Kalman filter gain matrix $\mathbf{K}$ determines the relative weights of the historical data and a new measurement. In Eq. (2.7), $\mathbf{P}^{-1}$ is often called the *information matrix*. Kalman filters are minimum-mean-square-error (MMSE)-optimal as the trace of $\mathbf{P}$ – characterizing the estimation error – at each step is minimized. A more thorough discussion of the Kalman filter basics can be found in [73].

2.2.2 Optimal Fusion of Fully Received Local State Estimates

Let us focus on the information form of the KF correction step Eq. (2.7). From the equation, we have

$$\mathbf{J}^i_k \triangleq (\mathbf{P}^i_{k|k})^{-1} - (\mathbf{P}^i_{k|k-1})^{-1} = (\mathbf{H}^i)^T(\mathbf{R}^i)^{-1}\mathbf{H}^i, \tag{2.8}$$

where $\mathbf{J}_k^i$ is defined as the *information gain matrix* from the sensor i at time k. This $\mathbf{J}$ matrix measures the increase of the information matrix $\mathbf{P}^{-1}$, and indirectly measures the reduction of error $\mathbf{P}$. Similarly, we can define

$$\mathbf{j}_k^i \triangleq (\mathbf{P}_{k|k}^i)^{-1}\hat{\mathbf{x}}_{k|k}^i - (\mathbf{P}_{k|k-1}^i)^{-1}\hat{\mathbf{x}}_{k|k-1}^i = (\mathbf{H}^i)^T(\mathbf{R}^i)^{-1}\mathbf{y}_k^i \tag{2.9}$$

as the *information gain vector* – the difference between the prior and posterior *weighted* estimates. It can be shown that $\mathbf{J}_k^i$ of a real system is semi-positive definite; besides, it is also stable since both $\mathbf{H}$ and $\mathbf{R}$ are time-invariant[3].

In multi-sensor fusion, the updates from the KFs run by the individual sensors are sent to the fusion center so that the global fusion can be performed. If we define the above parameters for the fusion center in a similar manner ("G" denotes "global"), the optimal fusion rule is simply to add up the information gain terms from the n sensors during the correction step:

$$\mathbf{J}_k^G \triangleq (\mathbf{P}_{k|k}^G)^{-1} - (\mathbf{P}_{k|k-1}^G)^{-1} = \sum_{i=1}^{n}\left((\mathbf{P}_{k|k}^i)^{-1} - (\mathbf{P}_{k|k-1}^i)^{-1}\right), \tag{2.10}$$

$$\mathbf{j}_k^G \triangleq (\mathbf{P}_{k|k}^G)^{-1}\hat{\mathbf{x}}_{k|k}^G - (\mathbf{P}_{k|k-1}^G)^{-1}\hat{\mathbf{x}}_{k|k-1}^G = \sum_{i=1}^{n}\left((\mathbf{P}_{k|k}^i)^{-1}\hat{\mathbf{x}}_{k|k}^i - (\mathbf{P}_{k|k-1}^i)^{-1}\hat{\mathbf{x}}_{k|k-1}^i\right). \tag{2.11}$$

Such results have been shown in works such as [19] and [80]; and in [14], this is named the centralized measurement fusion (CMF), as the estimate fusion here is in effect equivalent to the centralized scheme in which the fusion center has received all the raw measurements.

In our setting, the state estimates and their corresponding error covariances, both prior and posterior, are sent to the fusion center. Since the raw measurement data usually come in much larger volumes, sending them directly to the fusion center not only consumes more bandwidth but may also renders transmission more prone to the link delay and loss.

[3]This time-invariance serves as the basis for our selective fusion algorithms in Section 2.4, as the fusion center initially needs to use this steady information gain to determine the potential contribution of a missing packet.

Combining the above global correction step with the global prediction step, which has the very same form as in the one-sensor case (Eqs. (2.3) and (2.4)), we have the evolution of $\mathbf{P}^G$ and $\hat{\mathbf{x}}^G$ at the fusion center as follows:

$$\hat{\mathbf{x}}^G_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}^G_{k-1|k-1} \tag{2.12}$$

$$\mathbf{P}^G_{k|k-1} = \mathbf{F}\mathbf{P}^G_{k-1|k-1}\mathbf{F}^T + \mathbf{Q} \tag{2.13}$$

$$(\mathbf{P}^G_{k|k})^{-1} = (\mathbf{P}^G_{k|k-1})^{-1} + \sum_{i=1}^{n} \left((\mathbf{P}^i_{k|k})^{-1} - (\mathbf{P}^i_{k|k-1})^{-1}\right) \tag{2.14}$$

$$(\mathbf{P}^G_{k|k})^{-1}\hat{\mathbf{x}}^G_{k|k} = (\mathbf{P}^G_{k|k-1})^{-1}\hat{\mathbf{x}}^G_{k|k-1} + \sum_{i=1}^{n} \left((\mathbf{P}^i_{k|k})^{-1}\hat{\mathbf{x}}^i_{k|k} - (\mathbf{P}^i_{k|k-1})^{-1}\hat{\mathbf{x}}^i_{k|k-1}\right) \tag{2.15}$$

These equations constitute the optimal global fusion rule, again, when all the estimates generated by the sensors are successfully received by the fusion center.

2.3 Selective Fusion: Design Considerations

With the remote sensors' state estimates being fully received by the fusion center, the estimation error of the fused estimate is often much lower than that of the state estimates provided by individual sensors. However, the severe delay and loss inherent over the long-haul links may significantly limit the fusion gain. In this section, we propose selective fusion as a capable solution to balance the dual requirements of reporting accuracy and timeliness.

With incomplete data, Eqs. (2.14) and (2.15) can be rewritten as

$$(\mathbf{P}^G_{k|k})^{-1} = (\mathbf{P}^G_{k|k-1})^{-1} + \sum_{i=1}^{n} \mathbb{I}^i_k \mathbf{J}^i_k, \tag{2.16}$$

$$(\mathbf{P}^G_{k|k})^{-1}\hat{\mathbf{x}}^G_{k|k} = (\mathbf{P}^G_{k|k-1})^{-1}\hat{\mathbf{x}}^G_{k|k-1} + \sum_{i=1}^{n} \mathbb{I}^i_k \mathbf{j}^i_k, \tag{2.17}$$

where $\mathbb{I}^i_k = \{0, 1\}$ is the indicator function that describes whether the actual packet sent by Sensor i for time k – which is denoted by P^i_k – is delivered to the fusion center on time and

hereby contributes to the final fusion. One option is that the fusion center simply ignores those missing packets. As a result, with incomplete observation, fewer than n terms are incorporated in Eqs. (2.16)-(2.17) during the correction step and the resulting a posteriori $\mathbf{P}$ is higher – in terms of the elevated diagonal elements in P – than that in the full-observation case. Alternatively, the fusion center can substitute one- or multi-step predicted values – that is, the a priori estimates – for the missing data. However, the effect is exactly the same as that of simply ignoring the missing packets: $\mathbf{P}^i_{k|k} = \mathbf{P}^i_{k|k-1}$ and $\hat{\mathbf{x}}^i_{k|k} = \hat{\mathbf{x}}^i_{k|k-1}$ leading to $\mathbf{J}^i_k = 0$ and $\mathbf{j}^i_k = 0$, respectively (the same can be said for multi-step predictions). Consequently, prediction alone is equivalent to having zero information gain, and may cause the error variance of (some elements of) the state estimates to shoot up within a short amount of time if there are multiple missing packets. Also the unavailability of certain components in Eqs. (2.16)-(2.17) results in sub-optimality of the fuser [14].

Fortunately, the fusion center is often allowed to delay its reporting, up till the reporting deadline, by waiting for the delayed data to arrive. The reporting deadline D_{max} is often set in such a way to reflect the worst-case delay performance the system could tolerate. The fusion center can certainly hold off the finalization of the global estimate for an earlier time till all the packets from that time have arrived; however, the average waiting time could well approach D_{max} whenever there is a single packet loss. For many applications, in which near real-time performance is called for, it is desirable for the fusion center to report its fused estimate as early as possible before the deadline. Not only does waiting passively for any lost and/or delayed packets significantly increase the reporting delay, but there is also a good chance that some of the packets being awaited carry little information to improve the existing level of estimation accuracy.

On the other hand, if the fusion center prematurely terminates the process of waiting for some of the packets with a higher potential to reduce the estimation error, the fused estimate obtained may deviate too much from the true state to be acceptable. Therefore, to maintain a decent accuracy level while not incurring a long reporting delay, it would be more viable

for the fusion center to selectively wait for some missing packets before it decides to finalize the estimate of an earlier time instant. This can be stated in an alternate way: at any time step, the fusion center should decide whether each delayed packet is still "worth" waiting for. By making such online decisions for the pending data, the fusion center can dynamically balance the need for both reporting accuracy and timeliness to finalize the state estimate.

We note that such "*wait-versus-disregard*" decisions should be made online because they are largely determined by the current accuracy level at the fusion center. Generally speaking, if the current error covariance is higher, it is preferable that the fusion center waits longer for the recovery of those missing packets to achieve a higher confidence before sending out the final estimate. Suppose that the elevated error covariance due to an earlier missing packet P_k^i leads the fusion center to initially make a "waiting" decision, as other packets are subsequently received (including the packets from Sensor i itself but generated at different time instants other than k and those sent by other sensors), the information loss from P_k^i may be offset by the gain from these other packets. At a certain point, even when the said packet is still missing, the fusion center may decide to discontinue the wait.

For any missing packet, only by further observing the subsequent arriving packets can the fusion center *opportunistically* decide the cost vs. benefit of deferring its final reporting for one more time step[4]. In the next section, we propose a selective fusion scheme that enables the fusion center to make its fusion decisions based on the current level of estimation accuracy.

A final note is in place before we end this section: Although we loosely use "information gain" and "error reduction" interchangeably as the same concept, quantitative calculations should always follow the latter because in the KF evolution the trace of the error variance matrix is minimized, *not* that the trace of the information matrix is maximized.

[4]This can be regarded as one instance of the principle of *diminishing information return.*

2.4 Selective Fusion With Projected Differential Information Contribution (PRODIC) and Retrodiction

The fusion center apparently cannot predict whether a missing packet will eventually be received; it can, however, *project* the packet's past information gain to the current time and decide whether the potential information contribution to the current time still warrants further waiting for the missing packet. In other words, the information gain past due is measured from the perspective of the current time so that the decrease of information gain as time progresses, due to the arrival of other packets, is accounted for.

In this section, we propose using an information metric to guide the fusion center through the selective waiting and fusion process. In particular, we combine forward projected information gain and backward retrodiction so that the fusion center can obtain an accurate estimate much faster.

2.4.1 PRODIC: the Information Metric

For notational simplicity, we define a function h that links two successive a posteriori $\mathbf{P}$ together:

$$h(\mathbf{X}, \mathbf{Y}) \triangleq (\mathbf{F}\mathbf{X}^{-1}\mathbf{F}^T + \mathbf{Q})^{-1} + \mathbf{Y}. \tag{2.18}$$

Then from Eqs. (2.13) and (2.14), we have

$$\mathbf{P}_{k|k}^{-1} = h(\mathbf{P}_{k-1|k-1}^{-1}, \mathbf{J}_k). \tag{2.19}$$

We name the information metric Projected differential information contribution (PRODIC). It measures the potential information contribution of a delayed packet should it return now. The following steps calculate $\Delta\mathbf{P}_{k,k-d}^{i}$ at time k, which is the PRODIC of the missing packet from Sensor i with the time-stamp $k-d$, that is, P_{k-d}^{i}:

Step 1: Add the information gain[5] $\mathbf{J}^i_{k-d}$ of the missing packet to the information matrix $\mathbf{P}^G_{k-d}$:

$$(\mathbf{P}^G_{k-d,temp})^{-1} = (\mathbf{P}^G_{k-d})^{-1} + \mathbf{J}^i_{k-d}; \tag{2.20}$$

The "temp" in this and the following equations denotes that the associated $\mathbf{P}^G$ is only updated temporarily to obtain the PRODIC of the missing packet which has not actually arrived.

Step 2: Recursively propagate the change of $\mathbf{P}^G_{k-d}$ in Step 1, through the intermediate steps, to the current time k. From time $T_n = k-d+1$, $k-d+2$, ..., up to k, calculate

$$(\mathbf{P}^G_{T_n,temp})^{-1} = h\left((\mathbf{P}^G_{T_n-1,temp})^{-1}, \mathbf{J}^G_{T_n}\right); \tag{2.21}$$

In this step, all the existing $\mathbf{J}^G$ values of these intermediate time steps remain the same.

Step 3: Calculate the differential information gain

$$\Delta\mathbf{P}^i_{k,k-d} = \mathbf{P}^G_k - \mathbf{P}^G_{k,temp}. \tag{2.22}$$

After the recursion in Step 2 proceeds to the current time k, Eq. (2.22) measures the difference between $\mathbf{P}^G_{k,temp}$ – the updated $\mathbf{P}^G_k$ with the *supposed arrival* of the missing packet – and the current $\mathbf{P}^G_k$. Note that $\mathbf{P}^G_k$ has been calculated according to the existing arrivals; no matter how small the original information gain from the missing estimate is, the aggregate error in $\mathbf{P}^G_{k,temp}$ will be better than that in $\mathbf{P}^G_k$. In other words, $\Delta\mathbf{P}^i_{k,k-d} \succ 0$.

The lag d is in general a random variable that depends on the actual arrival history of the recent data; however, in the following analysis, we consider it as the *existing* lag of the next global estimate to be obtained. The PRODIC for each delayed packet of time $k-d$ is calculated separately. At any time instant, it is computationally expensive to consider all

[5]From now on, the conditions in the subscripts are dropped for simplicity since all the variables are considered a posteriori: e.g., $k|k$ is shortened to k.

the possible packet arrival patterns across multiple time steps for all the currently missing messages; there are as many as 2^{dn} different patterns in a d-lag enumeration. By singling out each sensor's contribution, we have reduced the complexity from a brute-force search to a linear order dn. After considering the PRODIC of one missing packet, the fusion center has found the least information to be gained among all the possible arrival patterns that include at least this particular pending packet. Therefore, the PRODIC is actually a conservative measure of the potential information gain from awaiting a particular pending packet.

Following the above calculation, the fusion center should compare the PRODIC value with a cutoff threshold th. If the PRODIC value is larger than the threshold, the fusion center still considers the information carried by the packet important for reducing the estimation error and will continue to wait for it. As long as the reporting deadline has not been reached, the pending global estimate for a time instant will be finalized only when the fusion center decides not to wait for any of the pending packet which contains the corresponding sensor estimate.

Determining the value of the threshold th is again a process of balancing the dual requirements of high reporting accuracy and low latency. In general, the higher the existing error variance is, the more potential information gain can be expected from the same missing packet. The threshold thus in principle should adapt according to varying levels of $\mathbf{P}^G$. For the convenience of implementation, however, it is better to normalize the threshold to a fixed value. We choose the desired proportion of error reduction by the fusion center as the threshold at any time step, so that

$$\frac{tr(\Delta \mathbf{P}^i_{k,k-d})}{tr(\mathbf{P}^G_k)} = 1 - \frac{tr(\mathbf{P}^G_{k,temp})}{tr(\mathbf{P}^G_k)} > th \tag{2.23}$$

implies the packet P^i_{k-d} can potentially improve the current $\mathbf{P}^G_k$ more than the threshold level and thus will be awaited.

The online decisions are largely affected by the availability of the packets from the sensors

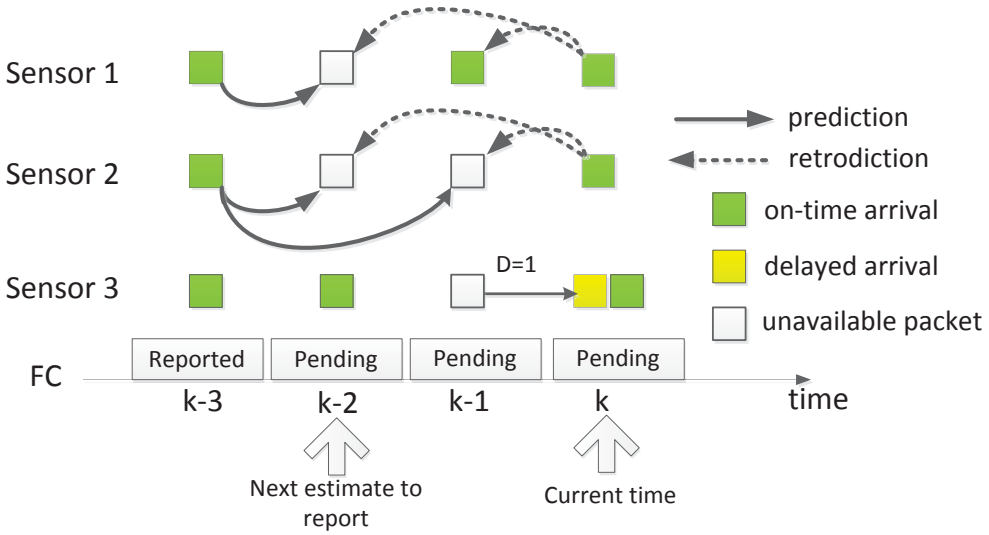


Figure 2.1: Selective fusion example with three sensors and forward prediction and backward retrodiction

with better accuracy guarantees. When the fusion center has received all or most packets from these sensors, any further improvement from the missing ones becomes increasingly small and thus unnecessary beyond a certain point. On the other hand, with the data from these better sensors missing, $\mathbf{P}^G$ can quickly inflate, thereby elevating the normalized PRODIC of these packets to a higher level; what often ensues is the decision to continue waiting for these missing packets. Hence, with heterogeneous sensors, a decision for a sensor with higher accuracy (e.g., "to wait") often overrules that from another one with lower accuracy (e.g., "to disregard") when the two decisions are contradictory.

2.4.2 Information Gain from Retrodiction

From the last subsection, sensors that yield more accurate estimates usually predominate the fusion center's selective waiting decisions. Because of their larger potential information gain, the fusion center generally has to wait longer in case an estimate sent from any of these sensors is delayed. In order to guarantee timely reporting, it is critical for the fusion center to reduce such passive waiting.

Estimation of a target state at a particular time based on measurements collected beyond that time is generally called retrodiction (or smoothing). Traditionally, an earlier *existing* estimate is retrodicted using subsequent measurements so that its accuracy is improved [73]. In this study, we propose a novel use of retrodiction to proactively *interpolate* intermediate missing data. Once one or more subsequent packets of an unavailable one have been received, the fusion center uses them to retrodict the preceding missing one. While waiting for missing packets, the fusion center applies retrodiction backward, from the current time k to time $k-d$ whose globe estimate is to be reported next, for *all* the packets – including the available ones – in between. This idea is illustrated in Fig. 2.1, where the current lag is $d = 2$. There are a total of three sensors. Since applying prediction-only estimates results in a higher error variance, to reduce the performance degradation, the fusion center may decide to wait for the delayed packet while applying the retrodiction scheme if subsequent packets arrive. Note both available and unavailable estimates are retrodicted during waiting. The retrodiction process is always run backward to the time instant for the next pending global estimate.

We apply the fixed-interval Rauch-Tung-Streibel (RTS) retrodiction algorithm [73], which is known to be computationally efficient. Besides, since measurements do not appear in the equations, the algorithm is especially suited for our scenario in which state estimates rather than raw data are sent directly to the fusion center. The following iterative steps propagate the newly gained information due to the on-time arrival of packet P_k^i backward to time $k-d$.
Step 0: Initialize the backward smoother.

$$\mathbf{P}_{k,retr}^i = \mathbf{P}_{k|k}^i; \tag{2.24}$$

$$\hat{\mathbf{x}}_{k,retr}^i = \hat{\mathbf{x}}_{k|k}^i; \tag{2.25}$$

From time $T_n = k$, $k-1$, ..., up to $k-d$, recursively calculate the values through the following three steps:

Step 1: calculate the backward smoothing gain

$$\mathbf{G}^i_{T_n-1} = \mathbf{P}^i_{T_n-1|T_n-1}\mathbf{F}^T(\mathbf{P}^i_{T_n|T_n-1})^{-1}; \tag{2.26}$$

Step 2: calculate the error $\mathbf{P}^i$ of the smoothed estimate

$$\mathbf{P}^i_{T_n-1,retr} = \mathbf{P}^i_{T_n-1|T_n-1} - \mathbf{G}^i_{T_n-1}(\mathbf{P}^i_{T_n|T_n-1} - \mathbf{P}^i_{T_n,retr})(\mathbf{G}^i_{T_n-1})^T; \tag{2.27}$$

Step 3: find the smoothed estimate

$$\hat{\mathbf{x}}^i_{T_n-1,retr} = \hat{\mathbf{x}}^i_{T_n-1|T_n-1} + \mathbf{G}^i_{T_n-1}(\hat{\mathbf{x}}^i_{T_n,retr} - \hat{\mathbf{x}}^i_{T_n|T_n-1}). \tag{2.28}$$

In these equations, $\mathbf{P}^i_{retr}$ denotes the a posteriori $\mathbf{P}^i$ after retrodiction. The algorithm is applied to each sensor separately, so that the process is also in line with the packet-level PRODIC calculation. As mentioned earlier, retrodiction in our scheme has the dual benefits of improving the accuracy of existing estimates and interpolating missing ones; consequently, the process has its distinct features, some of which are absent from those conventional retrodiction studies.

(1) *Interpolation of the missing estimates and the chain effect.* Retrodiction is only meaningful when a later estimate has an accuracy level at least as good as that of an earlier estimate. According to the equations, an unavailable packet itself has no impact on its preceding estimate during RTS retrodiction (that is, retrodiction is only effective with the presence of an actual estimate); however, if this unavailable estimate has been retrodicted from a subsequently available one, it can in turn improve its preceding one as well. This can be regarded as the "chain effect" of retrodiction. As a result, a string of missing estimates can be improved by just one single estimate subsequent to them all. For example, in Fig. 2.1, at time k, having been retrodicted by packet P^2_k, P^2_{k-1} can further retrodict P^2_{k-2}.

(2) *Handling the a priori mismatch as a result of loss.* Both the a priori estimates and

their error covariances appear in the RTS algorithm. In the full-observation case, sending a posteriori values is often enough, since the fusion center can always derive one-step predicted values for a priori values of the next step that are congruent with the predictions at the sensors. However, with missing data, the a priori values at the sensors and those at the fusion center may not match. The predicted values at the fusion center can be much more error-prone after multi-step losses. This has an effect on RTS retrodiction. For instance, when an actual estimate is received following at least a missing one, there exists a *mismatch* between the a priori values calculated from prediction and those (if any) sent directly by the sensors. We let the fusion center take the prior values from earlier prediction for the following two reasons: first, the a priori values may not actually be sent by the sensors in a real system or can be lost separately from the a posteriori data; second, the predicted values at the fusion center is a true reflection of accuracy level evolution associated with the global estimate.

(3) *Handling the delayed estimates.* What if the original estimate sent from a sensor arrives when the estimate has been interpolated from subsequent packet(s) but before the global estimate for the time of interest has been finalized? For simplicity, in our design, we have the original estimate replace the corresponding "partially retrodicted estimate(s)". Still better yet, the fusion center may reprocess the estimates so that both the original and its subsequent estimates are combined to yield a "fully retrodicted estimate", possessing an even better accuracy level than the original itself.

2.4.3 Selective Fusion Algorithm – PRODIC-RTS

Now we are ready to present the complete selective fusion algorithm, in which we have incorporated the RTS retrodiction into our recursive PRODIC calculation.

In a dynamic decision-making process like ours, there is more than one way to implement the selective fusion scheme. While we hope to improve both reporting accuracy and delay performances at the same time, the algorithm should be implemented as easily as possible.

Algorithm 1 Selective Fusion Algorithm with PRODIC and RTS Retrodiction

1: **Initialize**: At time k, the next estimate to be reported is $\hat{\mathbf{x}}^G_{k-d}$
2: **for** $T_n := d$ to 0 with decrement 1 **do**
3: Collect arriving packets $\mathcal{N}_k = \bigcup_{T=0}^{d} \mathcal{N}_{k,k-T}$.
4: Update $\mathcal{W}_{k,k-Tn}$ and $\mathcal{F}_{k,k-Tn}$:
5: $\mathcal{W}_{k,k-Tn} \leftarrow \mathcal{W}_{k-1,k-Tn} - \mathcal{N}_{k,k-Tn}$;
6: $\mathcal{F}_{k,k-Tn} \leftarrow \mathcal{F}_{k-1,k-Tn} \bigcup \mathcal{N}_{k,k-Tn}$;
7: **end for**
8: **for** $\forall i \in \mathcal{S}$ **do**
9: Calculate $\mathbf{P}^i_{k-d,retr}$ and $\hat{\mathbf{x}}^i_{k-d,retr}$ using Eqs. (2.24)-(2.28);
10: **for** $T_n := d$ to 0 with decrement 1 **do**
11: Calculate $\mathbf{J}^i_{k-T_n}$ using Eq. (2.8) (with retrodicted values);
12: $\mathbf{J}^G_{k-T_n} = \sum_{\text{packet } \mathsf{P}^i_{k-T_n} \in \mathcal{F}_{k,k-T_n}} \mathbf{J}^i_{k-T_n}$;
13: $(\mathbf{P}^G_{k-T_n|k-T_n})^{-1} = h((\mathbf{P}^G_{k-T_n-1|k-T_n-1})^{-1}, \mathbf{J}^G_{k-T_n})$;
14: **end for**
15: **end for**
16: **if** $\mathcal{W}_{k,k-d} = \emptyset$ or $d = D_{max}$ **then**
17: Finalize $\hat{\mathbf{x}}^G_{k-d}$ using Eq. (2.15) with all the elements in $\mathcal{F}_{k,k-d}$ and $\mathbf{P}^G_{k-d}$;
18: $d \leftarrow d - 1$;
19: **if** $d > 0$ **then**
20: Go to Line 8;
21: **end if**
22: **else**
23: **for** $\forall j \in \mathcal{S}$ such that $\mathsf{P}^j_{k-d} \in \mathcal{W}_{k,k-d}$ **do**
24: Calculate $\Delta\mathbf{P}^i_{k,k-d}$ using Eqs. (2.20)-(2.22);
25: Calculate the difference between the normalized PRODIC and the threshold th $Diff^j_{k-d} = \frac{tr(\Delta\mathbf{P}^j_{k,k-d})}{tr(\mathbf{P}^G_k)} - th = 1 - \frac{tr(\mathbf{P}^{G,j}_{k|k,temp})}{tr(\mathbf{P}^G_k)} - th$;
26: **end for**
27: **if** $\forall i \in \mathcal{S}$, $Diff^i_{k-d} \leq 0$ **then**
28: $\mathcal{W}_{k,k-d} \leftarrow \emptyset$;
29: Go to Line 16;
30: **else**
31: $d = d + 1$;
32: **end if**
33: **end if**

In our design, only arrivals from within the waiting window are considered; that is, delayed arrivals are disregarded if the corresponding global estimate has already been finalized and reported. Although it is possible that the fusion center still considers incorporating delayed arrivals (e.g., up to D_{max}) in this scenario, doing so may increase the computational cost as the size of the historical data grows. In addition, the benefit of incorporating the estimates with such a long delay is often negligible: the PRODIC criterion has already dictated that the previously disregarded packets carried little information, and even more so after the global estimate has been finalized, due again to the diminishing information over time.

Algorithm 1 describes the selective fusion rule performed by the fusion center at time k for a global estimate at time stamp $k-d$ (i.e., the current lag is d). The following sets are defined:

- $\mathcal{S}$ is the set of all sensors;
- $\mathcal{P}_k$ is the set of all the packets with time stamp k;
- $\mathcal{N}_{k,k-d}$ is the set of new arrivals at time k with time-stamp $k-d$;
- $\mathcal{W}_{k,k-d}$ consists of all the packets with time-stamp $k-d$ that the fusion center is still expecting at time k;
- $\mathcal{F}_{k,k-d}$ is the set of packets with time-stamp $k-d$ that are ready at time k for fusion.

In addition, we let $\mathcal{W}_{k,k-d} = \mathcal{P}_{k-d}$ and $\mathcal{F}_{k,k-d} = \emptyset$ if $d < 0$. Apparently, if at time k, $\mathcal{W}_{k,k-d} = \emptyset$, the fusion center is ready to finalize $\hat{\mathbf{x}}^G_{k-d}$ using all the elements in $\mathcal{F}_{k,k-d}$. Besides, a packet appearing in $\mathcal{N}_{k,k-d}$ ($d > 0$) must have appeared in $\mathcal{W}_{k-1,k-d}$ too. Therefore, the selective fusion decision-making process evolves as follows. After updating newly received packets, the fusion center first retrodicts the past estimates using newly received data; afterward, it calculates the PRODIC of each packet in $\mathcal{W}_{k,k-d}$ and decide whether further waiting is necessary. Only when all the packets in the set have been received, or when the pending packets are deemed no longer worthy of further waiting can the fusion center finalize $\hat{\mathbf{x}}^G_{k-d}$.

In the algorithm, the instantaneous $\mathbf{P}^G$ can be calculated immediately, as shown in Lines 8-12, after the fusion center has incorporated the packets that have just arrived and has retrodicted the estimates. Afterward, the PRODIC of all the pending packets is computed and normalized for comparison with a cutoff threshold th (Lines 23-26). There are two possible scenarios when the next pending estimate can be finalized: all the packets have been received (Line 16) or all the remaining pending packets are no longer deemed necessary and will be disregarded (Lines 27-28). Besides, when one global estimate is finalized, the above procedure is repeated immediately for the next pending estimate (Line 21) – for a lag of $d-1$ in this case – as more than one pending estimate may be finalized at the same time step.

Thanks to the information gain from retrodiction, the PRODIC of a missing packet becomes smaller and the fusion center can often terminate its waiting for the pending packets much earlier compared to the case without retrodiction. The level of improvement will be demonstrated via simulation studies in Section 2.6.

2.5 Expected Information Gain with Link Loss and Delay Profiles

Our earlier discussions highlighted the proposed selective waiting and fusion algorithm as an online scheme since the fusion center makes its decisions based solely on the actual message arrival instances. In this section, we consider the scenario where the fusion center has some knowledge about the packet loss and delay statistics. The deterministic information gain from a pending estimate can then be extended to the expected information gain for different time periods. Substituting the expected information gain for the original will also result in different fusion performance as to be discussed below.

2.5.1 Loss and Delay Profiles

The packet loss and delay characteristics are largely determined by the long-term condition of the long-haul link. In a real system, even when the fusion center is initially oblivious to these characteristics, the empirical packet arrival patterns can be measured and recorded over time and thus approximate profiles of link loss and delay can be constructed. For ease of exposition, here we assume that each packet sent by a sensor is lost during transmission with probability p independently of other packets. A pdf $f(t)$ and the corresponding cdf $F(t) = \int_0^t f(u)\,\mathrm{d}u$ can model the overall delay t – a continuous random variable – that a packet experiences to be successfully delivered to the fusion center. Additionally, the loss and delay are regarded as two independent processes.

2.5.2 Expected Information Gain

If we let time zero denote the time of interest, that is, the time when the packet containing the state estimate is generated and sent out by a sensor, then the probability that this packet is delivered by time t is $(1-p)F(t)$. Consider a pending packet with a current lag of d steps – in other words, a delay of at least dT, where T is the estimation interval – and the probability that the packet will be delivered within the next interval $[dT, dT+T]$ can be calculated as

$$\begin{aligned} p_{del,dT\to(d+1)T} &\triangleq \Pr\{t \le dT+T | t > dT\} \\ &= \frac{\Pr\{dT < t \le dT+T\}}{\Pr\{t > dT\}} \\ &= \frac{(1-p)[F(dT+T)-F(dT)]}{1-(1-p)F(dT)}. \end{aligned} \tag{2.29}$$

Eq. (2.29) describes the probability that the information contribution from a pending estimate will be realized in the next estimation interval. Then the *expected* information

contribution would be

$$\mathbf{J}_{k-d,E} = p_{del,dT\to(d+1)T}\mathbf{J}_{k-d}. \tag{2.30}$$

2.5.3 Effect of the Expected Gain on Selective Fusion

Eq. (2.30) is a statistical measure of the potential information gain in the first step of our selective fusion algorithm. We consider its effect for both heterogeneous and homogeneous link conditions.

Heterogeneous Link Loss and Delay Profiles

Since both the packet-level loss rate p and the delay distribution $F(t)$ appear in Eq. (2.30), any variations in either of them would cause a shift in the expected information gain even for the same value of $\mathbf{J}$. It is easy to show that for the same delay distribution, an increase in the packet loss rate would result in the decrease of the expected information gain at any step (that is, Eq. (2.29) monotonically decreases with p.); on the other hand, with the same loss rate, the expected information gain depends on the specific shape of the pdf $f(t)$. In our context, then, the fusion center would find it difficult to realize the potential information gain promptly for a link with a high loss rate and/or a long average arrival delay.

Homogeneous Link Loss and Delay Profiles

Even when the loss and delay statistics are homogenous across different communication links, differences in fusion performance still exist if the fusion center opts to use Eq. (2.30) as the potential information gain. In particular, since Eq. (2.29) effectively serves as a discount factor, the perceived information gain at each step becomes smaller than the original $\mathbf{J}$. Recall from the last section that the fusion center uses a cutoff threshold th to decide whether to continue waiting for a pending packet. As the information gain term to be plugged in the initial step becomes smaller, so does the one propagated to the current time. As such, for

the same th, the fusion center may alter some of the would-be "wait" decision so that the packet is then disregarded. This would help reduce the reporting delay significantly at the cost of slightly higher estimation errors.

2.6 Performance Evaluation

We evaluate the performance of our selective fusion algorithm through extensive simulations. We first introduce the target motion model, and then compare a few on-line fusion algorithms; and finally, we compare our scheme with two other OOSM fusion schemes that also exploit retrodiction. Despite our discussions on expected information gain in the last section, in our simulation studies, the fusion center does not assume any statistical knowledge about the loss and delay characteristics in its online decision making.

2.6.1 Simulation Setup

Target Motion Model

We consider tracking of a target whose motion follows the nearly-constant-acceleration model [44]; that is, the trajectory of the moving target follows Newtonian laws with independently incremented acceleration. In particular, we consider the white-noise jerk version of the model, in which the acceleration derivative (i.e., the "jerk") is an independent white noise process.

The target in general moves within the three-dimensional free space, but the trajectory can be mapped to orthogonal axes (e.g., the commonly used "east – north – up"). And here we single out the effect of one dimension (e.g., east) by mapping the trajectory onto this axis only. The target state then consists of the position r, velocity v, and acceleration a on this axis and the discretized state evolution is given by

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{w}_k,$$

Table 2.1: Simulation Parameters Default Setup

Parameter	Symbol	Value
sampling period (s)	T	0.5
process noise PSD (m^2/s^3)	S_w	0.5
loss rate	P_{loss}	0.1
normalized arrival delay	D_{arv}	3
reporting deadline	D_{max}	10
no. of sensors	n	3
measurement noise s.t.d. (m)	$\sqrt{R}$	50
PRODIC threshold	th	5%

where

$$\mathbf{x}_k = \begin{bmatrix} r_k \\ v_k \\ a_k \end{bmatrix}, \mathbf{F} = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix},$$

and

$$\mathbf{Q} = \mathrm{cov}(\mathbf{w}_k \mathbf{w}_k^T) = S_w \begin{bmatrix} T^5/20 & T^4/8 & T^3/6 \\ T^4/8 & T^3/3 & T^2/2 \\ T^3/6 & T^2/2 & T \end{bmatrix},$$

where S_w is the power spectral density of the continuous-time white noise, and T is the sampling/estimation period. Besides, the common measurement matrix at the individual sensors is

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

Parameters and Performance Metrics

Table 2.1 contains the list of our default parameters.

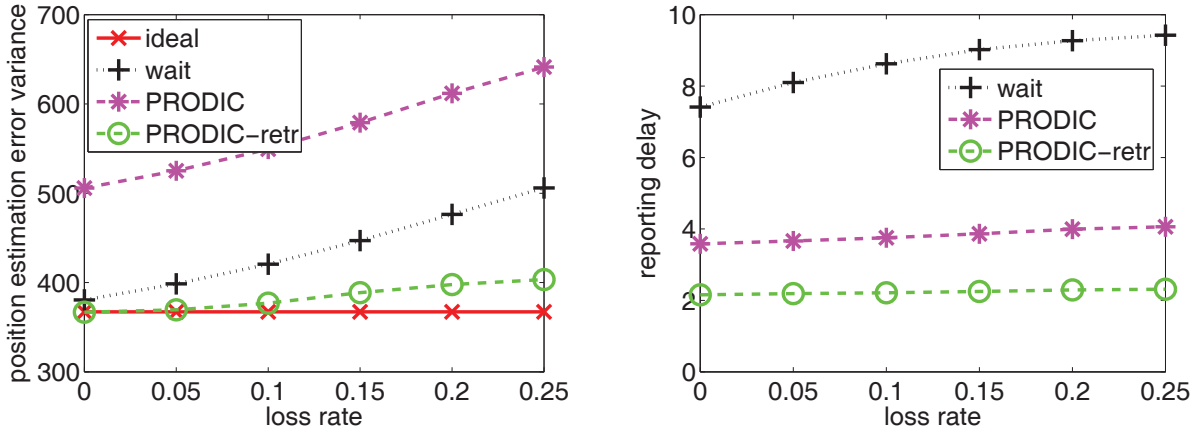


Figure 2.2: Loss rate vs. (a) position error variance and (b) normalized reporting delay

As our default setup, there are a total of three sensors, whose measurement noise standard deviations are all 50 m. The process noise PSD is 0.5 m^2/s^3; the sampling/estimation period $T = 0.5$ s and the associated time measures are subsequently "normalized" relative to this time (without units); e.g., the normalized sampling time is 1, whereas the normalized deadline is set as 10 (which is really 5 s). The PRODIC threshold is set to 5%. Again, in our study, packet loss and delay are generated as two independent processes. While the packet loss is generated as an independent Bernoulli process, delays follow (memoryless) exponential distribution. The default loss rate and normalized arrival delay are set to be 10% and 3, respectively.

The loss and delay statistics for a *given* long-haul network should be fairly stable over a period of time (that's the very reason that they can be learned over time). Nevertheless, we are interested in studying the impact on fusion performance from variable loss and delay profiles, for instance, when different types of networks are compared with one another. We study the impact of each factor separately by varying its values while keeping all other parameters at their default values. In addition, although we expect a long-haul network is well designed for scalability, we study the scenario when the number of deployed sensors is small (no more than five) as long-haul sensors are generally expensive to deploy; also, the

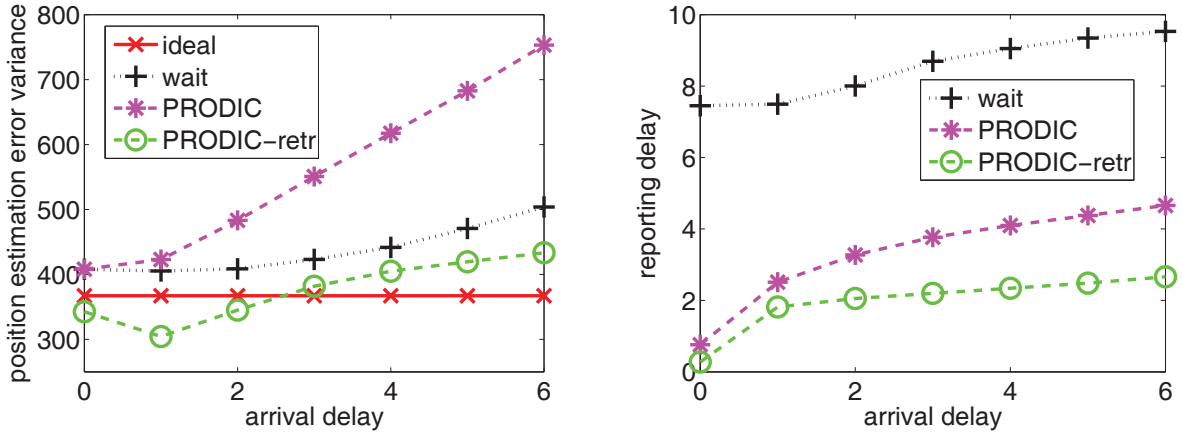


Figure 2.3: Normalized arrival delay vs. (a) position error variance and (b) normalized reporting delay

system is more vulnerable to failure and suboptimal performance with an insufficient number of sensors and thereby warrants special attention.

2.6.2 Comparison of Different Online Fusion Schemes

We compare the following schemes in our first group of study:

- Maximum waiting ("wait"): the FC finalizes the estimate after all missing estimates arrive or the reporting deadline is reached, whichever comes first;
- Selective fusion based on PRODIC but without retrodiction ("PRODIC"):
- Selective fusion based on PRODIC with modified RTS retrodiction[6] ("PRODIC-retr");
- We also consider the full-observation case ("ideal") as a baseline scenario for comparison with other schemes, in which the reporting delay is always zero and hence there is no retrodiction being implemented. In other words, this ideal scenario is not realistic since it always assumes perfect communications and instant data processing and reporting.

[6]This has been denoted as "PRODIC-RTS" earlier, which will be used in subsequent analysis as well.

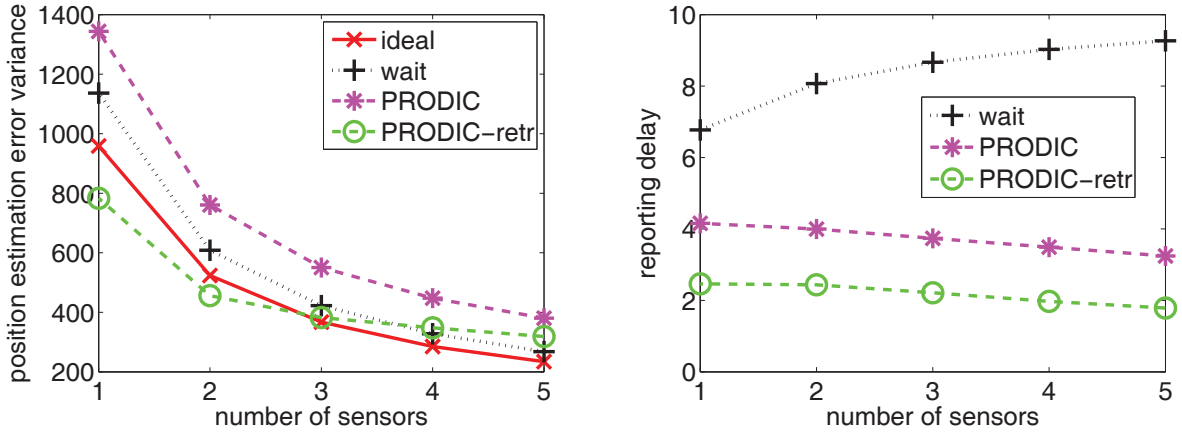


Figure 2.4: Number of sensors vs. (a) position error variance and (b) normalized reporting delay

Overall the goal is to reduce both the estimation MSE and reporting time, where the former is measured as the trace of the error covariance matrix.

Loss Rate

The packet loss rate is varied from 0 to 0.25, and the results are shown in Fig. 2.2. With its average reporting time approaching the deadline, the "wait" scheme takes advantage of the extra waiting time to collect delayed estimates and thus reduces the estimation error[7]. The PRODIC scheme is seen to effectively reduce the reporting delay; however, the estimation error is still relatively high. Compared to other schemes, PRODIC-RTS is less sensitive to changes in the loss rate. At the highest loss rate 25% in our study, the position error variance increases only by about 7% from the zero-loss case. This demonstrates the effectiveness of our design, which exploits retrodiction to reduce the estimation error upon packet loss. One of the main reasons for this improvement has been shown earlier: as long as one most recent packet is received, all the preceding missing estimates can be retrodicted. Besides, the change in reporting delays as the loss rate increases is negligible (it stays slightly above 2). From this perspective, our PRODIC-RTS is robust to packet loss.

[7]The unit for this and all subsequent position error variances is m^2.

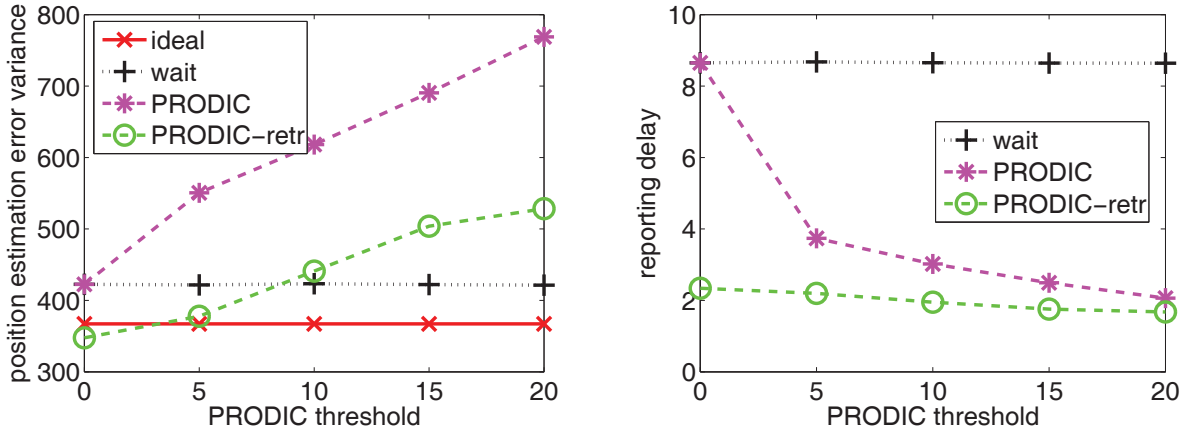


Figure 2.5: PRODIC threshold vs. (a) position error variance and (b) normalized reporting delay

Arrival Delay

We vary the normalized packet arrival delay from 0 to 6, and observe similar trends in Fig. 2.3 as in the previous case, with some minor exceptions. When there is no arrival delay, no retrodiction is performed so that the estimates can be reported immediately; as the arrival delay goes up to 1, the error variance decreases following PRODIC-RTS thanks to retrodiction. And then, as the arrival delay continues to increase, the error variance also increases, though not quite as fast as in other schemes. As can be seen again, retrodiction has effectively reduced the estimation error when there are significantly long delays.

Although it may first seem surprising that the reporting delay is well below the average arrival delay (e.g., when the arrival delay equals 6, the reporting delay is about 2.5), the result is attributed to both the randomness of the arrival delay and the PRODIC-based selective fusion process. There exist packets whose arrival delay is smaller than that of others and hence can retrodict other missing ones comparatively faster. With the improved estimates following retrodiction, the fusion center may decide to terminate its waiting much earlier by disregarding all the remaining pending packets. In contrast, the reporting delay in the "waiting" case, even under moderate loss and delay profiles, almost always approaches

the reporting deadline D_{max} due to the nearly constant presence of missing packets.

Number of Sensors

We vary the number of sensors from 1 to 5 to test its impact on fusion performance. From Fig. 2.4, we observe that when the number of sensors is small, the effect of retrodiction is more prominent. Because the error variance is usually much higher with an insufficient number of sensors – at the risk of violating the maximum tolerable errors – the targeted potential information gain (as specified by PRODIC threshold) from a missing packet can be realized only by waiting longer so that the actual packet or subsequent packets following this missing one can be received. If there is only one sensor, all the gains from the retrodiction would benefit the final estimate because no other sources – that is, packets from other sensors – exist that can compensate for the information loss due to an unavailable packet. On the other hand, the relative information gain from retrodiction with an increasing number of sensors being present becomes smaller and the advantage of applying PRODIC-RTS in terms of improved accuracy diminishes.

As for the reporting delay, opposite trends are observed. The "waiting" scheme is subject to the increase of waiting time when more sensors are present, as there is a higher possibility that a packet is absent. In contrast, PRODIC-based schemes experience reduced waiting time, thanks to the selective waiting process, in which the information gain from a missing packet diminishes as more sensors contribute to the final fusion.

PRODIC Threshold

In the above simulations, we have kept the PRODIC threshold at a conservative 5%; that is, only when a pending packet can potentially reduce the current estimate error by more than 5% would the fusion center decide to wait for it. In reality, the fusion center can tune the threshold according to the current accuracy level. In Fig. 2.5, the threshold is set to vary from 0% (i.e., to wait for all pending packets) to 20%. The plots can be easily interpreted:

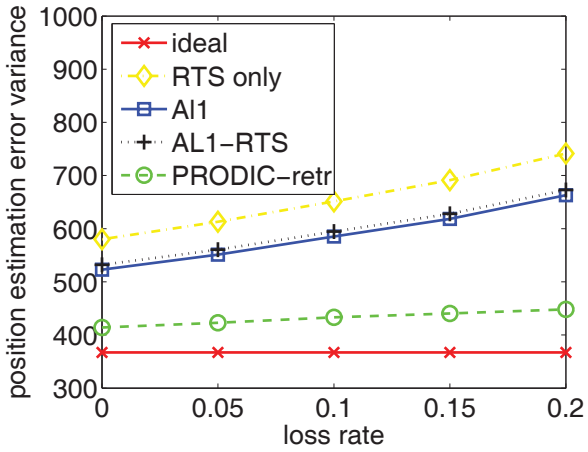


Figure 2.6: Loss rate vs. position error variance (with normalized reporting delay = 2)

When the threshold goes up, the requirement on each packet is relaxed, fewer packets need to be awaited, and reduced accuracy and reporting delay would follow; and vice versa. It is interesting to note that when the threshold is zero, the PRODIC scheme is reduced to the "waiting" case; PRODIC-RTS, on the other hand, does not have inflated waiting time thanks to the proactive nature of retrodiction.

2.6.3 Performance Comparison with OOSM Algorithms

We also compared our scheme with Al1 [11] and Al1-RTS [58] algorithms, both of which are designed to address OOSM issues. The main feature of these algorithms is that retrodiction is performed only after an OOSM actually arrives and the goal is to correct only the current estimate. The difference within the two cited algorithms exists during the retrodiction process: while Al1 uses an "equivalent measurement" to perform one-step retrodiction, Al1-RTS applies the standard RTS retrodiction algorithm.

The original schemes are proposed under relatively simple circumstances: single sensor, single l-step delay, no data loss, and no reporting deadline, among others. However, it is easy to extend the schemes to multi-OOSM [84] and multi-sensor case with data loss and reporting deadline. Besides, we also allow non-zero reporting delay so that the correction after retrodiction applies to an intermediate step as well, thereby improving the accuracy

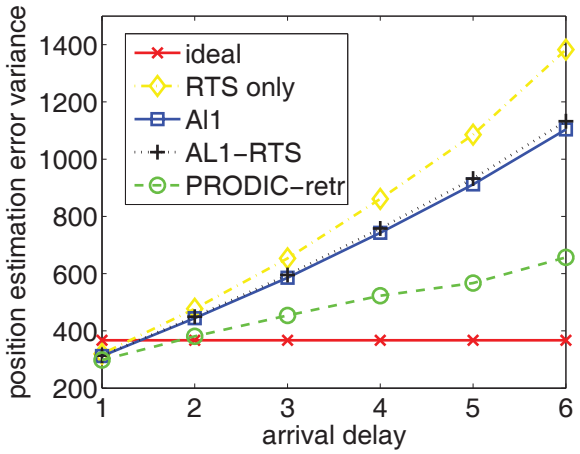


Figure 2.7: Normalized average arrival delay vs. position error variance (with normalized reporting delay = 2)

performance of not just the current estimate. This can somewhat guarantee fair comparisons among the algorithms.

Because all the schemes share retrodiction as a common means to reduce the error variance of an earlier estimate, we also compare with an "RTS-only" scheme in which no OOSM is processed; that is, data are either received on time or smoothed with subsequent ones whenever available. We test how the position error variance varies with different parameters (loss rate, arrival delay, and number of sensors) with a *given* normalized reporting delay of two (since the reporting delay in other schemes needs to be pre-assigned to run the algorithms). Note that the thresholds have been adjusted in our PRODIC-RTS scheme to guarantee the same reporting delay.

Loss Rate

From Fig. 2.6, as the loss rate increases, the increase in the position estimation error of PRODIC-RTS is less significant compared to others. When the loss rate is 20%, the error variances of both Al1 and Al1-RTS are 50% more than that of PRODIC-RTS. On the other hand, while the two OOSM algorithms reduce the error variance by about 10% compared to the case where OOSMs are simply ignored ("RTS only"), our scheme reduces the error

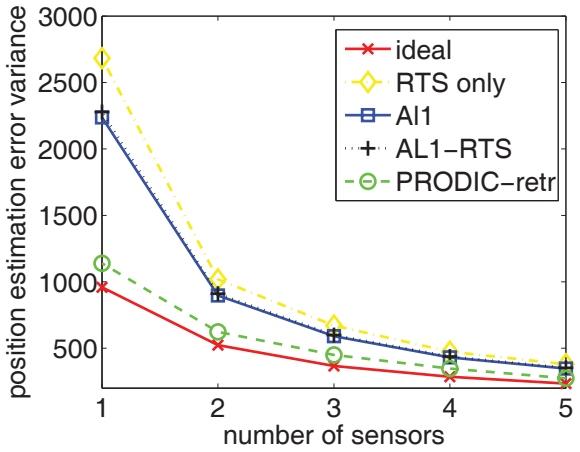


Figure 2.8: Number of sensors vs. position error variance (with normalized reporting delay = 2)

by about 40%. The reason is fairly simple: In other schemes, retrodiction is performed only when the missing packets actually arrive later, which may not be effective when the loss rate is high, resulting in larger estimation errors.

Arrival Delay

As can be seen in Fig. 2.7, with the increasing mean arrival delay, the position error variance in PRODIC-RTS does not shoot up as much as those in other schemes. When the arrival delay is 4, for example, Al1 already has its error variance nearly 50% higher than that of PRODIC-RTS; as it further reaches 6, the error variances of the two OOSM schemes are more than 75% higher. Similar to the case with varying loss rates, as the arrival delay increases, the fusion center has to wait longer to perform retrodiction. Meanwhile, the reporting deadline is fixed, allowing fewer estimates to be effectively retrodicted in time for the final reporting, thus leading to a higher error variance.

Number of Sensors

Again, we can see from Fig. 2.8 that the benefit of retrodiction is more significant when the number of sensors is small for all cases. With fewer sensors, all the other schemes suffer from much higher error variances than PRODIC-RTS. Our scheme benefits from the proactive

retrodiction where the correction doesn't have to occur only after an OOSM actually arrives. The other schemes must wait significantly longer to perform retrodiction, which may not be possible within the deadline if the missing packets do not arrive in time, and this becomes even more challenging when fewer sensors exist. The difference is especially significant when there is only one sensor.

To sum up the above results, our PRODIC-RTS scheme, combining features such as information gain projection, selective waiting, and proactive retrodiction, often yields accuracy performance comparable to that under the full-observation case while incurring very little reporting latency, demonstrating its robustness against degradation in transmission links such as severe loss and arrival delay.

2.7 Conclusion

In this chapter, to meet the stringent requirements on accuracy and timeliness, while accounting for the long latency and severe loss inherent over long-haul links that exert a negative impact on fusion performance, we have proposed an information metric PRODIC and a modied application of the RTS retrodiction algorithm, so that the fusion center can make its online selective waiting decisions on when to fuse its available sensor information. Simulation results of a target tracking application have validated the major advantages of our design under variable link loss and delay profiles.

Chapter 3

Message Retransmission and Retrodiction for Recovery of Missing Sensor Information

3.1 Introduction

Information loss is often an intrinsic feature of a long-haul sensor network. In this chapter, we consider how to recover some of the lost information over time. One way to counteract the effect of the lossy transmission link is to adopt certain transport protocols in which message retransmission is implemented and some lost messages can be recovered after one or multiple rounds of retransmission. Not to be overlooked, however, is another aspect of the system requirement – the delay performance. Owing to often near real-time requirements of the monitoring/tracking tasks, the system often allows for only a small time gap between the time of interest and the time when the estimate should be finally obtained and reported. This often comes as a predefined reporting deadline before which an estimate must be reported by the fusion center. Message retransmission may exacerbate the reporting delay performance by incurring extra time on top of the already relatively large propagation and transmission

latency. The fusion center may have to increase its reporting time significantly in order to recover the lost messages, even at the risk of violating the stipulated reporting deadline.

The transmission control protocol (TCP) implemented in wired Internet and wireless local area networks (WLANs) is still garnering research efforts that are too numerous to list. Analysis of TCP-like transport protocols over satellite links can be found in studies such as [5] and [33]. Commonly acknowledged are the difficulties in applying "conventional" TCP protocols to transmission over satellite links, mainly because of the very large propagation delay not encountered in other networks. The specificity of our application also somewhat distinguishes our analysis from the ones geared toward the voice- and video-based broadcasting and data-based Internet access, both of which have continuous data in flight. Also of note is that in our settings, state estimates from the remote sensors are generally intermittently sent over a wide-band satellite channel – with the interval possibly ranging from a few times within a second to once every few minutes – and thus congestion is not as much a concern as in conventional TCP applications. Hence, we assume a simplified transport protocol in which retransmission is performed on the message-level basis.

In many state estimation applications, retrodiction, also known as smoothing, serves as the "backward prediction" of an earlier estimate. Depending on the relationship between the length of data used and the time of interest, we can categorize retrodiction roughly into fixed-point, fixed-lag, and fixed-interval retrodiction [73]. Whereas the conventional retrodiction techniques are used mainly for improving estimates that have been obtained, for instance, in the context of out-of-order measurement (OOSM) problems [58, 83, 84], we are primarily interested in how missing estimates can be interpolated from retrodiction[1] and the corresponding improvement in estimation errors following such retrodiction.

Another important aspect of the multi-target tracking task is that prior to fusion, the data association algorithm determines the groups of estimates, so that each group is hypothesized to correspond to the same target. Many types of association and fusion algorithms for

[1] In the meantime, an available estimate is retrodicted by subsequent estimate(s) as well whenever applicable as in conventional retrodiction.

tracking and navigation applications have been studied [12]. However, our focus here is not on performance comparison among different algorithms; rather, of interest to us is the performance improvement from retransmission and retrodiction following a given simple set of association and fusion algorithms.

In this chapter, we provide analytical models to systematically study the impact from retransmission and retrodiction on target-tracking performance under variable loss and delay conditions in a long-haul sensor network. Our study is among the very first to link both communication (message retransmission) and computation (prediction, retrodiction) with state estimation performance, accounting for both data fusion and association in target tracking applications. Simulations of a coasting ballistic target tracking example are conducted and results under various conditions are shown in the end to validate our analysis.

The remainder of this chapter is organized as follows: We first provide analysis of the delivery rate of a message after retransmission in Section 3.2, and derive the arrival time distribution in Section 3.3. Following analytical studies on retransmission, we explore the effect of retrodiction on the estimation performance improvement in Section 3.4, where non-cooperative and cooperative types of retrodiction are discussed. Simulation results of a coasting ballistic target tracking application are presented and analyzed in Section 3.5 before we conclude the chapter in Section 3.6.

3.2 Message Retransmission

In a long-haul sensor network, a remote sensor sends out a message containing the state estimate; upon successful reception of this message, the fusion center sends back an acknowledgment (ACK) message to the sensor. A failed arrival of the ACK message before the expiration of the timeout T_{TO} – due to loss and/or long delay of the message itself or the ACK – will prompt the sensor to retransmit the message. Typically, T_{TO} could be several times the RTT of the connection, and over long-haul connections it could be of the order

of seconds. Setting T_{TO} too long could reduce the maximum number of retransmissions, thereby limiting the potential to recover the lost message; on the other hand, a short T_{TO} may incur many rounds of retransmission (often unnecessarily) when the sensor could have waited a bit longer to receive the ACK. Such retransmission continues till the acknowledgment is received by the sensor, or the retransmission window W expires. This window should ideally contain multiple T_{TO} periods so that under adverse link conditions, it's likely that the message can eventually be recovered after multiple tries.

In a real system, the reporting deadline may limit the potential gain from retransmission as the overall time before reporting can be very short. A cutoff time T_{CO} is defined to mark the end of the waiting at the fusion center. This cutoff on the one hand limits the total number of retransmissions, and on the other limits certain messages from being eventually delivered due to the randomness of the delay. In Fig. 3.1, the effect of this time cutoff is shown. The window W is set to be $3T_{TO}$ and hence there are a total of two rounds of retransmision (at T_{TO} and $2T_{TO}$). In the first case, T_{CO} is small so that the last round of retransmitted message cannot arrive in time for fusion. While in the last case, setting T_{CO} way beyond the end of the retransmission window is not likely to significantly increase the chance of receiving the message. Therefore, the system should guarantee that the retransmission window – at the sensors side – is commensurate with the cutoff time – at the fusion center, as in the second case in the figure, so that the fusion center could benefit from all rounds of retransmission while not wasting time attempting to recover the pending message after the retransmission has ended.

The message-level loss and delay characteristics are determined by the long-haul link conditions. We assume that each message sent by a sensor is lost during transmission with probability p_L independently of other messages. Normally, the latency that a message experiences before arriving at the fusion center consists of the initial detection and measurement delay, data processing delay by both the sensor and the fusion center, propagation delay,

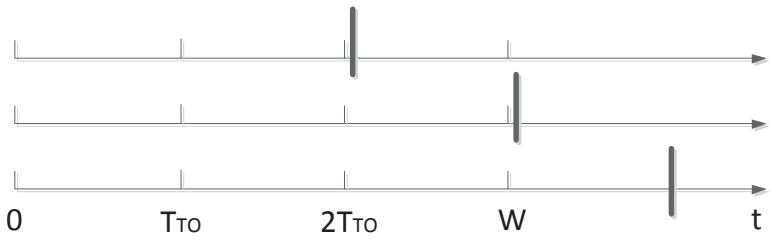


Figure 3.1: Timing of message retransmission: The timeout is T_{TO}, retransmission window is W, and different choices of the cutoff time T_{CO} are marked by bold lines.

and transmission delay, among others[2]. These are collectively considered as the minimum delay that a message must undergo to reach the fusion center, which is bound mostly by factors such as the distance of the satellite link, the transmission data rate, and length of the message. The extra random delay is often due to link conditions such as weather and terrain. We suppose a pdf $f(t)$ can model the overall delay t that a message experiences to be successfully delivered to the fusion center. One typical example is that of the shifted exponential distribution:

$$f(t) = \frac{1}{\mu_D} \exp^{-\frac{t-T}{\mu_D}}, \text{ for } t \geq T. \tag{3.1}$$

in which T serves as the common link and processing delay, and μ_D is the mean of the random delay beyond T. In a real system, the empirical values of the message delay can be measured over time and thus an approximate function $\tilde{f}$ can be estimated. In the following analysis, however, we still use the generic function $f(t)$ to model the arrival delay.

We are interested in the average probability of a message being successfully delivered by a certain time, that is, by the cutoff time T_{CO}. An estimate is only counted once even if it arrives multiple times due to retransmission. The duplicate messages received by the FC can simply be ignored as they will not contribute further to the fusion performance.

[2]The queueing delay is also minimal with little/no congestion.

With the time of interest being regarded as time zero in this section, the maximum number of retransmissions before the cutoff time T_{CO} is

$$K_{retx} = \left\lceil \frac{\min\{T_{CO}, W\}}{T_{TO}} \right\rceil - 1. \tag{3.2}$$

From the definition, $K_{retx} + 1$ is the total rounds of transmission, including the original and subsequent retransmissions.

We define $p^k_{del,t}$ as the probability that a message is delivered by time t after k rounds of retransmissions, and

$$T_{retx,k} = T_{CO} - kT_{TO}, \text{ for } k = 0, 1, ..., K_{retx} \tag{3.3}$$

as the duration of the period $[kT_{TO}, T_{CO}]$ in which the k-th retransmitted message is in flight and could be potentially delivered to the fusion center.

When there is no retransmission within $[0, t]$, the probability of a message being delivered by time t is

$$p^0_{del,t} = (1 - p_L)F(t), \tag{3.4}$$

in which $F(t) = \int_0^t f(u)\,\mathrm{d}u$ is the cdf of the arrival delay. Its complement, the probability that the original message is unavailable at time t, is denoted as

$$p^0_{loss,t} = 1 - p^0_{del,t} = p_L + (1 - p_L)\overline{F}(t), \tag{3.5}$$

in which $\overline{F}(t) = 1 - F(t)$ is the tail distribution. With these two probabilities, we can derive the message delivery rate $p^{K_{retx}}_{del,T_{CO}}$.

The original message is delivered by T_{CO} with probability

$$p^0_{del,T_{CO}} = p^0_{del,T_{retx,0}} = (1 - p_L)F(T_{retx,0}). \tag{3.6}$$

And with the first round of retransmission, the delivery probability totals

$$p^1_{del,T_{CO}} = p^0_{del,T_{CO}} + p^0_{loss,T_{retx,0}} p^0_{del,T_{retx,1}}. \tag{3.7}$$

In general, for the k-th ($0 < k \leq K_{retx}$) round of message retransmission, we have

$$p^k_{del,T_{CO}} = p^{k-1}_{del,T_{CO}} + p^0_{del,T_{retx,k}} \left(\prod_{i=0}^{k-1} p^0_{loss,T_{retx,i}} \right). \tag{3.8}$$

In other words, the extra delivery rate from the k-th round is realized when all the previous $k-1$ retransmissions and the original message are not available by T_{CO}. Subsequently, we can obtain the overall message delivery probability within time $[0, T_{CO}]$ by summing up all such probabilities:

$$\begin{aligned} p^{K_{retx}}_{del,T_{CO}} &= \sum_{k=0}^{K_{retx}} p^0_{del,T_{retx,k}} \left(\prod_{i=0}^{k-1} p^0_{loss,T_{retx,i}} \right) \\ &= (1-p_L) \sum_{k=0}^{K_{retx}} F(T_{retx,k}) \left\{ \prod_{i=0}^{k-1} [1-(1-p_L)F(T_{retx,i})] \right\}. \end{aligned} \tag{3.9}$$

3.3 Arrival Time Pattern Distribution

So far we have combined the message-level loss rate and one-way arrival delay distribution to obtain the message delivery probability with a certain deadline requirement as specified by T_{CO}. In practice, the fusion center should be afforded some flexibility in deciding its actual cutoff time that does not violate the systemic value T_{CO}. First, sometimes reporting an abnormal change promptly is more crucial than initially providing an accurate estimate because the alert level can be increased immediately that facilitates further investigation. As the maximum allowable delay, T_{CO} may nevertheless be too large for such rare but time-critical incidences. On the other hand, owing to the dynamic environment of the field, sometimes the fusion center may decide to reduce its waiting time for the retrans-

mitted messages because of increased computational effort to obtain the final estimate, due to an increased number of sensors or increased state dimensionality when multiple closely positioned targets are in clutter [14].

In this section, we aim to derive the distribution of the arrival time, and more particularly, the cdf of the first instance of arrival before T_{CO}. This provides us a view of the internal structure of the arrival process within $[0, T_{CO}]$, which could be explored for the above flexible scheduling of early cutoff.

3.3.1 Arrival Time: One-way Communication Analysis

We treat loss and delay as two independent processes, although a lost message can be regarded as having an arrival delay of infinity. In the previous section, only the one-way delay characterized by the pdf $f(t)$ is considered because we noticed the equivalence of the final delivery probability no matter how acknowledgments are actually received. In deriving the arrival time, however, we need to consider two-way delay as well: loss and latency of ACKs would affect the total number of retransmissions, which in turn decides the distribution of the arrival time. For ease of explication though, we first work on the case in which there are exactly K_{retx} retransmissions – as if no ACKs were ever sent back by the fusion center – and later extend the results to an arbitrary number of retransmissions.

First, we define the cdf of a truncated nonnegative random variable Y_T with the upper truncation point $b > 0$ as[3]

$$F_T^b(y) = \frac{F(y)}{F(b)}, \text{ for all } 0 \leq y \leq b. \tag{3.10}$$

And the associated pdf is

$$f_T^b(y) = \frac{f(y)}{F(b)}, \text{ for all } 0 \leq y \leq b. \tag{3.11}$$

[3]Note that the subscript "T" appearing in cdf's and pdf's denote the function describes a truncated random variable.

In our study, we are interested in a series of truncated cdfs and pdfs corresponding to different retransmission cycles. More specifically, we consider the k-th round of retransmitted message has a truncated cdf by time $T_{retx,k}$ as

$$F_T^{T_{retx,k}}(t) = \frac{F(t)}{F(T_{retx,k})}, \text{ for all } 0 \leq t \leq T_{retx,k}. \tag{3.12}$$

Let D denote the arrival time of the message, and more specifically, $D_k = d_k + kT_{TO}$ the arrival time of the k-th retransmitted message (or the original message when $k = 0$). Apparently, d_k denotes the arrival delay of the k-th retransmission.

We are interested in deriving the distribution of $D_{(1)}$ – the time of the first arrival – before the cutoff time T_{CO}. When there are a maximum number of K_{retx} retransmissions before T_{CO}, we let

$$D_{(1),T_{CO}}^{K_{retx}} = \min_{k=0,...,K_{retx}} D_{k,T_{CO}}^{K_{retx}} = \min_{k=0,...,K_{retx}} \{d_{k,T_{CO}}^{K_{retx}} + kT_{TO}\} \tag{3.13}$$

be the time of the first arrival among all $K_{retx}+1$ messages sent out by the sensor. We note

$$\Pr\{D_{(1),T_{CO}}^{K_{retx}} \leq T_{CO}\} = p_{del,T_{CO}}^{K_{retx}}, \tag{3.14}$$

where the right-hand side of the equation was given in Eq. (3.9). Our goal is to derive the distribution of $D_{(1),T_{CO}}^{K_{retx}}$, that is, $\Pr\{D_{(1),T_{CO}}^{K_{retx}} \leq t\}$ for any $0 < t < T_{CO}$. For ease of presentation, in the remainder of this section, we assume that a certain T_{CO} value has been specified along with the resulting K_{retx} and drop them from the notations unless otherwise specified.

One may first be tempted to directly apply the result of the distribution of the minimum of n random variables, which is a special case of the order statistics [65]. Despite the seemingly similar relationship, the problem at hand is more complicated. First, from Eq. (3.13), we need to find the minimum of D_k for $k = 0, 1, 2, ..., K_{retx}$, which are from different distributions

for different k. Although results for independently and non-identically distributed random variables have been studied in the literature [9], the operations involve substantial use of matrix manipulation, and one must enumerate all $2^{K_{retx}+1}$ possible arrival patterns since each would yield a distinct result for the distribution of the minimum. To circumvent the issue, we follow another approach by finding the probability that the k-th retransmitted message (and the original message when $k = 0$) is the earliest to arrive, denoted as $\Pr\{\mathbb{I}_{D_{(1)}} = k\}$, in which $\mathbb{I}$ is the indicator for the earliest arriving message. As stated above, we have

$$\Pr\{\mathbb{I}_{D_{(1)}} = k\} = \Pr\{D_k \leq D_j\} \text{ for all } j \neq k. \tag{3.15}$$

$K_{retx} = 0$

For the simplest case where there is no retransmission before the cutoff time T_{CO}, that is, if $K_{retx} = 0$, the cdf of the first instance of arrival is simply the truncated cdf as shown in Eq. (3.12) for $k = 0$. To show this, we first have (a) the delivery probability in Eq. (3.9) is $(1 - p_L)F(T_{retx,0})$; and (b) the associated probability that the arrival time – which happens to be the delay of the original message as well in this case – is no greater than t is $(1 - p_L)F(t)$. And the cdf can thus be obtained by having (b) divided by (a):

$$F_{D^0_{(1)}}(t) = \Pr\{D^0_{(1)} \leq t\} = \frac{(1 - p_L)F(t)}{(1 - p_L)F(T_{retx,0})} = F_T^{T_{retx,0}}(t). \tag{3.16}$$

$K_{retx} = 1$

When $K_{retx} = 1$, the sensor can retransmit the message at most once before the cutoff time. There are basically two scenarios for the first instance of arrival, namely:

a. the original message arrives first before T_{CO};

b. the retransmitted message arrives first before T_{CO}.

The first case can be further subdivided into

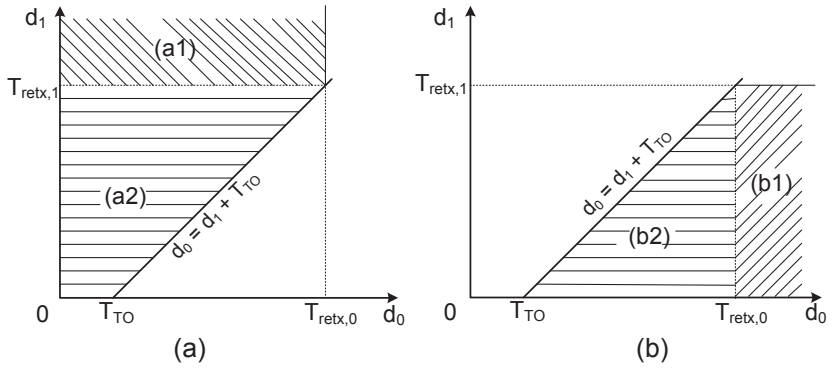


Figure 3.2: Distributions of d_0 and d_1

a1: the retransmitted message is not available by T_{CO};

a2: the retransmitted message is also delivered by T_{CO}, but its random arrival delay d_1 must satisfy $d_0 \leq d_1 + T_{TO}$.

Likewise, for the second scenario, we have

b1: the original message is not available by T_{CO};

b2: the original message is also delivered by T_{CO} with its random arrival delay d_0 satisfying $d_1 + T_{TO} < d_0$.

All different scenarios are illustrated in Fig. 3.2. Their probabilities are calculated as

$$\begin{aligned}\Pr\{\text{a1}\} &= p^0_{loss,T_{retx,1}} p^0_{del,T_{retx,0}} \\ &= (1-(1-p_L)F(T_{retx,1}))(1-p_L)F(T_{retx,0}).\end{aligned} \tag{3.17}$$

$$\begin{aligned}\Pr\{\text{a2}\} &= (1-p_L)^2 \Pr\{d_0 - T_{TO} \leq d_1 < T_{retx,1}\} \\ &= (1-p_L)^2 \int_0^{T_{retx,1}} \left(\int_0^{d_1+T_{TO}} f(d_0)\,\mathrm{d}d_0 \right) f(d_1)\,\mathrm{d}d_1 \\ &= (1-p_L)^2 \int_0^{T_{retx,1}} F(t+T_{TO}) f(t)\,\mathrm{d}t.\end{aligned} \tag{3.18}$$

Similarly, we have

$$\begin{aligned}\Pr\{\text{b1}\} &= p^0_{loss,T_{retx,0}} p^0_{del,T_{retx,1}} \\ &= (1-(1-p_L)F(T_{retx,0}))(1-p_L)F(T_{retx,1}). \end{aligned} \tag{3.19}$$

$$\begin{aligned}\Pr\{\text{b2}\} &= (1-p_L)^2 \Pr\{d_1 + T_{TO} < d_0 < T_{retx,0}\} \\ &= (1-p_L)^2 \int_{T_{TO}}^{T_{retx,0}} \left(\int_0^{d_0 - T_{TO}} f(d_1)\,\mathrm{d}d_1 \right) f(d_0)\,\mathrm{d}d_0 \\ &= (1-p_L)^2 \int_0^{T_{retx,1}} F(t) f(t+T_{TO})\,\mathrm{d}t. \end{aligned} \tag{3.20}$$

Note that the sum of Eqs. (3.18) and (3.20) is $p^0_{del,T_{retx,0}} p^0_{del,T_{retx,1}} = (1-p_L)F(T_{retx,0}) \cdot (1-p_L)F(T_{retx,1})$, the probability that both the original and the retransmitted messages are available at the cutoff time.

The resulting cdf of $D^1_{(1)}$ is hence

$$\begin{aligned} F_{D^1_{(1)}}(t) &= \Pr\{D^1_{(1)} \le t\} \\ &= \frac{\Pr\{\mathbb{I}_{D^1_{(1)}} = 0\} F_T^{T_{retx,0}}(t) + \Pr\{\mathbb{I}_{D^1_{(1)}} = 1\} F_T^{T_{retx,1}}(t - T_{TO})}{p^1_{del,T_{CO}}}, \end{aligned} \tag{3.21}$$

in which $\Pr\{\mathbb{I}_{D^1_{(1)}} = 0\}$ and $\Pr\{\mathbb{I}_{D^1_{(1)}} = 1\}$ are the sums of Eqs. (3.17) and (3.18), and Eqs. (3.19) and (3.20), respectively. From Eq. (3.9), we have the total delivery probability $p^1_{del,T_{CO}}$ for "normalizing" the probabilities to obtain the cdf.

$K_{retx} > 1$

When K_{retx} is any number greater than one, thanks to the independence of different retransmissions, we can carry out the above pairwise comparison for any arbitrary pair of arrivals. In fact, if we generalize the above results, we have for any pair i and j $(i < j)$ of effective

retransmissions

$$\Pr\{D_i \le D_j\} = (1-p_L)^2 \int_0^{T_{retx,j}} F\left(t + (j-i)T_{TO}\right) f(t)\,\mathrm{d}t + \left(1-(1-p_L)F(T_{retx,j})\right)(1-p_L)F(T_{retx,i}), \tag{3.22}$$

and

$$\Pr\{D_i > D_j\} = (1-p_L)^2 \int_0^{T_{retx,j}} F(t) f(t + (j-i)T_{TO})\,\mathrm{d}t + \left(1-(1-p_L)F(T_{retx,i})\right)(1-p_L)F(T_{retx,j}). \tag{3.23}$$

Repeating for all possible pairs, we have the probability of D_k being the minimum, that is, the k-th retransmitted message is received first, as

$$\begin{aligned}
&\Pr\{\mathbb{I}_{D_{(1)}^{K_{retx}}} = k\} \\
&= \prod_{\substack{j=0 \\ j\neq k}}^{K_{retx}} \Pr\{D_k \le D_j\} \\
&= (1-p_L)\cdot \prod_{\substack{j=0 \\ j\neq k}}^{K_{retx}} \Bigg\{ (1-p_L) \int_0^{T_{retx,\max\{k,j\}}} F\left(t + \max\{0,(j-k)T_{TO}\}\right) \cdot \\
&\qquad f\left(t + \max\{0,(k-j)T_{TO}\}\right)\,\mathrm{d}t \\
&\qquad + \left(1-(1-p_L)F(T_{retx,j})\right) F(T_{retx,k}) \Bigg\}.
\end{aligned} \tag{3.24}$$

This leads to the overall distribution of the $D_{(1)}^{K_{retx}}$:

$$\begin{aligned}
F_{D_{(1)}^{K_{retx}}}(t) &= \Pr\{D_{(1)}^{K_{retx}} \le t\} \\
&= \frac{\sum_{k=0}^{K_{retx}} \Pr\{\mathbb{I}_{D_{(1)}^{K_{retx}}} = k\} F_T^{T_{retx,k}}(t - kT_{TO})}{p_{del,T_{CO}}^{K_{retx}}}.
\end{aligned} \tag{3.25}$$

3.3.2 Arrival Time: Two-way Communication Analysis

Eq. (3.25) describes the distribution of the earliest arrival time under the condition that all K_{retx} rounds of retransmissions are sent out after the original message. In reality, though, the total number of retransmissions can be anywhere from 0 to K_{retx}. In this subsection, we consider the two-way communications that determines the number of the actual retransmissions, which in turn affects the overall distribution of the earliest arrival time before the cutoff.

For satellite systems with conventional bent pipe type of transponders [70], one uplink (sensor $\rightarrow$ satellite) and downlink (satellite $\rightarrow$ FC) pair is used for the forward link, and the reverse link similarly consists of the uplink (FC $\rightarrow$ satellite) and downlink (satellite $\rightarrow$ sensor) pair. Depending on specific channel allocation schemes (e.g., TDMA- or FDMA-based), that is, whether the forward and reverse channels are assigned the same frequency band, the delay distribution of the ACK could vary from that of the messages[4]. Regardless, we have the pdf of the sum of two random delay values being expressed as the convolution of their respective pdfs:

$$h(t) = f(t) \star g(t) = \int_{t=0}^{\infty} f(u)g(t-u)\,\mathrm{d}u, \tag{3.26}$$

in which f and g are the distributions of the forward and reverse links, respectively. Meanwhile, if the ACK message is lost over the reverse link with a probability $p_{L,ACK}$, the overall probability that the ACK message can be eventually delivered is $(1-p_L)(1-p_{L,ACK})$, and its complement

$$p_{L,T} = 1 - (1-p_L)(1-p_{L,ACK}) \tag{3.27}$$

is the "total" loss rate of the "super-message" that includes both the estimate message and

[4]Also the initial delay could be quite different too, owing to the usually much smaller size of the ACK messages.

ACK. With this loss rate and $h(t)$ function, we can derive a general form of the arrival time.

Probability of Having k_1 ($0 \leq k_1 \leq K_{retx}$) Retransmissions

First, we have the trivial case in which $K_{retx} = 0$ as $T_{CO} \leq T_{TO}$, then with probability one, there is no retransmission. Next we focus on the cases where $K_{retx} \geq 1$.

Having exactly k_1 ($0 \leq k_1 < K_{retx}$) retransmissions means that the earliest reception of the ACK message by the sensor occurs in the interval $[k_1 T_{TO}, (k_1 + 1)T_{TO})$. In other words, the first instance of the ACK arrival at the sensor

$$D^{k_1}_{T,(1)} = \min_{k=0,\ldots,k_1} \{D_{T,k}\} \tag{3.28}$$

must satisfy

$$k_1 T_{TO} \leq D^{k_1}_{T,(1)} < (k_1 + 1)T_{TO}, \tag{3.29}$$

in which $D^{k_1}_{T,(1)}$ is similarly defined as in Eq. (3.13), with the subscript T specifying that this is the arrival time accounting for the total delay from both forward and reverse links. Meanwhile, we define the delivery rate for the "super-message" described earlier as $p^{K_{retx}}_{T,del,t}$ with the maximum number of retransmissions K_{retx}. Then we have

$$\begin{aligned} &\Pr\{\text{There are exactly } k_1 \text{ retransmissions, } 0 \leq k_1 < K_{retx}\} \\ &= p^{K_{retx}}_{T,del,(k_1+1)T_{TO}} - p^{K_{retx}}_{T,del,k_1 T_{TO}}. \end{aligned} \tag{3.30}$$

The delivery probabilities can be similarly calculated as in Eq. (3.9), with p_L being replaced by $p_{L,T}$ and $F(t)$ by $H(t) = \int_0^t h(u)\,\mathrm{d}u$, respectively.

On the other hand, having K_{retx} retransmissions means that none of the ACKs have been

received by $K_{retx}T_{TO}$, and we have

$$\begin{aligned}
&\Pr\{\text{There are exactly } K_{retx} \text{ retransmissions}\} \\
&= \prod_{k=0}^{K_{retx}-1} \Pr\{D_{T,k}^{K_{retx}} > K_{retx}T_{TO}\} \\
&= \prod_{k=0}^{K_{retx}-1} \left[1-(1-p_{L,T})H((K_{retx}-k)T_{TO})\right]. \qquad (3.31)
\end{aligned}$$

Distribution of the Arrival Time for a Given T_{CO}

With K_{retx} in Eqs. (3.9) and (3.24) being replaced by any k_1 ($0 \le k_1 \le K_{retx}$), we can easily find the delivery rate $p_{del,T_{CO}}^{k_1}$ after k_1 rounds of retransmissions and the probability $\Pr\{\mathbb{I}_{D_{(1)}^{k_1}} = k\}$ that the k-th retransmission marks the earliest arrival among all k_1+1 sent out messages. And then we have the cdf with exactly k_1 retransmissions as

$$\begin{aligned}
F_{D_{(1)}^{k_1}}(t) &= \Pr\{D_{(1)}^{k_1} \le t\} \\
&= \frac{\sum_{k=0}^{k_1} \Pr\{\mathbb{I}_{D_{(1)}^{k_1}} = k\} F_T^{T_{retx,k}}(t-kT_{TO})}{p_{del,T_{CO}}^{k_1}}. \qquad (3.32)
\end{aligned}$$

Finally, we can combine Eqs. (3.30), (3.31), and (3.32) to obtain the distribution of the arrival time for any given T_{CO}:

$$F_{D_{(1)}}(t) = \sum_{k_1=0}^{K_{retx}} F_{D_{(1)}^{k_1}}(t) \Pr\{\text{There are } k_1 \text{ retransmissions}\}. \qquad (3.33)$$

3.4 State Estimation and Fusion with Retransmission and Retrodiction

We have seen that retransmission can effectively increase the message delivery rate, which in turn is expected to improve the estimation quality. However, at times, we wish to further expedite this recovery process so that the final estimate can be reported earlier; besides, the

system may impose rather stringent requirements on the estimation errors so that given the same allocated time for retransmission, we want more accurate estimates from the output of the fusion center. This section addresses these concerns by means of utilizing estimate retrodiction.

3.4.1 Estimate Retrodiction

Estimation of a target state at a particular time based on measurements collected beyond that time is generally called retrodiction or smoothing. Retrodiction improves the accuracy of the estimates, thanks to the use of more information, at the cost of extra delay. Nevertheless, the inherent link delay in a long-haul network occurring before the final reporting entails that the fusion center can exploit the opportunities for potential retrodiction to improve the accuracy of the fused estimate. Moreover, the randomness of the arrival delay of different messages also facilitates the fusion center to opportunistically interpolate the missing messages from the available ones for subsequent time periods.

Consider a discrete-time system in which estimates are generated regularly at a certain fixed period T_I. An important note here is that a subsequent estimate must be available in order for retrodiction to happen. As such, the retransmission window W and the cutoff time T_{CO} must contain at least one estimation interval T_I (and also the initial latency of the latest estimate to arrive at the fusion center) so that the next estimate could potentially arrive therein. This is shown in Fig. 3.3. In the first case, no retrodiction can be possibly performed because the cutoff time comes before the end of the same estimation interval; while in the second and third cases, a maximum of one and two, respectively, rounds of retrodiction can be possibly carried out. In many actual systems, the near real-time reporting requirement dictates that the estimation intervals are chosen fairly small. As a result, the next estimate can be generated and sent to the fusion center in time for retrodiction before the reporting deadline.

Let $\mathbf{x}_n$ denote the true target state at time $t = nT_I$ and $\hat{\mathbf{x}}_n$ its estimate. One note is

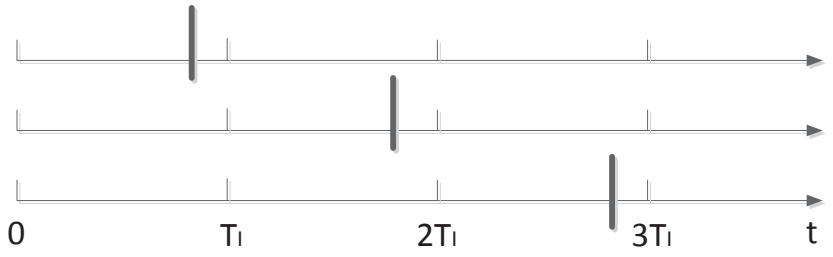


Figure 3.3: Timing of estimate retrodiction: The estimation interval is T_I and different choices of the cutoff time T_{CO} are marked by bold lines.

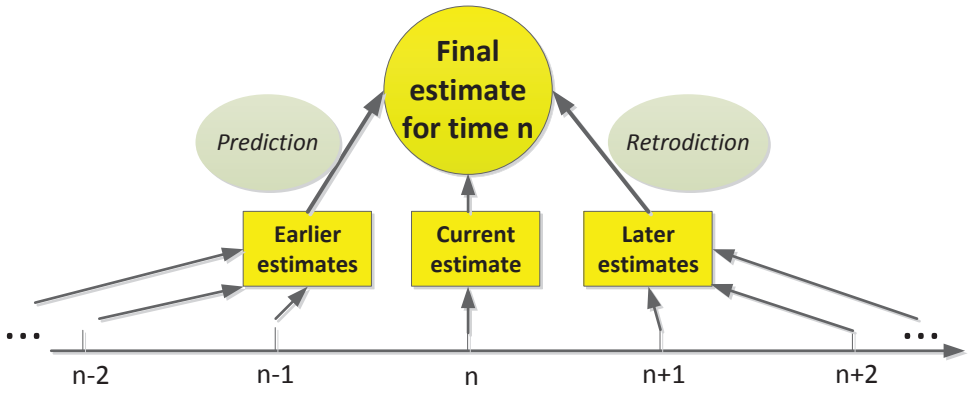


Figure 3.4: Prediction and retrodiction: The estimate at time n is to be obtained.

that since we have both continuous time – when addressing latency – and the discrete time indices for retrodiction analysis, all time subsequently labeled as "n" – the time of interest to us – should be understood as the continuous time nT_I and the retransmission parameters T_{TO} and W and the cutoff time T_{CO} are all defined relative to this time instant.

Fig. 3.4 demonstrates the effect of prediction and retrodiction on the quality of the state estimate. For each sensor, for example, we have the following types of estimates when retrodiction of *up to one step* is performed:

a. $\hat{\mathbf{x}}_n$, shorthand for $\hat{\mathbf{x}}_{n|n}$, the "default" estimate sent from the sensor;

b. $\hat{\mathbf{x}}_{n^-}$, the predicted estimate;

c. $\hat{\mathbf{x}}_{n^-|n+1}$, the predicted & retrodicted estimate; and

d. $\hat{\mathbf{x}}_{n^+|n+1}$, the retrodicted estimate.

In the cases a) and d), the estimate $\hat{\mathbf{x}}_n$ is received by the fusion center; whereas in both other cases, this estimate is missing and hence prediction of one or multiple steps is at first necessary. As is well-known in filtering theory, estimates derived from prediction alone generally have higher errors when system uncertainty exists; and the errors will increase with the number of prediction steps that have accrued. For example, $\hat{\mathbf{x}}_{n|n-2}$ is a worse estimate than $\hat{\mathbf{x}}_{n|n-1}$ in terms of accuracy. On the other hand, the presence of the subsequent estimate $\hat{\mathbf{x}}_{n+1}$ (in the last two cases) helps improve the quality of the estimate of time n. Of course, the improvement is on top of the predicted estimate in case c) but on the already received $\hat{\mathbf{x}}_n$ in case d). Of concern here is whether an interpolated estimate from retrodiction – such as that in case c) – can adequately substitute the default estimate in case a); and what is the probabilistic performance of obtaining these different types of estimates so that the system requirement on estimation errors can be met.

The quantitative performance of retrodiction also depends on the specific algorithm. In this study, we apply Rauch-Tung-Striebel (RTS) retrodiction [73] because not only the algorithm is easy to implement, with relatively low computational cost, but the algorithm involves only the state estimates and their covariances – rather than the raw measurement data – which is well suited in our settings when the fusion center needs to run the algorithm.

Based on the criteria that whether the sensors actively participate in retrodiction during message retransmission, we categorize our schemes into non-cooperative and cooperative retrodiction. In the former case, message retransmission is carried out in exactly the same way as before; it is up to the fusion center to opportunistically apply retrodiction whenever applicable. In contrast, cooperative retrodiction means that the sensors themselves, upon request, send out the retrodicted estimates during retransmission so that the fusion center can directly fuse such retrodicted values if successfully delivered. In what follows, we derive the delivery rates of different types of estimates during retransmission for both types of

retrodiction and consider their impact on the final error performance.

3.4.2 Non-Cooperative Retrodiction

The fusion center can opportunistically apply retrodiction whenever applicable; this can of course be combined with message retransmission introduced earlier. In what follows, we derive the delivery rates of different types of estimates following retransmission and retrodiction. Although here only one-step retrodiction is shown, analysis for retrodiction of two or more steps can be similarly obtained, albeit in a more exhaustive manner, as the number of possible scenarios grows exponentially with the number of steps[5].

We have the following probabilities at the cutoff time $nT_I + T_{CO}$:

- p_{n^-}, the probability that $\hat{\mathbf{x}}_{n^-}$ is reported (neither $\hat{\mathbf{x}}_n$ or $\hat{\mathbf{x}}_{n+1}$ is delivered by the cutoff);
- $p_{n|n}$, the probability that $\hat{\mathbf{x}}_{n|n}$ is reported ($\hat{\mathbf{x}}_{n+1}$ is not delivered yet);
- $p_{n^-|n+1}$, the probability that $\hat{\mathbf{x}}_{n^-|n+1}$ is reported ($\hat{\mathbf{x}}_{n+1}$ has been delivered but not $\hat{\mathbf{x}}_n$);
- $p_{n^+|n+1}$, the probability that $\hat{\mathbf{x}}_{n^+|n+1}$ is reported (both $\hat{\mathbf{x}}_n$ and $\hat{\mathbf{x}}_{n+1}$ have been delivered).

The analysis in Sec. 3.2 can be readily applied here, thanks to the independence of the transmission from different time intervals. Similar to Eq. (3.2), we have the maximum number of retransmissions during $t \in [nT_I + T_I, nT_I + T_{CO}]$ given as

$$K_{retx,retr_1} = \left\lceil \frac{\min\{T_{CO} - T_I, W\}}{T_{TO}} \right\rceil - 1, \tag{3.34}$$

in which the subscript $retr_1$ denotes that there is a maximum of one-step retrodiction. And the duration in Eq. (3.3) can also be defined likewise for the above time interval:

$$T_{retx,retr_1,k} = T_{CO} - T_I - kT_{TO}, \qquad \text{for } k = 0, 1, ..., K_{retx,retr_1}. \tag{3.35}$$

[5]Another caveat is that message-level loss and delay may worsen as significantly more data are sent simultaneously with increasing retrodiction steps.

To calculate the above probabilities, we need to consider the probability that $\hat{\mathbf{x}}_{n+1}$ is delivered at the cutoff time. This probability, denoted as $p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}$, follows the very same form as in Eq. (3.9), but with newly defined Eqs. (3.34) and (3.35) substituting the corresponding terms. Then we have

$$p_{n^-} = (1 - p_{del,T_{CO}}^{K_{retx}})(1 - p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}) \tag{3.36}$$

$$p_{n|n} = p_{del,T_{CO}}^{K_{retx}}(1 - p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}) \tag{3.37}$$

$$p_{n^-|n+1} = (1 - p_{del,T_{CO}}^{K_{retx}})p_{del,T_{CO}-T_I}^{K_{retx,retr_1}} \tag{3.38}$$

$$p_{n^+|n+1} = p_{del,T_{CO}}^{K_{retx}} p_{del,T_{CO}-T_I}^{K_{retx,retr_1}} \tag{3.39}$$

3.4.3 Cooperative Retrodiction

In this subsection, we provide similar analysis as above – that is, with a maximum of one step of retrodiction being performed – and focus on comparisons between cooperative and non-cooperative retrodiction techniques. In particular, two possible implementations of the cooperative retrodiction are studied, one in which we only consider one-way communications – as we have done so far – and the other requiring two-way analysis that addresses the delivery of the ACK messages as well.

Condition for One-Step Cooperative Retrodiction

In non-cooperative retrodiction, message retransmission is scheduled in the manner as described in Sec. 3.2 and a sensor is oblivious to the retrodiction process happening at the fusion center. In contrast, cooperative retrodiction requires the retrodicted estimates to be sent out during retransmission. In order for a sensor to actually send out its one-step retrodicted estimates, the retransmission window W should not have expired at the end of the estimation interval T_I; in fact, there should be at least one round of retransmission initiated by the sensor after T_I when the retrodicted estimate can be sent out by the sensor. Hence, compared to non-cooperative retrodiction, tighter conditions are in place for cooperative

retrodiction.

Cooperative Retrodiction: One-way Communications without ACK

During the time period $[nT_I + T_I, nT_I + T_{CO}]$, instead of the original estimate, the sensor sends out the retrodicted estimate $\hat{\mathbf{x}}_{n^+|n+1}$ directly. The total number of retransmission rounds for the original message during $[nT_I, nT_I + T_I]$ is reduced to

$$K_{retx,coop} = \left\lceil \frac{T_I}{T_{TO}} \right\rceil - 1, \tag{3.40}$$

while both the new state estimate $\hat{\mathbf{x}}_{n+1|n+1}$ and retrodicted estimate $\hat{\mathbf{x}}_{n^+|n+1}$ are sent after T_I. If $T_I = lT_{TO}$, $l = 1, 2, 3, ...$, both estimates will undergo

$$K_{retx,coop,retr_1} = K_{retx,retr_1} = \left\lceil \frac{T_{CO} - T_I}{T_{TO}} \right\rceil - 1, \tag{3.41}$$

rounds of retransmission, in which the subscripts "$coop, retr_1$" and "$retr_1$" denote cooperative and non-cooperative retrodiction of up to one step respectively. Similar to our earlier analysis, we can obtain the delivery probabilities of the original estimate $\hat{\mathbf{x}}_{n|n}$ as $p_{del,T_{CO}}^{K_{retx,coop}}$, of the subsequent estimate $\hat{\mathbf{x}}_{n+1|n+1}$ as $p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}$, and of the retrodicted estimate by the sensor $\hat{\mathbf{x}}_{n^+|n+1}$ as $p_{del,T_{CO}-T_I}^{K_{retx,coop,retr_1}}$. With an increased size of the state space, the probabilities of obtaining different types of estimates at the cutoff time can now be computed as

$$p_{n^-} = (1 - p_{del,T_{CO}-T_I}^{K_{retx,coop,retr_1}})(1 - p_{del,T_{CO}}^{K_{retx,coop}})(1 - p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}) \tag{3.42}$$

$$p_{n|n} = (1 - p_{del,T_{CO}-T_I}^{K_{retx,coop,retr_1}})p_{del,T_{CO}}^{K_{retx,coop}}(1 - p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}) \tag{3.43}$$

$$p_{n^-|n+1} = (1 - p_{del,T_{CO}-T_I}^{K_{retx,coop,retr_1}})(1 - p_{del,T_{CO}}^{K_{retx,coop}})p_{del,T_{CO}-T_I}^{K_{retx,retr_1}} \tag{3.44}$$

$$p_{n^+|n+1} = p_{del,T_{CO}-T_I}^{K_{retx,coop,retr_1}} + (1 - p_{del,T_{CO}-T_I}^{K_{retx,coop,retr_1}})p_{del,T_{CO}}^{K_{retx,coop}}p_{del,T_{CO}-T_I}^{K_{retx,retr_1}} \tag{3.45}$$

Note in Eq. (3.45) that the estimate $\hat{\mathbf{x}}_{n^+|n+1}$ can be obtained either directly from the sensor, or indirectly in the manner we discussed in the non-cooperative retrodiction case.

Cooperative Retrodiction: Two-way Communications with ACK

The analysis above is along the same line as that in Sec. 3.2, where only one-way communication is considered. This can also be seen as the extreme case where no ACK is ever sent back by the fusion center, since the sensors always have their one-step retrodicted estimates sent out after one estimation interval T_I. In reality, though, the ACK might have been successfully received by the sensor within T_I, thereby obviating the need for further retransmission. Next, we carry out two-way communication analysis to account for such scenarios.

Recall from Eq. (3.27) that $p_{L,T}$ is the loss rate of the "super-message" that includes both the estimate message and ACK. With this loss rate and $h(t)$ function in Eq. (3.26), we have the probability that the ACK is delivered by time t and hence no more retransmission occurs afterward:

$$\overline{p_{retx}}(t) = (1 - p_{L,T})H(t), \text{ for } t \in [0, \min\{T_{CO}, W\}], \tag{3.46}$$

in which $H(t) = \int_0^t h(u)f(u)\,\mathrm{d}u$ is the cdf of the two-way communications delay.

After the ACK has been successfully received within one estimation interval, the retrodicted estimate is no longer to be sent out, thereby reducing the chance that the best estimate $\hat{\mathbf{x}}_{n^+|n+1}$ is available at the fusion center. Subsequently, with probability $1 - \overline{p_{retx}}(T_I)$, Eqs. (3.42)–(3.45) hold true; on the other hand, with probability $\overline{p_{retx}}(T_I)$, only two types of estimates are possible to be used by the cutoff time – since $\hat{\mathbf{x}}_{n|n}$ has been received successfully – with probabilities $1 - p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}$ for $\hat{\mathbf{x}}_{n|n}$ and $p_{del,T_{CO}-T_I}^{K_{retx,retr_1}}$ for $\hat{\mathbf{x}}_{n^+|n+1}$. Using the law of total probability, we can easily incorporate them to calculate the overall probabilities of obtaining each type of estimate.

3.4.4 Impact of Retransmission and Retrodiction on Estimation Error

It's generally difficult to derive the exact error performance analytically. Note that in Eqs. (3.36) and (3.38), the time at which the last received estimate was generated is not specified; that is, the number of prediction steps leading up to $\hat{\mathbf{x}}_{n^-}$ is unknown. Even with known error profiles for $\hat{\mathbf{x}}_{n|n}$ and $\hat{\mathbf{x}}_{n^+|n+1}$, an exact evaluation of estimation error for $\hat{\mathbf{x}}_{n^-}$ and $\hat{\mathbf{x}}_{n^-|n+1}$ requires knowledge of the errors with an arbitrary number of prediction steps, which is generally unrealistic. If the off-line error performance can be obtained for the above original, predicted, and retrodicted estimates, an approximate estimation error can be calculated as the probabilistic sum of errors for different types of estimates. Next, via simulations of a ballistic target tracking application, we explore the impact of both retransmission and retrodiction on actual estimation performance.

3.5 Performance Evaluation

We carry out system-level tracking simulations of coasting ballistic targets. First, we focus on the one-target case where fusion is performed by the fusion center to combine the state estimates of a single target sent by the sensors. Then, the multi-target tracking scenario is investigated, where a probability association model is used to capture the performance of data association prior to the estimate fusion. We aim to explore the benefits and limitations of applying retransmission and retrodiction under variable loss and delay statistics.

3.5.1 Effect of Retransmission Schedules on Tracking Performance

Target Model

The trajectory of a ballistic target, from launch to impact, is divided into three basic phases: boost, coast, and reentry. In this dissertation we focus on the coast phase, which is an exo-

atmospheric, free-flight motion, continuing until the Earth's atmosphere is reached again. The states of a coasting ballistic target – whose motion is governed predominantly by gravity – are generated using the following state-space model [45]:

$$\dot{\mathbf{x}} \triangleq \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{v}} \end{bmatrix} = \mathbf{f}\left(\begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix}\right) \triangleq \begin{bmatrix} \mathbf{v} \\ \mathbf{a}_G(\mathbf{p}) \end{bmatrix}. \tag{3.47}$$

The target state vector $\mathbf{x} = \begin{bmatrix} \mathbf{p}^T & \mathbf{v}^T \end{bmatrix}^T$, where $\mathbf{p} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ and $\mathbf{v} \triangleq \dot{\mathbf{p}} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T$ are the target position and velocity vectors, respectively. $\mathbf{a}_G(\mathbf{p})$ is the gravitational acceleration under the spherical Earth model [45]:

$$\mathbf{a}_G(\mathbf{p}) = -\frac{\mu}{p^2}\mathbf{u}_p = -\frac{\mu}{p^3}\mathbf{p}, \tag{3.48}$$

where $\mathbf{p}$ is the vector from the Earth's center to the target, $p \triangleq \|\mathbf{p}\|$ is its length, $\mathbf{u}_p \triangleq \mathbf{p}/p$ is the unit vector in the direction of $\mathbf{p}$, and $\mu = 3.986012 \times 10^5$ km^3/s^2 is the Earth's gravitational constant.

The algorithm for state propagation can be found in [81] and the initial target state is [38]: $[113.75 \quad 3960 \quad 5150 \quad 0.988 \quad 3.33 \quad -6.01]^T$, in which the position and velocity values are in the units of km and km/s respectively.

Sensor and Noise Profiles

A total of $N = 5$ sensors are deployed for reporting their state estimates of the dynamic target defined above. Estimates are generated once per second. A state estimate $\hat{\mathbf{x}}_i(k)$ from sensor i at time k is generated by adding random Gaussian noise to the true states[6] as in [68]:

$$\hat{\mathbf{x}}_i(k) = \mathbf{x}(k) + \mathbf{n}_i(k), \tag{3.49}$$

[6]This is a simplified model for scenarios in which the dependency of estimation errors on the underlying states are negligible.

where $\mathbf{x}(k)$ is the true target state for a target at time k, and $\mathbf{n}_i(k) \sim \mathcal{N}(\underline{0}, \mathbf{\Sigma})$. $\mathbf{\Sigma}$ is a diagonal matrix:

$$\mathbf{\Sigma} = diag\left(\begin{bmatrix} \sigma_x^2 & \sigma_y^2 & \sigma_z^2 & \sigma_{\dot{x}}^2 & \sigma_{\dot{y}}^2 & \sigma_{\dot{z}}^2 \end{bmatrix}\right), \tag{3.50}$$

where σ_x^2, σ_y^2, and σ_z^2 are the position error variances, and $\sigma_{\dot{x}}^2$, $\sigma_{\dot{y}}^2$, and $\sigma_{\dot{z}}^2$ are the velocity error variances. The following state estimation errors are set commonly for all sensors: $\sigma_x^2 = \sigma_y^2 = \sigma_z^2 = 1$ and $\sigma_{\dot{x}}^2 = \sigma_{\dot{y}}^2 = \sigma_{\dot{z}}^2 = 10^{-4}$.

Fusion Rule

We apply the linear fuser defined as follows:

$$\mathbf{P}_F = \left(\sum_{i=1}^{L} \mathbf{P}_i^{-1}\right)^{-1}, \text{ and } \hat{\mathbf{x}}_F = \mathbf{P}_F \sum_{i=1}^{L} \mathbf{P}_i^{-1} \hat{\mathbf{x}}_i, \tag{3.51}$$

where $\hat{\mathbf{x}}_F$ is the fused estimate and $\mathbf{P}_F$ is its error covariance matrix. $\mathbf{P}_i$ and $\hat{\mathbf{x}}_i$ are similarly defined for sensor i. A total of L ($L \leq N$) state estimates are combined at the fusion center. This simple fuser is a special form of the track-to-track fuser [14] where the process noise is zero.

Communication Link Statistics

The default forward link loss rate is set to be $p_L = 0.5$, compared to that of the reverse link $p_{L,ACK} = 0.1$. The arrival delay of both directions satisfies the shifted exponential distribution defined in Eq. (3.1), with $\mu_{D,F} = 0.3$ s and $\mu_{D,R} = 0.2$ s for the forward and reverse links respectively and a common $T = 0.5$ s. The default T_{TO} and W are set to be 1.5 s and 4.5 s respectively, which are both integer multiples of the measured average RTT at 1.5 s.

MSE analysis

From Eq. (3.51), the mean-square-error (MSE) of the position and velocity estimates of the linearly fused estimate satisfies

$$|\hat{\mathbf{p}}_F - \mathbf{p}|^2 = |\hat{\mathbf{p}}_i - \mathbf{p}|^2/L, \text{ and } |\hat{\mathbf{v}}_F - \mathbf{v}|^2 = |\hat{\mathbf{v}}_i - \mathbf{v}|^2/L, \tag{3.52}$$

if the sensors have the same estimation error profiles (that is, the same MSEs). Suppose the system imposes its maximum tolerable errors of position and velocity estimates as $MSE_{max,\hat{\mathbf{p}}}$ and $MSE_{max,\hat{\mathbf{v}}}$ respectively, with independent transmissions from up to N sensors, the minimum delivery probability should satisfy

$$p_{del,min} = \frac{L_{min}}{N} = \frac{1}{N}\max\{\frac{|\hat{\mathbf{p}}_i - \mathbf{p}|^2}{MSE_{max,\hat{\mathbf{p}}}}, \frac{|\hat{\mathbf{v}}_i - \mathbf{v}|^2}{MSE_{max,\hat{\mathbf{v}}}}\}, \tag{3.53}$$

in which L_{min} is the minimum number of estimates that should be delivered. The actual delivery probability from message retransmission as expressed in Eq. (3.9) must be checked against this minimum delivery rate to ensure the MSE requirements are met.

The distribution of the arrival time, as in Eq. (3.33), can be used to derive an upper bound of the estimation error. For a given cutoff time T_{CO}, when the fusion center decides to finalize the estimate at an earlier time t, according to Eq. (3.51), the expected contribution of the position estimate from the sensor i is lower bounded by $\frac{\alpha}{|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}|^2}$, in which $\alpha = F_{D_{(1)}^{T_{CO}}}(t)$. The other part, weighted by $1-\alpha$, depends on how the fuser chooses to substitute the missing estimate, such as using one- or multi-step prediction. If we consider the error performance of the fused estimate, the upper bound – as in the worst-case scenario in which a missing estimate results in zero contribution – is

$$|\hat{\mathbf{p}}_F(t) - \mathbf{p}|^2_{max} = \frac{|\mathbf{p}_F(T_{CO}) - \mathbf{p}|^2}{F_{D_{(1)}^{T_{CO}}}(t)}, \text{ for } 0 < t \leq T_{CO}, \tag{3.54}$$

in which $\mathbf{p}_F(T_{CO})$ is the fused position estimate at time T_{CO}. A similar result can be found

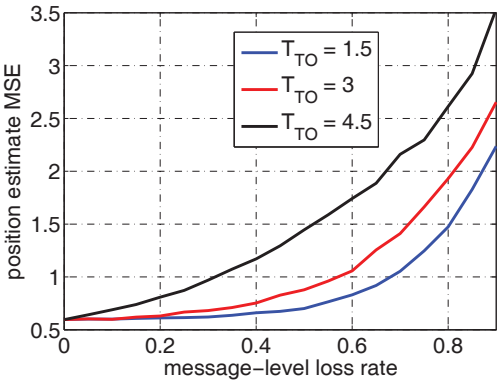


Figure 3.5: Position MSE (km^2) versus message-level loss rates with $T_{CO} = 5$ s and different T_{TO}

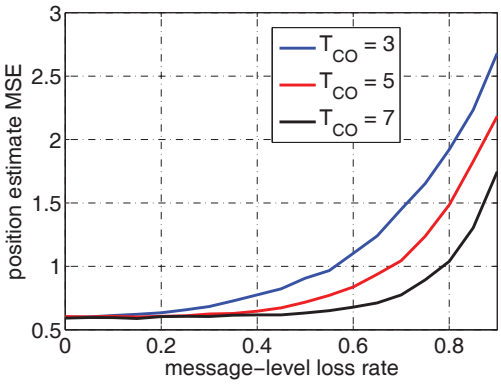


Figure 3.6: Position MSE (km^2) versus message-level loss rates with $T_{TO} = 1.5$ s and different T_{CO}

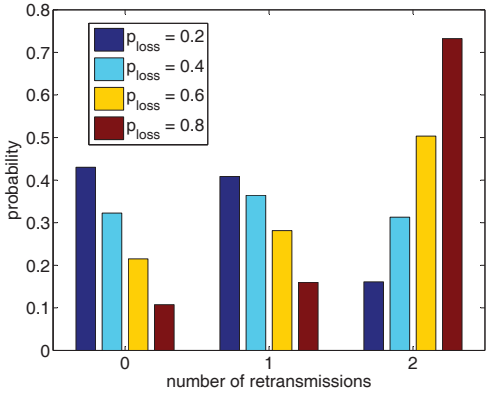


Figure 3.7: Probability of using different number of retransmissions with different loss rates and $K_{retx} = 2$

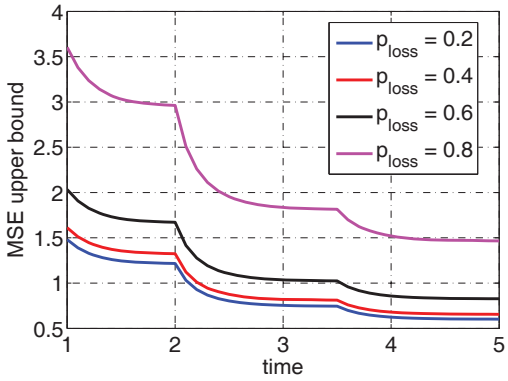


Figure 3.8: Upper bound of the position MSE (km^2) with variable message-level loss rates

for the velocity estimate.

Timeout T_{TO} and Cutoff T_{CO}

Figs. 3.5 and 3.6 demonstrate the effect of different retransmission timeout and fusion cutoff. There is no retransmission when $W = T_{TO} = 4.5$ s. When T_{TO} is set to be 1.5 s, however, two rounds of retransmissions can effectively reduce the MSE of the estimate. For example, when the loss rate is 60%, the error is reduced by more than 50%. Likewise, for a given T_{TO}, when the reporting deadline requirement is tightened, as reflected by decreasing cutoff time T_{CO}, the estimation MSE will increase given the same loss rate. On the other hand,

when read horizontally, the plots indicate that to meet the same MSE requirement, with increasing loss rates, T_{TO} should be reduced and/or T_{CO} should be increased. A good rule of thumb to determine the MSE performance is the ratio T_{CO}/T_{TO}, although this rule may at times fail due to the periodicity of the retransmission process, especially when the values being compared are close.

Retransmission Performance

In Fig. 3.7, we plot the proportion of different numbers of retransmissions with respect to various loss rates. Note that the last group (labeled as "2") does not indicate the message will be delivered within this round, but rather this is the last try as $K_{retx} = 2$. As expected, an increased message-level loss rate requires more rounds of retransmissions so that the message can be recovered with the same probability over a longer period of time.

Upper Bound of the MSEs From Early Cutoff

Finally, in Fig. 3.8, the upper bounds of the MSEs resulting from earlier-than-scheduled cutoff are shown. The singular points in these plots indicate the arrival of a new round of retransmission. The concavity of the bounds (excluding the singular points) implies that the "best" time for early fusion is roughly in the middle of each round (accounting for the link delay), where the deepest descent in this round has occurred. In addition, as time inches closer to the cutoff time T_{CO}, the bound also approaches the actual MSE obtained when the cutoff time is set at that point, and is a good indicator of the actual MSE performance.

3.5.2 Tracking of One Coasting Target

Target Model

The target model remains the same as that introduced in the previous subsection. We note the generated trajectory there – which happens to be the very last few seconds before the target enters the re-entry stage starting at an altitude of about 100 km – is very short. Hence,

we have backtracked a segment of the earlier trajectory so we can focus on the position state estimates during the last 30 seconds prior to the re-entry stage.

Sensor Profiles

A total of $N = 3$ sensors are deployed for reporting their state estimates with the common estimation interval set as $T_I = 2$ s. The measurements ($\mathbf{z}$) of the range (r), elevation (E), and azimuth (A) of the target are generated using the following measurement model [38]:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{v}, \tag{3.55}$$

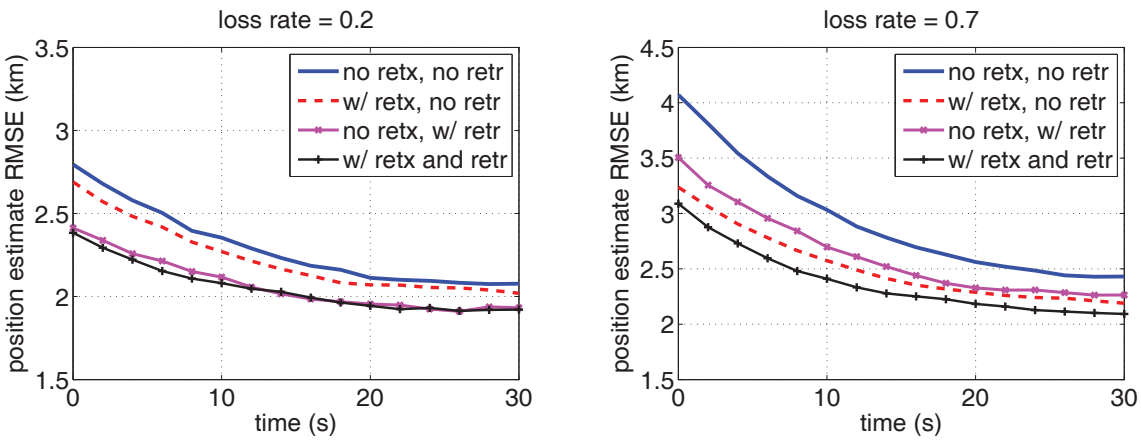
where the target state $\mathbf{x}$ is in Cartesian coordinates, but the measurement $\mathbf{z}$ and additive noise $\mathbf{v}$ are in the sensor spherical coordinates. If $[x \quad y \quad z]^T$ is the true position of the target, then the measurement[7] is given as

$$\mathbf{z} = \begin{bmatrix} r \\ E \\ A \end{bmatrix} + \mathbf{v} = \begin{bmatrix} \sqrt{x^2+y^2+z^2} \\ \tan^{-1}\left(\frac{z}{\sqrt{x^2+y^2}}\right) \\ \tan^{-1}\left(\frac{x}{y}\right) \end{bmatrix} + \mathbf{v}, \tag{3.56}$$

$$\mathbf{v} \sim \mathcal{N}(0, \mathbf{R}), \quad \mathbf{R} = \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_E^2 & 0 \\ 0 & 0 & \frac{\sigma_E^2}{\cos^2(E)} \end{bmatrix}. \tag{3.57}$$

A simplified radar model is used to define the range and elevation error standard deviations. The σ_r and σ_E values of a ballistic target/satellite tracking phased array radar, the Cobra Dane, are 15 ft and 0.05° respectively, according to [24]; and these values are assumed for all the sensors. The sensors apply the recursive best linear unbiased estimator (BLUE) proposed in [86] which improves upon the measurement-conversion approach [43]; that is, the output has the minimum MSE among all linear unbiased filters in Cartesian coordinates.

[7]Note that this measurement has been normalized to any sensor's own known location.



(a) (b)

Figure 3.9: Position estimate RMSE over time, with loss rates of (a) 0.2; and (b) 0.7

As in [86], the filter is initialized with an effectively large state error covariance and a highly inaccurate state estimate.

Fusion Rule

We apply a linear fuser similar to that defined in (3.51). With imperfect communications, the number of received state estimates for any time of interest is often less than the desired value N. The fusion center can apply two approaches: First, as in Eq. (3.51), it uses the actual number of available estimates L ($L \leq N$) and ignores the remaining, if any, unavailable tracks. Alternatively, it can substitute the predicted (or retrodicted) estimates in place of the original, if the latter is unavailable, and uses all N tracks for fusion. While the first approach is easier to implement, its estimation error is often worse; in other words, a predicted estimate still more or less provides error reduction for the fused estimate when compared to the case where the track is simply dropped. Therefore, in what follows, we pursue the second method of fusing the estimates in which all the tracks are used.

Communication Link Statistics

The following communication link statistics are used for the remainder of this section. Two link loss rates of 0.2 and 0.7, are studied, representing respectively the low and high loss

scenarios. The arrival delay satisfies the shifted exponential distribution defined in Eq. (3.1), with $\mu_D = 0.3$ s and the initial latency $T = 0.5$ s. With the cutoff time set as $T_{CO} = 3$ s – the same for the retransmission window W – we explore two options with the retransmission timeout period T_{TO} set as 1.5 s (up to one round of retransmission) and 3 s (no retransmission) respectively.

Position Estimate RMSE Performance

We run Monte Carlo simulations to test the position estimate root-mean-square-errors[8] (RMSEs) during the last 30 seconds of the coasting phase, and the results are shown in Fig. 3.9. Whereas the performance improvement of cooperative retrodiction over the non-cooperative counterpart has been shown in a numerical example in [54], the improvement is usually not as significant. We focus here on the case – often in practical implementation – where the sensors do not participate in cooperation. The estimation errors become noticeably higher with a higher message-level loss rate; but regardless of the loss rate, applying retransmission and retrodiction can effectively reduce the estimation errors. Given the communication parameters, according to Eq. (3.9), the total delivery probability of the original message at time cutoff T_{CO} is approximately $(1 - p_L^2)$ where p_L denotes the message-level loss rate. As such, the delivery rates for $p_L = 0.2$ and $p_L = 0.7$ at T_{CO} are approximately 96% and 50% respectively. The increased delivery rate effectively prevents the fusion center from using the predicted estimates – statistically worse than the original – thereby improving the estimation performance compared to the case without retransmission.

During this 30-second period[9], the sensors produce state estimates whose quality progressively improves over time – the favorable condition for applying retrodiction. Even with just one-step retrodiction, its effect on improving the accuracy can already be seen from the plots, where the RMSE is reduced generally by over 10%. Interestingly, retransmission

[8]The RMSEs are used more often in the tracking literature. We used MSEs earlier in Sec. 3.5.1 because the upper bounds were used to approximate fusion performance.

[9]The filters have been initialized prior to the "zero" time in the figures.

seems to play a more prominent role than retrodiction when the loss rate is high – note the two curves in the middle appear "flipped" when the loss changes from 0.2 to 0.7 – since the chance of applying retrodiction is rather small without retransmission under higher loss rates.

3.5.3 Tracking of Multiple Coasting Targets

In this subsection, we consider tracking of multiple ballistic targets coasting in close proximity to one another. The initial states for $M = 4$ different targets (in which Target 1 is the one we used in the previous subsection) are shown in Table 3.1. The trajectories are generated in the same manner as before.

Table 3.1: Initial target states

	Position	**Velocity**
Target	x, y, z (km)	$\dot{x}$, $\dot{y}$, $\dot{z}$ (km/s)
1	76.1, 3829.4, 5373.1	0.995, 3.54, -5.73
2	77.0, 3827.1, 5368.6	0.985, 3.48, -5.75
3	75.0, 3828.2, 5370.0	0.970, 3.52, -5.74
4	77.8, 3831.3, 5375.8	0.998, 3.51, -5.72

Association Properties

An important task prior to fusion is that every sensor estimate (track) must be associated with a certain target. Rather than focus on a particular type of data association algorithm, we explore the effect of available time on data association and on the final estimation performance by means of the probabilistic model [68] as follows.

The state estimate of Target i is assigned to Target j with probability a_{ij}:

$$a_{ij} = \begin{cases} 1-(M-1)p_A, & i=j \\ p_A, & i \neq j \end{cases}, \qquad i,j = 1,...,M, M \geq 2, \tag{3.58}$$

where M is the number of targets, and p_A is the probability that we assign a state estimate to any incorrect target[10]. We model p_A here to be the following function of the number of targets, M, and time τ:

$$p_A = \frac{1}{M(1+\beta\tau)}, \quad p_A \leq \frac{1}{M}, \tag{3.59}$$

where β is a scaling parameter. This function has the following properties: (a) as the number of targets M increases, the total probability of misassignment, $(M-1)p_A$, increases; (b) as τ increases, p_A decreases; (c) if $\tau = 0$, there is an equal probability of assigning a state estimate to any target; and (d) as $\tau \to \infty$, $p_A \to 0$. However, of note is that p_A is unlikely to be zero given an unlimited execution time for the correlator; more likely there may be some lower bound $p_{A,min}$ on the probability p_A, such that if $\tau \to \infty$, $p_A \to p_{A,min}$. This model is suited for the scenario in our study because the targets remain in close proximity and thus the distance between any two does not undergo dramatic change; as such, no target lies far apart from the rest from the view of a long-haul sensor. A fixed amount of time τ is allocated right before T_{CO} for computation, and the scaling factor β depends on the specific algorithms used. Here we set $\beta = 80$.

Position Estimate RMSE Performance

In Fig. 3.10, the position estimate RMSEs for Target 1 under different retransmission-retrodiction combinations are plotted for a message loss rate of $p_L = 0.7$. First, by comparing the results from each category with that in Fig. 3.9(b), we observe the estimation errors have increased due to mis-assigned tracks, although with the same allocated time τ, the improvement after retransmission and/or retrodiction is on the level as before. When the allocated time for computation is short, say $\tau = 0.05$, the degradation in error performance is much higher. As τ gradually goes up, the errors are seen to decrease, thanks to the improved

[10]For the case where $M = 1$, data association is not required since there is only one target that the state estimate can belong to (i.e., for $M = 1$, $p_A = 0$).

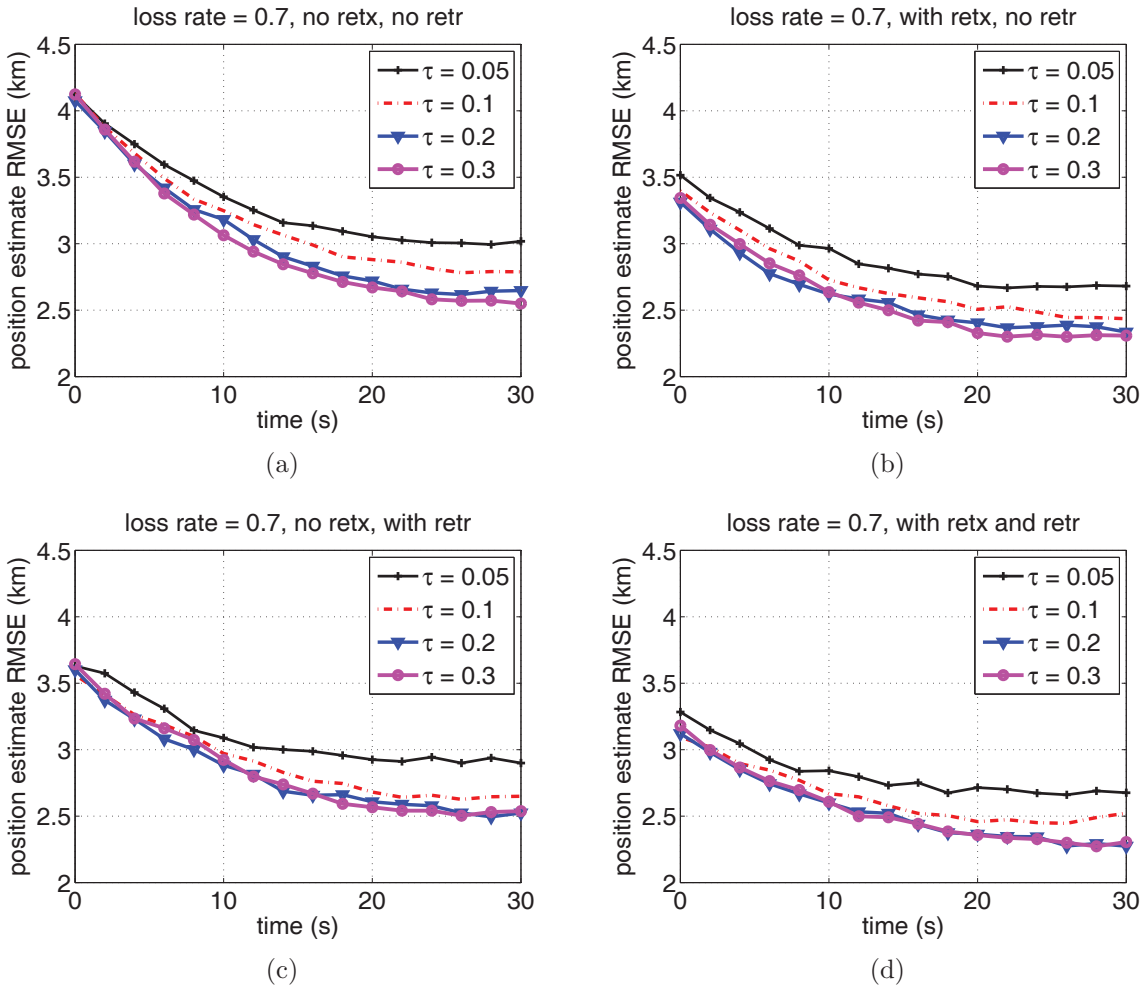


Figure 3.10: Multi-target position estimate RMSE over time with variable τ and the loss rate as 0.7: (a) no retransmission or retrodiction; (b) with retransmission, but no retrodiction; (c) with retrodiction, but no retransmission; (d) with both retransmission and retrodiction

performance of the correct association probabilities.

On the other hand, we observe that as τ is further increased, for example from 0.2 to 0.3, except in Case (a), where no retransmission or retrodiction is implemented, there is no clearly discernible performance improvement in accuracy. Actually, by this time, the probability for wrong associations has become very low. But as τ, the allocated time within T_{CO}, increases, the remaining time for communications is shortened. As such, when τ becomes higher, the slight improvement in computation can hardly offset reduced communication opportunities, which could either be the retransmitted messages (in (b)) or the next estimate itself used for retrodiction (in (c)) or both (in (d)). This demonstrates the trade-offs between computation

and communication when the total allocated time T_{CO} is fixed.

Discussions

In this subsection, we have assumed the communication parameters are given for a certain long-haul network. For a given deadline T_{CO}, however, if an existing schedule cannot meet the required or desired estimation accuracy levels, the fusion center can schedule more frequent retransmission and/or estimation intervals (at the cost of higher communication overhead for the sensors), to counteract the increasing demand for computation due to more sensors being deployed and/or more targets being monitored.

3.6 Conclusion

In this chapter, we have studied target tracking in a long-haul sensor network where communication loss and delay are severe enough to degrade the estimation accuracy performance significantly. Message-level retransmission is thus adopted to recover some of the missing sensor data. We have derived the improved delivery probabilities of estimate messages after retransmission and the distribution of the first arrivals among all retransmitted messages. We have also analyzed the probabilities of obtaining different types of estimates by the fusion center when retransmission and retrodiction techniques are applied. Simulation results for tracking of coasting ballistic targets – with the effect of track association and fusion being accounted for – demonstrate the effectiveness of our retransmission- and retrodiction-based mechanisms and the extent to which they can be applied so that the system requirements on estimation errors can be potentially met.

Chapter 4

Opportunistic Staggered Sensing Scheduling

4.1 Introduction

Sensing scheduling is a broad term that pertains to any action that can effect changes in the attributes or parameters during the sensing process [34]. Many of the existing studies on wireless sensor networks have considered energy and power savings as an essential task for prolonging the lifespan of the entire network. In these studies, myopic- and non-myopic sensor management schemes [42] have been proposed that rely on some predefined cost functions associated with certain sensing actions. However, these formulations do not lead to generally applicable solutions as they suffer from the "curse of dimensionality" due to the exponentially increasing computational complexity over time. An algorithm for scheduling and control of passive sensors is similarly proposed in [29] that aims to maximize an information measure in the sensor measurements. An optimization problem for estimation scheduling has been studied in [63] where a uniform scheduling technique is found to be optimal with steady-state sensors and an average error metric. Recently, [56] has proposed an optimization solution that strikes a balance between estimation accuracy and total sensor activations

over one period. None of these studies have accounted for varying tracking requirements and are generally not amenable to communication-constrained settings as all the sensor data must be present for the manager to make global decisions.

Our focus in this chapter, in contrast, is not on finding the optimal solutions (e.g., minimum errors) for every possible combination of network and sensor conditions in a communication- and computation-constrained environment; rather, of interest is how to opportunistically use these constraints to schedule sensing over time that can potentially improve the estimation and fusion performance. Different from conventional approaches, where the time instants to generate state estimates at the sensors and the fused state at the fusion center coincide, we consider *staggered* estimation and fusion where there exists a time shift – considered as the "staggered time" – between the two. The fusion center can take advantage of the time difference to perform *intra-state* prediction and retrodiction to improve the quality of the fused estimates at desired reporting time. Such scheduling can be carried out across multiple sensors as well. Correspondingly, the fusion center needs to fuse such multi-sensor state estimates with various time stamps. The main goal of this chapter is to investigate the effect of such asynchronous and staggered estimation/fusion on tracking for different types of target trajectories under variable communication loss/delay conditions and sensor bias levels.

The remainder of this chapter is organized as follows: Section 4.2 is an overview of the system model, ranging from the target and measurements, to generating the estimates at the sensors and fusing the estimates at the fusion center. Next in Section 4.3, the prediction and retrodiction performed at the fusion center is highlighted, which serves as the basis of staggered scheduling in subsequent sections. In Section 4.4, we introduce the basic ideas behind staggered estimation by the technique of intra-state prediction and retrodiction. In Section 4.5, we carry out a study using a simplified trajectory case that demonstrates the major advantages of such opportunistic staggered scheduling in improving estimation performance. Simulation results of a maneuver target tracking application are shown and

analyzed in Section 4.6. The chapter concludes in Section 4.7.

4.2 System Model

In this section we present the target and sensor measurement models as well as the estimation and fusion algorithms.

4.2.1 Target Model

The most popular motion models are the kinematic state models that are obtained by setting a certain derivative of the position to be equal to a zero-mean white noise [14]. The process noise in each model describes the motion uncertainty that complicates the estimation process. We consider a trajectory that consists of two basic types of motion: straight-line and turn movements. The straight line and turn components are described by the continuous white-noise acceleration (CWNA) and coordinated turn (CT) models respectively. Such a combination has been used in the literature, such as in [82].

In most scenarios, the motion along each coordinate (such as in the "east-north-up" coordinate system [14]) is typically assumed to be decoupled from the other coordinates. As such, the same model is used for each coordinate. In this study, for simplicity, we consider tracking occurs in horizontal dimensions (for both straight-line and turn segments). This is typically the case for motion over land or water; but even for aircraft motion, real trajectories suggest that the vertical motion can be easily decoupled from that in the horizontal plane [62].

Continuous White-Noise Acceleration (CWNA) Model

The discretized continuous white noise acceleration (CWNA) model is a simple, commonly used motion model in which an object moving in a generic coordinate ξ is assumed to be traveling at a near constant speed. The discrete-time state equation is given by $\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{w}_k$, where (dropping the time index k), $\mathbf{x} = [\xi \quad \dot{\xi}]^T$ here is a two-dimensional vector

representing the position and velocity, and $\mathbf{F}$ is known as the transition matrix and is given by $\mathbf{F} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$, where T is the scan rate of the sensor[1] (i.e., sampling period). The covariance of the discrete-time process noise $\mathbf{w}_k$ is $\mathbf{Q} = \tilde{q} \begin{bmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{bmatrix}$, where $\tilde{q}$ (often assumed to be constant over time) is the power spectral density (PSD) of the underlying continuous-time white stochastic process.

Since our focus is on 2-D tracking, we extend the above model to two coordinates ξ and η. The evolution of the state vector $\mathbf{x} = [\xi \quad \dot{\xi} \quad \eta \quad \dot{\eta}]^T$ is described by

$$\mathbf{x}_{k+1} = \begin{bmatrix} \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \end{bmatrix} \mathbf{x}_k + \mathbf{w}_k, \tag{4.1}$$

and the covariance matrix $\mathbf{Q}_k$ of the process noise $\mathbf{w}_k$ is

$$\mathbf{Q}_k = \begin{bmatrix} \tilde{q}_\xi \begin{bmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{bmatrix} & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \tilde{q}_\eta \begin{bmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{bmatrix} \end{bmatrix}. \tag{4.2}$$

Target Maneuver

The second type of motion occurs when the target performs a maneuver (i.e., a turn). A turn usually follows a pattern known as a *coordinated turn*, which is characterized by a constant turn rate and a constant speed. The turn rate Ω is incorporated into the motion model by augmenting the state vector for a horizontal motion model: $\mathbf{x} = \begin{bmatrix} \xi & \dot{\xi} & \eta & \dot{\eta} & \Omega \end{bmatrix}^T$, which

[1]Note that a superscript T always denotes the transpose of a vector or matrix.

gives rise to the discretized coordinated turn (CT) model [14], given by

$$\mathbf{x}_{k+1} = \begin{bmatrix} 1 & \frac{\sin\Omega(k)T}{\Omega(k)} & 0 & -\frac{1-\cos\Omega(k)T}{\Omega(k)} & 0 \\ 0 & \cos\Omega(k)T & 0 & -\sin\Omega(k)T & 0 \\ 0 & \frac{1-\cos\Omega(k)T}{\Omega(k)} & 1 & \frac{\sin\Omega(k)T}{\Omega(k)} & 0 \\ 0 & \sin\Omega(k)T & 0 & \cos\Omega(k)T & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k + \mathbf{w}_k, \tag{4.3}$$

and the covariance matrix of the process noise $\mathbf{w}_k$ is

$$\mathbf{Q}_k = \begin{bmatrix} \tilde{q}_\xi \begin{bmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{bmatrix} & \mathbf{0}_{2\times 2} & \begin{matrix} 0 \\ 0 \end{matrix} \\ \mathbf{0}_{2\times 2} & \tilde{q}_\eta \begin{bmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{bmatrix} & \begin{matrix} 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 \end{matrix} & \begin{matrix} 0 & 0 \end{matrix} & \tilde{q}_\Omega T \end{bmatrix}. \tag{4.4}$$

In contrast to the CWNA model, the CT model is a nonlinear one if the turn rate Ω is not a known constant. In practice, the linear acceleration noise PSD levels in both dimensions are assumed to be equal; i.e., $\tilde{q}_\xi = \tilde{q}_\eta$. The general guidelines for selecting an appropriates levels of these noise parameters can be found in [14].

4.2.2 Sensor Measurement Model

A sensor collects measurements of the target range and azimuth angle according to the following measurement model [82]:

$$\mathbf{z} = \begin{bmatrix} r \\ a \end{bmatrix} = \begin{bmatrix} \sqrt{x^2+y^2} \\ \tan^{-1}\left(\frac{y}{x}\right) \end{bmatrix} + \begin{bmatrix} w_r \\ w_a \end{bmatrix} \tag{4.5}$$

where w_r and w_a are independent zero mean Gaussian noises with standard deviations σ_r and σ_a, respectively. Note that this measurement has been normalized to the sensor's own known location.

4.2.3 Generating the State Estimates

Conversion of Measurements from Polar to Cartesian Coordinate

In practice the measurements are often reported in polar coordinates as in Eq. (4.5) with respect to the sensor location. Nevertheless, common motion models are given in Cartesian coordinates as shown earlier. Therefore, it is necessary that a sensor first converts the polar measurements to Cartesian ones before generating its state estimates. The standard conversion formulas are

$$z_\xi = r\cos a \qquad z_\eta = r\sin a, \tag{4.6}$$

where z_ξ and z_η denote the transformed measurement components along ξ and η axes respectively. However, it is noted that this conversion only applies in restricted conditions as it can introduce a bias after the conversion. A more general *unbiased* conversion rule is given by applying a correction factor as follows [59]:

$$z_\xi^u = e^{\sigma_a^2/2} r\cos a \qquad z_\eta^u = e^{\sigma_a^2/2} r\sin a, \tag{4.7}$$

where σ_a is the standard deviation of the polar azimuth measurement.

Filtering

Once a measurement in Cartesian coordinates is obtained, the sensors can then generate its state estimate using its prior knowledge about the target and this newest measurement. Conceptually, the goal of a state estimator is to extract the state information $\mathbf{x}$ from measure-

ment $\mathbf{z}$ that is corrupted by noise; this is done by running a filter that sequentially outputs the state estimate $\hat{\mathbf{x}}$ and its associated error covariance matrix $\mathbf{P}$. In most recursive state estimators, each step is comprised of two distinct phases, namely, "predict" and "update". In the "predict" phase, the state estimate from the previous time step is used to produce a rough estimate of the current state, which is also known as the a priori state estimate. In the subsequent "update" phase, this predicted value is combined with the latest measurement to refine the state estimate, resulting in the a posteriori estimate that is supposed to possess an improved quality in terms of a reduced estimation error. Normally, these two phases alternate and the system state at a certain time instant is predicted directly from the preceding posterior estimate. Also computed during each step are the prior and posterior error covariance matrices $\mathbf{P}$. Defined as $\mathbf{P}_k = \mathbb{E}[(\hat{\mathbf{x}}_k - \mathbf{x}_k)(\hat{\mathbf{x}}_k - \mathbf{x}_k)^T]$ but can be recursively computed without knowledge of the actual target state, $\mathbf{P}$ provides a theoretical accuracy measure of the estimator.

Kalman Filters and Extended Kalman Filters

The Kalman filters (KFs) are arguably the most well-known state estimators. They are linear minimum-mean-square-error (LMMSE)-optimal as the trace of $\mathbf{P}$ – characterizing the estimation error – at each step is minimized. The equations and explanations have been provided in Chapter 2. Of note is that KFs are the first options with linear system and measurement dynamics, i.e., where both system and measurement models are well-defined linear functions. To recall the motion models introduced above, we have the CWNA model as a good candidate for KFs as the transition matrix is well-defined and stable over time. Although the polar measurement equation is nonlinear as in Eq. (4.5), a sensor can regard the transformed Cartesian measurement in Eq. (4.6) or (4.7) as the direct measurement, which corresponds to a linear mapping from states to measurements.

On the other hand, when the state transition matrix contains elements of the current state, as in the CT model, the system dynamics are nonlinear. As such, extended Kalman

filters (EKFs) can be used to approximate the nonlinearity. To elaborate, when the state transition and measurement models are not linear functions of the state but differentiable functions

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}) + \mathbf{w}_k \tag{4.8}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k, \tag{4.9}$$

where $\mathbf{w}$ and $\mathbf{v}$ are again the process and measurement noises, the functions f and h can be used to compute the predicted state from the previous estimate and measurement respectively, but they cannot be applied to the covariances directly. Instead, their Jacobians are used to compute the state transition and measurement matrices[2]:

$$\mathbf{F}_{k-1} = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1|k-1}} \tag{4.10}$$

$$\mathbf{H}_k = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k|k-1}}. \tag{4.11}$$

At each time step, the Jacobians are evaluated with current states. These matrices can then be used in the Kalman filter equations. This process essentially linearizes the non-linear function around the current estimate.

Interacting Multiple Models

In many practical scenarios, the system characteristics can change over time. As an example, a fighter jet, which normally proceeds with stable flight dynamics, might commence rapid maneuvers when approached by a hostile missile. Such varying system characteristics are not easily described by just one model, thereby calling for a multiple-model approach. A multiple model (MM) provides a versatile tool for adapting the state estimation process in dynamic systems where a target can undergo different types of motion at different times. In particular,

[2]These are used for the first-order EKF. Higher order EKFs may be obtained by retaining more terms of the Taylor series expansions.

the interacting multiple-model (IMM) estimator is considered one of the most cost-effective dynamic MM algorithms. In IMM, at any time, the system state is assumed to be in a number of possible modes that are described by their probabilities. The structure of the system and/or the statistics of the noises can be different from mode to mode. The transition probabilities between modes from one estimation epoch to the next are assumed to follow a Markov chain. For each mode, the underlying filtering process is performed as described earlier, with the addition of evaluating the probabilities of different modes and interacting and mixing all the modes to generate an overall state estimate and error covariance. The equations and a detailed discussion of design parameters and implementation issues can be found in [14].

In our two-sensor settings, since KF and EKF can be used for CWNA and CT models respectively as described above, we consider Sensor 1 uses an IMM estimator that contains two modes, namely, KF and EKF, with appropriately selected noise levels. On the other hand, in [14], an argument has been made that during maneuvers, if the number of samples are relatively small (say, less than 10), a stand-alone KF with high process noise levels can be used for the maneuver. As such, we consider a heterogenous sensor setting where Sensor 2 uses a KF throughout its filtering process; in other words, it doesn't distinguish different types of motion. For both sensors, state estimates $\hat{\mathbf{x}}$ and the associated error covariances $\mathbf{P}$ are generated periodically and sent out, in a timely fashion, to the remote fusion center. Besides, when the fusion center combines the two estimates together according to the algorithms to be presented below, the component estimates should be of the same dimension; hence, at the time of fusion, the fusion center can simply drop the turn rate component from Sensor 1.

4.2.4 Fusers

It is a well known fact that the common process noise in measuring the motion of any target results in correlation among estimates generated by multiple sensors. The error cross-

covariance is the term that describes this spatial correlation. However, it is in general a very challenging task to derive the exact cross-covariance terms in practice. We consider two types of fusers where the fused estimate can be obtained directly in closed forms with no cross-covariance calculation needed.

Track-to-Track Fuser without Cross-Covariance

In tracking applications, the track-to-track fuser (T2TF) is a linear fuser that is optimal in the linear minimum mean-square error (LMMSE) sense. In general, the fused state estimate $\hat{\mathbf{x}}_F$ and its error covariance $\mathbf{P}_F$ are defined for two sensors [14] as:

$$\hat{\mathbf{x}}_F = \hat{\mathbf{x}}_1 + (\mathbf{P}_1 - \mathbf{P}_{12})(\mathbf{P}_1 + \mathbf{P}_2 - \mathbf{P}_{12} - \mathbf{P}_{21})^{-1}(\hat{\mathbf{x}}_2 - \hat{\mathbf{x}}_1) \tag{4.12}$$

$$\mathbf{P}_F = \mathbf{P}_1 - (\mathbf{P}_1 - \mathbf{P}_{12})(\mathbf{P}_1 + \mathbf{P}_2 - \mathbf{P}_{12} - \mathbf{P}_{21})^{-1}(\mathbf{P}_1 - \mathbf{P}_{21}) \tag{4.13}$$

where $\hat{\mathbf{x}}_i$ and $\mathbf{P}_i$ are the state estimate and error covariance from sensor i, respectively, and $\mathbf{P}_{ij} = \mathbf{P}_{ji}^T$ is the error cross-covariance between sensors i and j. However, when the sensor errors are correlated and the cross-covariance is unavailable, one may assume that the cross-covariance is zero in order to apply this linear fuser, even though the result will be suboptimal. The fuser would then be reduced to a simple convex combination of the state estimates:

$$\mathbf{P}_F = (\mathbf{P}_1^{-1} + \mathbf{P}_2^{-1})^{-1} \tag{4.14}$$

$$\hat{\mathbf{x}}_F = \mathbf{P}_F(\mathbf{P}_1^{-1}\hat{\mathbf{x}}_1 + \mathbf{P}_2^{-1}\hat{\mathbf{x}}_2) \tag{4.15}$$

Fast Covariance Intersection (CI) Algorithm

Another sensor fusion method is the covariance intersection (CI) algorithm. The intuition behind this approach comes from a geometric interpretation of the problem. If one were to plot the covariance ellipses for $\mathbf{P}_F$ (defined as the locus of points $\{\mathbf{y} : \mathbf{y}^T\mathbf{P}_F^{-1}\mathbf{y} = c\}$ where

c is some constant), the ellipses of $\mathbf{P}_F$ are found to always lie within the intersection of the ellipses for $\mathbf{P}_1$ and $\mathbf{P}_2$ for all possible choices of $\mathbf{P}_{12}$ [36]. The intersection is characterized by the convex combination of sensor covariances:

$$\mathbf{P}_F = (\omega_1 \mathbf{P}_1^{-1} + \omega_2 \mathbf{P}_2^{-1})^{-1} \tag{4.16}$$

$$\hat{\mathbf{x}}_F = \mathbf{P}_F \left(\omega_1 \mathbf{P}_1^{-1} \hat{\mathbf{x}}_1 + \omega_2 \mathbf{P}_2^{-1} \hat{\mathbf{x}}_2\right), \quad \omega_1 + \omega_2 = 1 \tag{4.17}$$

where $\omega_1, \omega_2 > 0$ are weights to be determined (e.g., by minimizing the determinant of $\mathbf{P}_F$).

[79] proposed a fast CI algorithm where the weights are found based on an information-theoretic criterion so that ω_1 and ω_2 can be solved for analytically as follows:

$$\omega_1 = \frac{D(p_1, p_2)}{D(p_1, p_2) + D(p_2, p_1)} \tag{4.18}$$

where $D(p_A, p_B)$ is the Kullback-Leibler (KL) divergence from $p_A(\cdot)$ to $p_B(\cdot)$, and $\omega_2 = 1 - \omega_1$. When the underlying estimates are Gaussian, the KL divergence can be computed as:

$$D(p_i, p_j) = \frac{1}{2}\left[\ln \frac{|\mathbf{P}_j|}{|\mathbf{P}_i|} + \mathbf{d}_X^T \mathbf{P}_j^{-1} \mathbf{d}_X + Tr(\mathbf{P}_i \mathbf{P}_j^{-1}) - k\right] \tag{4.19}$$

where $\mathbf{d}_X = \hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j$, k is the dimensionality of $\hat{\mathbf{x}}_i$, and $|\cdot|$ denotes the determinant. This fast-CI algorithm will be used for a quantitative comparison against the above T2TF with unavailable cross-covariances.

4.2.5 Target Trajectory

The initial state of the target in Cartesian coordinates (with the position in meters and velocity in m/s) is set to be [82] $\mathbf{x}(0) = [x(0) \quad \dot{x}(0) \quad y(0) \quad \dot{y}(0)]^T = [0 \quad 0 \quad 20000 \quad 250]^T$. At $t = 60$ s, the test target starts to take a left turn at a turn rate of 2°/s for 30 s, and then continues straight until $t = 150$ s. The sampling rate of the sensors is once every two seconds, i.e., $T = 2$ s.

4.3 Prediction and Retrodiction by the Fusion Center

4.3.1 Prediction by the Fusion Center

We consider prediction performed not by a sensor during its regular recursive filtering, but by the fusion center. The purpose is largely different even though the two may share the same "prediction" equations. Since the fusion center does not have access to measurements, it needs the sensors to communicate their processed state estimates for subsequent fusion. However, due to severe loss and delay, the desired state estimates are not always available; in this case, the fusion center may simply *interpolate* the unavailable estimates by plugging in its own predicted estimates from earlier ones, using known or learned state evolution models. Hence, the prediction by the fusion center is used to counteract the effect of communication constraints. Due to the system uncertainty characterized by process noise, prediction alone often results in higher estimation errors compared to the estimates generated and sent by the sensors (this is the very reason measurements have to be taken regularly at the sensors in order to maintain desired tracking performance). Nevertheless, to achieve the fusion gain, at the fusion center, often it is still preferable to use predicted estimates for a sensor rather than discard the sensor's potential information altogether.

4.3.2 Retrodiction by the Fusion Center

To recall, estimation of a target state at a particular time based on data collected beyond that time is called retrodiction or smoothing. Retrodiction improves the accuracy of the estimates, thanks to the use of more information, at the cost of extra delay. The vast majority of the existing literature studies have considered retrodiction only from the perspective of an individual sensor; the effect of retrodiction in the context of state fusion has been largely unexplored. Since retrodiction calls for the availability of subsequent data to the ones of interest, the inherent link delay over a long-haul network entails that the fusion center can exploit the opportunities for potential retrodiction to improve the accuracy of the fused

estimate. Conventionally, an estimate is retrodicted only when it actually arrives as in many OOSM-related studies [58,83,84]. However, in our design, the fusion center opportunistically interpolates the missing estimates – that is, to "fill in the blanks" – from prior estimates using prediction, and subsequent ones with retrodiction. Of course, an available estimate can be retrodicted using its following estimates too – as in the conventional use of retrodiction – as long as the associated fused estimate has not been finalized by the fusion center. This has the potential benefit to speed up the process of finalizing the global estimates – since the fusion center does not have to wait for the actual missing estimates to finally arrive – and hence can reduce the chance of missing the reporting deadline.

4.3.3 Selection of Prediction and Retrodiction Algorithms with Multiple Models

In the context of possibly heterogenous and multiple system models utilized by individual sensors, the fusion center faces the question of which model(s) to use when deriving its predicted/retrodicted values. As a matter of fact, a sensor is not likely to send every filtering parameter update to the fusion center (considering the complexity of IMM estimators); therefore, the fusion center can only use its best guess, or out of practicality, use simplified models to interpolate the missing values. Although data thereby generated may not match the quality level of that of the sensor originals, they might still provide enough information to drive up the fusion gain.

We consider the fusion center uses a linear form of prediction and retrodiction (i.e., KF-based) for simplicity, both of which can be easily computed with closed-form expressions. In particular, the fixed-interval Rauch-Tung-Striebel (RTS) retrodiction algorithm is adopted that can help fill in missing data as well as update existing ones using subsequent arrivals.

4.4 Staggered Estimation: An Overview

In this chapter, we consider *staggered estimation* that exploits the temporal relationship between adjacent estimates in order to improve the estimation performance. To formulate the estimation and fusion process, we consider that a stream of globally fused estimates are reported at a regular time interval of T, which also coincides with the estimation interval at the sensors as well. Suppose the (continuous) time of interest is nT, where n is a positive integer. Given the stationarity of the above interval T, in subsequent analysis, the time instant will also be conveniently referred to as the time (step) n. Based on the estimates sent by the sensors, the fusion center can perform prediction and retrodiction – if necessary – to form component state estimates for fusion. For a given sensor, depending on what estimates sent from this sensor have been received, the fusion center may report one of the following types of estimates, corresponding to time n:

a. $\hat{\mathbf{x}}_{n|n}$, the "default" estimate sent from the sensor;

b. $\hat{\mathbf{x}}_{n^-}$, the predicted estimate;

c. $\hat{\mathbf{x}}_{n^-|n+1}$, the predicted & retrodicted estimate; and

d. $\hat{\mathbf{x}}_{n^+|n+1}$, the retrodicted estimate.

Note that similar notations were first introduced in Section 3.4 along with their explanations. The conventional prediction and retrodiction techniques discussed therein are schematically shown in Fig. 4.1(a), where the estimation interval T and its integer multiples serve as the basic time units for prediction and retrodiction. We conveniently name this as *inter-state* prediction and retrodiction.

Instead of forming the reports ideally at the same time instants as those when the sensors generate the estimates, we consider staggered scheduling shown in Fig. 4.1(b). With this method, a sensor is scheduled to generate its estimates following the same estimation interval T but at time instants different from the ones at which the fusion center creates reports. As

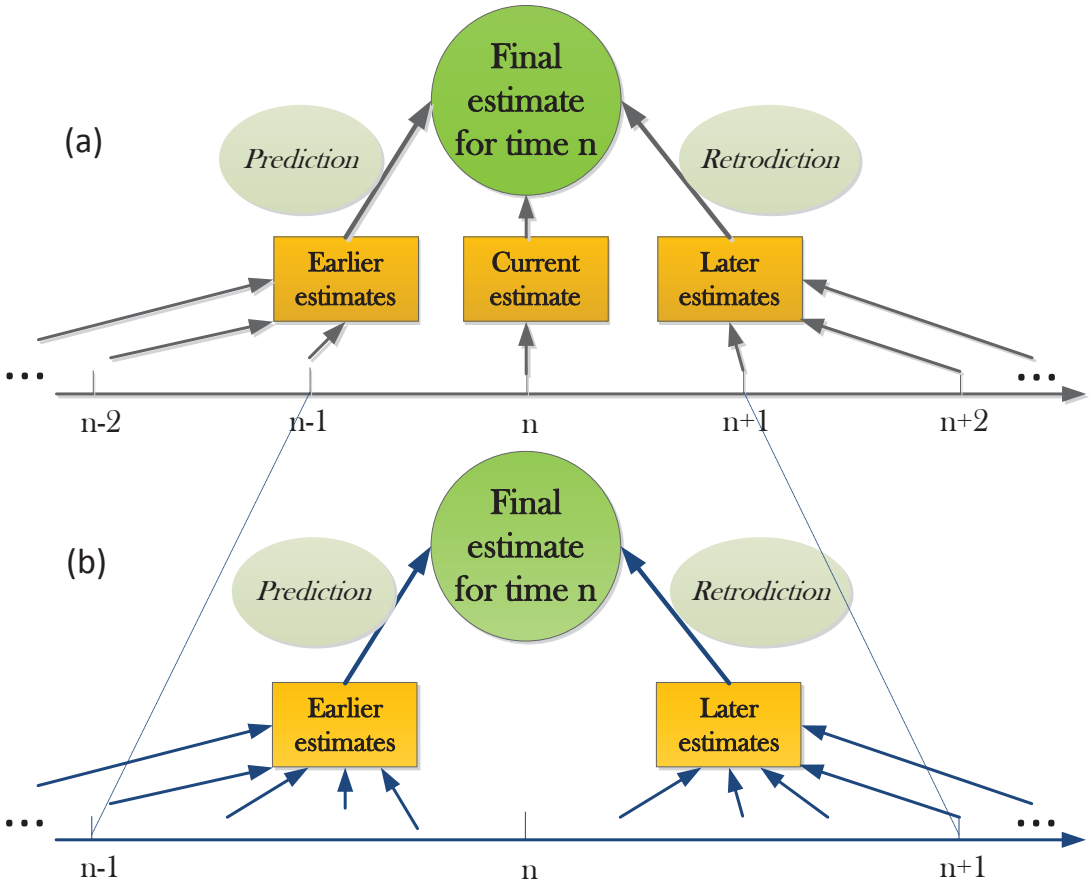


Figure 4.1: Prediction and retrodiction: The estimate at time n is to be obtained. (a) Conventionally, prediction and retrodiction occur over integer multiples of the estimation interval; (b) in staggered scheduling, prediction and retrodiction can be carried out over a fraction of the estimation interval.

a result, we allow both the prediction and retrodiction to be performed over a period of time that is a fraction of the estimation interval T. Hence, we have the *intra-state* prediction and retrodiction[3].

In Fig. 4.2, an example consisting of three different estimation schedules is shown. In the figure, the red dotted lines denote the common time instants of interest for the fusion center (i.e., the time instants whose state estimates are to be finalized and reported) and the green bars indicate the times where the estimates are generated. In (a), the standard

[3]Of course, the extension of this intra-state filtering can be realized by superimposing an estimation interval (and its integer multiples) on top of the fractional period τ.

estimation schedule is shown, where the estimation time at the sensors and fusion time at the fusion center always coincide[4]. In (b), an estimate with time-stamp $(n-0.2)T$ is sent out by the sensor. Upon initial reception, the fusion center can perform a 0.2-step prediction to form the estimate report for time instant nT; next, if the subsequent estimate from the same sensor – now with time-stamp $(n+0.8)T$ – arrives before the reporting deadline (which is assumed to be one estimation interval T here), the fusion center can further perform a 0.8-step retrodiction for an improvement in accuracy over the previous predicted estimate. On the other hand, the estimation time in (c) always lags its preceding fusion time by $0.2T$, resulting in a 0.8-step prediction and a 0.2-step retrodiction when both estimates are available. In the figure, τ (tau) values are shown in the figure as the gap – normalized to the estimation interval T – between the fusion time and the time stamp of the latest generated estimate. In all, when a sensor does not directly report its estimates for the time instants of interest but expects the fusion center to generate the corresponding estimates on its own for further fusion, we consider the scheduling as both "asynchronous" – from the perspective of the fusion center – and "staggered".

Albeit conceptually simple, the effects of this staggered scheduling on estimation and fusion performance are not readily predictable. Next we investigate its potential benefits for both one-sensor and two-sensor scenarios under variable communication loss and delay conditions during the straight-line motion.

4.5 Staggered Estimation and Fusion

In this section, we carry out a study of the staggered estimation and fusion under a simplified single-model motion scenario. To achieve this, we relax the target and sensor measurement settings as follows. We consider only the straight-line segment – as described by the CWNA model – in one generic coordinate ξ with the estimation interval set to be $T = 1$ s and the

[4]To avoid confusion, the term "fusion time" refers to the time of interest whose state estimate needs to be generated by the fusion center; whereas "reporting time" refers to the time when final fusion occurs.

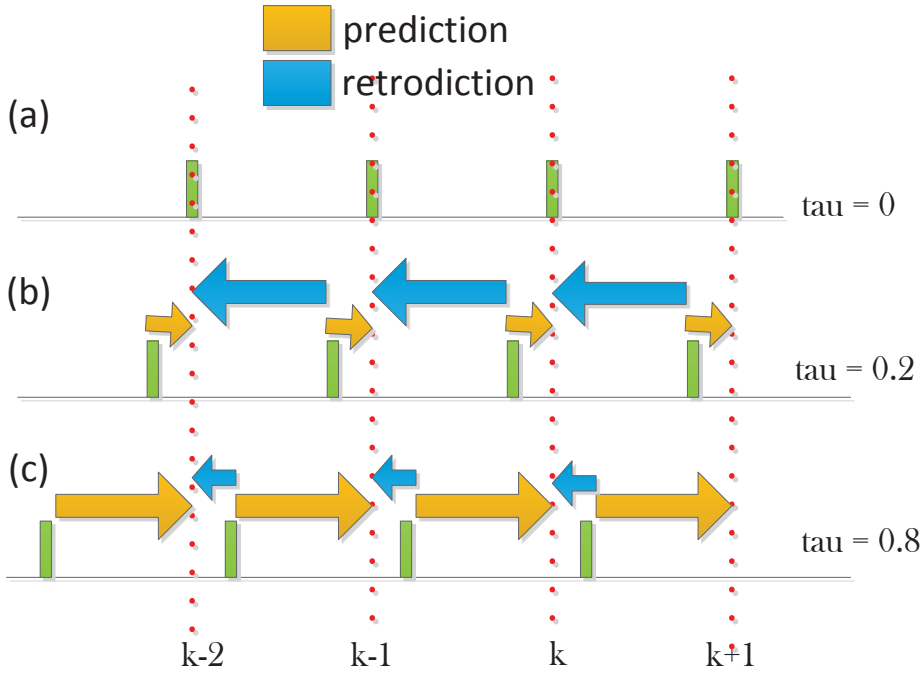


Figure 4.2: Staggered estimation scheduling: (a) the standard schedule; (b) staggered estimation, where the sensor takes measurements $0.2T$ earlier than the subsequent fusion time; (c) staggered estimation, where the sensor takes measurements $0.2T$ later than the preceding fusion time. The τ values are shown as the gap between the fusion time and the time stamp of the latest generated estimate.

noise PSD $\tilde{q}_\xi = 1$ m^2/s^3. The sensor measurement model has been simplified in which the sensor directly measures the Cartesian position of the target and hence the measurement $z_k = \mathbf{H}\mathbf{x}_k + v_k$ is available, where $\mathbf{H} = [1 \quad 0]$ is the measurement matrix, and the Gaussian measurement noise v has autocorrelation $\mathbb{E}[v_k v_j] = R\delta_{kj} \triangleq \sigma_v^2 \delta_{kj}$, where $\delta_{(\cdot)}$ is the Kronecker delta function. Both sensors have a $\sigma_v = 20$ m. Doing such a case study would allow us to focus on the effect of staggered scheduling on the estimation and fusion performance in a simplified setting, without considering the impact of such factors as target model uncertainty and sensor heterogeneity, among others.

4.5.1 Estimation Performance with One Sensor

We first explore the effect of staggered scheduling on one-sensor estimation. Had the communication between the sensor and the fusion center been perfect with the standard synchronous scheduling being used, the fusion center would simply "take over" the sensor state estimates. With communication loss and delay, however, the fusion center may have a different view of the state evolution from that of the sensor due to the use of prediction and retrodiction. The link loss and delay profiles have been introduced in Chapter 3. To recap, each message is lost with probability p that is independent of other messages. And the arrival delay of each message is modeled by a shifted exponential random variable with fixed initial delay T_I and mean random delay μ beyond T_I. We analyze the probabilities of generating different types of estimates by a certain deadline and show the impact of scheduling on estimation performance.

Probabilities for Obtaining Different Types of Estimates

We first consider the specific condition under which a certain number of retrodiction rounds can potentially take place. Suppose the interval between the time of interest and the preceding sensor estimation time is τ, where $0 \leq \tau < T$; in other words, the time stamp of the preceding sensor estimate is $nT - \tau$. Suppose the reporting deadline for time nT is $nT + D$ (i.e., with a maximum lag D); then in order to possibly perform at least one round of retrodiction, an estimate must be generated after time nT and arrive at the fusion center by $nT + D$. Since the time stamp of the estimate following time nT is $(n+1)T - \tau$, accounting for the fixed initial delay T_I, the earliest arrival time $(n+1)T - \tau + T_I$ should be no later than the deadline $nT + D$; on the other hand, to have only up to one round of retrodiction, the estimate generated at time $(n+2)T - \tau$ should arrive later than $nT + D$. Combining both constraints, we have the condition for both the reporting lag D and the scheduling lag τ with up to one round of retrodiction. In fact, this result can be easily extended to multi-round retrodiction: To have up to l ($l \geq 1$) rounds of retrodiction, the reporting lag

D should satisfy the following condition:

$$lT + T_I - \tau \leq D < (l+1)T + T_I - \tau. \tag{4.20}$$

Without loss of generality, in the following analysis, T and D are given as 1 s and 1.5 s respectively, with the common link communication and pre-processing delay T_I set as 0.5 s (and $\mu = 0.3$ s). This is the situation where in the standard scheduling scheme, the deadline for reporting one estimate happens to be the very earliest time the subsequent estimate arrives, namely, $D = T + T_I$. Also, it is easy to verify that $l = 1$.

Given the link statistics introduced earlier, the probability that a sensor estimate is successfully received by the fusion center within time t since being generated is $(1-p)F(t)$. It is easy to verify that the amounts of time it takes for the two estimates, one immediately preceding nT and the other following it, to be delivered to the fusion center before the deadline, are $D+\tau$ and $D-T+\tau$, respectively. As such, we have the following probabilities of using different types of estimates by the deadline:

a. $\hat{\mathbf{x}}_{n|n-\tau}$: $(1-p)F(D+\tau)(1-(1-p)F(D-T+\tau))$;

b. $\hat{\mathbf{x}}_{n^-}$: $(1-(1-p)F(D+\tau))(1-(1-p)F(D-T+\tau))$;

c. $\hat{\mathbf{x}}_{n^-|n+1-\tau}$: $(1-(1-p)F(D+\tau))(1-p)F(D-T+\tau)$; and

d. $\hat{\mathbf{x}}_{n^+|n+1-\tau}$: $(1-p)^2F(D+\tau)F(D-T+\tau)$.

Similar notations for these types of estimates were first introduced in Sec. 4.4, now with the exception that the staggered interval τ is added to the subscripts to reflect the time difference. Note that when $\tau = 0$, the results are simply reduced to those under standard scheduling. In the cases b) and c), the minus signs denote that the estimate generated at $n-\tau$ is not available at the fusion center; as such, these probabilities have also incorporated the scenarios where prediction over a longer time span by the fusion center has taken place.

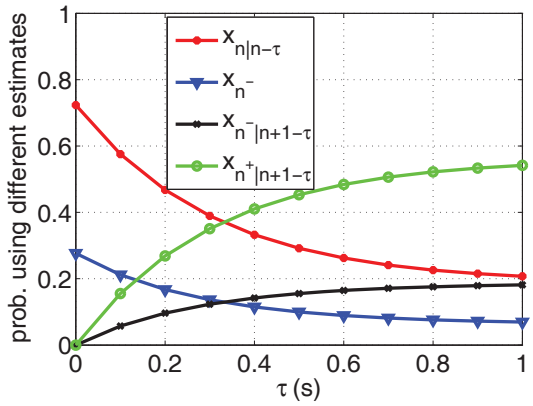


Figure 4.3: Probabilities of using different types of estimates at the deadline with variable staggered estimation intervals τ, where $p = 0.25$, $T = 1$, and $D = 1.5$ s.

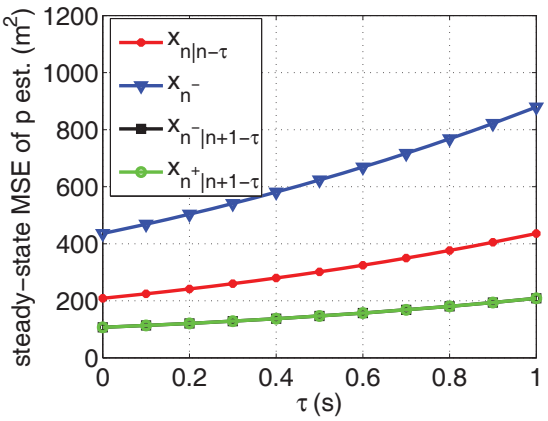


Figure 4.4: Steady-state position estimate mean-square-error (MSE) performance with variable staggered interval τ values

Given a pre-determined set of estimation interval T and deadline D values, the question of interest arises: how would different τ values impact the estimation performance at the fusion center? In Fig. 4.3, the probabilities of eventually using different types of estimates by the fusion center are plotted for a loss rate of $p = 0.25$. As can be easily seen in the figure, as the staggered interval τ moves from 0 all the way up to T, the probability of obtaining $\hat{\mathbf{x}}_{n^+|n+1-\tau}$ goes up while that of using the predicted state $\hat{\mathbf{x}}_{n^-}$ decreases. Among the four, these two represent the best and worst estimates respectively in terms of the estimation error. Increasing τ would then seem to improve the estimation performance when only these two types of estimates are considered. However, the other two types of estimate, $\hat{\mathbf{x}}_{n|n-\tau}$ and $\hat{\mathbf{x}}_{n^-|n+1-\tau}$, change in reverse directions too as τ shifts, and it is not immediately clear which of the two has overall better accuracy performance [54].

However, the above analysis does not capture the actual behavior of the estimate to be finalized by the fusion center, since intra-state prediction and/or retrodiction has to be applied when τ shifts away from zero, thereby affecting the behavior of all types of estimates. In what follows, we explore the error profiles with staggered scheduling under perfect communications. Then we will combine them with the above probabilistic analysis to derive the approximate estimation error performance.

4.5.2 Quantitative Results

Approximate Estimation Error

The fusion center applies the Rauch-Tung-Streibel (RTS) retrodiction algorithm [73] to obtain the retrodicted state estimates. With the previously established models, the steady-state behavior [73] of the sensor estimate can be found analytically via Ricatti equation recursion or more conveniently from simulations. In Fig. 4.4, the steady-state error performance of different types of estimates under variable τ values is displayed. Again, with our parameter setup, only the sensor estimates generated at $n+1-\tau$, if available, can be used by the fusion center for retrodiction. Another assumption used in generating the plots is that no bursty loss is present; that is, the number of prediction steps is constrained strictly under two. For example, in Cases b) and c) in the last subsection, the minus sign would mean that only the immediately preceding estimate is not received, but not the ones before.

From the plots, as τ gradually shifts away from 0, all types of estimates experience increasing steady-state estimation errors. Recall that under the steady-state condition, a sensor estimate has the same theoretical MSE guarantee regardless of its time of origin. Suppose two adjacent sensor estimates are successfully delivered to the fusion center (as in the case where $\hat{\mathbf{x}}_{n^+|n+1-\tau}$ can be obtained); as τ increases, the intra-state prediction step size is lengthened and retrodiction step size shortened, resulting in an increased estimation error. This relationship holds true for all other cases as well. Another interesting observation is that the two cases with $\hat{\mathbf{x}}_{n^+|n+1-\tau}$ and $\hat{\mathbf{x}}_{n^-|n+1-\tau}$ have nearly identical steady-state performance. This means that had the communications been perfect, the frequency that a sensor communicates its estimates (but with the same estimation frequency on tap) can be reduced by half without causing noticeable performance degradation.

Finally, we calculate the expected estimation MSE performance as the probabilistic combination of steady-state MSEs of different types of estimates. More specifically, the expected MSE with a certain τ choice is computed as the summation of the probabilities of obtain-

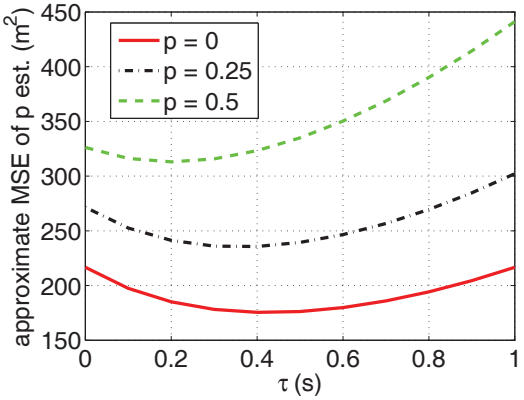


Figure 4.5: Approximate position estimate MSE performance with variable staggered interval τ and loss rate p values: only prediction across time of up to $2T$ is considered

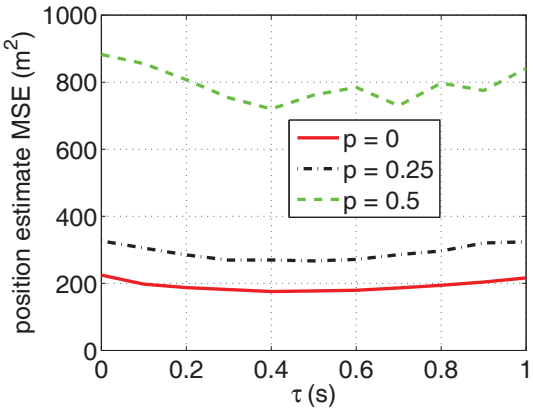


Figure 4.6: Actual position estimate MSE performance with variable staggered interval τ and loss rate p values

ing all four types of estimates, such as those shown in Fig. 4.3, times the corresponding steady-state position MSEs found in Fig. 4.4. This result is "approximate" at best in that the probabilities themselves may have included the cases where a string of losses occur. The results are plotted in Fig. 4.5 with three different link-level loss rates, namely 0%, 25%, and 50%. Interestingly, across all cases, the estimation errors decrease initially as τ shifts away from zero, and then increases. For validation of the results, however, we also need to test the *actual* estimation error performance via Monte-Carlo simulations.

Actual Estimation Error Performance

The same set of parameters are used to generate the actual position estimate MSE performance as shown in Fig. 4.6. Comparing it with Fig. 4.5, we can observe the following: First, the above approximation by probabilistic combination becomes increasingly erroneous as the loss rate increases. When there is no or little loss, the off-line probabilistic values serve as a good approximation of the actual error profile; however, as p increases, bursty losses become more commonplace, which was not reflected in the steady-state MSE values in Fig. 4.4, resulting in overly optimistic approximation when the loss becomes severe (as in the $p = 0.5$ case in the figure). Also the minimum estimation error is somewhat skewed in the

approximation. Nevertheless, a common time across cases where the minimum estimation errors can be found happens to be around $\tau = 0.4$ s. Here, at zero loss rate, the standard scheduling results in a nearly 30% higher estimation MSE compared to the value obtained at $\tau = 0.4$ s; even at a 25% loss rate, standard scheduling still yields 20% more errors compared to its staggered counterpart. As the loss becomes even higher, the improvement from staggered scheduling in terms of the percentage of error reduction becomes less prominent as the fusion center encounters more difficulties receiving estimates regardless of their time of origin. But overall, the error reduction performance showcases the major advantage of scheduling sensor estimation activity in a staggered manner.

4.5.3 Estimation and Fusion Performance with Two Sensors

Two sensors are assumed to have the same measurement noise profile. In this case, we need to consider all the combinations of different staggered intervals for both sensors – relative to the reporting time instants at the fusion center – denoted as τ_1 and τ_2, respectively. Probabilistic analysis similar to that in the last section can be carried out for both sensors. However, our focus here is to analyze the Monte Carlo simulation results as shown in Figs. 4.7 and 4.8, in which the position estimate MSEs for the linear T2TF and the fast-CI fuser are plotted respectively.

From the figures, all generated three-dimensional surfaces resemble a sheet with downward-curved center portions, the extension of the earlier one-sensor estimation performance. We observe that for nearly all cases, the fast-CI fuser outputs estimates that are of slightly worse quality than those generated by the simple linear fuser[5]. Also the fast-CI fuser is more sensitive to the changes in the loss rate. The increase in estimation MSE with a more lossy link is more dramatic in the CI fuser. Another common feature across the cases is that the standard scheduling $\tau_1 = \tau_2 = 0$ happens to result in the highest estimation error. Comparing the results here to those in the one-sensor case shown in Fig. 4.6, we can see

[5]Note that if $\mathbf{P}_1 = \mathbf{P}_2$, then $\omega_1 = \omega_2 = 0.5$, and the resulting fused estimate will be equivalent to that from Eq. (4.15) but with an inflated error covariance matrix (increased by a factor of 2).

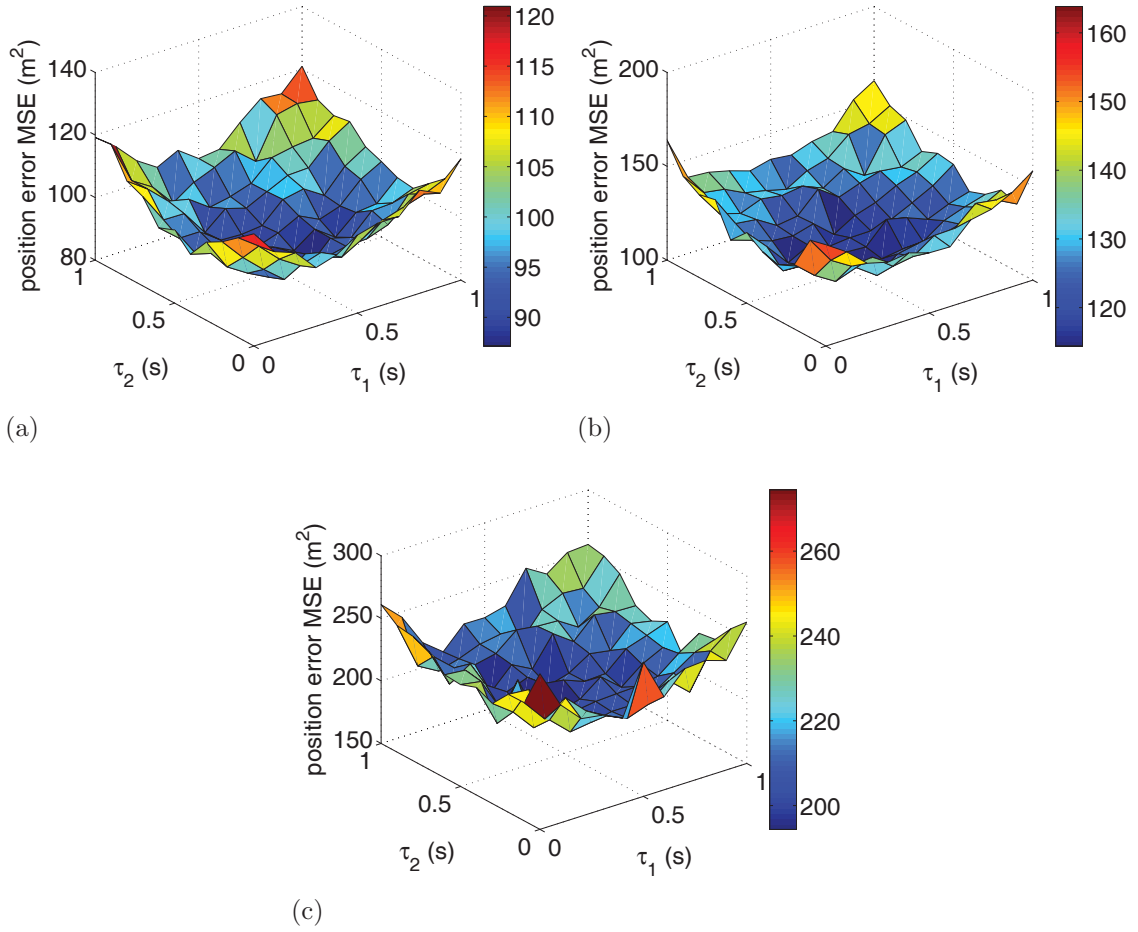


Figure 4.7: Actual position estimate MSE performance with two-sensor T2TF fusion and variable staggered interval τ and loss rate p values: (a) $p = 0$; (b) $p = 0.25$; (c) $p = 0.5$

that both fuser outputs have MSEs that are more than half of those values with one sensor only, reflecting the effect of the common process noise and cross-covariance.

Although not easily discernible in the figures here, a more "microscopic" examination of the numerical results reveals the effect of cross-covariance in staggered scheduling design. Individually, at $\tau = 0.4$ s, the fusion center can expect the least estimation error from either of the two sensors. However, the case where $\tau_1 = \tau_2 = 0.4$ s does not achieve the best fuser outputs; another point close by does. This observation can be construed as the reduction of cross-covariance by staggering the estimation time across sensors. If the two sensors take samples at the same time (even at optimal $\tau = 0.4$ s), the cross-covariance is the highest; as the time separation in between increases, so does the reduction of correlation. This reduction

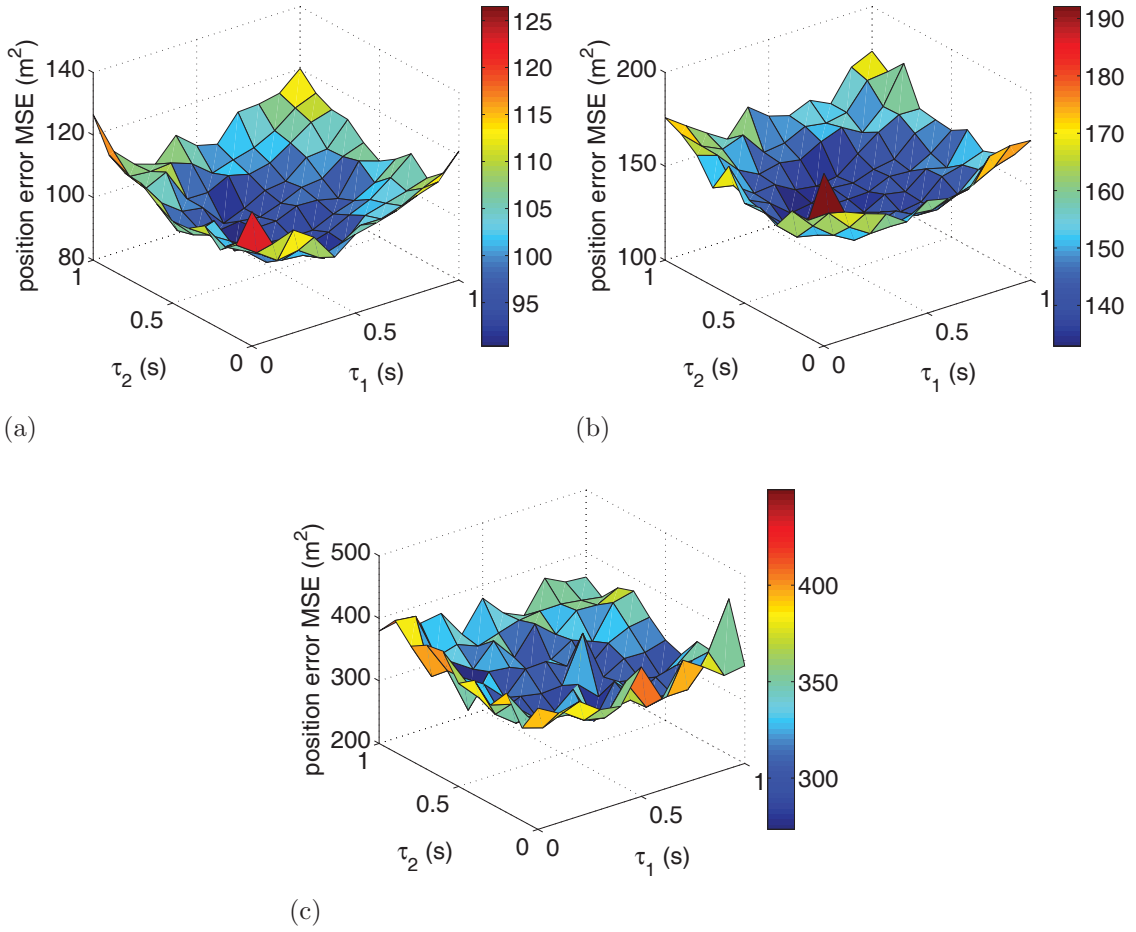


Figure 4.8: Actual position estimate MSE performance with two-sensor fast-CI fusion and variable staggered interval τ and loss rate p values: (a) $p = 0$; (b) $p = 0.25$; (c) $p = 0.5$

of cross-covariance over time was also observed in one numerical study [54] and is especially of interest for further investigation.

4.6 Performance Evaluation

In this section we evaluate the impact of variable staggered schedules on estimation and fusion performance in tracking a target undergoing a more complex set of motions as described in Section 4.2.

4.6.1 Sensor Behaviors

Whereas Sensor 1 uses an IMM estimator consisting of KF and EKF components whose parameters match the noise profiles of the CWNA and CT models respectively, Sensor 2 uses a KF with a much higher noise density level throughout to account for the uncertainty in the target trajectory. In Fig. 4.9(a), we observe the following behaviors of the individual sensors: The IMM estimator at Sensor 1 yields more accurate estimates – compared to the KF estimator at Sensor 2 – initially during the straight-line motion; but after the turn begins at $t = 60$ s, its error gradually increases and at around $t = 80$ s, shoots up rapidly till after the maneuver ends at $t = 90$ s. Afterward, the error decreases, less precipitately compared to the previous phase, and doesn't fall back to the pre-maneuver level until around $t = 140$ s. The relatively poor performance of the IMM estimator during and after maneuver in this case is due mainly to the selection of parameters for component modes. On the other hand, the error of the KF filter output at Sensor 2 remains stable throughout the process, higher than that of the IMM during the initial straight-line motion, but much lower as the other experiences inflated errors during and after the turn. Such sensor heterogeneity will be reflected in the effect of sensing scheduling on fusion as demonstrated below.

4.6.2 Staggered Sensing Scheduling without Sensor Bias

We now shift our focus to the fusion performance. Again in Fig. 4.9(a), the performance of the T2TF and CI fusers under conventional scheduling is shown for both $p = 0$ and $p = 0.5$ cases. Note the selection of fusers or different link profiles do not change the sensor estimation performance. As has been observed in the case study from the previous section, the CI fuser yields somewhat higher errors during the initial straight-line motion; however, during and shortly after the turn, the T2TF experiences higher errors than its CI counterpart. Also of note is that when loss is severe enough, say, when $p = 0.5$, for a short period of time after the turn is initiated, between $t = 80$ s and 90 s, both fusers yield even worse performance than the individual sensors. In this case, the CI fuser is able to pull back from its elevated

errors much faster than the T2TF, as can be observed by the somewhat narrower error peak around $t = 90$ s.

Next we consider staggered scheduling that spans different choices of τ_1 and τ_2 values. As there are more variables in this case, for better visualization, we use combinations of a few different values of both τ_1 and τ_2 and plot the performance under these settings in separate figures. In Fig. 4.9, starting from (a) the standard scheduling, where $\tau_1 = \tau_2 = 0$, τ_1 and τ_2 are incremented by 0.5 s from top to bottom, and left to right, respectively. The deadline is chosen to be $D = 2$ s. From the plots, while the improvements during the straight-line segments are not very obvious, we can easily observe that when τ_1 is 1 s, regardless of the τ_2 choices, the peak levels of both T2TF and CI drop significantly, especially for the $p = 0.5$ case. For example, the position RMSE is about 20% less when $\tau_1 = 1$ s compared to that following the default scheduling. This demonstrates by staggering the estimation times of the more error-prone Sensor 1, the fusion center can reduce the uncertainty in target motion by applying intra-state prediction and retrodiction. In contrast, different choices of τ_2 don't seem to affect the fusion performance as much. Besides, the improvement at the beginning of the maneuver, if any, is rather limited, since even using retrodiction doesn't help much during the phase of inflating errors at Sensor 1.

Next in Fig. 4.10, the same set of plots are shown, with the exception that the deadline is decreased to one half of the previous value, now set at 1 s. The reduced deadline apparently reduces the probability of receiving a sensor estimate at the fusion time; but more importantly, it also changes the dynamics of the staggered estimation process by reducing the retrodiction opportunities and increasing the chances of using prediction only. From the plots, it's easily observed that the previous $\tau_1 = 1$ s cases now experience even higher errors than the standard scheduling during and after the maneuver, so do the cases where $\tau_1 = 1.5$ s. Under these settings, the prediction steps are long, and due to the short deadline, there is not enough time for the subsequent sensor estimate to arrive, thereby greatly reducing the chances for applying retrodiction.

4.6.3 Staggered Sensing Scheduling with Sensor Bias

Next we study the case where the IMM estimator at Sensor 1 experiences measurement bias. In Fig. 4.11, the plots are shown for all above staggered schedules with a consistent positive bias term where the range and azimuth angle measurement bias values are $\sigma_r/\sqrt{5}$ and $\sigma_a/\sqrt{20}$ respectively. From the plots, the performances with staggered scheduling largely follow the trends discovered earlier in Fig. 4.9 and cases with $\tau_1 = 1$ s yield improved tracking accuracy levels compared to others. In addition, the performance gap between the T2TF and CI fusers following the maneuver becomes more pronounced, especially for the case where $p = 0.5$. This demonstrates that to some extent, the CI fuser is more tolerant of bias than its T2TF counterpart as the distances between the estimates (which reflects the bias) also appear in the weights which can somewhat mitigate the effect of more biased estimates.

Finally, we consider an increasing measurement bias occurs at Sensor 1, where the range and azimuth biases rise steadily from 0 to $3\sigma_r/\sqrt{5}$ and $3\sigma_a/\sqrt{20}$ respectively toward the end of the 150-second trajectory. From the plots in Fig. 4.12, the estimates from Sensor 1 are now even worse than before, as the errors never return to the pre-maneuver level toward the end of the trajectory. However, both fusers are able to retain similar tracking performance as in the previous steady-bias case, even when only half of the original sensor data are available.

4.7 Conclusion

.

In this chapter, we have studied how the fusion center can exploit staggered scheduling and opportunistically apply intra-state prediction and retrodiction to improve the fusion performance based on link-level loss and delay conditions. Tracking performances of a maneuvering target under variable network and sensor profiles as well as sensing schedules demonstrate the major potentials of such staggered estimation in reducing the tracking errors in the presence of communication and computation constraints.

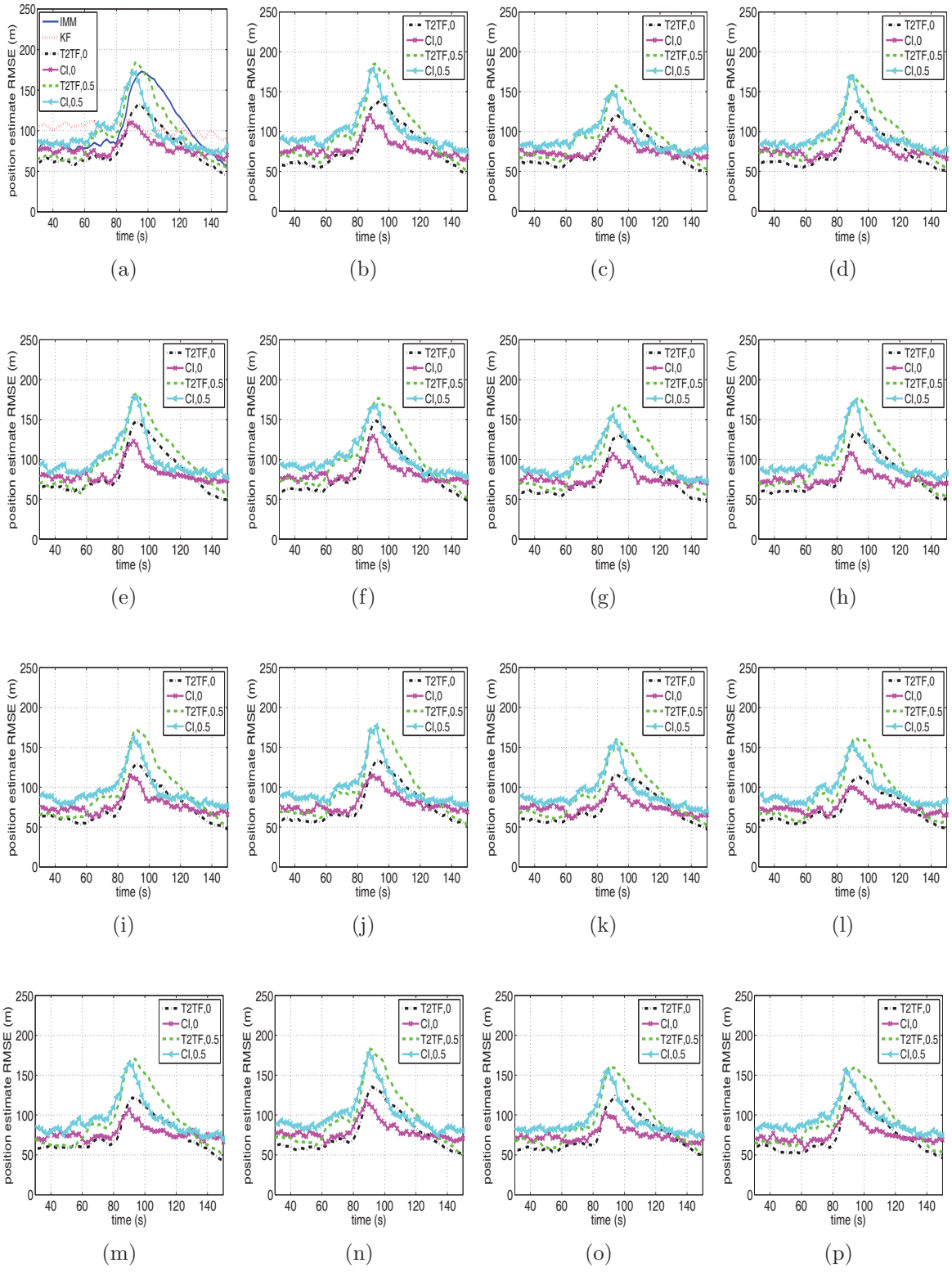


Figure 4.9: Position estimate RMSE performance with variable staggered intervals. Deadline is 2 s. From top left ($\tau_1 = \tau_2 = 0$), τ_1 and τ_2 are increased by 0.5 s along each column and row respectively (e.g., in (g), $\tau_1 = 0.5$ s and $\tau_2 = 1$ s).

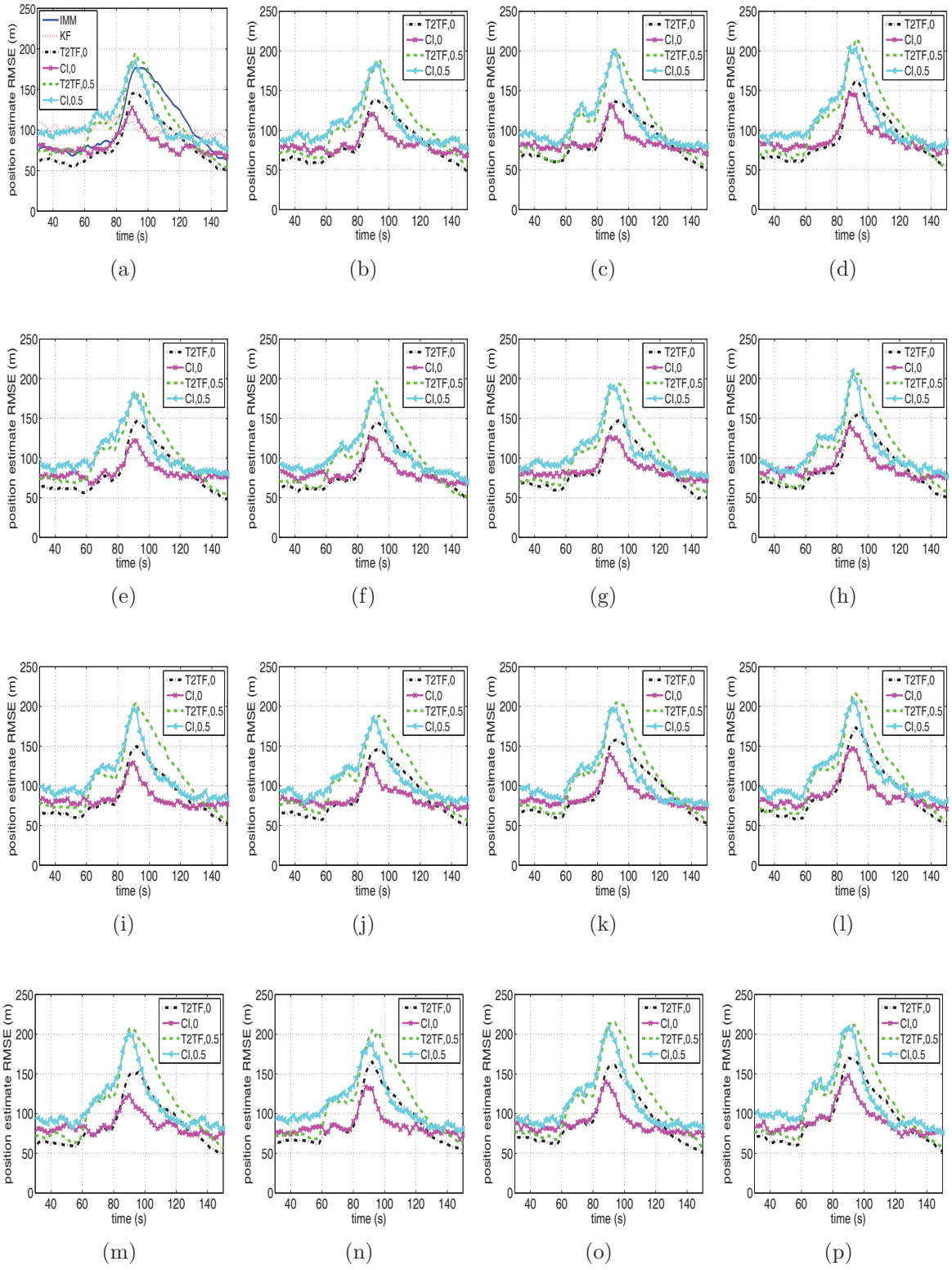


Figure 4.10: Position estimate RMSE performance with variable staggered intervals. Deadline is 1 s. From top left ($\tau_1 = \tau_2 = 0$), τ_1 and τ_2 are increased by 0.5 s along each column and row respectively.

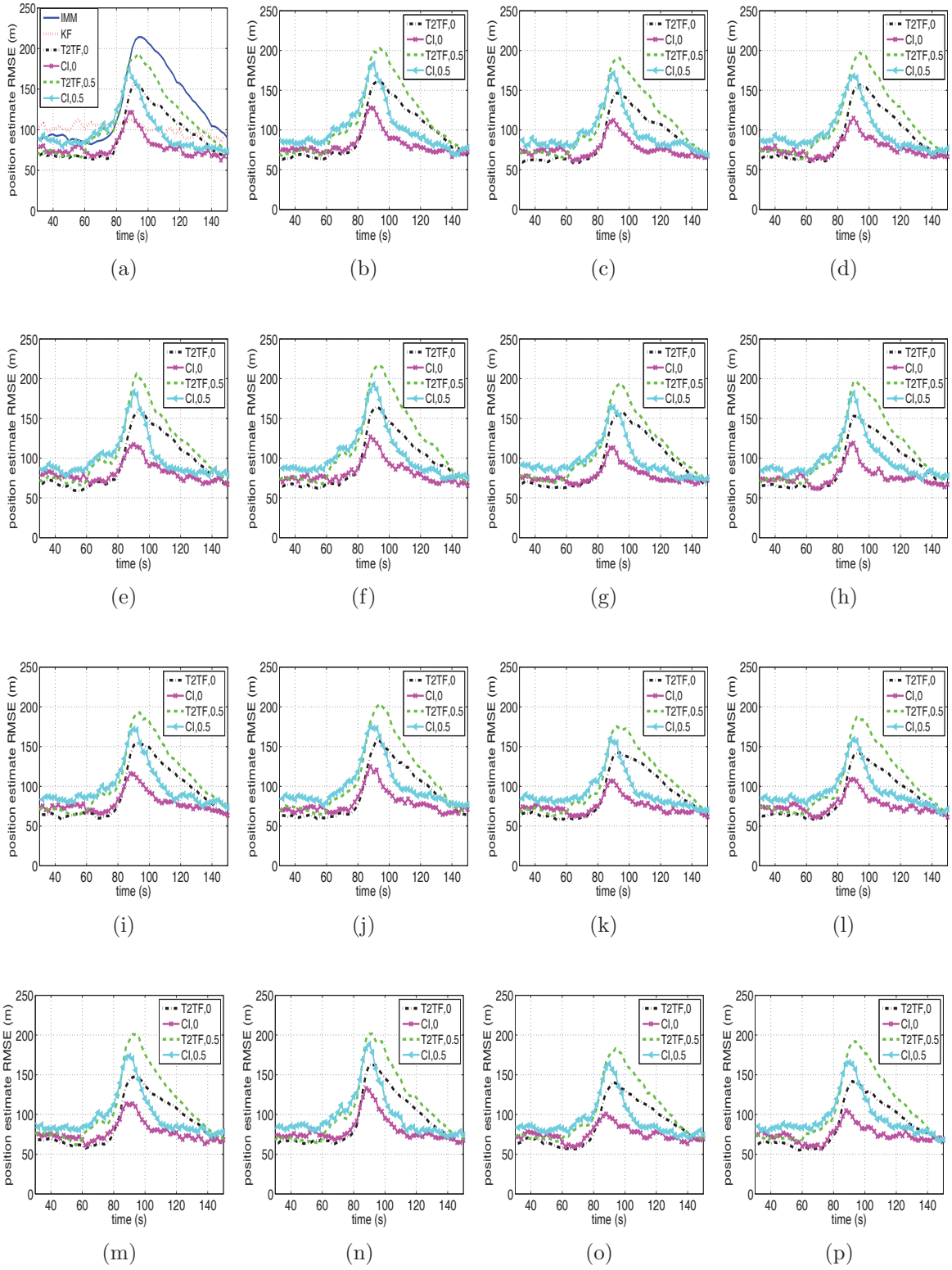


Figure 4.11: Position estimate RMSE performance with variable staggered intervals and steady measurement bias at Sensor 1 (IMM). Deadline is 2 s. From top left ($\tau_1 = \tau_2 = 0$), τ_1 and τ_2 are increased by 0.5 s along each column and row respectively.

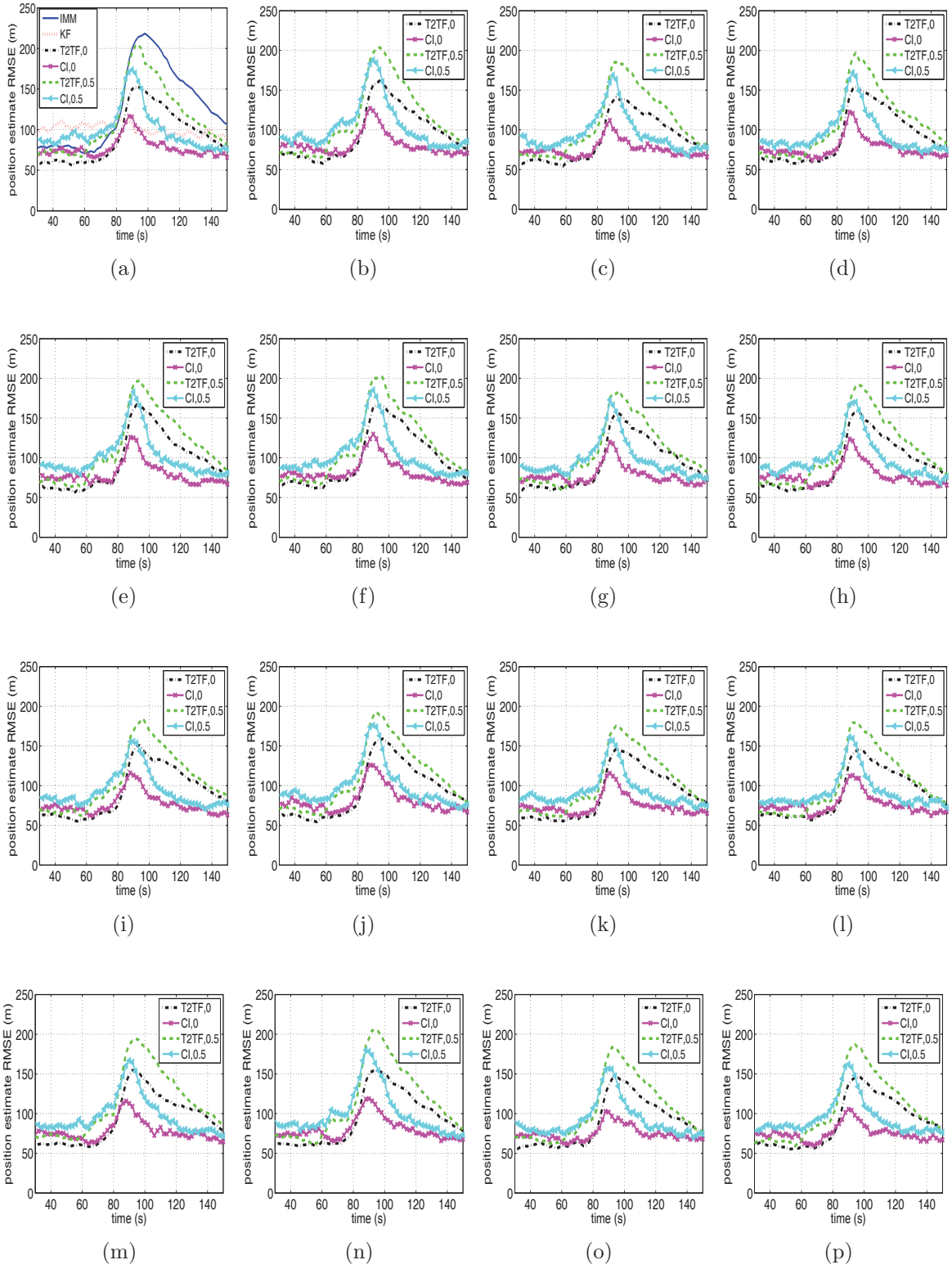


Figure 4.12: Position estimate RMSE performance with variable staggered intervals and increasing measurement bias at Sensor 1 (IMM). Deadline is 2 s. From top left ($\tau_1 = \tau_2 = 0$), τ_1 and τ_2 are increased by 0.5 s along each column and row respectively.

Chapter 5

Information Feedback and Fusion

5.1 Introduction

Despite all the recent research efforts in networked sensing, one aspect of state estimation and fusion in a communication-constrained environment has not received enough attention, namely, information feedback from the fusion center to the individual sensors. Feedback is central to closed-loop control-based design [40], where among others, system stability and convergence is one of the major benefits. In the literature, a number of studies have been conducted in the domain of wireless sensor networks that pertain to the role of feedback: In [8] and [16], for instance, feedback is studied from an information-theoretic perspective, which is used to mitigate the effect of the fading channels; in [74], on the other hand, a control-theoretical calibration algorithm is developed where the feedback is used to enhance signal detection performance in a surveillance network. Despite its importance, there have been very limited studies on the effect of information feedback in estimation and fusion performance. Whereas [76] has considered different information configurations, the study is limited in scope as only feedback to steady-state sensors under largely ideal communication or computation conditions is studied. In the simplest form without communication constraints, the global estimates and associated error covariances generated by the FC are always sent

back to the individual sensors so that the global values can supersede the local counterparts [14].

The main goal of this chapter is to investigate the effect of feedback on estimation/fusion in tracking different types of target trajectories under variable communication loss/delay conditions and sensor bias profiles. Conventionally the fusion center simply serves as the information collector and integrator. As the fusion center is expected to output more accurate estimates under normal circumstances, we are interested in exploring when feedback of a "better" estimate can improve – and at other times worsen – the estimation/fusion performance with or without the presence of link-level communication constraints. Our major contributions include probabilistic analysis of feedback message delivery within a given time-frame and its projected effect on information fusion, and investigation, via simulation studies, of the fusion performance in a two-sensor environment with variable link-level conditions, sensor biases, as well as feedback configurations and schedules. Two types of fusers are studied and their performances under variable conditions are compared in various settings. The system model descriptions, including those of target and measurement models, ways to generate the state estimates at the sensors, and the fusion algorithms at the fusion center, remain largely the same as those in Chapter 4 and are therefore not repeated here.

The remainder of this chapter is organized as follows: In Section 5.2, we explore the effect of information feedback on sensor fusion – via the study of a simplified tracking case – under the ideal assumption that the communication link is perfect. The results therein are further developed in Section 5.3, where communication link loss, delay, and the combination of the two come into play and the resulting impact of information degradation on feedback and fusion is shown. Simulation results of a maneuvering target tracking application, whose settings largely remain the same as those in Chapter 4 – are shown and analyzed in Section 5.4 before the chapter concludes in Section 5.5.

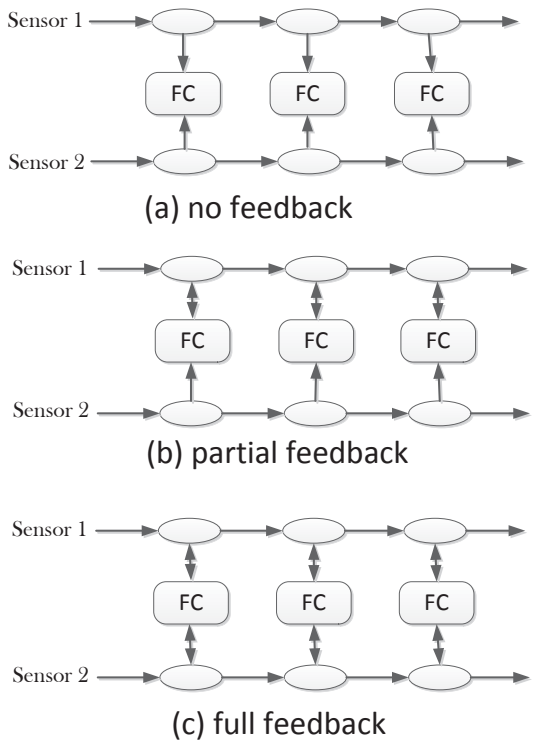


Figure 5.1: Different information feedback mechanisms with two sensors: (a) no feedback; (b) partial feedback (to Sensor 1 only); and (c) full feedback. The horizontal arrows indicate time evolution, whereas the vertical ones indicate the direction of information flow.

5.2 Information Feedback and Its Impact on Fusion without Communication Constraints

In this section, we consider information feedback under the assumption that the communication medium is ideal; that is, every message can be successfully delivered to its intended destination with no or negligible delay. As such, we can single out the effect of instantaneous feedback on information fusion, and focus on the effects of computation constraints in the form of sensor estimation bias, which are demonstrated using a simplified tracking scenario.

5.2.1 Feedback Configurations

In terms of to which sensor(s) a feedback message is being sent back, there exist different feedback configurations. For ease of presentation, here we consider only a two-sensor fusion scenario in this study, although the ideas can be similarly extended to multi-sensor fusion as well. In Fig. 5.1, a two-sensor scenario is shown with the ovals representing sensor estimates and rectangles fused estimates. In (a), as in conventional fusion settings, information exchange is one-way, meaning that no information feedback from the fusion center to the individual sensors exists. In contrast, the partial feedback to Sensor 1 in (b) and full feedback to both sensors in (c) serve to pass the global information generated by the fusion center to the individual sensor(s). Upon receiving a feedback message (containing the fused estimate and its associated error covariance matrix), a sensor substitutes the globally fused information for its own, which is further used as the input for its next-step filtering. An important note here is that under the ideal communication assumption, the fusion center can immediately obtain a fused estimate; in addition, any feedback message can be received by the sensor(s) with no delay[1] as well, thereby affecting its subsequent filtering.

Having introduced different feedback configurations, next we investigate the effect of information feedback with perfect communications. Variable sensor bias profiles that reflect the computational constraints are studied. To achieve this, we somewhat relax the target and sensor measurement settings as follows. We consider only the straight-line segment – as described by the CWNA model – in one generic coordinate ξ with the estimation interval set to be $T = 1$ s and the noise PSD $\tilde{q}_\xi = 1$ m^2/s^3. The sensor measurement model has been simplified in which the sensor directly measures the Cartesian position of the target and hence the measurement $z_k = \mathbf{H}\mathbf{x}_k + v_k$ is available, where $\mathbf{H} = [1 \quad 0]$ is the measurement matrix, and the Gaussian measurement noise v has autocorrelation $\mathbb{E}[v_k v_j] = R\delta_{kj} \triangleq \sigma_v^2 \delta_{kj}$, where $\delta_{(\cdot)}$ is the Kronecker delta function. Both sensors have $\sigma_v = 20$ m.

[1] Or at least this delay is negligible compared to the estimation interval.

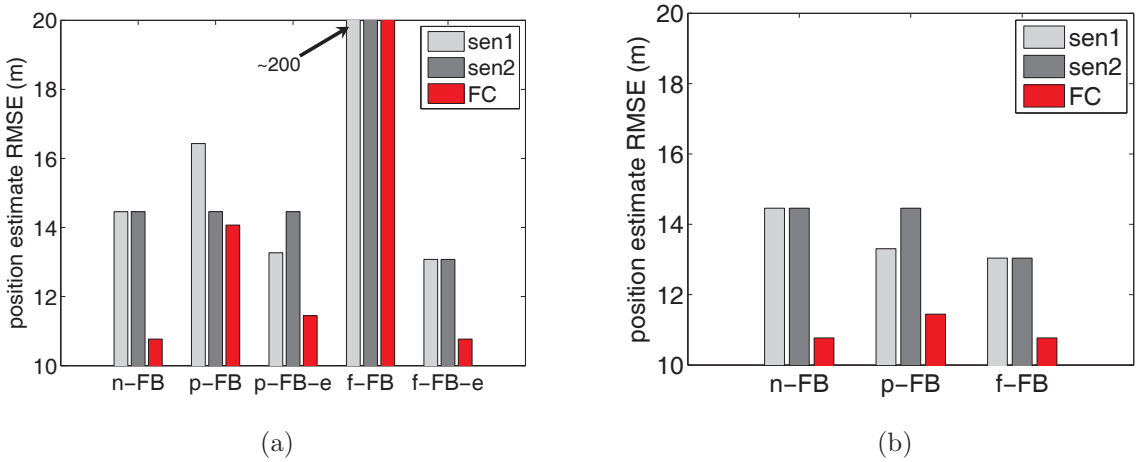


Figure 5.2: Position estimate RMSE performance under variable feedback configurations without communication and computation constraints for (a) T2TF and (b) fast-CI fuser

5.2.2 Feedback with Unbiased Sensor Data

We first consider the case with both sensors yielding unbiased estimates. The effects on position RMSE estimation performance of the sensors and on fusion performance at the fusion center are shown in Fig. 5.2.

In (a), the performance of the track-to-track fuser (T2TF) is shown. The "n", "p", and "f" in the labels stand for no, partial, and full feedback configurations, respectively. When there is no feedback, the fused estimate has on average 25.5% less position RMSE than the individual sensors, or equivalently, 44.5% less MSE after fusion[2]. On the other hand, in both "p-FB" and "f-FB" cases, where a fused estimate $\hat{\mathbf{x}}_F$ and the associated error covariance $\mathbf{P}_F$ are sent back, the resulting fusion performance is poor, especially for the full feedback case (with an RMSE approaching 200). It can be shown that this feedback mechanism, as in the partial feedback (to Sensor 1) case, is equivalent to artificially reducing the $\mathbf{P}$ of the other sensor (Sensor 2) to half of its value at each fusion step; in the full feedback case, $\mathbf{P}$ of both sensors are artificially halved, thereby leading to rapid filter divergence and extremely large fusion errors. As such, two modified feedback schemes are introduced, namely, "p-FB-e" and "f-FB-e", where "e" means that an elevated $\mathbf{P}$ – in this case, $2\mathbf{P}_F$ – is sent back by the

[2]Note that in the ideal case without error cross-covariance, this value would be 50%.

fusion center along with the fused estimate $\hat{\mathbf{x}}_F$.

Next in (b), the performance with the fast-CI fuser is similarly shown. The default setup without feedback yields basically the same error performance as in the T2TF case. Interestingly, this fuser doesn't undergo the above-described filter divergence problem when $\hat{\mathbf{x}}_F$ and $\mathbf{P}_F$ are taken by the sensor(s). In addition, as has been observed in the T2TF case (with elevated $\mathbf{P}$), feedback effectively reduces the sensor estimation errors; however, the fusion performance becomes slightly worse with partial feedback to sensor 1 and remains nearly the same with full feedback to both sensors. This effect can be somewhat explained by the geometric meaning of the CI fuser. Since the fusion result is described by an ellipse that contains the intersection of the two ellipses corresponding to errors of both sensors [14, 77], when a fused estimate is initially sent back and supplants data of Sensor 1, the area corresponding to the next-step fused estimate would at least contain the intersection of error ellipses of Sensor 2 and the fusion center[3], now an even larger ellipse, hence an increased error. In the full-feedback case, the expanding and shrinking of ellipses occur simultaneously, albeit in different directions, resulting in nearly identical intersection areas.

In the case with no information feedback, both sensors and the fusion center are in their respective steady state, meaning statistically the estimation errors remain stable over time. However, feedback subjects the sensor(s) taking the feedback messages and the fusion center to their respective transient quasi-steady state, since once the feedback is withdrawn, the errors will soon revert to their true steady-state values. It is to be understood that all performances under feedback are non-steady state behaviors, since in the steady state, as long as one sensor's error performance is improved, the fused estimate will experience reduced errors as well. From above, during the straight-line motion, with unbiased state estimates, feedback serves to disrupt the steady state of the individual sensors; although the sensor error performance is somewhat improved, there's no immediate benefit in improving the quality of fused estimates.

[3]In fact the updated Sensor 1 estimate would be described by a different ellipse than the fusion center's from the previous step. But the net effect is still a larger new intersection region.

5.2.3 Feedback with Biased Sensor Data

Now we focus on the case where the estimate quality of one sensor (Sensor 1 in this study) is compromised due to persistent measurement biases. In particular, a biased measurement z_k^b at time k consists of both deterministic and random elements: $z_k^b = \mathbf{H}\mathbf{x}_k + v_k + b_k + v_k^b$, where b_k is the fixed bias and v_k^b is the random element such that $\mathbb{E}[(v_k^b)^2] = R^b$ and R^b is the associated bias variance. To measure the effect of both deterministic and random biases, we consider a joint bias term $\tilde{v}_k$ that satisfies $\tilde{v}_k^2 = b_k^2 + (v_k^b)^2$ as the metric for bias. In our case study example, b_k is selected to be 96% of the average bias $\tilde{v}_k$ and the random bias term v_k^b can be easily generated as well. An important point here is that Sensor 1 is not aware of its measurement bias; therefore, $\mathbf{P}_1$ it generates is an over-optimistic description of its actual estimation error.

Fig. 5.3 shows the RMSE performance under variable Sensor 1 measurement bias levels and feedback configurations. It is interesting to note that the T2TF (with elevated error covariances sent back) and fast-CI fusers yield the same performance and therefore they are plotted together. As expected, the estimation error increases rapidly as the overall bias level goes up. The resulting fused estimates also experience degraded error performance, which after a certain point, becomes even worse than using Sensor 2 alone. With information feedback from the fusion center, the error performance of Sensor 1, again, can be improved significantly. Unlike the no-bias case, as the bias level increases, the fused estimates start to possess a better quality when the feedback is sent back to Sensor 1 only (labeled as "p-FB-1") than in the no- or full-feedback case. This demonstrates the major benefit of information feedback is to counteract sensor measurement/estimation bias. Of course, the premise is that feedback has to be selectively sent to the right sensor (i.e., biased sensor); otherwise, it can actually worsen the performance when feedback is sent to the better sensor (i.e., Sensor 2). The potential bias at any sensor can be possibly learned from historical data. Another interesting observation is that although the fusion performance under different partial and full feedback configurations varies, the effects on the individual sensors remain largely the

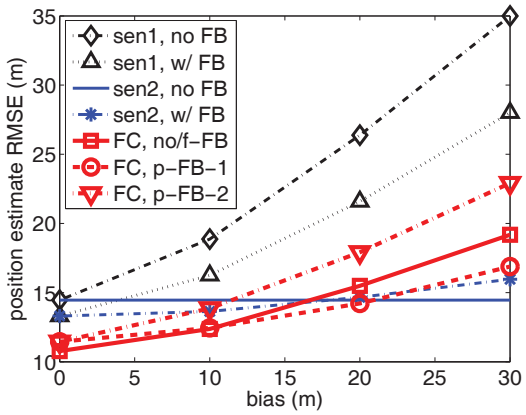


Figure 5.3: Simplified tracking long-term position estimate RMSE performance under variable feedback configurations and bias levels without communication constraints (both T2TF and fast-CI fusers)

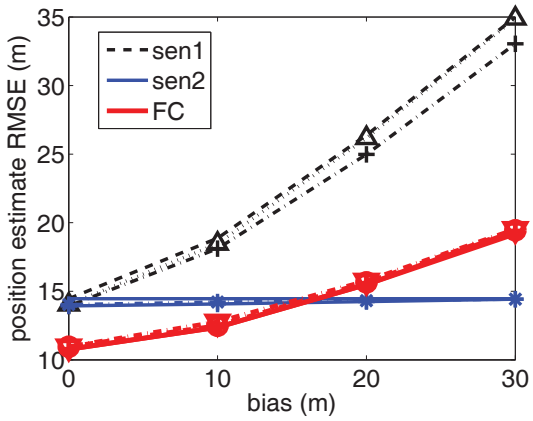


Figure 5.4: Position estimate RMSE performance under variable feedback configurations and bias levels with fixed RTT = 1 s (both T2TF and fast-CI fusers)

same; that is, the effect of feedback on Sensor 1 is independent of that on Sensor 2 and vice versa.

5.3 Information Feedback with Communication Constraints

Now we turn our attention to information feedback in the presence of communication constraints. We assume that the sensors can take measurements and then in turn generate and send out their state estimates in a timely manner; it is the imperfect communication link between any sensor and the fusion center that may result in unavailable state estimates at the latter. Doing so allows us to focus mainly on evaluating different information feedback schemes at the fusion center that can potentially improve estimation/fusion performance.

5.3.1 Case Study: Fixed Delay

Before we consider feedback under the independent loss and shifted-exponential delay settings used for the satellite-based links, we explore the simplified case in the last section one

step further by adding in fixed communication delay in both directions. More specifically, the RTT of the network is set to be a fixed 1 s, with each way (sensor $\rightarrow$ FC or FC $\rightarrow$ sensor) taking up exactly 0.5 s. Following this setup, half a second after both sensor estimates are sent out, the fusion center receives and combines them to obtain a fused estimate. Immediately, this estimate is sent back to the desired sensor(s). Suppose all processing delays are negligible, a sensor will receive the fused estimate one second after sending out its own. Because the estimation interval is also 1 s, the sensor needs to update this fused estimate (for the previous estimation epoch) to its current estimate. We now investigate how fusion performance will change with such delayed feedback and response.

Fig. 5.4 shows the estimation and fusion performance under variable feedback configurations and bias profiles as in the previous section. Compared to the results in Fig. 5.3, changes in fusion performance after feedback become much less visible regardless of the bias level. The main reason for this result is that in order to update the previous feedback message forward by 1 s, the sensor has used its latest measurement – which is exactly the biased one feedback attempts to improve. Another way to examine the effect is to count how many times a biased measurement is used in the sensor estimation process. Without communication delay, it is used just once for generating a new estimate; with RTT = 1 s, however, it is used an additional time to update the feedback message over time. As a result, the fusion performance is hardly affected by feedback.

Alternatively, to bypass the biased measurement, Sensor 1 can use one-step prediction to update the delayed feedback message. In Fig. 5.5, the performance comparison between updates using measurements ("-meas") and those using prediction ("-pred") is shown under different feedback configurations and a measurement bias = 20 m. Of interest to us is the case where feedback is sent back to Sensor 1, which in turns uses prediction to forward the feedback message by one second. It turns out this approach yields better fusion performance than others, and the difference will become more apparent with even higher biases.

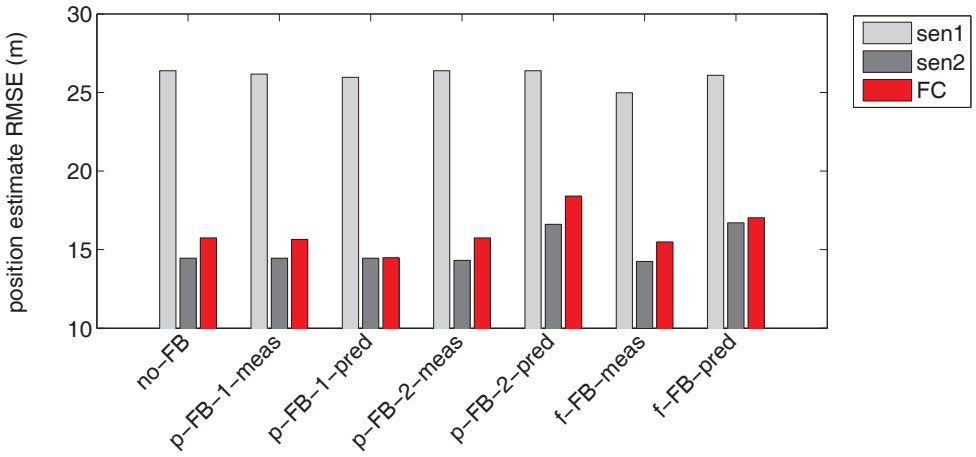


Figure 5.5: Position estimate RMSE performance with variable feedback configurations and update methods, measurement bias of Sensor 1 = 20 m, RTT = 1 s (both T2TF and fast-CI fusers)

5.3.2 Timing of Feedback with Random Loss and Delay

From the above case study, we noted that delayed response may reduce the potential of the information feedback to mitigate the effect of sensor bias. We are primarily interested in two aspects related to feedback when communication constraints exist: (1) when is the feedback message sent back by the fusion center and what's the quality of the message? (2) when is the message received by the sensor and how will the message affect the quality of its subsequent estimate(s)?

Suppose $f_1(\cdot)$ and $f_2(\cdot)$ represent the forward and reverse link delay pdf's, with their respective mean random delay being μ_1 and μ_2:

$$f_1(t) = \frac{1}{\mu_1} \exp^{-\frac{t-T_I}{\mu_1}}, \; f_2(t) = \frac{1}{\mu_2} \exp^{-\frac{t-T_I}{\mu_2}}, \text{ for } t \geq T_I. \tag{5.1}$$

Then the pdf $h(\cdot)$ of the total time for a feedback message to return to a sensor (starting from the time when the estimate was originally sent out by the sensor to the fusion center) can be found as

$$h(t) = \frac{1}{\mu_1 - \mu_2}(e^{-\frac{t-2T_I}{\mu_1}} - e^{-\frac{t-2T_I}{\mu_2}}) = \frac{\mu_1}{\mu_1 - \mu_2} e^{\frac{T_I}{\mu_1}} f_1(t) - \frac{\mu_2}{\mu_1 - \mu_2} e^{\frac{T_I}{\mu_2}} f_2(t), \text{ for } t \geq 2T_I,$$

(5.2)

and the distribution function can be found by replacing $f_1(\cdot)$ and $f_2(\cdot)$ with their respective cdf's respectively. With this distribution, the probability mass function (pmf) of the time when a subsequently updated estimate is sent out by the sensor can be obtained by taking the ceiling of t, and in turn the pdf of the time when the updated message takes effect at the fusion center by superimposing the above time on another forward-link communication delay.

With the presence of communication loss, the feedback process becomes much less certain in terms of the quality of any fused estimate to be sent back. If in the case of a lost message from a sensor, the fusion center applies prediction from the previously available estimates for that sensor, then a fused estimate at time k could be obtained from (1) available $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$ at k; (2) available $\hat{\mathbf{x}}_1$ but a predicted estimate from an earlier time for Sensor 2; (3) available $\hat{\mathbf{x}}_2$ but a predicted estimate from an earlier time for Sensor 1; and (4) predicted estimates from (possibly different) earlier times for both sensors.

The above observations are easily made; however, the effect of communication delay and loss should be examined more closely both at the fusion center and at the sensor to which a message is fed back. The communication delay profile apparently determines the arrival time of any feedback message, but the length of the sampling interval T and the epoch of feedback time T_F – that is, the time instant when the fusion center sends back its fused estimate – also plays a role in the feedback performance. Suppose a reporting deadline D is defined as the maximum allowable time for the fusion center to fuse the available sensor estimates. Note that the fusion center can decide to feedback a message earlier than D, i.e., $T_F \leq D$. The question arises: what message is fed back?

In our two-sensor case, it can be derived that in a lossless scenario, an average amount of $\mathbb{E}[\max(T_1, T_2)] = T_I + 3\mu_1/2$ time is needed before the fusion center receives both sensor estimates. However, if the fusion center restricts the fusion time within a certain range, the

total amount of time it takes for the feedback message can be found to follow the distribution

$$\Pr\{t \leq D\} = \iint_0^D f_2(t|T_F)g(T_F)\mathrm{d}t\mathrm{d}T_F, \tag{5.3}$$

where $f_2(t|T_F)$ is the conditional pdf contingent on a certain T_F value, and $g(\cdot)$ is the prior pdf of the fusion/feedback time. If the fusion time is chosen uniformly from the interval $[T_I, D - T_I]$ (accounting for the propagation delay), i.e., $g(T_F) = 1/(D - 2T_I)$ for $T_F \in [T_I, D - T_I]$ and 0 otherwise, then the distribution in Eq. (5.3) can be further derived as

$$\Pr\{t \leq D\} = \frac{1}{D - 2T_I}\int_{T_I}^{D-T_I}(1 - e^{-\frac{D-T_I-t}{\mu_2}})\mathrm{d}t = 1 - \frac{\mu_2}{D - 2T_I}F_2(D - T_I), \tag{5.4}$$

which is a further improvement compared to the distribution derived with Eq. (5.2) alone, i.e., without the additional prior knowledge about T_F.

On the other hand, with loss, the probability that the fusion center can obtain both sensor estimates by the feedback time T_F is $(1-p)^2F_1^2(T_F)$ where p is the link-level loss rate. With a longer T_F, it is more likely that the fusion center has obtained a better estimate by the feedback time, after fusing the latest arrivals from both sensors. However, the subsequent reverse-link communication would have a delayed start, thereby affecting the arrival time of the feedback message.

As in the previous case study, when the feedback message happens to arrive after an estimation interval T, then the sensor would have to update this fused estimate (for the previous estimation epoch) to its current time. To forward this feedback estimate by T, the sensor can recycle its latest measurement and re-generate a state estimate that is based on the feedback message. Therefore, the newest measurement is used twice in the context of feedback: (a) to generate a sensor state estimate as usual; and (b) to reincorporate a delayed feedback message and to generate an updated state estimate. Unfortunately, the new measurement is more likely to be another biased one that feedback attempts to improve

in the first place. Using it twice is hence likely to counteract the goal of reducing sensor bias. Alternatively, to bypass the biased measurement, the sensor can instead use one-step prediction to update the delayed feedback message. However, as prediction alone often brings higher estimation errors in the presence of model uncertainty, this will likely diminish the gain from feedback as well.

More generally, when there's no communication loss, the average amount of time it takes for the feedback message to arrive is $T_I + \mu_2$; with the initial period T_F, the total average latency between the time when the feedback message is received and the (prior) time instant the message describes is $T_F + T_I + \mu_2 \leq 2T_I + 3\mu_1/2 + \mu_2$. If this duration exceeds the estimation interval T, the sensor has already generated a new estimate before the feedback message arrives; then it would have to either use prediction or measurements to project this delayed fused estimate to its next pending estimate, with an average of no less than $\lceil \frac{2T_I+3\mu_1/2+\mu_2}{T} \rceil$ prediction steps, or alternatively, using each measurement $\lceil \frac{2T_I+3\mu_1/2+\mu_2}{T} \rceil$ times.

In summary, to reduce the potential negative effect of estimation error increase due to prediction and/or measurement bias following delayed feedback, it is preferable to use a smaller T_F. However, with an early time cutoff, the fusion center is using less information for fusion, which may have a more adverse effect on tracking performance as a decrease in T_F can easily reduce the probability of having new arrivals from both sensors. Accounting for the timing factors analyzed above, although it is possible to probabilistically combine the long-term error values for these different scenarios (such as those found in the simplified example in the previous section) to approximate the errors after fusion, in what follows, we will mainly resort to simulation studies for an evaluation of the tracking performance of a maneuvering target with different communication statistics, sensor measurement profiles as well as feedback configurations and schedules.

5.4 Performance Evaluation

In this section we evaluate the impact of variable feedback schedules on estimation and fusion performance in tracking a target undergoing a more complex set of motions again as described in Section 4.2. To recap, a turn is in place from $t = 60$ s to 90 s in between straight-line movements. Two sensors, with Sensor 1 using an IMM estimator and Sensor 2 a KF, take measurements and generate their state estimates once every $T = 2$ s.

5.4.1 Feedback on Estimation and Fusion without Bias

To begin, Fig. 5.6 shows the estimation and fusion performance where no feedback occurs[4]. Loss rates of $p = 0$, 25%, 50%, as well as a fusion deadline of $D_F = 2$ s and 1 s are considered. Similar to what we have seen in the previous chapter, when there's no loss, the CI fuser performs slightly worse during the initial straight-line segment. During the peak of the maneuver and for a period of time after the maneuver is over, on the other hand, it performs much better than its T2TF counterpart. However, this superiority is easily offset with a higher loss rate, as the CI fuser is more susceptible to loss compared to T2TF. As shown in (c) and (f), when the loss rate reaches 50%, the output of the CI fuser is worse than those from both the IMM track and the T2TF.

Next we consider the effect of feedback under the same set of conditions. Note that in the remainder of the chapter, we consider only partial feedback to Sensor 1 (using the original error covariances) as we try to minimize the peak error performance during the maneuver. In Figs. 5.7 and 5.8, tracking performances with feedback from the output of the T2TF and the CI fuser are shown respectively. Loss rates of $p = 0$ and 50% are considered, along with three feedback schedules with forward and reverse link deadlines set as (1) $D_F = 2$ s, $D_R = 1.5$ s; $D_F = 1.5$ s, $D_R = 1.2$ s; and $D_F = 1.2$ s, $D_R = 0.8$ s respectively. The deadlines for the reverse link are smaller than the associated forward-link deadlines as the average

[4]Note the results look slightly different from those shown in the previous chapter, more noticeably for the case where $p = 0.5$. This is because we have adopted a slightly different implementation at the fusion center where it keeps track of the turn rate component as well.

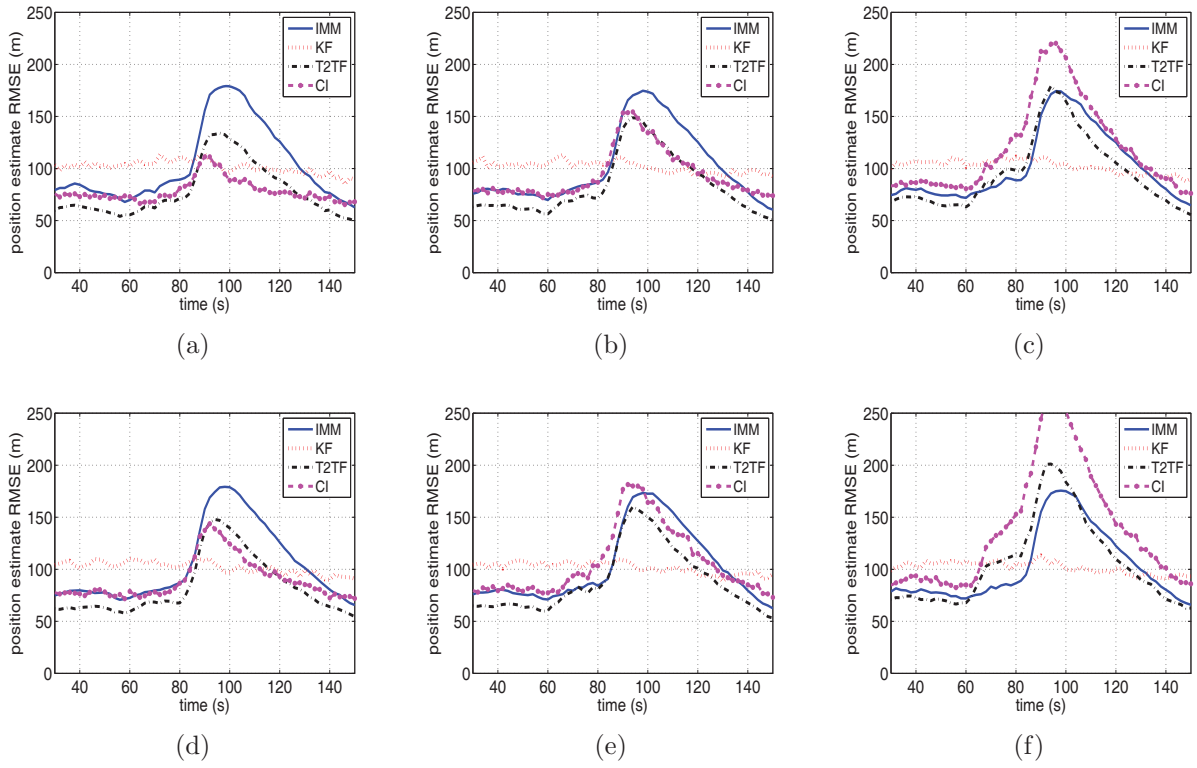


Figure 5.6: Position estimate RMSE performance with variable link-level loss rates and deadlines, no measurement bias or feedback. First row: $D_F = 2$ s; second row: $D_F = 1$ s. Loss rates are 0, 25%, and 50% from left to right for each row.

communication delay on the former is also smaller. From the plots, we observe the feedback messages are able to largely reshape the error curves of Sensor 1 around the maneuver time period. In particular, the peaks of the error bell curves for Sensor 1 are significantly narrowed by the feedback, although the new peaks are slightly higher than the original. Across all the cases, although there are some variations in the shape and position of the IMM error curves, the tracking performance for the fuser itself remains more or less stable under the same loss condition. On the other hand, feedback from the CI fuser results in even more drastic changes in the estimation and fusion behavior. Now the IMM curves (when $p = 0$) are largely flattened and there's hardly any noticeable increase in the errors of the IMM or fuser outputs during the maneuver. However, such desired performance does not hold for the $p = 0.5$ case, as the CI fuser itself already yields even worse estimates than the IMM, and feeding back such estimates may temporarily improve the IMM estimator performance,

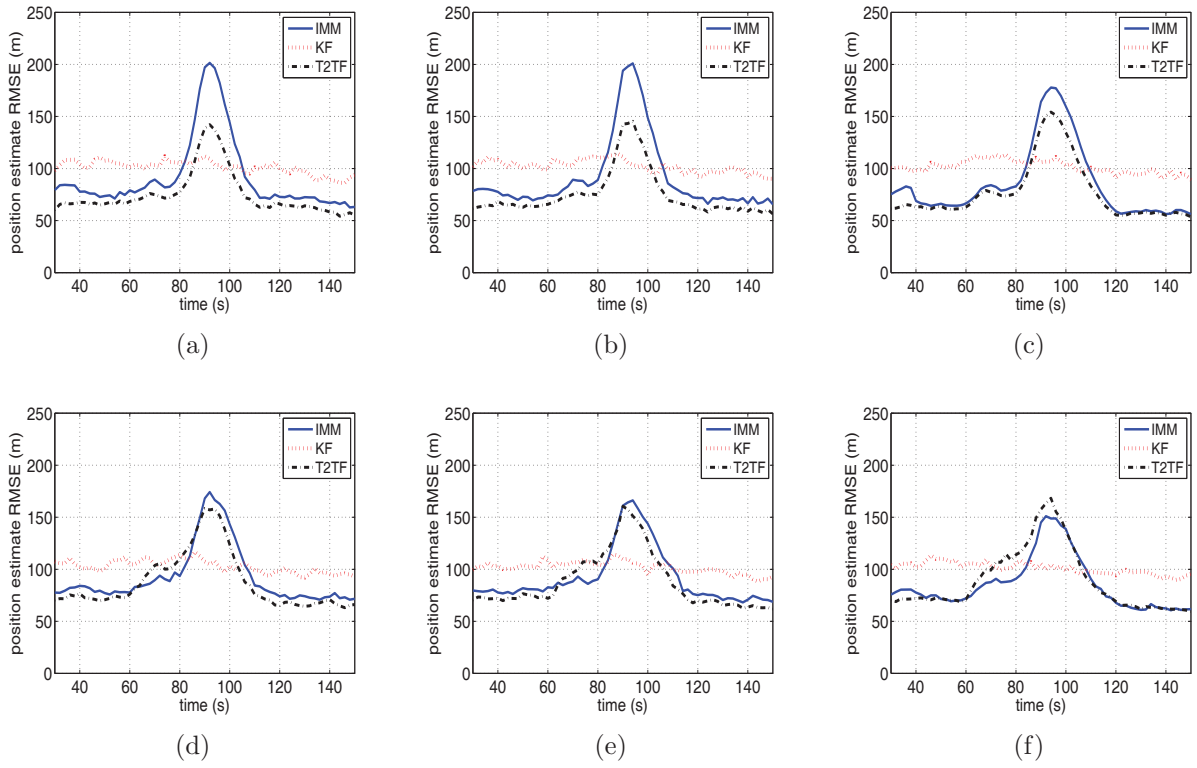


Figure 5.7: Position estimate RMSE performance with variable feedback schedules and link-level loss rates for the T2TF fuser, no measurement bias. The first row: $p = 0$; second row: $p = 0.5$. From left to right are different schedules: $D_F = 2$ s, $D_R = 1.5$ s; $D_F = 1.5$ s, $D_R = 1.2$ s; $D_F = 1.2$ s, $D_R = 0.8$ s.

but it does little to pull the CI error curves down. Worse still, the fuser outputs tend to become unstable after the maneuver is over. These observations demonstrate (1) feedback can also benefit the estimation and fusion performance when there's no bias; and (2) there exist performance trade-offs when deciding which fuser to use for feedback, as different fusers may have different responses to communication loss and delay.

5.4.2 Feedback on Estimation and Fusion with Bias

Since we discovered that one major benefit of feedback is to mitigate sensor bias, now we investigate how tracking performances change with feedback sent from either type of fuser. In Figs. 5.9, 5.10, and 5.11, estimation and fusion performances without feedback, with feedback from the T2TF, and from the CI fuser are shown respectively for the case where

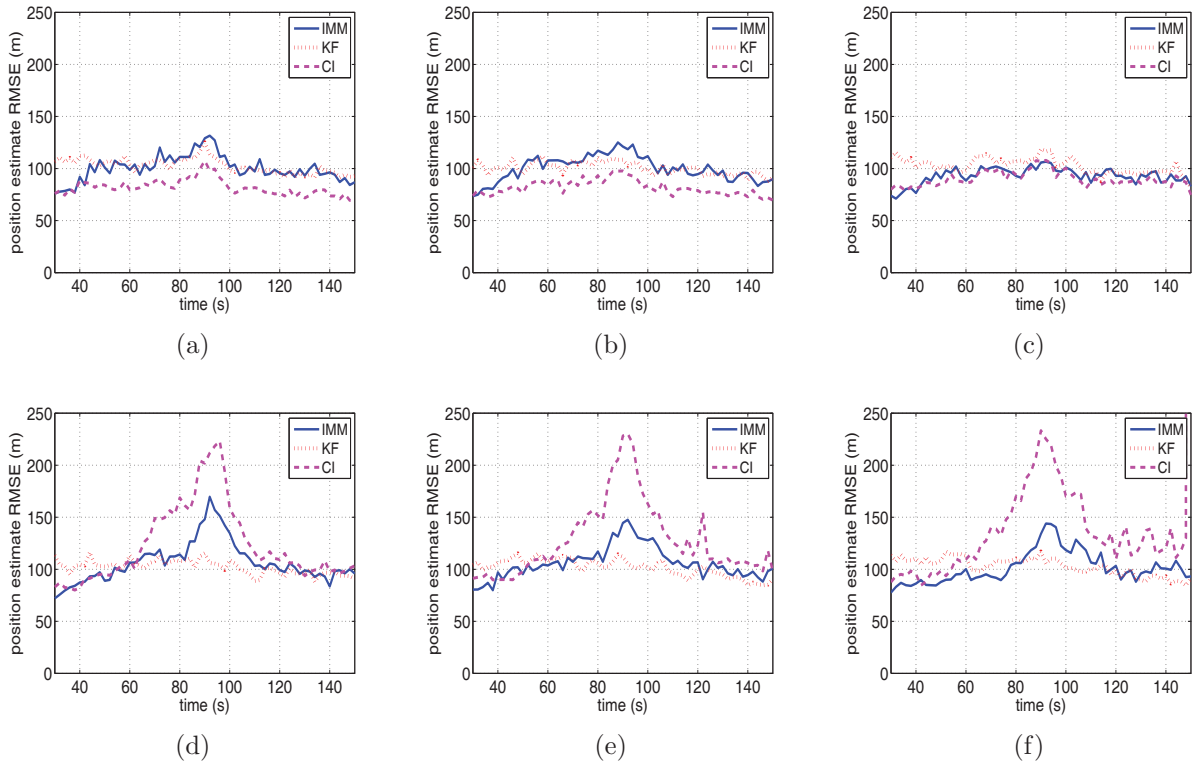


Figure 5.8: Position estimate RMSE performance with variable feedback schedules and link-level loss rates for the CI fuser, no measurement bias. The first row: $p = 0$; second row: $p = 0.5$. From left to right are different schedules: $D_F = 2$ s, $D_R = 1.5$ s; $D_F = 1.5$ s, $D_R = 1.2$ s; $D_F = 1.2$ s, $D_R = 0.8$ s.

there is a positive bias term with the measurement bias values as $\sigma_r/\sqrt{5}$ for the range and $\sigma_a/\sqrt{20}$ for the azimuth. As before, feedback from the T2TF can reduce the Sensor 1 IMM estimation error much faster at the peak of and after the maneuver, which in turn improves the fusion performance to some extent as well. However, the CI fuser again experiences much improved track performance where there's no loss, and is likely to become unstable with half of the original sensor data missing. Such instability may exacerbate the subsequent IMM filtering as well if the sensor replaces its own data with the feedback without discretion.

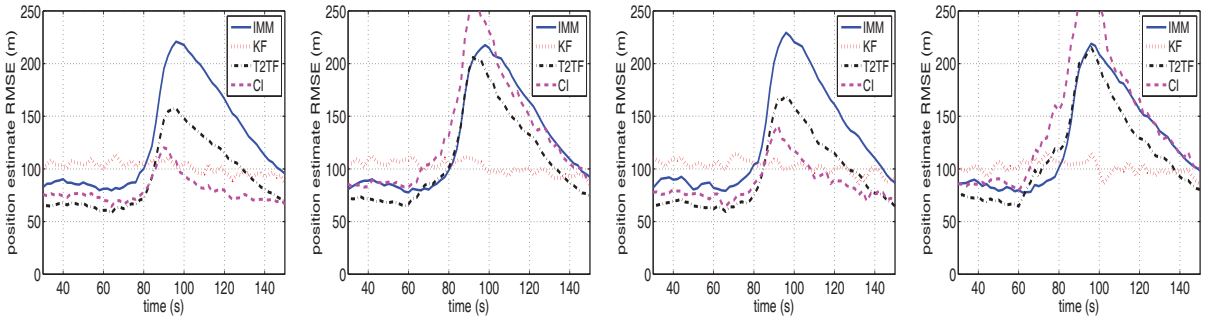
Finally, the same set of simulations are run for the case with increasing measurement bias over time and the results are plotted in Figs. 5.12, 5.13, and 5.14. In these simulations, the range and azimuth biases rise steadily from 0 to $3\sigma_r/\sqrt{5}$ and $3\sigma_a/\sqrt{20}$ respectively toward the end of the 150-second trajectory. We can see the curves largely follow the same trend

as in the previous case. From these results, it's fair to say that when there is no loss or if the loss rate is rather small, it's better to run the CI fuser to reduce the peak error during maneuver; otherwise, the T2TF is preferred for its stability and superior performance under a high level of loss.

5.5 Conclusion

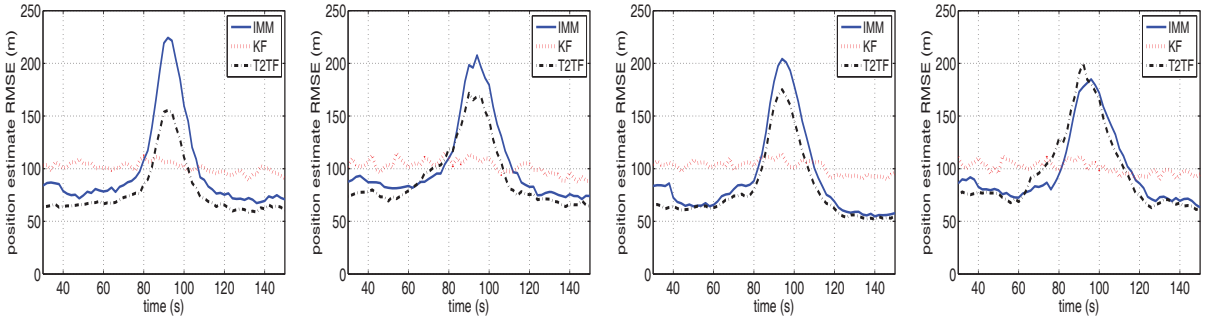
.

In this chapter, we have investigated the effect of different information feedback configurations and schedules in the context of variable communication link-level loss and delay conditions as well as sensor bias levels. Numerical and simulation studies with a two-sensor fusion scenario are used to demonstrate the major benefits and limitations of information feedback as applied in a long-haul tracking application. We have observed that although information feedback has potentials to reduce sensor bias, and sometimes even reduce the estimation and fusion errors without the presence of sensor bias, the communication link-level communication loss and delay can pose a major challenge to fulfill such potentials. As such, information feedback should be used in conjunction with other techniques that can reduce the effect of loss and delay.



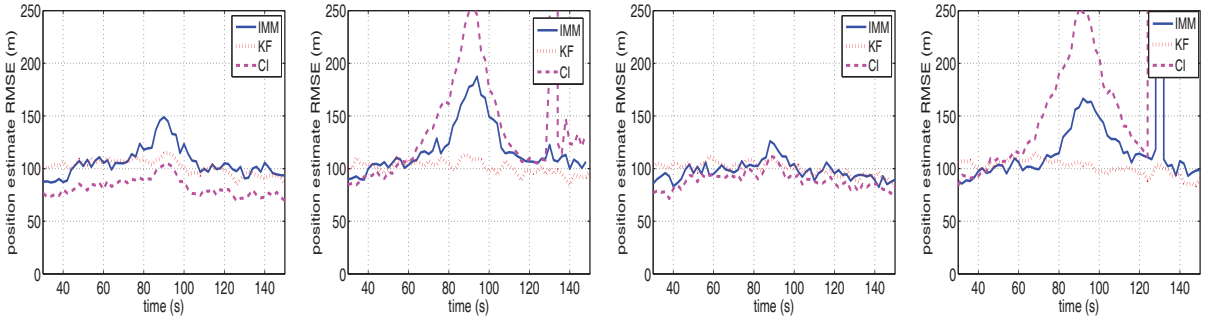
(a) (b) (c) (d)

Figure 5.9: Position estimate RMSE performance with variable link-level loss rates and deadlines, steady measurement bias, and no feedback. (a) $p = 0$, $D_F = 2$ s; (b) $p = 0.5$, $D_F = 2$ s; (c) $p = 0$, $D_F = 1.2$ s; (d) $p = 0.5$, $D_F = 1.2$ s.



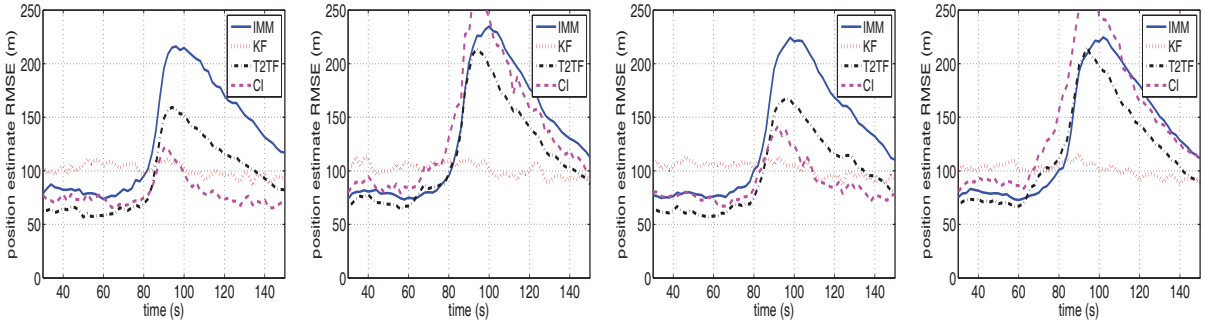
(a) (b) (c) (d)

Figure 5.10: Position estimate RMSE performance with variable link-level loss rates and feedback schedules, steady measurement bias, and no feedback. T2TF. (a) $p = 0$, $D_F = 2$ s, $D_R = 1.5$ s; (c) $p = 0.5$, $D_F = 2$ s, $D_R = 1.5$ s; (b) $p = 0$, $D_F = 1.2$ s, $D_R = 0.8$ s; (d) $p = 0.5$, $D_F = 1.2$ s, $D_R = 0.8$ s.



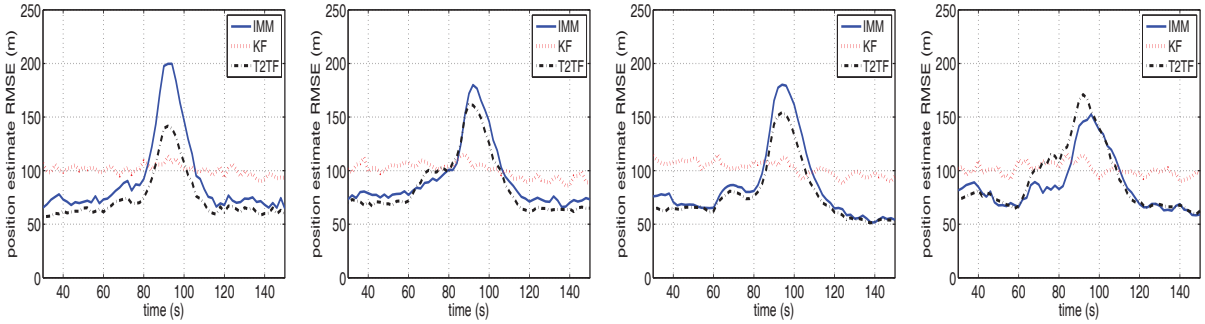
(a) (b) (c) (d)

Figure 5.11: Position estimate RMSE performance with variable link-level loss rates and feedback schedules, steady measurement bias, and no feedback. CI fuser. (a) $p = 0$, $D_F = 2$ s, $D_R = 1.5$ s; (b) $p = 0.5$, $D_F = 2$ s, $D_R = 1.5$ s; (c) $p = 0$, $D_F = 1.2$ s, $D_R = 0.8$ s; (d) $p = 0.5$, $D_F = 1.2$ s, $D_R = 0.8$ s.



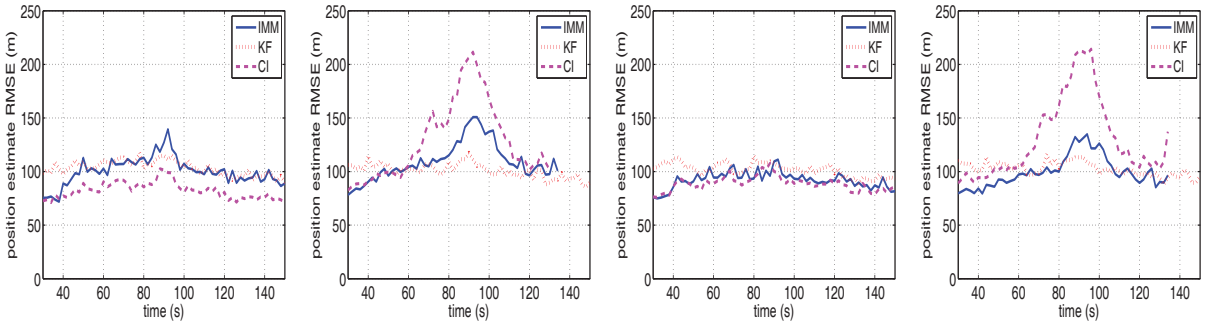
(a) (b) (c) (d)

Figure 5.12: Position estimate RMSE performance with variable link-level loss rates and deadlines, increasing measurement bias, and no feedback. (a) $p = 0$, $D_F = 2$ s; (b) $p = 0.5$, $D_F = 2$ s; (c) $p = 0$, $D_F = 1.2$ s; (d) $p = 0.5$, $D_F = 1.2$ s.



(a) (b) (c) (d)

Figure 5.13: Position estimate RMSE performance with variable link-level loss rates and feedback schedules, increasing measurement bias, and no feedback. T2TF. (a) $p = 0$, $D_F = 2$ s, $D_R = 1.5$ s; (c) $p = 0.5$, $D_F = 2$ s, $D_R = 1.5$ s; (b) $p = 0$, $D_F = 1.2$ s, $D_R = 0.8$ s; (d) $p = 0.5$, $D_F = 1.2$ s, $D_R = 0.8$ s.



(a) (b) (c) (d)

Figure 5.14: Position estimate RMSE performance with variable link-level loss rates and feedback schedules, increasing measurement bias, and no feedback. CI fuser. (a) $p = 0$, $D_F = 2$ s, $D_R = 1.5$ s; (c) $p = 0.5$, $D_F = 2$ s, $D_R = 1.5$ s; (b) $p = 0$, $D_F = 1.2$ s, $D_R = 0.8$ s; (d) $p = 0.5$, $D_F = 1.2$ s, $D_R = 0.8$ s.

Chapter 6

Artificial Neural Network Learning-Based Information Fusion

6.1 Introduction

A number of data fusion methods have been developed over the years, with a primary goal of taking in the data from multiple sensors and combining them to produce a condensed set of meaningful information with the highest possible degree of accuracy and certainty [14, 78]. Whereas most conventional state fusion approaches produce fused estimates by linearly combining the available sensor data, the use of nonlinear fusers is still fairly unexplored. In contrast to the closed-form linear fusers such as the T2TF and fast-CI fusers used extensively in the preceding chapters, in this chapter we are primarily interested in using nonlinear functions to fuse sensor data that may potentially yield better results than with linear fusion.

Essentially, information fusion is a regression problem, and there are many existing regression analysis techniques that can be applied. In many applications, field tests may be performed a priori using the available sensor networks to collect test measurements. Such field tests with known true target states facilitate learning-based fuser design as many types

of regression analysis methods exist and can be used to learn or compute the parameters of the fusing function we wish to estimate from these field tests. In this chapter, we look at the use of Artificial Neural Networks for fusing the state estimates as they are known to be able to approximate any continuous function given sufficient parameters.

Artificial neural networks (ANNs) have been applied to tasks such as pattern classification, clustering/categorization, function approximation, and prediction/forecasting, among others [30, 35]. In the literature, [21] and [25] have proposed using ANNs for sensor fusion, where the neural networks are used to determine the weights for linearly combining sensor state estimates. More recently, [17] has proposed nonlinear fusion using ANNs. This study of ANN-based fusers accounts for both communication and computation constraints and aims to yield improved long-haul target tracking and monitoring performance under these constraints. In particular, besides performing the core function of learning from true target trajectories and sensor data and then applying such learned patterns to testing data to be combined by the fusion center, we also consider how to perform such tasks effectively with (1) very limited training data; (2) lost data in both training and testing stages; and (3) sensor biases. Of concern throughout our study is the performance of generalization capabilities from training to testing data under variable communication and computation constraints as discussed earlier. A ballistic target tracking application is used to demonstrate the performance of our learning-based design.

The remainder of this chapter is organized as follows: In Section 6.2, the fundamentals of ANNs and the learning algorithm called backpropagation are briefly overviewed. Next in Section 6.3, the enhanced variations of a standard backpropagation algorithm are considered that could improve the generalization capabilities of the training-based fuser to new data. We consider the communication and computation constraints in Sections 6.4 and 6.5 respectively, highlighting the effect of missing data and bias on the training and testing stages. Simulation results of a ballistic target tracking application are presented and analyzed in Section 6.6 before we conclude this chapter in Section 6.7.

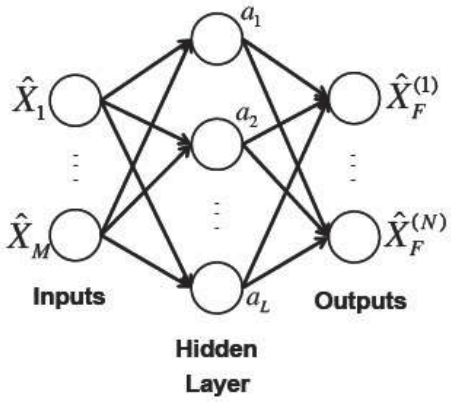


Figure 6.1: An example of a three-layer feedforward neural network

6.2 Artificial Neural Networks (ANNs) and Backpropagation

6.2.1 Structure of an ANN

Over the past few decades, advances have been made in developing intelligent systems, some of which – notably the artificial neural networks (ANNs) – have been inspired by biological neural networks. Broadly speaking, ANNs are a class of statistical models that consist of sets of adaptive weights (i.e., numerical parameters) that are tuned by a learning algorithm, and are capable of approximating non-linear functions of their inputs. There are different types of ANNs, but we focus on the simplest feedforward neural networks where connections between the units do not form a directed cycle or loop (i.e., no feedback exists in the network). The structure of such a three-layer neural network is shown in Fig. 6.1. This network consists of an input layer, a hidden layer, and an output layer, interconnected by weights (to be determined) which are represented by the arrows between the layers. Apparently, information moves forward in one direction, from the input nodes, through the hidden nodes, and to the output nodes.

In our settings, the inputs $\hat{x}^{(1)}$, ..., $\hat{x}^{(N_i)}$ can be the state estimates from the sensors, and

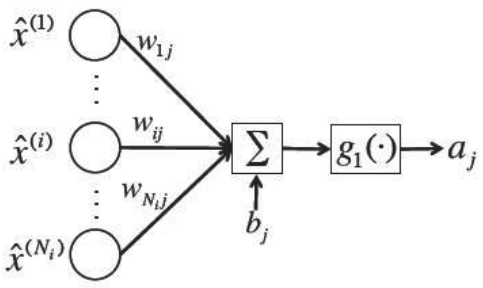


Figure 6.2: Weights from N_i inputs to the hidden node j

the outputs $\hat{x}_F^{(1)}$, ..., $\hat{x}_F^{(N_o)}$ are the global (fused) state estimates. There is also a bias unit (not shown in the figure) that is connected to each node in addition to the input nodes. The output of the j^{th} hidden node, a_j, is given by

$$a_j = g_1\left(\sum_{i=1}^{N_i} w_{ij}\hat{x}^{(i)} + b_j\right), \tag{6.1}$$

where the parameters w_{ij} and b_j are typically called the weights and biases, respectively. $\hat{x}^{(i)}$ is an input feature (e.g., a state estimate from a sensor), and $g_1(\cdot)$ is a nondecreasing function called the activation function, which is typically a bounded function such as the sigmoid. A simple diagram illustrating this node function is shown in Fig. 6.2. If we concatenate all of the hidden node outputs a_j into a vector $\mathbf{a} = [a_1, ..., a_L]^T$, then we can write the hidden node outputs as

$$\mathbf{a} = g_1(\mathbf{W}_H^T\hat{\mathbf{x}} + \mathbf{b}_H), \tag{6.2}$$

where $\mathbf{W}_H = [w_{ij}]_{N_i \times L}$ is the matrix of weights whose transpose is multiplied by the input vector $\hat{\mathbf{x}} = [x^{(1)}, ..., x^{(N_i)}]^T$, and $\mathbf{b}_H = [b_1, ..., b_L]^T$ is a vector of the biases for each hidden node. The fused output of our network, $\hat{\mathbf{x}}_F$, which is an N_o-dimensional vector, is then given by

$$\hat{\mathbf{x}}_F = g_2(\mathbf{W}_o^T\mathbf{a} + \mathbf{b}_o), \tag{6.3}$$

where $\mathbf{W}_o = [w_{ij}^o]_{L \times N_o}$ is another weight matrix, $\mathbf{b}_o = [b_1^o, ...b_{N_o}^o]^T$ is the vector of biases for each output, and $g_2(\cdot)$ is another activation function.

6.2.2 Backpropagation

When the target outputs are available, a well-known approach to determining the neural network parameters is called backpropagation. Backpropagation is based on gradient descent; the weights are initialized with random values and are iteratively updated to reduce the error (according to some user-defined error function, e.g., the mean-squared error). Once the network parameters are learned (from training data), new inputs can simply be fed into the neural network to obtain fused outputs.

The Levenberg-Marquardt (LM) algorithm [27] is a backpropagation method that is used in this study to train the ANNs as it can be implemented efficiently and is considered to be one of the faster training methods with relatively good performance. It is a second-order method in that it uses both the first and second derivatives of the error function to find a set of optimum weights. The formulation is as follows. Assume we want to minimize some function $S(\mathbf{w})$ with respect to a r-dimensional parameter vector $\mathbf{w}$. Then the Gauss-Newton method for updating $\mathbf{w}$, which is an iterative method that uses the first and second derivatives of a function to find a point where the derivative is zero, would be

$$\Delta \mathbf{w} = -[\nabla^2 S(\mathbf{w})]^{-1} \nabla S(\mathbf{w}), \tag{6.4}$$

where $\nabla^2 S(\mathbf{w})$ and $\nabla S(\mathbf{w})$ are the Hessian and the gradient, respectively, of $S(\mathbf{w})$. If we let $S(\mathbf{w})$ be a sum of squares function over m training patterns, e.g.,

$$S(\mathbf{w}) = \sum_{k=1}^{m} (y^{(k)} - f(\hat{\mathbf{x}}^{(k)}, \mathbf{w}))^2 = \sum_{k=1}^{m} (e_k(\mathbf{w}))^2 = \mathbf{e}(\mathbf{w})^T \mathbf{e}(\mathbf{w}), \tag{6.5}$$

where $y^{(k)}$ is the true target state in the k^{th} training pattern, $e_k(\mathbf{w}) = y^{(k)} - f(\hat{\mathbf{x}}^{(k)}, \mathbf{w})$ and

$\mathbf{e}(\mathbf{w}) = [e_1(\mathbf{w}), ..., e_m(\mathbf{w})]^T$, then we can approximate the Hessian and the gradient with the Jacobian, $\mathbf{J}$, of the error vector $\mathbf{e}(\mathbf{w})$ and rewrite Eq. (6.4) as

$$\Delta\mathbf{w} = -(\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{e}(\mathbf{w}), \tag{6.6}$$

where

$$\mathbf{J} = \begin{bmatrix} \frac{\partial e_1(\mathbf{w})}{\partial w_1} & \cdots & \frac{\partial e_1(\mathbf{w})}{\partial w_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial e_m(\mathbf{w})}{\partial w_1} & \cdots & \frac{\partial e_m(\mathbf{w})}{\partial w_d} \end{bmatrix}. \tag{6.7}$$

The Levenberg-Marquardt modification[1] to the Gauss-Newton method is

$$\Delta\mathbf{w} = -(\mathbf{J}^T\mathbf{J} + \eta\mathbf{I})^{-1}\mathbf{J}^T\mathbf{e}(\mathbf{w}) \tag{6.8}$$

where $\mathbf{I}$ is the identity matrix, and $\eta > 0$ is a damping factor, which is adjusted at each iteration of the weight update. If a step (i.e., weight update) results in an increased $S(\mathbf{w})$, then η is multiplied by some factor ν, and if a step results in a decreased $S(\mathbf{w})$, then η is divided by ν. The Jacobian of $\mathbf{e}(\mathbf{w})$ can be computed using the backpropagation approach as described in [27], where $\mathbf{w}$ is a vector containing all of the neural network parameters as follows:

$$\mathbf{w} = [\mathbf{W}_H(1,1), \mathbf{W}_H(1,2), ..., \mathbf{W}_H(L,N_i), \mathbf{b}_H(1), ..., \mathbf{b}_H(L), \mathbf{W}_o(1,1), ..., \mathbf{b}_o(N_o)]^T. \tag{6.9}$$

If we have N_i network inputs, L hidden nodes, and N_o network outputs, then the dimension of $\mathbf{w}$ is $d = L(N_i + 1) + N_o(L + 1)$. In this study, the state estimates from each sensor are used as a network input, so $N_i = Ns$ since there are N sensors generating s-dimensional

[1]There are other variations of the algorithm presented here. For example, there are different ways to select the initial damping factor η and increase/decrease its value. Also, the matrix consisting of only the diagonal elements of $\mathbf{J}^T\mathbf{J}$ may be used in place of the identity matrix $\mathbf{I}$.

state estimates, and $N_o = s$ so that the neural network outputs a s-dimensional fused state estimate.

6.3 Improving the Generalizability of the ANN Training

Overfitting is one of the most critical issues during neural network training, or more broadly, for any learning-based design. The error on the training set is driven to a small value, but when new testing data are input to the learned network, the error can be potentially large. In other words, the network has memorized the training examples, but it has not learned to generalize to new situations. In this section, we consider a few techniques for improving the generalizability of the LM-based ANN training.

6.3.1 Multiple ANNs

It is preferable to train several networks to ensure that a network with good generalization is found. Simple as it sounds, using multiple neural networks to train the same set of data[2] and averaging their outputs might yield superior performance by diversifying the training process. Typically each backpropagation training session starts with different initial weights and biases and these conditions may lead to very different solutions for the same problem. In addition, the network structure can be diversified as well by using a different number of hidden nodes or even hidden layers. Just a slight change in the structure would result in a different neural network with a completely new set of parameters. This approach can be especially helpful for a small and noisy dataset.

[2]In our setting, only a small set of training data from previous field tests are available and the testing stage occurs in the form of real-time tracking, hence we don't pursue cross-validation [3] approaches where data are iteratively divided into training, validation, and testing sets to improve the generalization capabilities.

6.3.2 Bayesian Regularization

Another method for improving generalization capabilities of an learning algorithm is called regularization. This involves modifying the performance function, which is normally chosen to be the sum of squares of the network errors on the training set, as shown in Eq. (6.5). An additional term is added to the original performance function, usually in the form of a cost or penalty function for complexity, such as restrictions for smoothness or bounds on the vector space norm [26].

MacKay [57] proposed a Bayesian framework to overcome the problem in interpolating noisy data, which can be directly applied to the neural network learning problem. In this framework, the weights and biases of the network are assumed to be random variables with specified distributions. The regularization parameters are related to the unknown variances associated with these distributions. Another goal is to reduce the effective number of parameters actually used by the model - in this case, the number of network weights actually needed to solve a particular problem. Hence, MacKay's Bayesian regularization expands the original sum-of-squares cost function to search for the minimum error using minimum weights by introducing two Bayesian hyperparameters, α and β, to balance the dual needs during the learning process. In particular, the objective function to be minimized is in the form of

$$S(\mathbf{w}) = \frac{\beta}{2}\sum_{k=1}^{m}(y^{(k)} - f(\hat{\mathbf{x}}^{(k)}, \mathbf{w}))^2 + \frac{\alpha}{2}\sum_{k=1}^{d} w_i^2 = \beta \cdot SSE + \alpha \cdot SSW. \tag{6.10}$$

It can be seen that in addition to the sum of squared errors term, an additional term, namely, the sum of squared weights (SSW), is added to the objective function. The SSW is simply the square of the $\mathbf{L}^2$ norm of the d-dimensional weight vector $\mathbf{w}$ introduced in Eq. (6.9). The hyperparameters are updated as

$$\alpha = \frac{\gamma}{2 \cdot SSW}, \text{ and } \beta = \frac{m - \gamma}{2 \cdot SSE}, \tag{6.11}$$

where

$$\gamma = d - \alpha \ \mathrm{tr}((\mathbf{J}^T\mathbf{J})^{-1}) \tag{6.12}$$

is considered as the number of "effective" parameters (weights and biases) being used by the network, thus giving an indication on how complex the network should be. These parameter update steps[3] can be easily incorporated into each iteration of the LM weight update equation as in Eq. (6.8).

6.3.3 Regularization Using Error Covariance Matrices

It is noted that the sensors also provide additional information regarding the state estimates: the error covariances. It is still an open question of how to best utilize this information, if at all, when designing or using these nonlinear fusers. It is proposed here that these error covariance estimates can be used when training the neural network to improve the neural network's generalization capability. If we assume that the state estimates from the sensors are equal to the true states plus noise, that is,

$$\hat{\mathbf{x}}_i = \mathbf{x} + \mathbf{n}_i \tag{6.13}$$

where i is the sensor index, and $\mathbf{n}_i$ is zero-mean white Gaussian noise with covariance $\mathbf{P}_i$, then in fusing these state estimates, the neural network will also transform and fuse the noise. We can therefore try to further reduce the variance of the output in addition to the overall error by adding the output variance to the objective function that we minimize when determining the neural network parameters. But if one were to use a nonlinear activation function such as the tangent hyperbolic sigmoid function, $f(x) = \frac{2}{1+e^{-2x}} - 1$, then the variance becomes

[3]Poland [66] provided another way to update α as

$$\alpha = d/(2.0 \cdot SSW + \ \mathrm{tr}((\mathbf{J}^T\mathbf{J})^{-1}))$$

which has been shown to provide more smooth and robust updates compared to Eq. (6.11).

quite difficult to determine analytically. However, from [18], we have the following upper bound on the variance of the function of a random variable X, if $X \sim \mathcal{N}(\mu, \sigma^2)$:

$$\mathbb{V}\text{ar}[g(X)] \leq \sigma^2 \mathbb{E}[g'(X)]^2 \tag{6.14}$$

We can write the output $\hat{\mathbf{x}}_F$ as a function of the noise $\mathbf{n} = [\mathbf{n}_1^T, ..., \mathbf{n}_N^T]^T$, where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{P})$. $\mathbf{P}$ is a block diagonal matrix where the blocks are $\mathbf{P}_1$ through $\mathbf{P}_N$:

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_1 & \mathbf{0}_{s\times s} & \cdots & \mathbf{0}_{s\times s} \\ \mathbf{0}_{s\times s} & \mathbf{P}_2 & \cdots & \mathbf{0}_{s\times s} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{s\times s} & \cdots & \mathbf{0}_{s\times s} & \mathbf{P}_N \end{bmatrix} \tag{6.15}$$

where s is the number of states in the true target state $\mathbf{x}$. For the ANN fuser, if we let the activation function for each hidden node be $g_1(x) = \frac{2}{1+e^{-2x}} - 1$ and the output activation function $g_2(x) = x$, then the ANN fuser output would be given by

$$\begin{aligned} \hat{\mathbf{x}}_F &= \mathbf{W}_o^T \left(\frac{2}{1 + e^{-2(\mathbf{W}_H^T \hat{\mathbf{x}} + \mathbf{b}_H)}} - 1 \right) + \mathbf{b}_o \\ &= \mathbf{W}_o^T \left(\frac{2}{1 + e^{-2(\mathbf{W}_H^T \mathbf{x} + \mathbf{b}_H)} e^{-2\mathbf{W}_H^T \mathbf{n}}} - 1 \right) + \mathbf{b}_o \end{aligned} \tag{6.16}$$

Now if we let $\boldsymbol{\theta} = \mathbf{W}_H^T \mathbf{n}$, which is simply a linear transformation of the Gaussian random variable $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{P})$, then $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \mathbf{W}_H^T \mathbf{P} \mathbf{W}_H)$. Therefore, we can write Eq. (6.14) as

$$\mathbb{V}\text{ar}[\hat{\mathbf{x}}_F(\boldsymbol{\theta})] \leq \mathbb{E}\left[\frac{\partial \hat{\mathbf{x}}_F(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right] \cdot \mathbf{W}_H^T \mathbf{P} \mathbf{W}_H \cdot \mathbb{E}\left[\frac{\partial \hat{\mathbf{x}}_F(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right]^T \tag{6.17}$$

However, even determining $\mathbb{E}\left[\partial \hat{\mathbf{x}}_F(\boldsymbol{\theta}) / \partial \boldsymbol{\theta}\right]$ is still quite complicated. Suppose $\boldsymbol{\theta}_0 = \mathbf{W}_H^T \mathbf{x} + \mathbf{b}_H$, then we can obtain the following relationship:

$$\mathbb{V}\text{ar}[\hat{\mathbf{x}}_F(\boldsymbol{\theta})] < 4 \mathbf{W}_o^T \exp(\boldsymbol{\Theta}) \mathbf{W}_H^T \mathbf{P} \mathbf{W}_H \exp(\Theta) \mathbf{W}_o, \tag{6.18}$$

in which $\mathbf{\Theta} = 8\mathbf{W}_H^T\mathbf{P}\mathbf{W}_H \odot \mathbf{I}_L + 4\boldsymbol{\theta}_0^D$, where $\odot$ denotes the Hadamard product, $\mathbf{I}_L$ is the $L \times L$ identity matrix, $\boldsymbol{\theta}_0^D$ is the diagonal matrix whose diagonal elements are the elements of the vector $\boldsymbol{\theta}_0$, and finally $\exp(\cdot)$ denotes the exponential of a diagonal matrix. The proof of Eq. (6.18) can be found at the end of this chapter. However, this multiplicative relationship is still not neat enough for use in the additional term for error regularization, considering the complexity of the matrix Θ. Therefore, to simplify, an extended form of the term $\mathbf{W}_H^T\mathbf{P}\mathbf{W}_H$ will be used to add to the error function for minimization. We can modify the LM algorithm described in the previous section to incorporate this additional term using a tradeoff parameter λ:

$$S(\mathbf{w}) = \sum_{k=1}^{m}(y^{(k)} - f(\hat{\mathbf{x}}^{(k)}, \mathbf{w}))^2 + \lambda \mathbf{w}^T \mathbf{\Sigma} \mathbf{w} \tag{6.19}$$

where

$$\mathbf{\Sigma} = \left[\begin{array}{c|c} \mathbf{P}_R & \mathbf{0}_{NLs\times(d-NL)s} \\ \hline \mathbf{0}_{(d-NL)s\times NLs} & \mathbf{0}_{(d-NL)s\times(d-NL)s} \end{array}\right], \tag{6.20}$$

and $\mathbf{P}_R \in \Re^{(NLs)\times(NLs)}$ is a block diagonal matrix with each block consisting of the matrix $\mathbf{P}$, repeated L times (once for each hidden node):

$$\mathbf{P}_R = \begin{bmatrix} \mathbf{P} & \mathbf{0}_{Ns\times Ns} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{P} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{P} \end{bmatrix} \tag{6.21}$$

Recall that N is the number of sensors and s is the number of states, so each block diagonal matrix $\mathbf{P}$ is of size $Ns \times Ns$. Following the LM approach, when we update $\mathbf{w}$ with $\Delta\mathbf{w}$, we

get the following update to our error function $S(\mathbf{w})$:

$$\begin{aligned}
& S(\mathbf{w}+\Delta\mathbf{w}) \\
=& \sum_{k=1}^{m}(y^{(k)} - f(\hat{\mathbf{x}}^{(k)}, \mathbf{w}+\Delta\mathbf{w}))^2 + \lambda(\mathbf{w}+\Delta\mathbf{w})^T\boldsymbol{\Sigma}(\mathbf{w}+\Delta\mathbf{w}) \\
=& \sum_{k=1}^{m}(e_k(\mathbf{w}+\Delta\mathbf{w}))^2 + \lambda(\mathbf{w}+\Delta\mathbf{w})^T\boldsymbol{\Sigma}(\mathbf{w}+\Delta\mathbf{w}) \\
=& \left[\mathbf{e}(\mathbf{w}+\Delta\mathbf{w})\right]^T \mathbf{e}(\mathbf{w}+\Delta\mathbf{w}) + \lambda(\mathbf{w}+\Delta\mathbf{w})^T\boldsymbol{\Sigma}(\mathbf{w}+\Delta\mathbf{w}) \\
\approx& \left[\mathbf{e}(\mathbf{w}) + \nabla\mathbf{e}(\mathbf{w})\Delta\mathbf{w}\right]^T \left[\mathbf{e}(\mathbf{w}) + \nabla\mathbf{e}(\mathbf{w})\Delta\mathbf{w}\right] + \lambda(\mathbf{w}+\Delta\mathbf{w})^T\boldsymbol{\Sigma}(\mathbf{w}+\Delta\mathbf{w})
\end{aligned} \tag{6.22}$$

where the following substitutions are made: $y^{(k)} - f(\hat{\mathbf{x}}^{(k)}, \mathbf{w}+\Delta\mathbf{w}) = e_k(\mathbf{w}+\Delta\mathbf{w})$ (line 1 to line 2), and $\mathbf{e}(\mathbf{w}+\Delta\mathbf{w}) = [e_1(\mathbf{w}+\Delta\mathbf{w}), ..., e_m(\mathbf{w}+\Delta\mathbf{w})]^T$ (line 2 to line 3). In the fourth line of Eq. (6.22), we approximate and substitute $\mathbf{e}(\mathbf{w}+\Delta\mathbf{w})$ with its first-order Taylor expansion about $\mathbf{w}$. Furthermore, we know that at the minimum of $S(\mathbf{w}+\Delta\mathbf{w})$, the gradient with respect to $\Delta\mathbf{w}$ is zero, so we have

$$\begin{aligned}
\frac{\partial S(\mathbf{w}+\Delta\mathbf{w})}{\partial\Delta\mathbf{w}} &\approx 2\left[\nabla\mathbf{e}(\mathbf{w})\right]^T\left[\mathbf{e}(\mathbf{w}) + \nabla\mathbf{e}(\mathbf{w})\Delta\mathbf{w}\right] + 2\lambda\boldsymbol{\Sigma}(\mathbf{w}+\Delta\mathbf{w}) \\
&= 2\mathbf{J}^T\mathbf{e}(\mathbf{w}) + 2\mathbf{J}^T\mathbf{J}\Delta\mathbf{w} + 2\lambda\boldsymbol{\Sigma}\mathbf{w} + 2\lambda\boldsymbol{\Sigma}\Delta\mathbf{w}
\end{aligned} \tag{6.23}$$

where $\mathbf{J}$ is the Jacobian of $\mathbf{e}(\mathbf{w})$.

The weight update equation can then be computed as: $\Delta\mathbf{w} = -(\mathbf{J}^T\mathbf{J} + \lambda\boldsymbol{\Sigma})^{-1}(\mathbf{J}^T\mathbf{e}(\mathbf{w}) + \lambda\boldsymbol{\Sigma}\mathbf{w})$, and with the LM modification to include the damping factor, we obtain the final weight update equation which incorporates the error covariance estimates into the training using the LM method:

$$\Delta\mathbf{w} = -(\mathbf{J}^T\mathbf{J} + \lambda\boldsymbol{\Sigma} + \eta\mathbf{I})^{-1}(\mathbf{J}^T\mathbf{e}(\mathbf{w}) + \lambda\boldsymbol{\Sigma}\mathbf{w}) \tag{6.24}$$

6.4 Training and Testing with Missing Data

Having provided the training algorithm and techniques for generalization, we now focus on the communication and computation constraints and their impact on the learning process. First, the communication link loss and delay further reduces the amount of data that can be effectively used for training and testing purposes, which must be accounted for by the fusion center in devising efficient learning-based fusers.

6.4.1 Data Loss during Training

As training is performed based on data gathered from historical field tests, usually the true target states are obtained via some other sources that are not subject to the same link loss and delay conditions as from the sensors, whereas some sensor estimates may have never arrived. As such, the fusion center simply has a smaller set of training data that are input to an ANN. Suppose the link-level loss rate is p_L across all training patterns, then the effect this loss has on training can be interpreted in two different ways. First, the actual number of available training patterns $\tilde{m}$ is often less than m, and thus the training objective simply uses SSE terms for the available $\tilde{m}$ patterns. In other words, in Eq. (6.5), based on the actual pattern availability, we have

$$S(\mathbf{w}) = \sum_{k=1}^{\tilde{m}} (e_k(\mathbf{w}))^2 = \tilde{\mathbf{e}}(\mathbf{w})^T \tilde{\mathbf{e}}(\mathbf{w}), \tag{6.25}$$

where $\tilde{\mathbf{e}}$ is the error vector containing $\tilde{m}$ elements. Alternatively, the number of available patterns follows the following binomial distribution: $\tilde{m} \sim \mathcal{B}(m(1-p_L), mp_L(1-p_L))$. As a result, the original SSE term in Eq. (6.5) can now be expressed as

$$S(\mathbf{w}) = \sum_{k=1}^{m} i_k (e_k(\mathbf{w}))^2 = \mathbf{e}(\mathbf{w})^T \mathbf{\Lambda} \mathbf{e}(\mathbf{w}), \tag{6.26}$$

where i_k is the indicator function in which $i_k = 1$ with probability $1-p_L$ and 0 otherwise, and the diagonal matrix $\mathbf{\Lambda} = \mathbf{i}^D$ where $\mathbf{i} = [i_1, i_2, ..., i_m]^T$. These results can be easily extended to patterns with different link loss rates.

6.4.2 Data Loss and Delay during Testing

The testing phase of the learning process, namely, to apply the learned ANN parameters to new sensor inputs, coincides with the actual online tracking process. The loss and delay inherent over the communication link, as a result, plays a somewhat different role compared to the off-line training phase.

In most tracking tasks that impose nearly real-time performance, the fusion center is required to finalize its fused state within a very tight deadline, by which time one or more sensor estimates may have not yet arrived (although they might arrive later). Thus, in contrast to the off-line training phase, the communication delay is a major limiting factor in the desired performance.

The structure of the ANN-based learning requires that the input data be complete in order to generate the desired output. Whereas it is in general difficult to interpolate missing values in the training phase as the underlying time-domain relationship between the training data is not readily available, it is possible that the fusion center uses adjacent sensor estimates that are available to interpolate the missing ones, based on the knowledge it has on the possible trajectory of the underlying target being tracked. Therefore, when some of the inputs are missing, the fusion center can use prediction and retrodiction (when applicable) techniques that utilize previous and subsequent sensor estimates, respectively, to manually fill in the missing inputs. Of course, should all other methods fail (as under extremely lossy link conditions), the fusion center can still bypass the neural network altogether by using prediction on its previously fused states to generate the immediate fused value, although this is likely to compromise the accuracy performance by a large margin.

6.5 Sensor Bias and Error Regularization

Another concern arising out of the long-haul tracking is the sensor information quality. Notably, sensor biases could potentially degrade the fusion performance. To be answered are questions such as how the ANN-based fuser would perform with the presence of variable bias levels and whether the learning process could potentially improve the training and testing data quality. In this section, we discuss bias-related issues pertaining to the ANN training and testing.

An important fact about estimation biases is that a sensor i might not be aware of its biases. Sometimes the biases are an integral part of the filtering process, due to linearization and coordinate conversion; while other times it is due to poor calibration or environmental factors that result in sensor measurement biases. As such, the error covariance matrix $\mathbf{P}_i$ – which is also supplied to the fusion center – is likely to be an optimistic measure of the actual estimation error. Fortunately, thanks to the available true target states during the training phase, the fusion center can evaluate the potential estimation biases from the sensor data and "correct" the error covariance matrix for its training.

To illustrate, consider a generic sensor estimate as $\hat{\mathbf{x}}_i$ whereas the true target state is $\mathbf{x}$. Then the error covariance matrix is defined as $\mathbf{P}_i = \mathbb{E}[(\hat{\mathbf{x}}_i - \mathbf{x})(\hat{\mathbf{x}}_i - \mathbf{x})^T]$. Now suppose $\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_u + \mathbf{b}_i$ where $\hat{\mathbf{x}}_u$ is an unbiased estimate for $\mathbf{x}$ and $\mathbf{b}_i$ is the bias vector, then we have

$$\begin{aligned}
\mathbf{P}_i &= \mathbb{E}[(\hat{\mathbf{x}}_i - \mathbf{x})(\hat{\mathbf{x}}_i - \mathbf{x})^T] \\
&= \mathbb{E}[(\hat{\mathbf{x}}_u - \mathbf{x} + \mathbf{b}_i)(\hat{\mathbf{x}}_u - \mathbf{x} + \mathbf{b}_i)^T] \\
&= \mathbb{E}[(\hat{\mathbf{x}}_u - \mathbf{x})(\hat{\mathbf{x}}_u - \mathbf{x})^T] + \mathbb{E}[\mathbf{b}_i\mathbf{b}_i^T] + 2\mathbb{E}[(\hat{\mathbf{x}}_u - \mathbf{x})\mathbf{b}_i^T] \\
&= \mathbf{P}_u + \mathbb{E}[\mathbf{b}_i\mathbf{b}_i^T] + 2\mathbb{E}[(\hat{\mathbf{x}}_u - \mathbf{x})\mathbf{b}_i^T] \\
&\approx \mathbf{P}_u + \overline{\mathbf{b}_i\mathbf{b}_i^T}, \qquad (6.27)
\end{aligned}$$

where $\mathbf{P}_u$ is the error covariance matrix for an unbiased estimate. In the extreme case

when a sensor is fully oblivious to its bias, this is the matrix it communicates to the fusion center. The last line has used two approximations, the first one being that the bias is largely uncorrelated with the true target state, and the second being a running average (as denoted by the overline) is used for the expectation of the outer (tensor) product of the bias vector and itself.

In an ideal scenario, the estimation bias at a sensor is deterministic and consistent, and the fusion center can easily obtain the bias vector and subtract it from the associated sensor estimates during the testing stage. However, with more complex forms of bias, instead of correcting the sensor estimates, the fusion center may want to use Eq. (6.27) for error regulation so that the updated matrices more closely match the actual error levels of the sensor estimates. Doing so might be of better benefit to the prediction (and retrodiction) when the fusion center encounters missing data during the online tracking. The performance comparison using different error regularization methods will be shown in the next section.

6.6 Performance Evaluation

In this section, the performance of the ANN-based fusers is evaluated for a coasting ballistic target tracking application via simulations. Different configurations during the learning process are considered, so are variable communication and computation constraints. The tracking performance with ANN-based fusers is also compared against that using the track-to-track fuser (T2TF) and the fast-CI fuser that have been studied extensively in this dissertation.

6.6.1 Simulation Models

We model two sensors tracking a ballistic target in the exo-atmospheric coast phase whose trajectory is determined by a nonlinear state-space model. The target state and sensor measurement models have been presented in Section 3.5.2. As in previous chapters, an

independent link loss p_L and a delay pdf $f(t) = \exp\left((T-t)/\mu_D\right)/\mu_D$ for $t \geq T$ (where T is the fixed initial delay and μ_D is the mean of the additional random delay) is considered. We have $T = 0.5$ s and $\mu_D = 0.3$ s. The fusion deadline D_F describes the time by which the fusion center must have combined the individual sensor estimates and generated a fused estimate.

The initial states of the training and test targets are randomly generated from a normal distribution with the mean set to the following (in km for position, and km/s for velocity):

$$\begin{aligned}\mathbf{x}(0) &= [x(0) \quad \dot{x}(0) \quad y(0) \quad \dot{y}(0) \quad z(0) \quad \dot{z}(0)]^T \\ &= [71.31,\ 0.946,\ 3794.5,\ 3.577,\ 5413.0,\ -5.676]^T,\end{aligned}$$

with a standard deviation of 500 m and 5 m/s for each position and velocity component, respectively. In our default setting, two target trajectories are available for training. Measurement data are generated according to the sensor measurement model introduced in Eqs. (3.55)-(3.57), and state estimates (and the associated error covariance matrices) – also available in the training data set – are computed from these measurements for each trajectory. Data are available for each trajectory segment that lasts 30 seconds.

The ANN used for training and testing has one hidden layer with a number of hidden nodes. The number of hidden nodes needed depends on the complexity of the function we are trying to estimate. Using too few hidden nodes may yield a poor approximation to the actual function. Using too many hidden nodes results in overfitting the data so that while the neural network may precisely give the desired outputs for the training data, it may not generalize well to unseen data. Unfortunately, there is no precise method that provides the optimal number of hidden nodes needed to properly model the data. But a good rule of thumb [31] is that it lies between the number of input and output nodes. Hence, we select three hidden nodes in our default setting. The tangent hyperbolic sigmoid function is used as the activation function from the input to hidden layers.

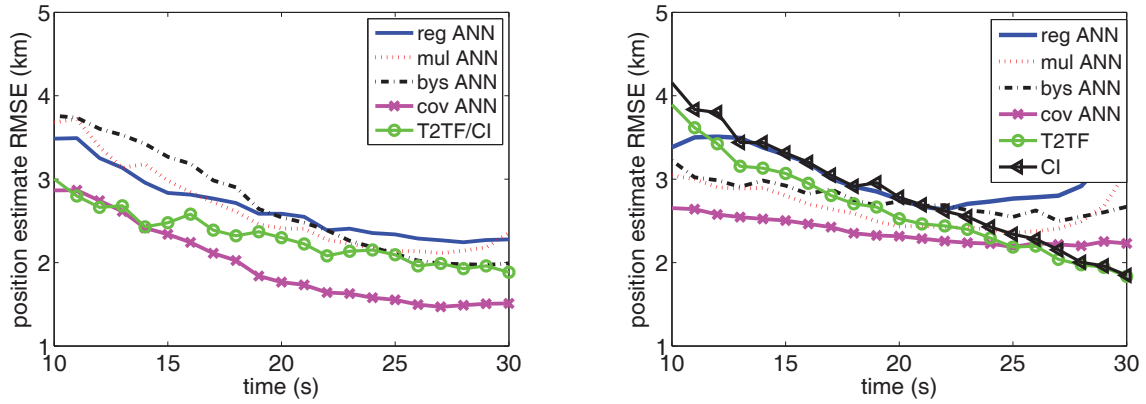
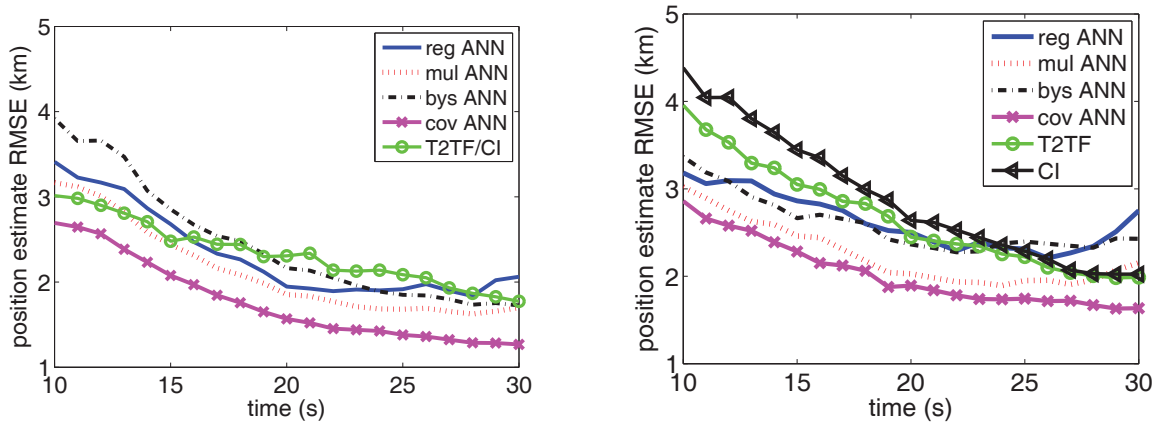


Figure 6.3: Position estimate RMSE over time with ANN-based fusers as well as T2TF and fast-CI fuser, two training trajectories, three hidden nodes: (a) $p_L = 0$, $D_F = 2$ s; (b) $p_L = 0.5$, $D_F = 1$ s

6.6.2 Fusion Performance without Sensor Biases

In Fig. 6.3(a), tracking performances using variations of the ANN-based fuser, including the regular ANN ("reg-ANN"), multiple ANN with 5 component ANNs of the same structure ("mul-ANN"), ANN with Bayesian regularization ("bys-ANN"), and ANN with error covariance regularization ("cov-ANN"), along with that of the T2TF/CI fuser[4] are plotted. From these plots, we observe that although the original ANN fuser doesn't yield estimates as good as those from the T2TF/CI by using only two available training trajectories, some of the enhanced variations do, especially "cov-ANN". In practice, Bayesian regularization is often applied when combined with other techniques, such as multiple ANNs; as a stand-alone technique, its performance is not always superior to that of the regular ANN as the Bayesian formulation depends on how the trade-off parameters are selected. From our simulations, there are a total of 33 weights to be determined in the regular ANN, and after applying regularization about 3 or 4 parameters are removed, which means the performance of the two should not differ by much under normal circumstances. In general, the multiple ANN approach would outperform the original ANN-based fuser by variable levels, depending on

[4]When $p_L = 0$, due to the lack of process noise, these two fusers yield the same performance.

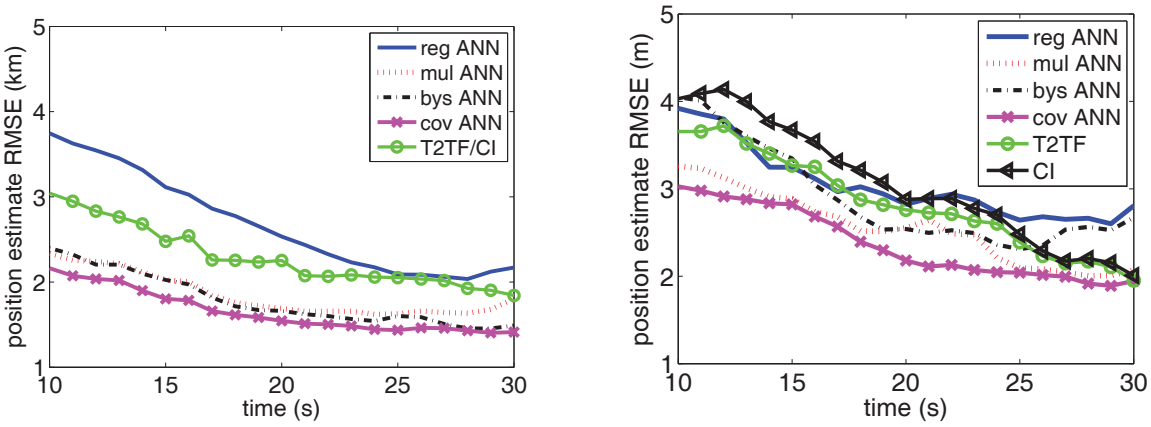


(a) (b)

Figure 6.4: Position estimate RMSE over time with ANN-based fusers as well as T2TF and fast-CI fuser, five training trajectories, three hidden nodes: (a) $p_L = 0$, $D_F = 2$ s; (b) $p_L = 0.5$, $D_F = 1$ s

such factors as how the weights are initialized in each component network. The covariance error regularization method can effectively use the extra information from the covariance matrices supplied by the sensors and provide improved performance compared to all other techniques.

As the link-level loss goes up, there's an increasing chance that the fusion center can't receive both sensor estimates for any time epoch during the testing stage. In addition, the training data are also likely to be obtained from past testing data – that experienced similar losses – with the addition of the true target states being revealed later, but not the sensor estimates. The communication delay also reduces the chance that a message is successfully received by the fusion center by a certain deadline. Therefore, with incomplete sensor data, the fusion center could use predicted estimates it derives for the individual sensors, and once these predicted values are obtained, the inputs to the ANNs are complete. Training data may contain such predicted estimates too. As a result, all the training algorithms can be run with complete data, even when a portion of the training data are not from the sensors, but rather derived by the fusion center during earlier tracking. Fig. 6.3(b) shows the performance with 50% sensor data loss in both training and testing data. In the previous



(a) (b)

Figure 6.5: Position estimate RMSE over time with ANN-based fusers as well as T2TF and fast-CI fuser, two training trajectories, six hidden nodes: (a) $p_L = 0$, $D_F = 2$ s; (b) $p_L = 0.5$, $D_F = 1$ s

case, a fusion deadline of $D_F = 2$ s can effectively reduce any effect of early cutoff; in contrast, here the deadline $D_F = 1$ s, resulting in even more unavailable original sensor data by the deadline. Comparing different cases, there is an increase in the tracking error in non-learning based T2TF and CI fusers compared to their respective no-loss case. However, the ANN learning-based approaches are largely able to retain most of their respective performance under the lossless situation, thereby demonstrating the major benefit of adopting these fusers in counteracting the negative effect of communication constraints.

Next, we explore how the learning-based fusion performance can be potentially improved with the use of more training data and/or an ANN of a larger size. In Fig. 6.4, the plots for the same settings as before but with five trajectories are shown. From these plots, the error curves for ANN-based fusers are variably lower compared to those in Fig. 6.3, reflecting the major benefit of training more data. On the other hand, with the same amount of training data (two trajectories), we increase the number of hidden nodes from three to six, and the error plots are shown in Fig. 6.5. It can be seen here again that using more hidden nodes can, by and large, improve the error performance somewhat for both loss cases, although the training process becomes more complex with more network weights to be determined. But

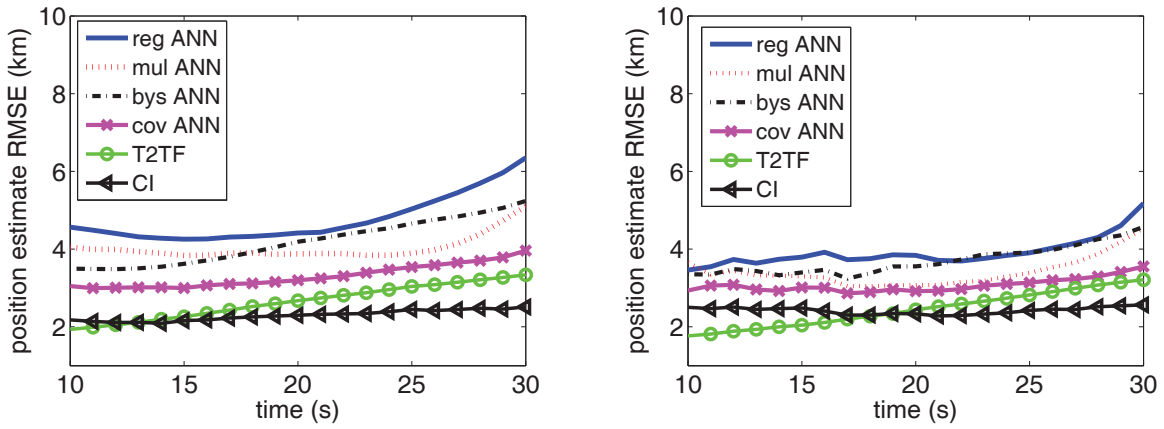


Figure 6.6: Position estimate RMSE over time with ANN-based fusers as well as T2TF and fast-CI fuser with unlearned bias: (a) $p_L = 0$, $D_F = 2$ s; (b) $p_L = 0.5$, $D_F = 1$ s

regardless of the chosen data or network size, a consistent observation is the superiority of the proposed covariance-based ANN fuser over others in terms of tracking accuracy.

6.6.3 Fusion Performance with Sensor Biases

We examine the cases where the training and/or testing data contain biased estimates. First, suppose in the test data, a positive uniform bias is added to the (unbiased) measurement noise at one of the sensors, with its mean magnitude set at 0.1% of the noise level. The fusion performance is shown in Fig. 6.6. Even such a small bias results in the error curves now trending upward across the board. Due to the mismatch between the training and testing data in terms of bias profiles, the unlearned bias (i.e., the bias not present in the training data) reduces the generalization capability of the trained parameters, resulting in elevated estimation errors across all learning-based cases. It's interesting to note that the learning-based fusers with more than half the original sensor data missing would yield even better tracking performance than those of the same type in the lossless case thanks to the reduced effect of the mismatch between the training and testing data.

Next, suppose in one of the training trajectories data share the same bias profile as

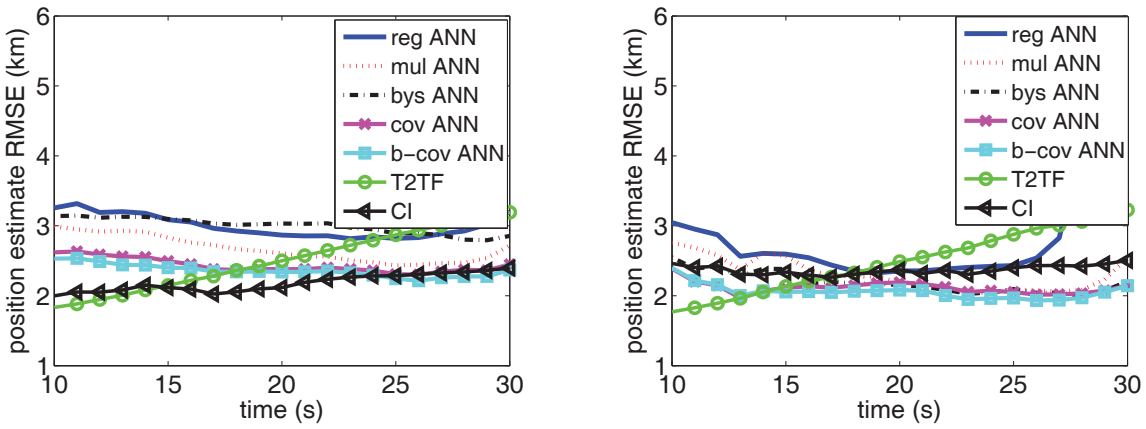


Figure 6.7: Position estimate RMSE over time with ANN-based fusers as well as T2TF and fast-CI fuser with bias: (a) $p_L = 0$, $D_F = 2$ s; (b) $p_L = 0.5$, $D_F = 1$ s

described above whereas estimates in the other remain unbiased. The fusion performance is shown in Fig. 6.7, where performance of the "cov ANN" under the bias correction method introduced earlier has been plotted as well ("b-cov ANN"). The improved generalization performance can now be seen for both loss settings compared to the previous case with bias mismatch. With an additional bias correction step, the covariance regularization method is able to output even better estimates, although this improvement is somewhat limited and can be reduced when a higher loss rate is encountered.

Finally, all others kept equal, the mean bias level is now increased to 0.2% of the unbiased measurement noise, and the estimation errors are plotted in Fig. 6.8. Whereas the T2TF and the CI fuser output increasingly error-prone estimates (more so for the T2TF), the ANN-based fusers are able to retain much of their error levels as in the previous case, thereby demonstrating their superiority in tracking with potentially more biased sensor data. Nevertheless, with such an elevated bias level and over half the original sensor data missing, the improvement in tracking accuracy over the non-learning fusers is reduced, as a result of the increased mismatch level (from extended prediction over biased estimates) between the training and testing data. Thus, from the above analysis, in order to improve the

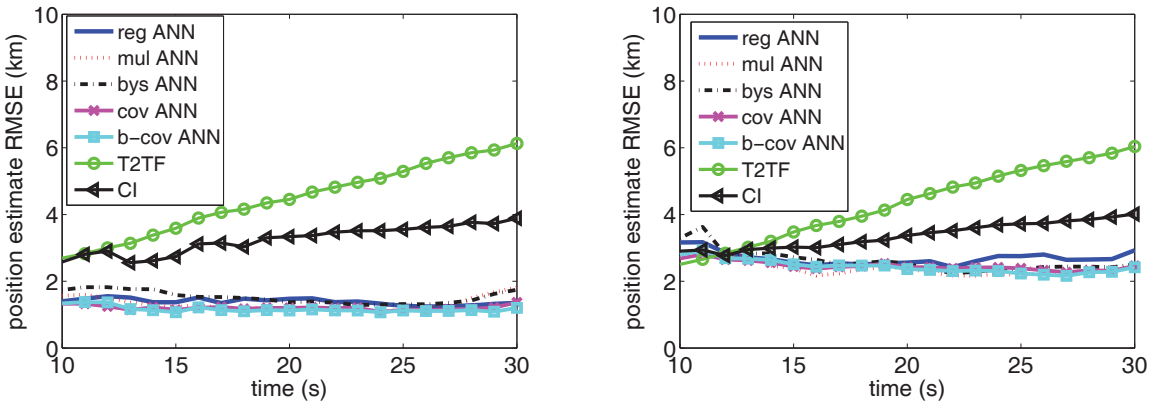


Figure 6.8: Position estimate RMSE over time with ANN-based fusers as well as T2TF and fast-CI fuser with a higher level of bias: (a) $p_L = 0$, $D_F = 2$ s; (b) $p_L = 0.5$, $D_F = 1$ s

generalizability of the training process, it is essential to guarantee the training and testing data match well in features such as bias levels.

6.7 Conclusion

In this chapter, we have provided a quantitative study on the potential benefits of artificial neural network learning-based fusers for sensor fusion in target tracking over long-haul sensor networks. In particular, the effects of variable communication link-level loss and delay conditions as well as sensor bias profiles on the performance of a variety of ANN implementations have been investigated. The extra covariance information provided by the sensors has been shown to provide better generalization guarantees in the presence of the above communication and computation uncertainties.

Proof of Eq. (6.18)

Proof. From Eq. (6.16), let

$$\boldsymbol{\xi}(\boldsymbol{\theta}) = \frac{2}{1 + e^{-2(\boldsymbol{\theta}+\boldsymbol{\theta}_0)}} - 1 \tag{6.28}$$

be the output of the hidden nodes. Then its derivative with respect to $\boldsymbol{\theta}$, $\partial\boldsymbol{\xi}(\boldsymbol{\theta})/\partial\boldsymbol{\theta}$, is a $L \times L$ diagonal matrix with entries $-2\left(1 + \exp(-2(\theta_k + \theta_{0,k}))\right)^{-2}$, $k = 1, 2, ..., L$, where θ_k and $\theta_{0,k}$ denote the k^{th} entry of the vector θ and θ_0, respectively. Next we focus on finding the expected value of $(1 + \exp(-2(\theta_k + \theta_{0,k})))^{-2}$:

$$\begin{aligned} & \mathbb{E}[(1 + \exp(-2(\theta_k + \theta_{0,k})))^{-2}] \\ < \;& \mathbb{E}[(\exp(-2(\theta_k + \theta_{0,k})))^{-2}] \\ = \;& \mathbb{E}[\exp(4(\theta_k + \theta_{0,k}))]. \end{aligned} \tag{6.29}$$

It can be shown that θ_k is a Gaussian random variable with $\theta_k \sim \mathcal{N}\left(0, (\mathbf{W}_{H,[:,k]})^T\mathbf{P}\mathbf{W}_{H,[:,k]}\right)$, where $\mathbf{W}_{H,[:,k]}$ denotes the k^{th} column of matrix $\mathbf{W}_H$. Consequently, $\exp(4(\theta_k + \theta_{0,k}))$ is a log-normal random variable. Recall the mean of a log-normal random variable $Y \sim \text{Lognormal}(\mu, \sigma^2)$ is $\exp(\mu + \sigma^2/2)$. Consequently, the mean of $\exp(4(\theta_k + \theta_{0,k}))$ is found to be $\exp(4\theta_{0,k} + 8(\mathbf{W}_{H,[:,k]})^T\mathbf{P}\mathbf{W}_{H,[:,k]})$. To aggregate the result for all entries, after some matrix manipulation, we have

$$\mathbb{E}\left[\partial\left(-\boldsymbol{\xi}(\boldsymbol{\theta})/2\right)/\partial\boldsymbol{\theta}\right] < \exp(8\mathbf{W}_H^T\mathbf{P}\mathbf{W}_H \odot \mathbf{I}_L + 4\boldsymbol{\theta}_0^D), \tag{6.30}$$

where $\odot$ denotes the Hadamard product, $\mathbf{I}_L$ is the $L \times L$ identity matrix, $\boldsymbol{\theta}_0^D$ is the diagonal matrix whose diagonal elements are the elements of the vector $\boldsymbol{\theta}_0$, and finally $\exp(\cdot)$ denotes the exponential of a diagonal matrix. Accounting for the coefficient -2 left out earlier, we let $\boldsymbol{\Theta} = 8\mathbf{W}_H^T\mathbf{P}\mathbf{W}_H \odot \mathbf{I}_L + 4\boldsymbol{\theta}_0^D$, plug everything into Eq. (6.17) and then obtain Eq. (6.18). □

Chapter 7

Conclusions

The purpose of this dissertation is to account for the communication and computation constraints in long-haul target tracking applications and to improve the overall tracking performance (accuracy and timeliness) using a suite of algorithms and schemes. More specifically, we have made contributions in the following aspects by studying

- A dynamic online selective fusion algorithm (PRODIC-RTS) that balances the dual requirements on tracking accuracy and timeliness [49, 52];
- A combined use of retransmission and retrodiction that speeds up the recovery of lost or delayed sensor information [53–55];
- Staggered and asynchronous estimation and fusion schedules that opportunistically use the network delay to mitigate the negative effect of cumulative process noise and target uncertainty on tracking [48, 50];
- Feedback of fused information back to the sensors to improve the data quality, counter the bias, and improve the overall tracking performance [47, 51];
- Artificial neural network learning-based fusers that utilize historical tracking data to reduce the effect of loss/delay and bias on new sensor data [46].

The research contained in this dissertation combines algorithm design, case studies, and evaluation of tracking performance via mathematical analysis as well as numerical and simulation studies. We expect the research results to be of interest to those working in the field and can guide future development of advanced estimation and fusion systems.

The research in this dissertation can be extended in many ways. The selective fusion algorithm can be incorporated into a system with more complex target and sensor dynamics and the metric itself be transformed into a range of values that account for such additional uncertainties. The retransmission scheduling can be studied in a cross-layer context where the effect of retransmission on higher-layer performance is considered along with the estimation error metric. Both staggered sensing scheduling and information feedback can be studied in a more dynamic environment where the fusion center keeps track of the evolving system dynamics and make adaptive changes to its existing sensing schedules or feedback configurations/schedules. The learning-based fusers can be reexamined in a setting with a much larger training dataset but with heterogenous trajectories and/or variable data quality levels so that prior to training, a classification process needs to be carried out in order to increase the potential match between the training and testing datasets and improve the generalizability of the learned patterns. Beyond the above extensions to the approaches considered in this dissertation, also of interest are other information-based performance metrics (besides MSE and RMSE) [22], more advanced estimation and fusion techniques (e.g., by utilizing multiple IMMs at the sensors and/or fusing the outputs of a number of fusers at the fusion center), and tracking under more complex network communication and computation constraints (such as in the presence of time-correlated link conditions and heterogeneous bias profiles across sensors).

Bibliography

[1] Earth systems research laboratory, national oceanic and atmospheric administration. http://www.esrl.noaa.gov/gmd/ccgg/index.html.

[2] Network telescope research, 2011. http://www.caida.org/research/security/telescope/.

[3] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H. T. Lin. *Learning from Data – A Short Course*. AMLbook, 2012.

[4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102–114, Aug. 2002.

[5] M. Allman, C. Hayes, H. Kruse, and S. Ostermann. Tcp performance over satellite links. In *Proceedings of Fifth International Conference on Telecommunications Systems*, Nashville, TN, Mar. 1997.

[6] T. Alpcan and T. Basar. *Network Security: A Decision and Game Theoretic Approach*. Cambridge University Press, 2011.

[7] D. E. Archer, B. R. Beauchamp, J. G. Mauger, et al. Adaptable radiation monitoring system and method, 2006. U.S. Patent 7,064,336 B2.

[8] M. K. Banavar, C. Tepedelenlioglu, and A. Spanias. Estimation over fading channels with limited feedback using distributed sensing. *IEEE Transactions on Signal Processing*, 58(1):414–425, 2010.

[9] R. B. Bapat and M. I. Beg. Order statistics for nonidentically distributed variables and permanents. *Sankhy: The Indian Journal of Statistics, Series A*, 51(1):79–93, Feb. 1989.

[10] Y. Bar-Shalom. Update with out-of-sequence measurements in tracking: exact solution. *Aerospace and Electronic Systems, IEEE Transactions on*, 38(3):769–778, Jul. 2002.

[11] Y. Bar-Shalom, H. Chen, and M. Mallick. One-step solution for the general out-of-sequence measurement problem in tracking. *Aerospace and Electronic Systems, IEEE Transactions on*, 40(1):27–37, Jan. 2004.

[12] Y. Bar-Shalom, T. Kirubarajan, and X. R. Li. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, NY, 2002.

[13] Y. Bar-Shalom and X. R. Li. *Multisensor Multitarget Tracking: Principles and Techniques*. YBS Publishing, 1995.

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

[15] W. Boord and J. B. Hoffman. *Air and Missile Defense Systems Engineering*. CRC Press, 2014.

[16] S. Borade and L. Zheng. Wideband fading channels with feedback. *IEEE Transactions on Information Theory*, 56(12):6058–6065, 2010.

[17] K. Brigham, B. V. K. Vijaya Kumar, and N. S. V. Rao. Learning-based approaches to nonlinear multisensor fusion in target tracking. In *Proc. 16th International Conference on Information Fusion (FUSION 2013)*, pages 1320–1327, Jul. 2013.

[18] T. Cacoullos and V. Papathanasiou. On upper bounds for the variance of functions of random variables. *Statistics and Probability Letters*, 3(4):175–184, Jul. 1992.

[19] H. Chen and X. R. Li. On track fusion with communication constraints. In *Information Fusion, 2007 10th International Conference on*, pages 1–7, Quebec, Canada, Jul. 2007.

[20] A. Chiuso and L. Schenato. Information fusion strategies from distributed filters in packet-drop networks. In *Proc. IEEE 47th Decision and Control Conference*, pages 1079 – 1084, Cancun, Mexico, Dec. 2007.

[21] F. Chowdhury. A neural approach to data fusion. In *Proc. the American Control Conference*, pages 1693–1697, Jun. 1995.

[22] O. E. Drummond. Information exchanged between fusion tracker and other fusion functions. In *Proc. SPIE Signal and Data Processing of Small Targets 2011*, volume 8137, Sep. 2011.

[23] Z. Duan and X. R. Li. Lossless linear transformation of sensor data for distributed estimation fusion. *IEEE Trans. on Signal Processing*, 59(1):362–372, Jan. 2011.

[24] E. Filer and J. Hartt. Cobra dane wideband pulse compression system. In *Proc. IEEE EASCON*, pages 26–29, 1976.

[25] L.-W. Fong and C.-Y. Fan. Multisensor fusion algorithms for maneuvering target tracking. In *Proc. 1st IEEE International Conference on E-Learning in Industrial Electronics*, pages 80–84, Dec. 2006.

[26] F. D. Foresee and M. T. Hagan. Gauss-newton approximation to bayesian learning. In *Neural Networks, International Conference on*, volume 3, pages 1930–1935, Jun. 1997.

[27] M. T. Hagan and M.-B. Menhaj. Training feedforward networks with the marquardt algorithm. *Neural Networks, IEEE Transactions on*, 5(6):989–993, Nov. 1994.

[28] D. Han, J. Zhang, Y. Zhang, and W. Gu. Convergence of sensor networks/internet of things and power grid information network at aggregation layer. In *Proc. International Conference on Power System Technology (POWERCON)*, pages 1–6, Oct. 2010.

[29] T. Hanselmann, M. Morelande, B. Moran, and P. Sarunic. Sensor scheduling for multiple target tracking and detection using passive measurements. In *Proc. 11th International Conference on Information Fusion*, pages 1–8, Jun. 2008.

[30] S. Haykin. *Neural Networks and Learning Machines*. Prentice Hall, 2008.

[31] J. Heaton. The number of hidden layers, 2008. http://www.heatonresearch.com/node/707.

[32] J. Heidemann, M. Stojanovic, and M. Zorzi. Underwater sensor networks: applications, advances and challenges. *Phil. Trans. of Royal Society*, 370(1958):158–175, 2012.

[33] T. R. Henderson and R. H. Katz. Transport protocols for internet-compatible satellite networks. *IEEE Journal on Selected Areas in Communications*, 17:326–344, 1999.

[34] A. O. Hero and D. Cochran. Sensor management: Past, present, and future. *Sensors Journal, IEEE*, 11(12):3064 –3075, Dec. 2011.

[35] A. K. Jain, J. Mao, and K. M. Mohiuddin. Artificial neural networks: a tutorial. *Computer*, 29(3):31–44, Mar. 1996.

[36] S. J. Julier and J. K. Uhlmann. *General Decentralized Data Fusion with Covariance Intersection, ser. Handbook of Multisensor Data Fusion*. CRC Press, 2001.

[37] M. K. Kalandros, L. Trailovic, Lucy Y. Pao, and Y. Bar-Shalom. Tutorial on multisensor management and fusion algorithms for target tracking. In *Proc. IEEE American Control Conference*, volume 5, pages 4734–4748, Jun. 2004.

[38] T. H. Kerr. Streamlining measurement iteration for ekf target tracking. *IEEE Transactions on Aerospace and Electronic Systems,*, 27(2):408–421, Mar. 1991.

[39] A. Khan, D. Schaefer, L. Tao, et al. Low power greenhouse gas sensors for unmanned aerial vehicles. *Journal of Remote Sensing*, 4(5):1355–1368, 2012.

[40] D. E. Kirk. *Optimal Control Theory: An Introduction.* Dover Books on Electrical Engineering Series. Dover Publications, 2004.

[41] W. Koch. Perspectives for a backbone technology: tracking in real-world applications. *Aerospace and Electronic Systems Magazine, IEEE*, 25(11):11–16, Nov. 2010.

[42] C. Kreucher, A. O. Hero, K. Kastella, and D. Chang. Efficient methods of non-myopic sensor management for multitarget tracking. In *Decision and Control, 43rd IEEE Conference on*, volume 1, pages 722–727, Dec. 2004.

[43] X. R. Li and V. P. Jilkov. A survey of maneuvering target tracking, part iii: Measurement models. In *Proceedings of the 2001 SPIE Conference on Signal and Data Processing of Small Targets*, volume 4473, pages 423–446, San Diego, CA, Jul–Aug. 2001.

[44] X. R. Li and V. P. Jilkov. Survey of maneuvering target tracking. part i: dynamic models. *IEEE Transactions on Aerospace and Electronics Systems*, 39(4):1333–1364, Jan. 2003.

[45] X. R. Li and V. P. Jilkov. Survey of maneuvering target tracking. part ii: Motion models of ballistic and space targets. *Aerospace and Electronic Systems, IEEE Transactions on*, 46(1):96–119, Jan. 2010.

[46] Q. Liu, K. Brigham, X. Wang, N. S. V. Rao, and B. V. K. Vijaya Kumar. Artificial neural network-based state fusion in long-haul sensor networks. submitted.

[47] Q. Liu, X. Wang, and N. S. V. Rao. Information feedback in estimation and fusion over long-haul sensor networks. submitted.

[48] Q. Liu, X. Wang, and N. S. V. Rao. Staggered scheduling of estimation and fusion over long-haul sensor networks. submitted.

[49] Q. Liu, X. Wang, and N. S. V. Rao. Fusion of state estimates over long-haul sensor networks under random delay and loss. In *Proceedings of 31st IEEE International Conference on Computer Communications (INFOCOM)*, pages 2968–2972, Orlando, FL, Mar. 2012.

[50] Q. Liu, X. Wang, and N. S. V. Rao. Staggered scheduling of estimation and fusion in long-haul sensor networks. In *Proc. Information Fusion (FUSION), 2013 16th International Conference on*, pages 1699–1706, Istanbul, Turkey, Jul. 2013.

[51] Q. Liu, X. Wang, and N. S. V. Rao. Information feedback for estimation and fusion in long-haul sensor networks. In *Proc. Information Fusion (FUSION), 2014 17th International Conference on*, Salamanca, Spain, Jul. 2014.

[52] Q. Liu, X. Wang, and N. S. V. Rao. Fusion of state estimates over long-haul sensor networks with random loss and delay. *IEEE/ACM Transactions on Networking*, 2014, to appear.

[53] Q. Liu, X. Wang, N. S. V. Rao, K. Brigham, and B. V. K. Vijaya Kumar. Effect of retransmission and retrodiction on estimation and fusion in long-haul sensor networks. submitted.

[54] Q. Liu, X. Wang, N. S. V. Rao, K. Brigham, and B. V. K. Vijaya Kumar. Fusion performance in long-haul sensor networks with message retransmission and retrodiction. In *9th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, Las Vegas, NV, Oct. 2012.

[55] Q. Liu, X. Wang, N. S. V. Rao, K. Brigham, and B. V. K. Vijaya Kumar. Performance of state estimate fusion in long-haul sensor networks with message retransmission. In *Proc. Information Fusion (FUSION), 2012 15th International Conference on*, pages 719–726, Singapore, Singapore, Jul. 2012.

[56] S. Liu, M. Fardad, E. Masazade, and P. K. Varshney. Optimal periodic sensor scheduling in networks of dynamical systems. *Signal Processing, IEEE Transactions on*, 62(12):3055–3068, Jun. 2014.

[57] D. J. Mackay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–492, May 1992.

[58] M. Mallick and K. Zhang. Optimal multiple-lag out-of-sequence measurement algorithm based on generalized smoothing framework. In *Proc. SPIE, Signal and Data Processing of Small Targets*, San Diego, CA, Apr. 2005.

[59] L. Mo, X. Song, Y. Zhou, Z. Sun, and Y. Bar-Shalom. Unbiased converted measurements for tracking. *Aerospace and Electronic Systems, IEEE Transactions on*, 34(3):1023–1027, Jul. 1998.

[60] M. Moayedi, Y. K. Foo, and Y. C. Soh. Adaptive kalman filtering in networked systems with random sensor delays, multiple packet dropouts and missing measurements. *IEEE Trans. on Signal Processing*, 58(3):1577–1588, Mar. 2010.

[61] R. N. Murty and M. Welsh. Towards a dependable architecture for internet-scale sensing. In *Proc. 2nd Workshop on Hot Topics in System Dependability - Volume 2*, Seattle, WA, Nov. 2006.

[62] N. Nabaa and R. H. Bishop. Validation and comparison of coordinated turn aircraft maneuver models. *Aerospace and Electronic Systems, IEEE Transactions on*, 36(1):250–259, Jan. 2000.

[63] R. Niu, P. K. Varshney, K. Mehrotra, and C. Mohan. Temporally staggered sensors in multi-sensor target tracking systems. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(3):794–808, Jul. 2005.

[64] University of Southampton. Global network of new-generation telescopes will track astrophysical events as they happen, *ScienceDaily*, Jan. 2011.

[65] A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes*. McGraw-Hill series in electrical and computer engineering. McGraw-Hill, 2002.

[66] J. Poland. On the robustness of update strategies for the bayesian hyperparameter α, 2001. http://www-alg.ist.hokudai.ac.jp/ jan/alpha.pdf.

[67] Y. Rachlin, R. Negi, and P. Khosla. Sensing capacity for discrete sensor network applications. In *Information Processing in Sensor Networks (IPSN). Fourth International Symposium on*, pages 126 – 132, Apr. 2005.

[68] N. S. V. Rao, K. Brigham, B. V. K. Vijaya Kumar, Q. Liu, and X. Wang. Effects of computing and communications on state fusion over long-haul sensor networks. In *Proc. Information Fusion (FUSION), 2012 15th International Conference on*, pages 1570–1577, Singapore, Singapore, Jul. 2012.

[69] M. A. Richards. *Fundamentals of Radar Signal Processing*. Mcgraw-Hill, 2005.

[70] D. Roddy. *Satellite Communications*. McGraw-Hill, 2006.

[71] H. Sawant, J. Tan, and Q. Yang. A sensor networked approach for intelligent transportation systems. In *Intelligent Robots and Systems(IROS). Proceedings IEEE/RSJ International Conference on*, volume 2, pages 1796 – 1801, Sep.-Oct. 2004.

[72] L. Shi, K. H. Johansson, and R. M. Murray. Kalman filtering with uncertain process and measurement noise covariances with application to state estimation in sensor networks. In *IEEE Intl' Conference on Control Applications*, pages 1031 – 1036, Singapore, Singapore, Oct. 2007.

[73] D. Simon. *Optimal state estimation: Kalman, H [infinity] and nonlinear approaches*. Wiley-Interscience, 2006.

[74] R. Tan, G. Xing, X. Liu, J. Yao, and Z. Yuan. Adaptive calibration for fusion-based wireless sensor networks. In *Proc. IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–9, 2010.

[75] D. K. Tasoulis, N. M. Adams, and D. J. Hand. Selective fusion of out-of-sequence measurements. *Journal of Information Fusion*, 11(2):183–191, Apr. 2010.

[76] X. Tian and Y. Bar-Shalom. Track-to-track fusion configurations and association in a sliding window. *Journal of Advances in Information Fusion*, 4(2):146 – 164, Dec. 2009.

[77] J. K. Uhlmann. Covariance consistency methods for fault-tolerant distributed data fusion. *Information Fusion, Elsevier*, 4(3):201–215, Sep. 2003.

[78] P. K. Varshney. *Distributed Detection and Data Fusion*. Springer-Verlag, 1997.

[79] Y. Wang and X. Li. Distributed estimation fusion with unavailable cross-correlation. *Aerospace and Electronic Systems, IEEE Transactions on*, 48(1):259–278, Jan. 2012.

[80] A. S. Willsky, M. Bello, D. A. Castanon, B. C. Levy, and G. C. Verghese. Combining and updating of local estimates and regional maps along sets of one-dimensional tracks. *Automatic Control, IEEE Transactions on*, 27(4):799–813, Aug. 1982.

[81] M. Yeddanapudi, Y. Bar-Shalom, K. R. Pattipati, and S. Deb. Ballistic missile track initiation from satellite observations. *IEEE Transactions on Aerospace and Electronic Systems*, 31(3):1054–1071, Jul. 1995.

[82] T. Yuan, Y. Bar-Shalom, and X. Tian. Heterogeneous track-to-track fusion. In *Information Fusion (FUSION), 14th International Conference on*, pages 1–8, Jul. 2011.

[83] K. Zhang, X. R. Li, and Y. Zhu. Optimal update with out-of-sequence measurements. *Signal Processing, IEEE Transactions on*, 53(6):1992–2004, Jun. 2005.

[84] S. Zhang and Y. Bar-Shalom. Optimal update with multiple out-of-sequence measurements with arbitrary arriving order. *Aerospace and Electronic Systems, IEEE Transactions on*, 48(4):3116–3132, Oct. 2012.

[85] S. Zhang, Y. Bar-Shalom, and G. Watson. Tracking with multisensor out-of-sequence measurements with residual biases. *Journal of Advances in Information Fusion*, 6(1):3–23, Jun. 2011.

[86] Z. Zhao, X. R. Li, and V. P. Jilkov. Best linear unbiased filtering with nonlinear measurements for target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 40(4):1324–1336, Oct. 2004.