A FORWARD PREDICTION BASED OUT OF SEQUENCE MEASUREMENT PROCESSING METHOD FOR KALMAN AND IMM FILTERING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

OZAN EREL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2022

Approval of the thesis:

A FORWARD PREDICTION BASED OUT OF SEQUENCE MEASUREMENT PROCESSING METHOD FOR KALMAN AND IMM FILTERING

submitted by **OZAN EREL** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Umut Orguner
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Emre Özkan
Electrical and Electronics Engineering, METU _____

Prof. Dr. Umut Orguner
Electrical and Electronics Engineering, METU _____

Prof. Dr. Mehmet Kemal Leblebicioğlu
Electrical and Electronics Engineering, METU _____

Assoc. Prof. Dr. Mustafa Mert Ankaralı
Electrical and Electronics Engineering, METU _____

Prof. Dr. Murat Efe
Electrical and Electronics Engineering, Ankara University _____

Date: 13.12.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Ozan Erel

Signature :

ABSTRACT

A FORWARD PREDICTION BASED OUT OF SEQUENCE MEASUREMENT PROCESSING METHOD FOR KALMAN AND IMM FILTERING

Erel, Ozan
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Umut Orguner

December 2022, 92 pages

Processing measurements from the same target which arrive at the processing center not in the order they were obtained (i.e., out of sequence) due to delayed communications is a challenging (OOSM) problem in target-tracking applications. If this problem is not given special care, the quality of the tracking may degrade rather than improve. Therefore, this thesis is focused on the study of forward prediction based OOSM processing methods for linear Gaussian systems and their extension for jump Markov linear systems. The main idea of these methods is to fuse a forward-predicted version of a past target track which incorporates the OOSM, with the current track. Among the proposed methods for linear Gaussian systems, the best-performing one turns out to be a preferable option since it performs close to existing solutions and has similar computational cost and data storage requirements with them. In the case of jump Markov linear systems, as well, the best-performing proposed method seems to be a good option for practical applications when the optimization parameter used in its fusion step can be chosen smaller to reduce the computational cost without experiencing considerable performance degradation.

Keywords: out of sequence measurement, track fusion, interacting multiple filter

ÖZ

KALMAN VE IMM FİLTRELEME İÇİN İLERİYE YÖNELİK TAHMİN TABANLI SIRA DIŞI ÖLÇÜM İŞLEME YÖNTEMİ

Erel, Ozan
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Umut Orguner

Aralık 2022 , 92 sayfa

Aynı hedeften elde edilen ve gecikmeli iletişim nedeniyle elde edilme sırasına göre işlem merkezine ulaşmayan (yani sıra dışı) ölçümlerin işlenmesi, hedef takip uygulamalarında zorlu bir (OOSM) problemdir. Bu probleme özel bir önem verilmezse, takip kalitesi artmak yerine düşebilir. Bu nedenle, bu tez doğrusal Gauss sistemleri için ileri tahmin tabanlı OOSM işleme yöntemleri ve bunların atlamalı Markov doğrusal sistemleri için genişletilmesi üzerine odaklanmıştır. Bu yöntemlerin ana fikri, OOSM'yi içeren geçmiş bir hedef izinin ileriye dönük tahmin edilen bir versiyonunu güncel hedef iziyle birleştirmektir. Doğrusal Gauss sistemleri için önerilen yöntemler arasında en iyi performans göstereni, mevcut çözümlere yakın performans göstermesi ve onlarla benzer hesaplama maliyeti ve veri depolama gereksinimlerine sahip olması nedeniyle tercih edilebilir bir seçenek olarak ortaya çıkmaktadır. Atlamalı Markov doğrusal sistemler söz konusu olduğunda da, önerilen en iyi performans gösteren yöntem, füzyon adımında kullanılan optimizasyon parametresinin kayda değer bir performans düşüşü yaşamadan hesaplama maliyetini düşürmek adına daha küçük seçilebildiği pratik uygulamalar için iyi bir seçenek gibi görünmektedir.

Anahtar Kelimeler: sıra dışı ölçüm, iz füzyonu, etkileşimli çoklu model

To my future self, who I hope will be able to assess the consequences of his decisions and their impact on him beforehand..

ACKNOWLEDGMENTS

I have to admit that before I started writing my thesis, I never imagined that there would be so many people I would want to mention in this section, so the lesson learned here is not to count the chickens before they hatch.

First of all, I would like to express my sincere gratitude to my professor Umut Orguner, who always provided me with the help and feedback I needed. I would also like to express my eternal gratitude to my colleague, my first and always polar star Görkem Abla for the constant guidance and support she has given me with her 30 years of experience, I don't know what I would have done without her divine interventions. I would also like to share my special thanks to my beloved girlfriend Beste, who guided me through difficult times with her exceptional and, according to her, effortless speeches. I would also like to thank my brothers Köse, Kemal and Berkay, who have been there to support me throughout this whole process and made me realize the value of our friendship once again. To Hande and Çağrı, whose joy and fun they brought to our workplace helped me loosen up during this stressful period, I promise you that I will do my best to get my mention in your acknowledgments. My dear friends Mert and Şimşek, I want you to show that I always carry the regret of not writing to you in the high school yearbook and that you are in my mind by mentioning you here.

Last but not least, to my parents, who have the biggest contribution to raising me to become who I am today, and to my biggest role model, the most caring sister a sibling could wish to have, whose footsteps I have always followed in my quest to be a better person in life, this thesis is as much your work as it is mine, I love you all very much.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS x

TABLE OF CONTENTS xi

LIST OF TABLES xiv

LIST OF FIGURES xv

LIST OF ABBREVIATIONS xviii

CHAPTERS

1 INTRODUCTION 1

1.1 Literature Review 1

1.2 The Outline of the Thesis 3

2 BACKGROUND 5

2.1 OOSM Problem for Linear Gaussian Systems 5

2.1.1 OOSM Problem Definition 5

2.1.2 In Sequence Estimation (Kalman Filter) 8

2.1.3 Existing Solutions for the OOSM Problem 9

2.1.3.1 Bl1 Algorithm 9

2.1.3.2 Forward Prediction Fusion and Decorrelation (FPFD) . 14

2.2 OOSM Problem for Jump Markov Linear Systems 19
2.2.1 OOSM Problem Definition 20
2.2.2 In Sequence Estimation (IMM filter) 21
2.2.3 Existing Solutions for the OOSM Problem 24
2.2.3.1 Bl1IMM Algorithm 24
2.3 Correlation Independent Fusion Methods 27
2.3.1 Naive Fusion 29
2.3.2 Covariance Intersection (CI) 30
2.3.3 Largest Ellipsoid Algorithm (LEA) 31
3 PROPOSED METHOD 39
3.1 Proposed Solution for Linear Gaussian Systems 39
3.1.1 Algorithm 39
3.1.2 Data Storage Requirement 42
3.1.3 Computational Cost 43
3.2 Proposed Solution for Jump Markov Linear Systems 44
3.2.1 Algorithm 45
3.2.1.1 Naive Fusion applied in FPFIMM 47
3.2.1.2 CI Fusion applied in FPFIMM 48
3.2.1.3 LEA Fusion applied in FPFIMM 51
3.2.2 Data Storage Requirement 51
3.2.3 Computational Cost 52
3.3 Advantages and Disadvantages 53
3.3.1 Advantages 53

3.3.2 Disadvantages 53
4 NUMERICAL RESULTS 55
4.1 Performance Testing of FPF Methods 55
4.1.1 2-D Linear Measurement Model Example 55
4.1.1.1 Single-Lag Scenario 57
4.1.1.2 Multi-Lag Scenario 63
4.1.2 2-D Nonlinear Measurement Model Example 68
4.2 Performance Testing of FPFIMM Methods 71
4.2.1 Model Match Case 71
4.2.2 Model-Mismatch Case 75
4.2.2.1 Effect of the OOSM Lag 76
4.2.2.2 Effect of the OOSM Time 77
5 CONCLUSIONS AND FUTURE WORK 87
REFERENCES 89

LIST OF TABLES

TABLES

Table 3.1 Number of scalars required to be stored in terms of l_{max} for a state vector with 6 elements. 43

Table 3.2 Number of operations required to be performed for any l. 44

Table 3.3 Number of scalars required to be stored in terms of l_{max} when the number of state variables in x_k is 4 and number of IMM filter models is 2. 52

Table 3.4 Number of operations required to be performed in IMM application for any l_{max}. 52

LIST OF FIGURES

FIGURES

Figure 2.1 The illustration of the OOSM problem. 7

Figure 2.2 The illustration of one-step lag OOSM problem. 10

Figure 2.3 The illustration of the equivalent measurement concept. 12

Figure 2.4 The illustration of the equivalent measurement processing concept. 13

Figure 2.5 The illustration of the FPFD method for OOSM track update. . . 18

Figure 2.6 The illustration of the resulting ellipsoid after CI fusion. 32

Figure 2.7 The illustration of the steps of LEA between the two estimates x_1 and x_2. 36

Figure 2.8 The illustration of the resulting ellipsoids after CI and LEA fusion. 37

Figure 3.1 The illustration of the proposed method for OOSM track update. 41

Figure 4.1 Position RMSE comparison of FPF methods and existing solutions for different process noise intensities. 59

Figure 4.2 Velocity RMSE comparison of different FPF methods and existing solutions for different process noise intensities. 60

Figure 4.3 NEES comparison of FPF methods and existing solutions for different process noise intensities. 61

Figure 4.4 NEES comparison of FPF methods and existing solutions for different process noise intensities (Zoomed version of Fig. 4.3). 62

Figure 4.5 Position RMSE comparison of FPF methods and existing solutions for different OOSM lags. 64

Figure 4.6 Velocity RMSE comparison of different FPF methods and existing solutions for different OOSM lags. 65

Figure 4.7 NEES comparison of different FPF methods and existing solutions for different OOSM lags. 66

Figure 4.8 NEES comparison of different FPF methods and existing solutions for different OOSM lags (Zoomed version of Fig. 4.7). 67

Figure 4.9 Time-averaged RMS position errors for different lags. 68

Figure 4.10 Time-averaged RMS velocity errors for different lags. 69

Figure 4.11 Position RMSE Values. 71

Figure 4.12 Velocity RMSE Values. 72

Figure 4.13 NEES Values. 73

Figure 4.14 NEES Values (Zoomed version of Fig. 4.13). 74

Figure 4.15 Position RMSE for 1 lag OOSMs. 75

Figure 4.16 Position RMSE for 3 lag OOSMs. 76

Figure 4.17 Velocity RMSE for 1 lag OOSMs. 77

Figure 4.18 Velocity RMSE for 3 lag OOSMs. 78

Figure 4.19 Mean NEES for 1 lag OOSMs. 78

Figure 4.20 Mean NEES for 3 lag OOSMs. 79

Figure 4.21 Mean NEES for 1 lag OOSMs (Zoomed version of Fig. 4.19). . . 79

Figure 4.22 Mean NEES for 3 lag OOSMs (Zoomed version of Fig. 4.20). . . 80

Figure 4.23 True and In-Seq estimate trajectories of the target in the model-mismatch scenario. The red dot denotes the starting point of the target trajectory. 80

Figure 4.24 Position RMSE for 1 lag OOSMs. 81

Figure 4.25 Position RMSE for 3 lag OOSMs. 81

Figure 4.26 Velocity RMSE for 1 lag OOSMs. 82

Figure 4.27 Velocity RMSE for 3 lag OOSMs. 82

Figure 4.28 Mean NEES for 1 lag OOSMs. 83

Figure 4.29 Mean NEES for 3 lag OOSMs. 83

Figure 4.30 Position RMSEs with respect to OOSM times for different OOSM lags. 84

Figure 4.31 Position RMSEs with respect to OOSM times for different OOSM lags (Zoomed version of Fig. 4.30). 85

LIST OF ABBREVIATIONS

CI	Covariance Intersection
FPF	Forward Prediction based Fusion
FPFD	Forward Prediction Fusion and Decorrelation
GMTI	Ground Moving Target Indication
IMM	Interacting Multiple Model
JMLS	Jump Markov Linear System
KF	Kalman Filter
LEA	Largest Ellipsoid Algorithm
MC	Monte Carlo
MSE	Mean Square Error
NEES	Normalized Estimation Error Squared
OOSM	Out of Sequence Measurement
RMS	Root Mean Square
RMSE	Root Mean Square Error

CHAPTER 1

INTRODUCTION

In this section, we first give a brief literature review about the Out of Sequence Measurement (OOSM) problem, and then we give the outline of the thesis.

1.1 Literature Review

In multisensor tracking systems that operate in a centralized manner [1], i.e., the systems in which the processing of the measurements from all the sensors is done at a single center, there are usually different time delays in the arrival of the measurement data from the various sensors to the center. This may arise from many factors, such as sensor diversity, and/or random delays over communication networks. This leads to a possibility that the measurements from the same target may not be received in the order they have originated, in other words out of sequence. Therefore, it may occur that a measurement at a given time arrives at the processing center after the corresponding track has already been updated with one, or more, more recently collected measurements. The resulting problem of how to update the current state estimate with an "older" measurement has appeared in the literature under various names such as the Out Of Sequence Measurements (OOSM) problem [2, 3], the problem of tracking with random sampling and delays [4, 5, 6], the problem of negative-time measurement update [2], and the problem of incorporating random time delayed measurements [7].

A direct solution to the OOSM problem is simply to ignore the OOSM in the processing. This solution leads obviously to a loss of the information contained in the OOSM. In order to avoid such a drawback, one can simply reprocess all measurements that are collected from the OOSM time to the current time in chronological

order and achieve the optimal solution. Nevertheless, this solution turns out to be inefficient due to its high computation and storage requirements, especially in a case where data association is handled by a complex algorithm like a multiple hypothesis tracker (MHT). Thus, instead of reprocessing an entire sequence of measurements, such measurements have to be processed as OOSMs.

Several methods have been proposed for Linear Gaussian systems in the literature to deal more efficiently with the OOSM problem. Firstly, the work in [2] developed the optimal procedure for incorporating a 1 lag measurement into a Kalman filter (KF) and showed that a simpler algorithm, designated as B (originated in [4]), is nearly optimal for processing such an OOSM. Algorithm Bl [8] generalized algorithm B for an l lag OOSM, however, requires l steps to carry out the update. Subsequently, [9] and [3] showed that an l lag OOSM can be processed similarly to a 1 lag OOSM, in one single step via another algorithm called Bl1, rather than in l steps that would be required by algorithm Bl. In [10] the Forward Prediction Fusion and Decorrelation (FPFD) method which is based on forward prediction and decorrelation to overcome the OOSMs problem is suggested, without the need for backward prediction. The proposed method has proved to compare favorably to the aforementioned algorithms. There are also some methods for updating the state estimate optimally with an out-of-sequence measurement (OOSM) by using augmented state smoothing [11, 12]. The works of [13, 14] solve for the update with an arbitrary lag OOSM, but they both require an iteration back for l steps and a considerable amount of storage.

Since the state-of-the-art tracker for real (maneuvering) targets is the interacting multiple model (IMM) estimator, algorithms for incorporating OOSM into an IMM estimator have also been developed in the literature. For maneuvering target tracking problems, it is popular to consider multiple model-based approaches as the maneuver behavior of a target can be more precisely described by using multiple models. The standard interacting multiple model (IMM) algorithm, proposed in [15] is the most effective suboptimal multiple model algorithm and has been adapted to many applications for maneuvering target tracking [16]. An IMM-PDA (probabilistic data association) filter for processing an OOSM was presented in [17] using state augmentation. The algorithm for incorporating OOSMs into the IMM estimator via the Bl1 approach [4, 5], is discussed in [18]. In the realistic GMTI examples presented

there, the IMM estimator that processes the OOSM via the Bl1 approach performs practically as well as the IMM which reorders and reprocesses the measurements in sequence, and, consequently, this is a practical approach suitable for real-time implementation. In this thesis, we propose a forward prediction-based OOSM processing solution which can be applicable to both linear Gaussian and jump Markov linear systems.

1.2 The Outline of the Thesis

Chapter 2 gives the OOSM problem definition and describes two of the existing solutions in the literature along with some of the well-known track-to-track fusion techniques. Chapter 3 discusses our proposed solutions as forward prediction-based OOSM processing methods which can be applied to JMLSs as well as linear Gaussian systems. In Chapter 4, the performance of our proposed methods is tested under various simulation scenarios and compared with existing solutions based on the numerical results obtained. Chapter 5 provides concluding remarks on the proposed methods and comments on the work presented.

CHAPTER 2

BACKGROUND

2.1 OOSM Problem for Linear Gaussian Systems

In this section, the definition of the OOSM problem in linear Gaussian systems is given first. Then it is explained how the estimation is performed through Kalman Filter for the in-sequence measurement(ISM) case in which the OOSM problem is not encountered. Finally, methods that adapt this KF estimation procedure to the OOSM problem are presented.

2.1.1 OOSM Problem Definition

The problem is defined under the framework in which the state x_k of a target of interest evolves over time according to the linear stochastic dynamics given below:

$$x_k = F_{k,k-1}x_{k-1} + w_{k,k-1} \tag{2.1}$$

where $F_{k,k-1}$ is the state transition matrix from time t_{k-1} to t_k and $w_{k,k-1}$ represents the process noise accumulated over this interval. Notice here that, instead of the more familiar index-only notation, the one with two arguments is adopted for clarity.

The measurement is modeled with the following equation:

$$z_k = H_k x_k + v_k, \tag{2.2}$$

where H_k is the measurement matrix and v_k is the measurement noise. It is assumed

that the noises are zero mean and the noise sequences are independent, i.e., white. In addition, the covariances of process and measurement noises are given as follows:

$$E[w_{k,j}(w_{k,j})^T] = Q_{k,j}, \qquad E[v_k v_k^T] = R_k. \tag{2.3}$$

The process and measurement noise sequences are assumed to be uncorrelated as expressed below:

$$E[w_{k,j} v_m^T] = 0. \tag{2.4}$$

The OOSM collected at time t_τ is given as:

$$z_\tau = H_\tau x_\tau + v_\tau, \tag{2.5}$$

where τ denotes the time instant OOSM originated at. Assume that at time t_k before the OOSM arrives, the tracking system computes the state estimate $\hat{x}_{k|k}$ and its corresponding estimated error covariance matrix $P_{k|k}$, defined as below:

$$\hat{x}_{k|k} \triangleq E[x_k|Z^k], \qquad P_{k|k} \triangleq cov[x_k|Z^k], \tag{2.6}$$

where Z^k which stands for the cumulative set of measurements excluding the out-of-sequence measurement z_τ is defined as below:

$$Z^k \triangleq \{z_i\}_{i=1}^k. \tag{2.7}$$

Subsequently, the measurement z_τ arrives upon the calculation of the state estimate (2.6). The problem which is illustrated in Fig. 2.1 arises then as to how to update the current estimate with an older measurement (2.5), namely, how to calculate

$$\hat{x}_{k|k,\tau} = E[x_k|Z^k, z_\tau], \qquad P_{k|k,\tau} = cov[x_k|Z^k, z_\tau]. \tag{2.8}$$

To summarize the discussion above, the OOSM problem emerges when a measurement arrives to update the current track but the time at which it is collected shown by t_τ falls behind the last update time t_k, i.e., $t_\tau < t_k$.

In standard (in-sequence) processing which will be discussed in the following subsection, however, an estimator which is used interchangeably with a tracker expects the next measurement to be collected later than that of the last measurement updating the track. For this reason, a measurement that arrives in a way that disrupts this sequential flow is referred to as the out-of-sequence measurement, and incorporating this measurement into the current track (estimate) is called the out-of-sequence measurement problem.

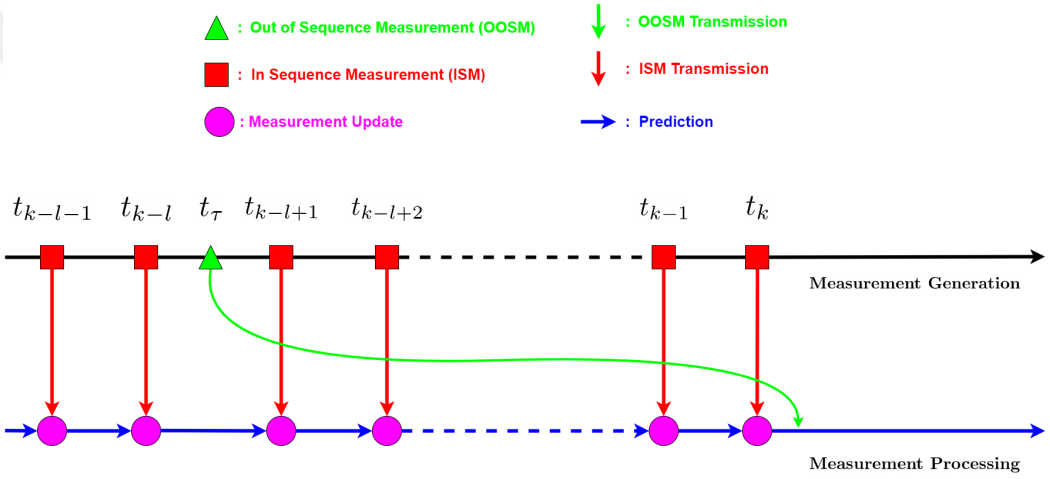


Figure 2.1: The illustration of the OOSM problem.

Furthermore, the concept of step lag which can be defined as the number of state updates made after t_τ, is introduced. Step lag l can be found according to the expression below:

$$t_{k-l} < t_\tau < t_{k-l+1}. \tag{2.9}$$

Depending on the values that the step lag (delay) can take, the OOSM problem is divided into two categories. In the literature, when the lag is equal to one, the problem is referred to as one-step lag or single-step lag OOSM. Otherwise, it is referred to as multi-step (lag) or multi-lag OOSM.

2.1.2 In Sequence Estimation (Kalman Filter)

KF (Kalman Filter) is an optimal (in MMSE sense) solution to the state estimation problem for linear Gaussian systems modeled by (2.1) and (2.2). In the KF framework, the posterior distribution is represented by a Gaussian density as follows:

$$p(x_k|Z^k) = \mathcal{N}(x_k; \hat{x}_{k|k}, P_{k|k}), \tag{2.10}$$

where $\hat{x}_{k|k}$ and $P_{k|k}$ are defined in (2.6).

As a Bayesian recursive filter, KF calculates the current posterior distribution $p(x_k|Z^k)$ from the prior distribution $p(x_{k-1}|Z^{k-1})$ by propagating the previous estimate $\hat{x}_{k-1|k-1}$ and the corresponding covariance $P_{k-1|k-1}$. The propagation takes place in two steps which are time update and measurement update. In the time update step, the predicted state estimate and covariance are obtained as follows:

$$\hat{x}_{k|k-1} = F_{k,k-1}\hat{x}_{k-1|k-1}, \tag{2.11a}$$

$$P_{k|k-1} = F_{k,k-1}P_{k-1|k-1}(F_{k,k-1})^T + Q_{k,k-1}. \tag{2.11b}$$

In the measurement update step, the predicted estimate $\hat{x}_{k|k-1}$ and covariance $P_{k|k-1}$ are updated with measurement z_k to yield the estimate $\hat{x}_{k|k}$ and covariance $P_{k|k}$ as follows:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - \hat{z}_{k|k-1}), \tag{2.12a}$$

$$P_{k|k} = P_{k|k-1} - K_k S_k K_k^T, \tag{2.12b}$$

where the predicted measurement $z_{k|k-1}$, the innovation covariance S_k and the Kalman gain K_k, are found as below:

$$\hat{z}_{k|k-1} = H_k \hat{x}_{k|k-1}, \tag{2.13a}$$

$$S_k = H_k P_{k|k-1} (H_k)^T + R_k, \tag{2.13b}$$

$$K_k = P_{k|k-1} (H_k)^T (S_k)^{-1}. \tag{2.13c}$$

Thus, the updated state estimate and the corresponding covariance which characterize the posterior distribution $p(x_k|Z^k)$ are obtained. Notice that for optimal processing, the next measurement z_k should be collected later than z_{k-1}, the last measurement updating the distribution, i.e, z_k and z_{k-1} are ISMs (In Sequence Measurements).

Since the parts of the KF filter are going to be used as building blocks in the following section, we define the following functions.

- $[\bar{x}, \bar{P}]$ = **KFPrediction**(x, P):
  This is the functional representation of the above-described KF filter time update (2.11) which takes the previous state estimate and covariance $\hat{x}_{k-1|k-1}$, $P_{k-1|k-1}$ and returns the predicted state estimate and covariance $\hat{x}_{k|k-1}$, $P_{k|k-1}$.
- $[x, P]$ = **KFUpdate**$(\bar{x}, \bar{P}, z)$:
  This is the functional representation of the above-described KF filter measurement update (2.12) which takes the predicted state estimate and covariance $\hat{x}_{k|k-1}$, $P_{k|k-1}$ along with the measurement z_k and returns the updated state estimate and the covariance $\hat{x}_{k|k}$, $P_{k|k}$.

2.1.3 Existing Solutions for the OOSM Problem

In this subsection, two algorithms in the literature developed for the linear Gaussian systems for the OOSM problem will be presented.

2.1.3.1 Bl1 Algorithm

Bl1 is a generalized version of B1 which is an approximate solution to the one-step-lag problem illustrated in Fig. 2.2, for the multi-step case [3]. Notice that the letter l

in Bl1 refers to step-lag l as it generalizes the previous one-step lag B1 to an arbitrary lag. Since Bl1 is the multi-step lag extension to the sub-optimal one-step-lag algorithm B1, derived in [2], the discussion about Bl1 starts with the explanation of the latter algorithm.

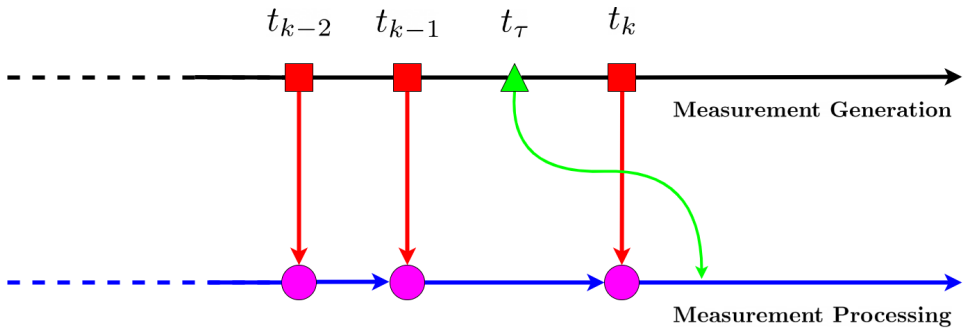


Figure 2.2: The illustration of one-step lag OOSM problem.

B1, which is the simplified and suboptimal version of the A1 algorithm, as mentioned before is developed for one-step lag, i.e., a special case of the OOSM problem. The procedure followed in the B1 algorithm consists of backward prediction (state retrodiction), filter gain computation, and update of the state estimate with the OOSM. The equations of the first step which yields retrodicted state and the corresponding (filter-calculated) covariance are given below:

$$\hat{x}_{\tau|k} = F_{\tau,k}\hat{x}_{k|k}, \tag{2.14}$$

$$P_{\tau|k} = F_{\tau,k}[P_{k|k} + P^{ww}_{k,\tau|k} - P^{xw}_{k,\tau|k} - (P^{xw}_{k,\tau|k})^T]F^T_{\tau,k}, \tag{2.15}$$

where

$$P^{ww}_{k,\tau|k} \triangleq cov[w_{k,\tau}|Z^k] = Q_{k,\tau}, \tag{2.16}$$

$$P^{xw}_{k,\tau|k} \triangleq cov[x_k, w_{k,\tau}|Z^k] = Q_{k,\tau} - P_{k|k-1}H^T_k S^{-1}_k H_k Q_{k,\tau}. \tag{2.17}$$

The innovation covariance $S_{\tau|k}$ and the cross-covariance P^{xz} between x and z, which are needed to compute the filter gain, are obtained by the expressions below:

$$S_{\tau|k} = H_\tau P_{\tau|k} H_\tau^T + R_\tau, \tag{2.18}$$

$$P^{xz}_{k,\tau|k} = (P_{k|k} - P^{xw}_{k,\tau|k}) F^T_{\tau,k} H^T_\tau. \tag{2.19}$$

Subsequently, the filter gain to be used in the state update is found as follows:

$$W_{k,\tau|k} = P^{xz}_{k,\tau|k} S^{-1}_{\tau|k}. \tag{2.20}$$

As the last step, the update with the OOSM is carried out using the KF measurement update equations (2.12) as below:

$$\hat{x}_{k|k,\tau} = \hat{x}_{k|k} + W_{k,\tau|k}(z_\tau - H_\tau \hat{x}_{\tau|k}), \tag{2.21}$$

$$P_{k|k,\tau} = P_{k|k} - W_{k,\tau|k} S_\tau (W_{k,\tau|k})^T. \tag{2.22}$$

However, these equations yield only a sub-optimal solution because the dependence of the process noise $w_{k,\tau}$ on the measurement z_k is fully ignored in (2.14) and partially ignored in (2.15) while performing the state retrodiction step. However, there indeed exists a correlation because z_k is dependent on x_k which contains $w_{k,\tau}$. This dependency is shown by the following expression where x_k is written in terms of x_τ by substituting the variable τ for $k-1$ in the state transition equation (2.1)

$$x_k = F_{k,\tau} x_\tau + w_{k,\tau}. \tag{2.23}$$

In the optimal version of the one-step lag solution B1, referred to as Algorithm A in [2], instead of assuming zero, $w_{k,\tau|k}$ is estimated and then compensated for. The reason for such a simplification on Algorithm A also called A1 in [3] is to avoid

the storage need for innovation value, which does not appear in the equations given above.

Bl1 solution, which is in the framework of algorithm B1, makes use of the equivalent measurement concept to adopt the solution to the multi-step lag case. In this approach, all the measurements from z_{k-l+1} to z_k given as

$$Z_{k-l+1}^{k} = \{z_{k-l+1}, ..., z_k\} \tag{2.24}$$

are replaced by an equivalent measurement at time t_k as illustrated in Fig. 2.3. Then the OOSM falls in the interval between t_{k-l} and t_k in which the only measurement is the equivalent measurement at time t_k. In this manner, the OOSM with an l-step lag problem turns into a single-step lag one and B1 becomes applicable for updating the state estimate at time t_k with the l-step-lag OOSM in a single giant leap [3]. The resulting algorithm solving the multi-step OOSM problem in one step is referred to as Bl1[3].

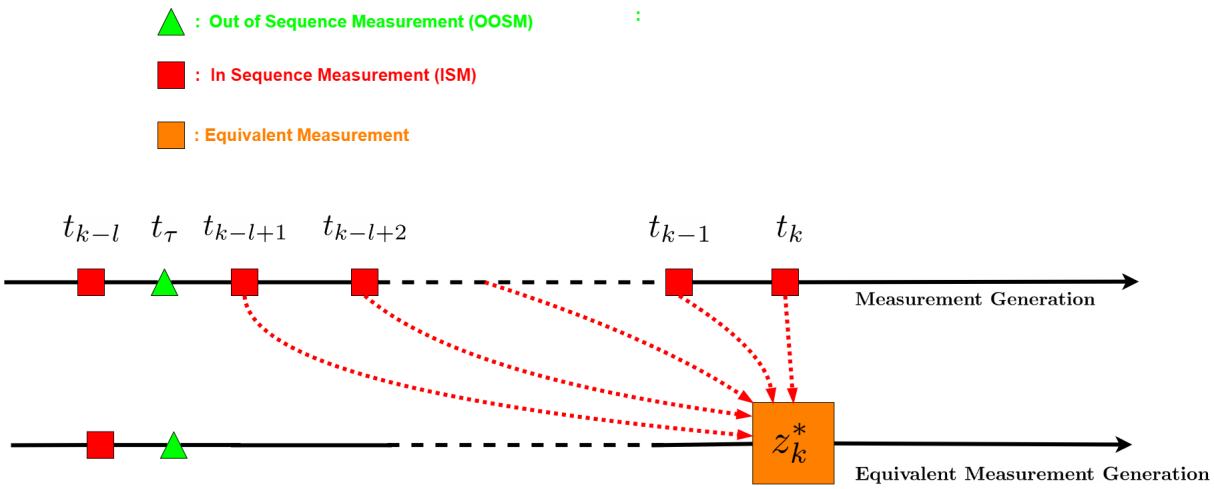


Figure 2.3: The illustration of the equivalent measurement concept.

The equivalent measurement at discrete time instant k can be defined as below:

$$z_k^* = H_k^* x_k + v_k^* \tag{2.25}$$

with covariance R_k^*.

Assuming that at discrete time k one has the state estimate $\hat{x}_{k|k}$ and its covariance $P_{k|k}$, R_k^* can be found by using the measurement update equation in the information form as below:

$$P_{k|k}^{-1} = P_{k|k-l}^{-1} + H_k^{*T} R_k^{*-1} H_k^*. \tag{2.26}$$

Choosing H_k^* as an identity matrix, R_k^* is found by rearranging (2.26) as follows:

$$R_k^{*-1} = P_{k|k}^{-1} - P_{k|k-l}^{-1}. \tag{2.27}$$

Lastly, in order to apply the procedure of algorithm B1, the equivalent innovation covariance S_k^* corresponding to S_k in (2.16) is calculated as below.

$$S_k^* = P_{k|k-l} + R_k^*. \tag{2.28}$$

After this substitution, the procedure of algorithm B1 can be applied in the same manner as illustrated in Fig. 2.4. One can notice the resemblance of this diagram with the one in Fig. 2.2. The difference is that since no actual transmission of the equivalent measurement occurs in reality, it is shown by a dashed line in the latter figure.

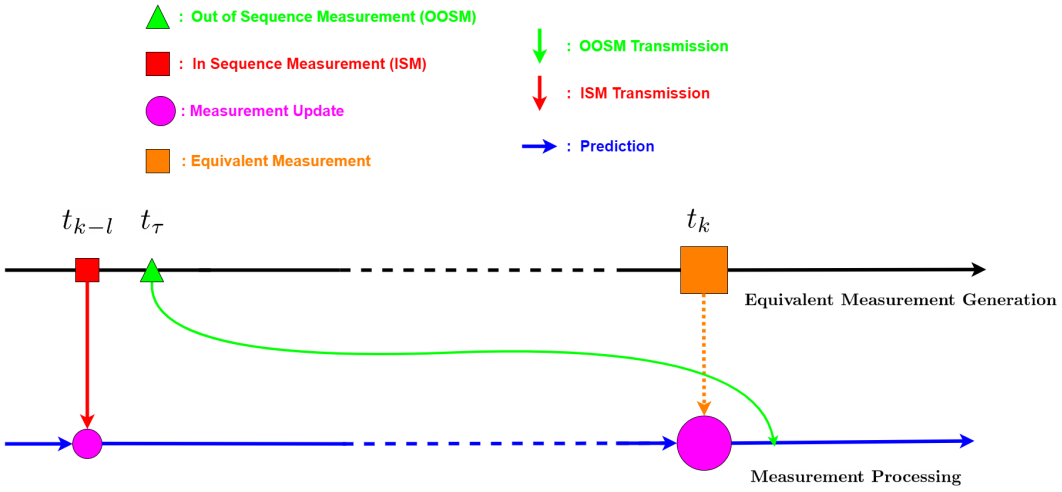


Figure 2.4: The illustration of the equivalent measurement processing concept.

2.1.3.2 Forward Prediction Fusion and Decorrelation (FPFD)

Unlike the Bl1, the FPFD method is not based on retrodiction, where the estimated current state is predicted backwards in order to incorporate the OOSM. The approach followed in this method, as its name implies, is based on forward prediction and decorrelation. In this method, instead of bringing the current track to the time OOSM measurement is obtained, as the retrodiction methods do, the aim is to bring the OOSM itself to the last track update time in a sense.

This task is accomplished by first utilizing an auxiliary filter which is initialized with a track estimate calculated prior to the OOSM being acquired (i.e.,t_τ) and then updating this filter with OOSM while performing forward prediction to the last track update time. This track is called pseudo-track (tracklet) in [10]. However, according to the definition in [19], it is not convenient to call as such because of the fact that its errors still are cross-correlated with the errors of the actual track. This correlation is caused by the prior estimate of the actual track which is used for initializing the new track. Thus, until the uncorrelated version of it is obtained which is the next step of the algorithm, it is referred to as the OOSM track as it contains the OOSM information.

As a next step, in order to make these tracks uncorrelated, i.e., for obtaining a tracklet from the OOSM track, similar decorrelation is applied as in channel filter [20]. Previous communication can be the cause of common information in a fusion architecture [21]. To make an analogy with that, the initialization of the OOSM track corresponds to the fact that the actual (principle) track gives feedback to the newly created one. For finding this redundant common information another auxiliary filter is utilized as before but this time without getting updated with OOSM. Then decorrelation operation simply turns into a subtraction of this information from that of the OOSM track in the information domain. Since the redundant information inside of the OOSM track is discarded after this, it can now be called a tracklet. As a final step, the decorrelated information sources are fused by performing addition in the information space. This algorithm lays the groundwork for the proposed solution to be discussed in the next chapter.

To summarize, instead of approaching the problem as retrodiction methods which

suggest directly incorporating OOSM into the current state estimate, i.e., measurement to track fusion, FPFD proposes to pass the OOSM information to a track whose main purpose is to carry this information and to combine this track with the current state estimate, i.e., track-track fusion. Consequently, FPFD and other retrodiction methods represent two OOSM processing methodologies which differ in the type of fusion they perform.

The abovementioned methodology for overcoming the OOSM problem is formulated next. For this purpose, assume that at time t_k the tracking system computes the state estimate $\hat{x}_{k|k}$ and its covariance matrix $P_{k|k}$. In addition to that, let the statistics of the OOSM track obtained after the forward prediction and prior to the decorrelation be represented similarly to (2.8) as follows:

$$\hat{x}_{k|b,\tau} \triangleq E[x_k|Z^b, z_\tau], \qquad P_{k|b,\tau} \triangleq cov[x_k|Z^b, z_\tau], \tag{2.29}$$

where b represents the discrete index of t_b the time at which the auxiliary filters are initialized. To recall the definition of the OOSM track at time t_k, derivation of its statistics can be given based on the state estimate calculated at time t_b prior to the OOSM, i.e., $t_b < \tau$. The corresponding state estimate and its error covariance matrix are denoted as $\hat{x}_{b|b}$ and $P_{b|b}$ respectively.

On the way to achieving the OOSM track at time t_k, one needs to update the auxiliary track with the OOSM itself. To that end, first $x_{b|b}$ is predicted to the OOSM time which can be carried out through (2.11) as follows:

$$\hat{x}_{\tau|b} = F_{\tau,b}\hat{x}_{b|b}, \tag{2.30}$$

$$P_{\tau|b} = F_{\tau,b}P_{b|b}(F_{\tau,b})^T + Q_{\tau,b}. \tag{2.31}$$

The next step is to take OOSM into account, which corresponds to obtaining the posterior state estimate $\hat{x}_{\tau|b,\tau}$. One way of achieving this is to perform KF measurement

update (2.12). The other alternative utilized in FPFD is to apply the counterpart of these equations in the information form by the so-called information filter (inverse covariance filter). The equations carrying out the same operations in a simpler and more intuitive way are given as follows:

$$P^{-1}_{\tau|b,\tau} = P^{-1}_{\tau|b} + H^T_\tau R^{-1}_\tau H_\tau, \tag{2.32}$$

$$P^{-1}_{\tau|b,\tau}\hat{x}_{\tau|b,\tau} = P^{-1}_{\tau|b}\hat{x}_{\tau|b} + H^T_\tau R^{-1}_\tau z_\tau. \tag{2.33}$$

Finally, applying the prediction procedure as before, the OOSM track is obtained as follows:

$$\hat{x}_{k|b,\tau} = F_{k,\tau}\hat{x}_{\tau|b,\tau}, \tag{2.34}$$

$$P_{k|b,\tau} = F_{k,\tau}P_{\tau|b,\tau}F^T_{k,\tau} + Q_{k,\tau}. \tag{2.35}$$

However, as mentioned before, since this track shares the same history $\hat{x}_{b|b}$ with the actual track, for an optimal fusion (i.e., for reaching the same estimate as the in-sequence measurements reprocessing method) one must take great care not to use the same information twice, i.e., double counting. Hence before fusing these two sources, the information provided by the common history at time t_k needs to be estimated and then compensated for. In FPFD this is achieved by estimating the redundant information, denoted $x_{k|b}$, as predicted $x_{b|b}$ to t_k and subsequently removing it using information decorrelation approach [22]. The mean and the covariance statistics of $x_{k|b}$ can be calculated from $x_{b|b}$ as follows:

$$\hat{x}_{k|b} = F_{k,b}\hat{x}_{b|b}, \tag{2.36}$$

$$P_{k|b} = F_{k,b} P_{b|b} F_{k,b}^T + Q_{k,b}. \tag{2.37}$$

Hereupon, FPFD obtains OOSM incorporated estimate by first applying decorrelation and then fusion. However, since applying these steps separately may lead to implementation issues such as incompatible covariance problems mentioned in [23], an equivalent step in which these operations are performed together is given as follows. This information decorrelation approach is also referred to as a generalized information matrix filter, a generalization of the information filter for asynchronous tracklets, or simply a channel filter [24].

$$P_{k|k,\tau}^{-1} = P_{k|k}^{-1} + P_{k|b,\tau}^{-1} - P_{k|b}^{-1}, \tag{2.38}$$

$$P_{k|k,\tau}^{-1} \hat{x}_{k|k,\tau} = P_{k|k}^{-1} \hat{x}_{k|k} + P_{k|b,\tau}^{-1} \hat{x}_{k|b,\tau} - P_{k|b}^{-1} \hat{x}_{k|b}. \tag{2.39}$$

One shall notice that in this way the calculation of the problematic decorrelated OOSM track (tracklet) term which would otherwise be calculated explicitly is avoided.

As mentioned previously, by incorporating OOSM into the current estimate using pseudo-track, FPFD builds its solution on track-to-track fusion, which is illustrated in Fig. 2.5. Since subsequent fusion leads to an optimal result only when the tracks to be fused are independent of each other, the decorrelation step carried out with (2.38) and (2.39) plays an important role in obtaining the optimal solution achieved by sequential reprocessing of all measurements, including OOSM. In FPFD, the excess information to be discarded is assumed to be $x_{k|b}$ which represents the piece of information common to both the current estimate and the OOSM track. The assumption that the common history or more generally previous communication is the only correlation source, however, may not match the reality due to the process noise in the state transition model which goes into estimation errors of both [25]. Thus except for the single lag case where not taking the process noise into account does not harm the optimality, optimal decorrelation cannot be achieved with this method in general as

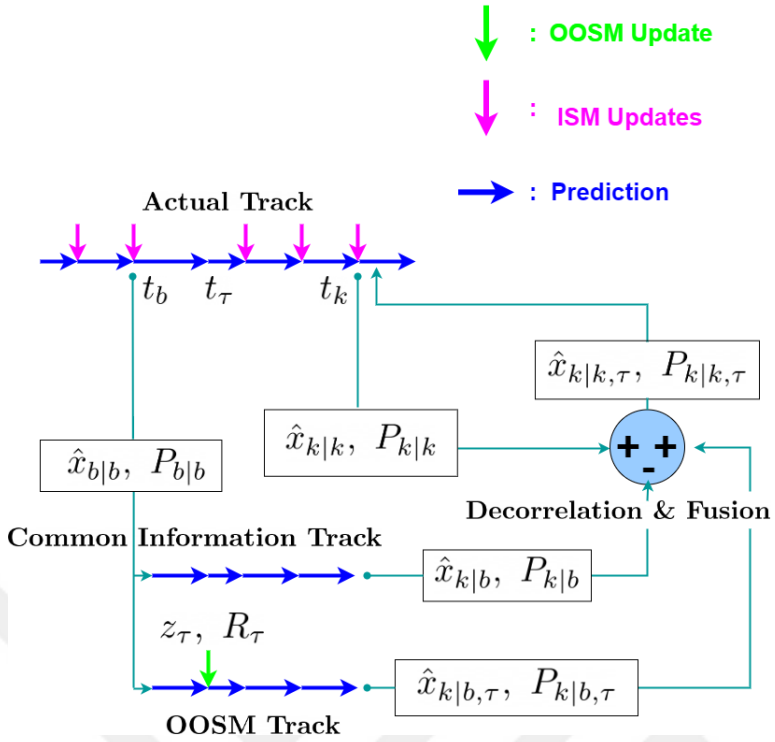


Figure 2.5: The illustration of the FPFD method for OOSM track update.

only the correlation caused by the common past is compensated. The reason behind this can be explained from the information-theoretic point of view as follows.

For one lag case, the current track estimate can be written as a combination of the predicted prior estimate information and the last measurement prior to OOSM information as follows:

$$P_{k|k}^{-1} = P_{k|b}^{-1} + H_k^T R_k^{-1} H_k, \tag{2.40}$$

$$P_{k|k}^{-1}\hat{x}_{k|k} = P_{k|b}^{-1}\hat{x}_{k|b} + H_k^T R_k^{-1} z_k, \tag{2.41}$$

which allows us to rewrite (2.38) and (2.39) by substituting the measurement-related terms for the ones related to the difference between the current estimate and the predicted prior estimate as below:

$$P_{k|k,\tau}^{-1} = P_{k|b,\tau}^{-1} + H_k^T R_k^{-1} H_k, \tag{2.42}$$

$$P_{k|k,\tau}^{-1} \hat{x}_{k|k,\tau} = P_{k|b,\tau}^{-1} \hat{x}_{k|b,\tau} + H_k^T R_k^{-1} z_k. \tag{2.43}$$

One can notice that the obtained equations are the information filter counterpart of the measurement update equations similar to (2.32) and (2.33), which reveals another way of interpreting the fusion carried out in FPFD. In this interpretation, it is considered that the solution is reached by updating the OOSM track with the equivalent measurement which is obtained by subtracting the predicted prior estimate information from actual track information rather than fusing the OOSM tracklet with the actual track. As a special case, in one lag scenario, the missing piece of information for the OOSM track to yield the optimal solution turns out exactly to be this extracted measurement information. For all the other cases, however, since the equivalent measurement doesn't satisfy (2.4) for some process noises, one of the fundamental KF assumptions is violated, thus the aforementioned update yields a suboptimal solution only. In other words, the performance of algorithm FPFD does degrade as the number of step-lag increases, compared to the in-sequence reprocessing of the measurements.

2.2 OOSM Problem for Jump Markov Linear Systems

In this subsection, firstly, the definition of the OOSM problem in JMLS (Jump Markov Linear Systems) is given, and then IMM (Interacting Multiple Model) filter, a standard estimation approach for JMLS when this problem does not exist (the measurements are in sequence), is described. Finally, the IMM application of the aforementioned Bl1, which adapts the standard estimation procedure of the IMM to the OOSM problem is presented.

2.2.1 OOSM Problem Definition

The problem is defined under the framework in which the state of a target of interest along with the measurement evolves over time according to linear stochastic dynamics which undergo a change in time also. The model of the described dynamics is referred to as Jump Markov Linear System(JMLS) and is expressed as follows:

$$x_k = F^{r_k}_{k,k-1} x_{k-1} + w^{r_k}_{k,k-1}, \tag{2.44a}$$

$$z_k = H^{r_k}_k x_k + v^{r_k}_k, \tag{2.44b}$$

where x_k and $r_k \in \{1, ..., N_r\}$ stand for the base state (i.e., continuous-valued states) and the mode state modeled by a homogeneous Markov chain respectively. All the terms with subscript r_k can be defined the same as in linear Gaussian Systems, emphasizing that they are mode dependent, i.e., depending on the mode state r_k. For example, the term $w^{r_k}_{k,k-1}$ denotes a mode dependent process noise with a covariance $Q^{r_k}_{k,k-1}$ whereas the term $v^{r_k}_k$ represents a mode dependent measurement noise with a covariance $R^{r_k}_k$.

The OOSM collected at time t_τ is given as:

$$z_\tau = H^{r_\tau}_\tau x_\tau + v^{r_\tau}_\tau, \tag{2.45}$$

where τ denotes the discrete-time instant OOSM originated at. Assume that at time t_k before the OOSM arrives, the state estimator for JMLS computes the posterior mode probabilities $\mu^i_{k|k}$ and, the mode-matched state estimates $\hat{x}^i_{k|k}$ and state covariances $P^i_{k|k}$, defined as below:

$$\mu^i_{k|k} \triangleq P(r_k = i|Z^k), \tag{2.46a}$$

$$\hat{x}^i_{k|k} \triangleq E[x_k|r_k = i, Z^k], \tag{2.46b}$$

$$P^i_{k|k} \triangleq E[(x_k - \hat{x}^i_{k|k})(x_k - \hat{x}^i_{k|k})^T|r_k = i, Z^k], \tag{2.46c}$$

where $i = 1, ..., N_r$.

Subsequently, the measurement z_τ arrives upon calculation of (2.46) which characterizes the joint posterior density. The problem emerges then as to how to update this density with an older measurement (2.45), namely, how to calculate

$$\mu^i_{k|k,\tau} = P(r_k = i|Z^k, z_\tau), \tag{2.47a}$$

$$\hat{x}^i_{k|k,\tau} = E[x_k|r_k = i, Z^k, z_\tau], \tag{2.47b}$$

$$P^i_{k|k,\tau} = E[(x_k - \hat{x}^i_{k|k})(x_k - \hat{x}^i_{k|k})^T|r_k = i, Z^k, z_\tau]. \tag{2.47c}$$

for $i = 1, ..., N_r$.

As in the linear Gaussian case, in order for ISM processing which will be discussed in the following subsection to be performed optimally, the next measurement is required to be collected later than that of the last measurement updating the joint density. For this reason, incorporating the older measurement into the current density again falls into the OOSM problem category.

2.2.2 In Sequence Estimation (IMM filter)

IMM filter is a suboptimal state estimation algorithm developed for the JMLS in (2.44) where the mode state process r_k is modeled by a homogeneous Markov chain with a transition probability matrix $\Pi_{k,k-1} = [\pi^{ji}_{k,k-1} \triangleq P(r_k = i|r_{k-1} = j)]$.

One can notice that, when $N_r = 1$, i.e., the JMLS consists of only a single model, the solution to the estimation problem turns out to be the Kalman filter in which the posterior distribution is represented by a single Gaussian. When $N_r > 1$, on the other hand, the posterior distribution $p(x_k|Z^k)$ is approximated by the IMM filter as a Gaussian mixture as follows:

$$p(x_k|Z^k) = \sum_{i=1}^{N_r} \mu^i_{k|k} \mathcal{N}(x_k; \hat{x}^i_{k|k}, P^i_{k|k}), \tag{2.48}$$

where $\hat{x}^i_{k|k}$ and $P^i_{k|k}$ represent mode-conditioned state estimate and covariance respectively while $\mu^i_{k|k}$ denotes the approximate posterior mode probability as before.

IMM filter iteratively calculates the current summary statistics $\{\mu^i_{k|k}, \hat{x}^i_{k|k}, P^i_{k|k}\}_{i=1}^{N_r}$ from the previous statistics $\{\mu^i_{k-1|k-1}, \hat{x}^i_{k-1|k-1}, P^i_{k-1|k-1}\}_{i=1}^{N_r}$, through the prediction update and measurement update steps as the KF. However, because of the nature of IMM running multiple Bayesian filters in an interactional manner, both steps are needed to be modified with additional operations. The modified IMM prediction update can be divided into three steps which are mode probability prediction, mixing, and mode-matched prediction update. In the first step the predicted mode probabilities $\{\mu^i_{k|k-1}\}_{i=1}^{N_r}$ are calculated as follows:

$$\mu^i_{k|k-1} = \sum_{j=1}^{N_r} \pi^{ji}_{k,k-1} \mu^j_{k-1|k-1} \tag{2.49}$$

for $i = 1, ..., N_r$.

In the mixing step, mixing probabilities $\mu_{ji}(k-1)$ to realize the interaction between models are calculated first as below:

$$\mu^{ji}_{k-1} = \frac{\pi^{ji}_{k,k-1} \mu^j_{k-1|k-1}}{\mu^i_{k|k-1}} \tag{2.50}$$

for $i, j = 1, ..., N_r$.

Then the interaction in which the estimate and the covariance of each mode contribute to another depending on the mixing probabilities is carried out. The mixed estimates $\{\hat{x}^{0i}_{k-1}\}_{i=1}^{N_r}$ and covariances $\{P^{0i}_{k-1}\}_{i=1}^{N_r}$ are calculated as follows:

$$\hat{x}^{0i}_{k-1} = \sum_{j=1}^{N_r} \mu^{ji}_{k-1} \hat{x}^j_{k-1|k-1}, \tag{2.51a}$$

$$P^{0i}_{k-1} = \sum_{j=1}^{N_r} \mu^{ji}_{k-1} [P^j_{k-1|k-1} + (\hat{x}^j_{k-1|k-1} - \hat{x}^{0i}_{k-1})(\cdot)^T] \tag{2.51b}$$

for $i = 1, ..., N_r$. In (2.51b) the notation $(x)(\cdot)^T$ denotes the outer product xx^T.

As the last step of the IMM prediction, mode-matched prediction update is performed for each model through KF time update equations (2.11) and predicted estimate $\hat{x}^i_{k|k-1}$ and $P^i_{k|k-1}$ are calculated from the mixed estimate and covariance as follows:

$$\hat{x}^i_{k|k-1} = F^i_{k,k-1}\hat{x}^{0i}_{k-1}, \tag{2.52a}$$

$$P^i_{k|k-1} = F^i_{k,k-1}P^{0i}_{k-1}(F^i_{k,k-1})^T + Q^i_{k,k-1} \tag{2.52b}$$

for $i = 1, ..., N_r$.

IMM measurement update also can be divided into two steps which are mode-matched measurement update and mode probability update. Mode-matched measurement update is carried out for each model through standard KF measurement update equations (2.12) using the corresponding mode parameters and updated estimate $\hat{x}^i_{k|k}$ and covariance $P^i_{k|k}$ are calculated from the predicted estimate $\hat{x}^i_{k|k-1}$ and covariance $P^i_{k|k-1}$ as below:

$$\hat{x}^i_{k|k} = \hat{x}^i_{k|k-1} + K^i_k(z_k - z^i_{k|k-1}), \tag{2.53a}$$

$$P^i_{k|k} = P^i_{k|k-1} - K^i_k S^i_k (K^i_k)^T, \tag{2.53b}$$

$$z^i_{k|k-1} = H^i_k \hat{x}^i_{k|k-1}, \tag{2.53c}$$

$$S^i_k = H^i_k P^i_{k|k-1}(H^i_k)^T + R^i_k, \tag{2.53d}$$

$$K^i_k = P^i_{k|k-1}(H^i_k)^T S^{i\,-1}_k \tag{2.53e}$$

for $i = 1, ..., N_r$.

In the second step, predicted mode probabilities found prior to the mixing are updated with a measurement z_k as follows:

$$\mu^i_{k|k} = \frac{\mu^i_{k|k-1}\mathcal{N}(z_k; z^i_{k|k-1}; S^i_k)}{\sum_{l=1}^{N_r} \mu^l_{k|k-1}\mathcal{N}(z_k; z^l_{k|k-1}; S^l_k)} \tag{2.54}$$

for $i = 1, ..., N_r$.

Now that the current summary statistics are obtained, the overall output estimate $\hat{x}_{k|k}$ and covariance $P_{k|k}$ of the posterior distribution of the base state in (2.48) can be calculated as follows:

$$\hat{x}_{k|k} = \sum_{i=1}^{N_r} \mu_{k|k}^i \hat{x}_{k|k}^i, \tag{2.55}$$

$$P_{k|k} = \sum_{i=1}^{N_r} \mu_{k|k}^i [P_{k|k}^i + (\hat{x}_{k|k}^i - \hat{x}_{k|k})(\cdot)^T]. \tag{2.56}$$

Since the parts of the IMM filter are going to be used as building blocks in the following section, we define the following functions.

- $\big[\{\bar{\mu}^i, \bar{x}^i, \bar{P}^i\}_{i=1}^{N_r}\big]$ = **IMMPrediction**$\big(\{\mu^i, x^i, P^i\}_{i=1}^{N_r}\big)$:
  This is the functional representation of the above-described IMM filter prediction (2.49)-(2.52) which takes the previous mode probabilities, estimates, and covariances $\{\mu_{k-1|k-1}^i, \hat{x}_{k-1|k-1}^i, P_{k-1|k-1}^i\}_{i=1}^{N_r}$ and returns the predicted mode probabilities, estimates, and covariances $\{\mu_{k|k-1}^i, \hat{x}_{k|k-1}^i, P_{k|k-1}^i\}_{i=1}^{N_r}$.
- $\big[\{\mu^i, x^i, P^i\}_{i=1}^{N_r}\big]$ =**IMMUpdate**$\big(\{\bar{\mu}^i, \bar{x}^i, \bar{P}^i\}_{i=1}^{N_r}, z\big)$:
  This is the functional representation of the above-described IMM filter update (2.53)-(2.54) which takes the predicted mode probabilities, estimates, and covariances $\{\mu_{k|k-1}^i, \hat{x}_{k|k-1}^i, P_{k|k-1}^i\}_{i=1}^{N_r}$ along with the measurement z_k and returns the updated mode probabilities, estimates, and covariances $\{\mu_{k|k}^i, \hat{x}_{k|k}^i, P_{k|k}^i\}_{i=1}^{N_r}$.

2.2.3 Existing Solutions for the OOSM Problem

In this subsection, we illustrate the existing solution for the OOSM problem for JMLS in the literature, which is the adaptation of the Bl1 algorithm to the IMM filter.

2.2.3.1 Bl1IMM Algorithm

Bl1IMM, the application of the Bl1 algorithm to IMM estimator [18], discussed hereafter. The main idea of Bl1IMM is to apply Bl1 to each mode of the IMM filter. The

procedure followed in the algorithm consists of state retrodiction, measurement retrodiction, mode likelihood function calculation, filter gain computation, and update of the state estimate and the current mode probabilities with the OOSM. The equations of the first step which yield mode matched retrodicted state and the corresponding covariance are given below:

$$\hat{x}^i_{\tau|k} = F^i_{\tau,k}\hat{x}^i_{k|k}, \tag{2.57a}$$

$$P^i_{\tau|k} = F^i_{\tau,k}[P^i_{k|k} + P^{ww,i}_{k,\tau|k} - P^{xw,i}_{k,\tau|k} - (P^{xw,i}_{k,\tau|k})^T](F^i_{\tau,k})^T \tag{2.57b}$$

for $i = 1, ..., N_r$.

The covariances related to the state retrodiction given above are expressed as follows:

$$P^{ww,i}_{k,\tau|k} = Q^i_{k,\tau}, \tag{2.58a}$$

$$P^{xw,i}_{k,\tau|k} = Q^i_{k,\tau} - P^i_{k|k-l}(S^{*i}_k)^{-1}Q^i_{k,\tau}, \tag{2.58b}$$

where

$$(S^{*i}_k)^{-1} = (P^i_{k|k-l})^{-1} - (P^i_{k|k-l})^{-1}P^i_{k|k}(P^i_{k|k-l})^{-1} \tag{2.59}$$

for $i = 1, ..., N_r$.

Recall that, the expression above, which follows from (2.28), is the essence of the Bl1 which reduces the l-step lag OOSM problem to the 1-step lag problem.

In the next step, the mode-matched retrodicted measurement and the mode-matched innovation covariance are calculated as follows:

$$\hat{z}^i_{\tau|k} = H^i_\tau \hat{x}^i_{\tau|k}, \tag{2.60a}$$

$$S^i_{\tau|k} = H^i_\tau P^i_{\tau|k} H^{iT}_\tau + R^i_\tau \tag{2.60b}$$

for $i = 1, ..., N_r$.

Now that we have $\hat{z}^i_{\tau|k}$ and $S^i_{\tau|k}$, the likelihood function of each mode, which is used for updating the mode probabilities is calculated as below:

$$\Lambda^i_\tau = \mathcal{N}(z_\tau; \hat{z}^i_{\tau|k}, S^i_{\tau|k}) \tag{2.61}$$

for $i = 1, ..., N_r$.

The equations related to mode-matched base state update are formulated as follows. First, mode-matched filter gain is obtained as below:

$$W^i_{k,\tau} = P^{xz,i}_{k,\tau|k}(S^i_{\tau|k})^{-1}, \tag{2.62}$$

where

$$P^{xz,i}_{k,\tau|k} = [P^i_{k|k} - P^{xw,i}_{k,\tau|k}](F^i_{\tau,k})^T(H^i_\tau)^T \tag{2.63}$$

for $i = 1, ..., N_r$.

Then using the mode-matched filter gain, the mode-matched current state estimate $\hat{x}^i_{k|k}$ and the corresponding covariance $P^i_{k|k}$ are updated with the OOSM by the following expressions:

$$\hat{x}^i_{k|k,\tau} = \hat{x}^i_{k|k} + W^i_{k,\tau}[z_\tau - z^i_{\tau|k}], \tag{2.64a}$$

$$P^i_{k|k,\tau} = P^i_{k|k} - P^{xz,i}_{k,\tau|k}(S^i_{\tau|k})^{-1}(P^{xz,i}_{k,\tau|k})^T \tag{2.64b}$$

for $i = 1, ..., N_r$.

Using the likelihood functions and the transition probability matrix $\pi^{ji}_{\tau,k}$, the current

mode probability for each mode $\mu^i_{k|k}$ are updated as follows:

$$\mu^i_{k|k,\tau} = \frac{1}{c}\sum_{j=1}^{N_r} \Lambda^j_\tau \pi^{ji}_{\tau,k} \mu^i_{k|k}, \tag{2.65}$$

where

$$c = \sum_{l=1}^{N_r}\sum_{j=1}^{N_r} \Lambda^j_\tau \pi^{jl}_{\tau,k} \mu^l_{k|k} \tag{2.66}$$

for $i = 1, ..., N_r$.

Now that all the sufficient statistics characterizing the joint posterior distribution $p(x_k, r_k|Z^k, z_\tau)$ are acquired, the overall output estimate $\hat{x}_{k|k,\tau}$ and covariance $P_{k|k,\tau}$ of the posterior distribution after the incorporation of OOSM can be calculated by marginalizing r_k out as follows:

$$\hat{x}_{k|k,\tau} = \sum_{i=1}^{N_r} \mu^i_{k|k,\tau} \hat{x}^i_{k|k,\tau}, \tag{2.67}$$

$$P_{k|k,\tau} = \sum_{i=1}^{N_r} \mu^i_{k|k,\tau} [P^i_{k|k,\tau} + (\hat{x}^i_{k|k,\tau} - \hat{x}_{k|k,\tau})(\cdot)^T]. \tag{2.68}$$

Bl1 algorithm discussed above is based on backward prediction. There is no forward prediction-based OOSM processing method such as FPFD applied to JMLSs in the literature. The aim of the thesis is to propose such a method.

2.3 Correlation Independent Fusion Methods

In this section, along with a detailed discussion of the correlation-independent fusion methods, an overview of the general fusion solutions to the correlation problem is given. These solutions whose aim is to fuse the information coming from two sources by removing the common information caused by two main correlation sources, previous communication, and process noise, can be categorized into three approaches. In the first category, the main idea is to extract the correlation information and utilize this in the fusion as in [26] in which the previous communication caused correlation

term is not addressed. For some fusion architectures, such a case where the correlation is dominated by the previous communication may occur. In that case, finding the transferred information may not be feasible because it necessitates possessing the knowledge of the fusion architecture and information flow, which may not be possible in practice. Moreover, for complex architectures, keeping track of the communicated information might be challenging too. Storage requirement is also a major concern since every piece of information that has been transmitted need to be stored. In addition, for some cases, correlation matrix calculations need to be made in every measurement update between fusion times (not just when there is fusion), which brings additional computational costs. For example, for the fusion architecture given in [23], even though the fusion takes place once in 3 measurement updates, computations are carried on because of the fact that the correlation matrix is calculated in a recursive manner according to [25].

The second category consists of methods developed to remove the correlations originating from the previous communication. In fusion architectures with memory or feedback, the fusion takes place between two tracks that share common information from the past. Examples can be given as the architectures called hierarchical with memory and hierarchical with feedback without memory diagrams from [21]. These methods apply a procedure for subtracting, i.e., decorrelating the common part incorporated into both tracks at the time they interacted in the past so that no double counting of the data occurs. This subtraction is performed on the transmitted or received information. If it is performed on the transmitted information, the decorrelated quantities are called equivalent measurements or tracklets (a pseudo track).

The last category involves the methods which provide a consistent fused estimate even though there is no correlation information available. The most well-known methods in this category are covariance intersection [27], ellipsoidal intersection [28], and LEA [29]. These methods assume the worst case at the cost of the independent information to preserve consistency regardless of correlations. As a result, depending on the fusion architecture and the information flow, they can yield overly conservative results with lower performance than the other two approaches by which correlations can be estimated, and compensated. As the proposed solution applies a method from the last category for fusion operation, these algorithms are discussed below.

2.3.1 Naive Fusion

Naive fusion, due to the optimistic assumption made within, is the most simplistic solution among all the correlation-independent fusion methods. The naive fusion on densities to be fused is composed of direct multiplication and normalization steps, which can be written as below:

$$\bar{f}(x) = \frac{f_1(x)f_2(x)}{\int f_1(x)f_2(x)dx}, \tag{2.69}$$

where $f_1(x)$ and $f_2(x)$ are the two probability density functions to be fused.

When both probability densities are Gaussian with mean and covariance $\hat{x}_1$,Σ_1 and $\hat{x}_2$,Σ_2 respectively, one can use the below multiplication rule in Gaussian algebra to obtain the fused density with mean $\hat{\bar{x}}$ and covariance $\bar{\Sigma}$:

$$\mathcal{N}(x;\hat{x}_1,\Sigma_1)\mathcal{N}(x;\hat{x}_2,\Sigma_2) = \mathcal{N}(\hat{x}_1;\hat{x}_2,\Sigma_1+\Sigma_2)\mathcal{N}(x;\hat{\bar{x}},\bar{\Sigma}), \tag{2.70}$$

where

$$\bar{\Sigma}^{-1} = \Sigma_1^{-1} + \Sigma_2^{-1}, \tag{2.71}$$

$$\bar{\Sigma}^{-1}\hat{\bar{x}} = \Sigma_1^{-1}\hat{x}_1 + {\Sigma_2}^{-1}\hat{x}_2. \tag{2.72}$$

After normalization, the first term on the right-hand side of (2.70) is canceled out and the fused density is obtained as $\mathcal{N}(x;\hat{\bar{x}},\bar{\Sigma})$. Regarding the inverse covariance as the information, since no subtraction of common prior information is performed, it is assumed that the maximum information is obtained with (2.71). In other words, when it comes to the dependency of the sources the naive approach optimistically assumes the best case in which the information sources are independent, i.e., there is no correlation between them at all. However, this assumption does not hold in general due to the process noise caused by the motion of the target under observation common to both of the information sources. When the independence can not be assured, the fused covariance could be much smaller than in the reality. As a result of this, the

obtained estimate might have a large error and this error may grow unbounded over time, i.e., divergence. The reason for that is by counting on the estimates to be independent the fused estimate gets overconfident and starts ignoring measurements in subsequent filtering.

2.3.2 Covariance Intersection (CI)

The unreliability of the naive fusion due to the performance and consistency issues mentioned above motivates the development of the Covariance Intersection. With this motivation, CI is designed to generate a consistent solution to the fusion problem under unknown correlations [27].

As CI is generalized by the CF (Chernoff Fusion) [30], which can be applied to all random variables, first CF is expressed mathematically:

$$f(x) = \frac{f_1(x)^{w^*} f_2(x)^{(1-w^*)}}{\int f_1(x)^{w^*} f_2(x)^{(1-w^*)} dx}, \tag{2.73}$$

where $w^* \, \epsilon \, [0, 1]$ denotes an optimization parameter minimizing some cost function which will be discussed below while $f_1(x)$, $f_2(x)$ are two probability density functions being fused.

In the case where the densities are Gaussian, the fusion takes the special name Covariance Intersection and (2.73) can be simplified to the analytical expressions below:

$$\bar{\Sigma}^{-1} = w^* \Sigma_1^{-1} + (1 - w^*) \Sigma_2^{-1}, \tag{2.74}$$

$$\bar{\Sigma}^{-1} \hat{\bar{x}} = w^* \Sigma_1^{-1} \hat{x}_1 + (1 - w^*) {\Sigma_2}^{-1} \hat{x}_2. \tag{2.75}$$

The optimization procedure applied for finding w^* is given as follows:

$$\bar{\Sigma}(w)^{-1} = w \Sigma_1^{-1} + (1 - w) \Sigma_2^{-1}, \tag{2.76}$$

$$w^* = \arg \min_{w \epsilon [0,1]} g(\bar{\Sigma}(w)). \tag{2.77}$$

In the above, optimization parameter w is found by minimizing the cost function denoted by $g(.)$ which can either be the trace or determinant operator. Moreover, as these functions are convex with respect to w, there is only one minimum within the interval and thus almost any optimization method can be applied. After finding w^* the fused estimate is obtained by taking a convex combination as in (2.74) and (2.75).

After giving the mathematical procedure of the CI, the discovery by which the CI is inspired is discussed here. The discovery is that the fused estimate's covariance ellipsoid (for a covariance matrix Σ this is the level curve points s : $s^T \Sigma^{-1} s$= c where c is a constant) is always inside of the intersection of the ellipsoids of the covariances to be fused regardless of what the cross-correlation matrix is. Thus, the presented method comes up with a procedure so that the covariance of the fused estimate always encircles the intersection region as illustrated in Fig. 2.6. In this way, for all choices of the cross-correlation covariance matrix, a consistent result is achieved which is proven in Appendix A of [27]. This is the reason why the CI method is referred to as pessimistic in its estimation since it always expects the worst in terms of the correlation, i.e., full correlation case. Notice here that naive fusion and CI represent the two edges of the fusion approaches in terms of the correlation amount they assume.

Contrary to naive fusion, CI assumes the worst-case scenario in which some of the independent information is not used by compromising the quality of the estimate for the sake of consistency which is arguably more important. Since the method provides a consistent solution, which means the fused covariance is larger than or equal to (in a positive semi-definite sense) the true covariance of the fused estimate, it tackles the divergence problem which may occur when applying naive fusion.

2.3.3 Largest Ellipsoid Algorithm (LEA)

As mentioned earlier, while naive fusion underestimates the actual covariance resulting in inconsistency issues, CI finds an upper bound for the actual covariance in order

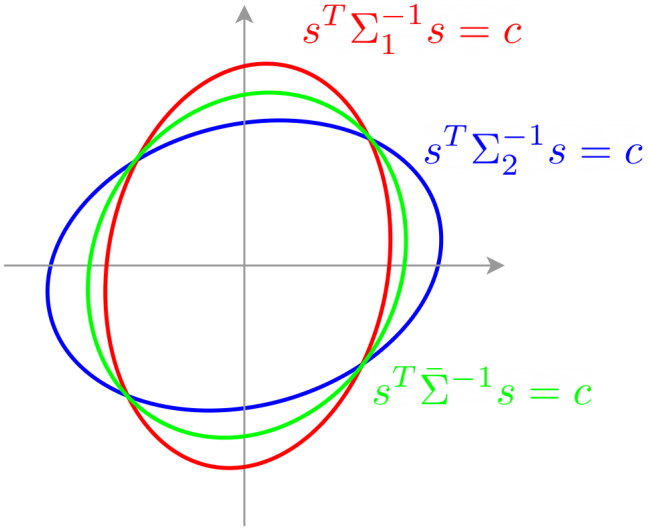


Figure 2.6: The illustration of the resulting ellipsoid after CI fusion.

to bring a robust solution to the fusion problem. Nonetheless, although inconsistency is avoided with the latter approach, since the fusion is performed conservatively, independent information is not used. Moreover, as the fused covariance matrix is greater than the actual one, in subsequent local filtering the measurements contribute to the estimate more than they should, i.e., the estimate is contaminated by the measurement noise which leads to a decrease in performance. In order to improve the performance, the largest ellipsoid algorithm is introduced, which is less conservative but consistent and based on the estimation of the intersection region of covariance matrices.[29].

The fact that the intersection region defined by the intersection of the covariance ellipsoids of the fused sources represents an upper limit for the actual covariance matrix and as CI overestimates means that there is room for improvement. LEA decreases this overestimated region by computing the largest ellipsoid contained within this intersection region, which leads to an improvement in performance. The procedure for the largest ellipsoid algorithm consists of decoupling the states in the estimates and keeping the most informative one from each estimate as the corresponding state of the fused estimate. Since the orientation of the covariance ellipsoids is different in general, a direct comparison of states in the estimates is not possible. To solve this problem, both covariance matrices are diagonalized jointly which will be discussed next.

Allow x_1 and x_2 to be the correlated estimates of x, for which mean and covariances are $\hat{x}_1$,Σ_1 and $\hat{x}_2$,Σ_2 respectively. In the first step, a linear transformation obtained by eigendecomposition of Σ_1 is applied to make the states in the estimate independent as follows:

$$\Sigma_1 = U_1 \Lambda_1 U_1^{-1}, \tag{2.78}$$

where U_1 is the matrix columns of which are composed of eigenvectors, and Λ_1 is the diagonal matrix whose diagonal elements are the eigenvalues.

As U_1^T = U_1^{-1} due to Σ_1 being a symmetric matrix, a linear transformation T_1 which decorrelates x_1 can be defined as below:

$$T_1 = \Lambda_1^{-1/2} U_1^T. \tag{2.79}$$

The transformed estimate x_1^1 with mean $\hat{x}_1^1$ and covariance Σ_1^1 is given as follows:

$$\hat{x}_1^1 = T_1 \hat{x}_1, \tag{2.80}$$

$$\Sigma_1^1 = T_1 \Sigma_1 T_1^T, \tag{2.81}$$

where Σ_1^1 is an identity matrix which means the states of the transformed estimate x_1^1 are now decoupled and the covariance ellipsoids (level curves) turn into circles. The same transformation is also applied to the estimate x_2 as follows:

$$\hat{x}_2^1 = T_1 \hat{x}_2, \tag{2.82}$$

$$\Sigma_2^1 = T_1 \Sigma_2 T_1^T, \tag{2.83}$$

where Σ_2^1 is the covariance matrix of the transformed estimate x_2^1, whose states are still correlated. In the second step, another transformation is applied to make the states in x_2 independent. For this purpose, eigendecomposition of Σ_2^1 is obtained as below:

$$\Sigma_2^1 = U_2 \Lambda_2 U_2^T. \tag{2.84}$$

The second transformation which only consists of rotation, i.e., there is no scaling as in (2.38), is defined as below:

$$T_2 = U_2^T. \tag{2.85}$$

By applying T_2 to the estimates transformed by T_1, jointly diagonalized estimates $\hat{x}_1^{12}$ and $\hat{x}_2^{12}$ are obtained as follows:

$$\hat{x}_1^{12} = T_2 \hat{x}_1^1, \tag{2.86a}$$

$$\Sigma_1^{12} = T_2 \Sigma_1^1 T_2^T = I, \tag{2.86b}$$

$$\hat{x}_2^{12} = T_2 \hat{x}_2^1, \tag{2.86c}$$

$$\Sigma_2^{12} = T_2 \Sigma_2^1 T_2^T = \Lambda_2. \tag{2.86d}$$

Notice that the states of x_1^{12} remain independent after the rotational transformation T_2 as they are equally uncertain in all directions prior to the operation. After joint diagonalization, since the states in the estimates become orthogonal to each other and hence are ready to be compared, the most informative estimate can be found for each direction separately (independently), i.e., by comparing the diagonal elements and choosing the smallest one. This procedure is written mathematically as follows:

$$[\hat{\bar{x}}^{12}]_i \triangleq \begin{cases} [\hat{x}_1^{12}]_i & [\Lambda_2]_{ii} \geq 1 \\ [\hat{x}_2^{12}]_i & [\Lambda_2]_{ii} < 1 \end{cases}, \tag{2.87}$$

$$[\bar{\Sigma}^{12}]_{ij} \triangleq \begin{cases} [\Sigma_1^{12}]_{ii} & i = j, [\Lambda_2]_{ii} \geq 1 \\ [\Sigma_2^{12}]_{ii} & i = j, [\Lambda_2]_{ii} < 1 \\ 0 & i \neq j \end{cases} \tag{2.88}$$

for $i, j \in \{1, ..., n_x\}$ where n_x denotes the number of elements in the state vector.

where $\hat{\bar{x}}^{12}$ and $\bar{\Sigma}^{12}$ represent the mean and the covariance of the fused estimate in the transformed space respectively. Thus, the fused estimate in the initial (original) space is obtained by applying the reverse of the two transformations, where the mean is computed according to [24].

$$\hat{\bar{x}} = T^{-1}\hat{\bar{x}}^{12}, \tag{2.89a}$$

$$\bar{\Sigma} = T^{-1}\bar{\Sigma}^{12}T^{-T}, \tag{2.89b}$$

where the overall transformation denoted as T is written as below:

$$T = T_2 T_1, \tag{2.90}$$

$$= U_2^T \Lambda_1^{-1/2} U_1^T. \tag{2.91}$$

Consequently, the largest ellipsoid algorithm yields an estimate with a covariance enclosing the intersection region tighter than the overestimated one computed by CI as can be seen in Fig. 2.8, and thus, a considerable amount of useful information is recovered. The overall process steps are illustrated in Fig. 2.7.

Since the described fusion methods are going to be used as building blocks in the next section, we define the following functions.

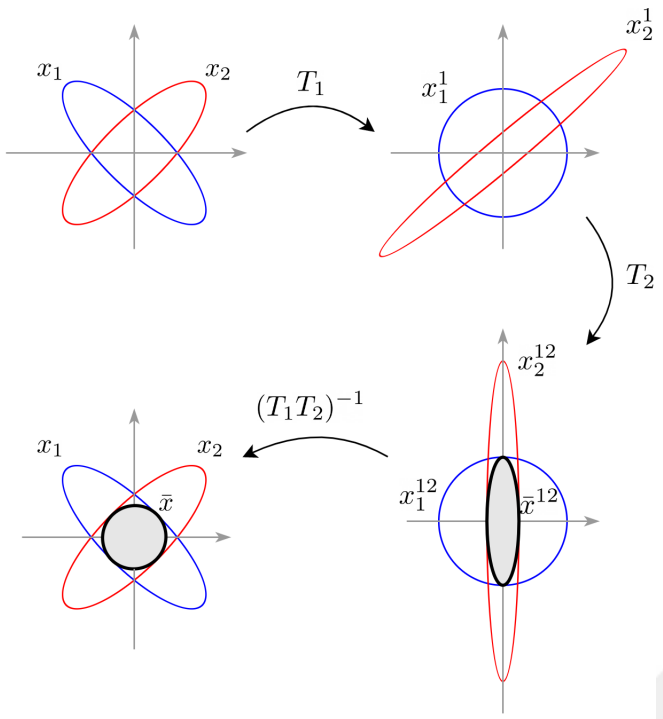


Figure 2.7: The illustration of the steps of LEA between the two estimates x_1 and x_2.

- $[\bar{x}, \bar{\Sigma}]$ = **Naive**(x_1, Σ_1, x_2, Σ_2):
  This is the functional representation of the above-described Naive fusion (2.71)-(2.72) which takes the state estimate and covariance of the Gaussian densities to be fused x_1, Σ_1, x_2, Σ_2 and returns the fused state estimate and covariance $\bar{x}$, $\bar{\Sigma}$.

- $[\bar{x}, \bar{\Sigma}, w^*]$ = **CI**(x_1, Σ_1, x_2, Σ_2):
  This is the functional representation of the above-described CI fusion (2.74)-(2.77) which takes the state estimate and covariance of the Gaussian densities to be fused x_1, Σ_1, x_2, Σ_2 and returns the fused state estimate and covariance $\bar{x}$, $\bar{\Sigma}$ along with the optimization parameter w^* found by (2.77).

- $[\bar{x}, \bar{\Sigma}]$ = **LEA**(x_1, Σ_1, x_2, Σ_2):
  This is the functional representation of the above-described LEA fusion (2.78)-(2.91) which takes the state estimate and covariance of the Gaussian densities to be fused x_1, Σ_1, x_2, Σ_2 and returns the fused state estimate and covariance $\bar{x}$, $\bar{\Sigma}$.

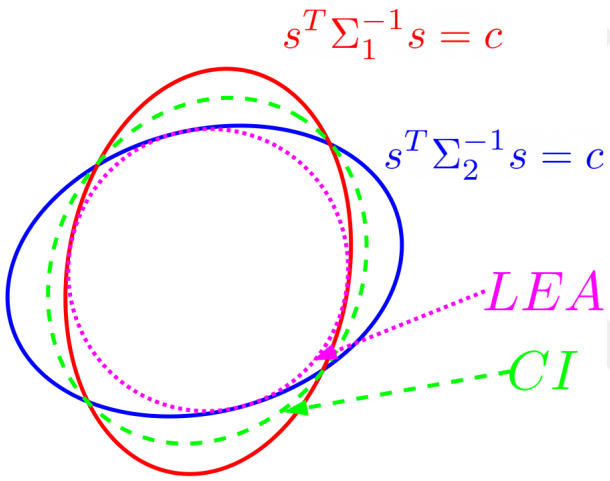


Figure 2.8: The illustration of the resulting ellipsoids after CI and LEA fusion.

CHAPTER 3

PROPOSED METHOD

In this chapter, we propose a forward prediction-based OOSM processing method such as FPFD, which can be applied to linear Gaussian systems as well as to JMLSs. Since the direct adaptation of FPFD to JMLSs is not feasible due to the decorrelation problem, which will be discussed in detail in the following sections, the main goal of the proposed solution is to tackle this correlation problem.

The proposed method overcomes the aforementioned challenge by eliminating the separate decorrelation step and utilizing a fusion algorithm in which the decorrelation is handled within it. For this reason, we refer to our proposed solution as FPF (Forward Prediction and Fusion). In the following, the proposed solution FPF will be described first for linear Gaussian and then for jump Markov linear systems.

3.1 Proposed Solution for Linear Gaussian Systems

In this section, first, the algorithm of the proposed solution for linear Gaussian systems is given. Data storage need and computational cost of the FPF algorithm will be presented and compared with the existing solutions next.

3.1.1 Algorithm

The algorithm of the proposed solution for linear Gaussian systems which consists of two parts, i.e., forward prediction and fusion is given as follows.

FPF Algorithm

- **Given:** $\{\hat{x}_{b|b}, P_{b|b}, \hat{x}_{k|k}, P_{k|k}, z_\tau\}$.
- **Initialization:**

$$\hat{\bar{x}}_{b|b} = \hat{x}_{b|b}, \tag{3.1a}$$

$$\bar{P}_{b|b} = P_{b|b}. \tag{3.1b}$$

- **Forward Prediction:**

$$[\hat{\bar{x}}_{\tau|b}, \bar{P}_{\tau|b}] = \text{KFPrediction}(\hat{\bar{x}}_{b|b}, \bar{P}_{b|b}), \tag{3.2a}$$

$$[\hat{\bar{x}}_{\tau|b,\tau}, \bar{P}_{\tau|b,\tau}] = \text{KFUpdate}(\hat{\bar{x}}_{\tau|b}, \bar{P}_{\tau|b}, z_\tau), \tag{3.2b}$$

$$[\hat{\bar{x}}_{k|b,\tau}, \bar{P}_{k|b,\tau}] = \text{KFPrediction}(\hat{\bar{x}}_{\tau|b,\tau}, \bar{P}_{\tau|b,\tau}). \tag{3.2c}$$

- **Fusion:**

  $if \quad Fusion\ Method\ is\ Naive$

$$[\hat{x}_{k|k,\tau}, P_{k|k,\tau}] = \text{Naive}(\hat{\bar{x}}_{k|b,\tau}, \bar{P}_{k|b,\tau}, \hat{x}_{k|k}, P_{k|k}). \tag{3.3}$$

  $elseif \quad Fusion\ Method\ is\ CI$

$$[\hat{x}_{k|k,\tau}, P_{k|k,\tau}] = \text{CI}(\hat{\bar{x}}_{k|b,\tau}\bar{P}_{k|b,\tau}, \hat{x}_{k|k}, P_{k|k}). \tag{3.4}$$

  $elseif \quad Fusion\ Method\ is\ LEA$

$$[\hat{x}_{k|k,\tau}, P_{k|k,\tau}] = \text{LEA}(\hat{\bar{x}}_{k|b,\tau}, \bar{P}_{k|b,\tau}, \hat{x}_{k|k}, P_{k|k}). \tag{3.5}$$

  end

In the given algorithm, forward prediction is performed almost as in FPFD. The only difference is, compared to the forward prediction of FPFD which uses two auxiliary filters, here only one auxiliary filter is used. In the first step of the algorithm, this auxiliary filter is initialized as in (3.1) with an estimate $\hat{x}_{b|b}$ calculated by the principal filter before OOSM measurement is collected. This estimate is then predicted (3.2a) to the time t_τ at which the OOSM arrives. In the next step, the predicted estimate is updated with OOSM through (3.2b).

Finally, the OOSM track is predicted to the current time (3.2c). The forward prediction is illustrated at the bottom of Fig. 3.1. Since the decorrelation is carried out implicitly by a fusion algorithm, the knowledge of how much two tracks are correlated is not required, which, as mentioned previously, distinguishes our solution from FPFD. For this reason, the second auxiliary filter employed for decorrelation purposes in the FPFD is not utilized, which reduces the computational burden of the FPF.

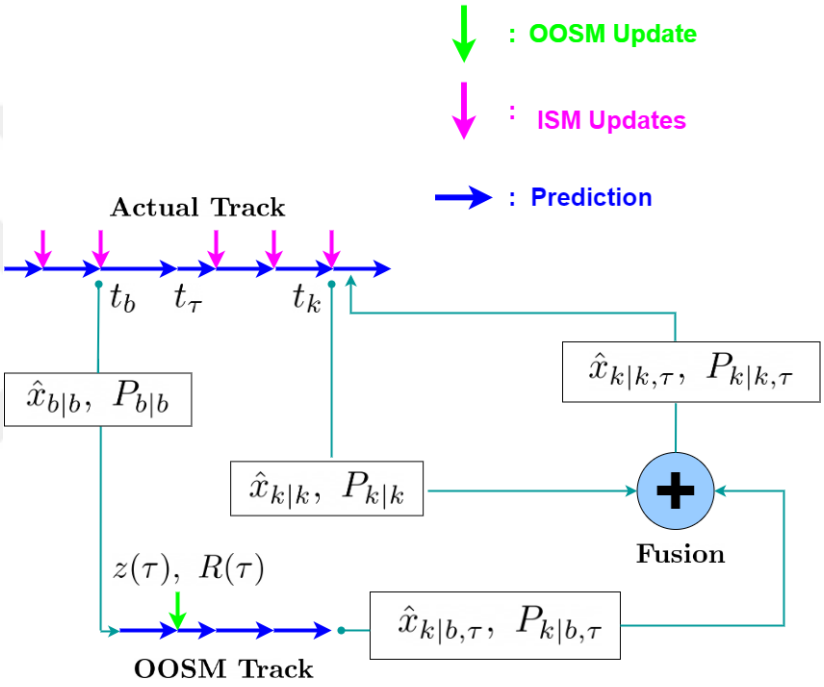


Figure 3.1: The illustration of the proposed method for OOSM track update.

In the second and final part, one of the previously discussed correlation-independent fusion algorithms, Naive, CI, and LEA, is chosen to combine the predicted OOSM track with the current track. As a result of the fusion operation, the estimate $\hat{x}_{k|k,\tau}$ which contains the OOSM information is obtained. In Fig. 3.1, the OOSM incorporation process of FPF is illustrated. In the following sections, depending on which fusion method it applies, the algorithm may be referred to as "FPF followed by the name of that method", e.g., FPF CI.

3.1.2 Data Storage Requirement

Even though the quantities needed to be stored may not seem large while processing the OOSM for a single target, as the number of tracked targets increases the data storage might become an important issue. Therefore, in this subsection, the storage requirements of the proposed methods and the existing solutions are analyzed and then compared considering the number of scalars they need to hold in memory.

To begin with, all the methods necessitate storing the last update time t_k along with the estimate $\hat{x}_{k|k}$ and the covariance $P_{k|k}$ calculated at that time. This amounts to 1, n, and $n(n+1)/2$ number of scalars respectively, where n stands for the number of state variables. In addition to the current quantities, forward prediction-based FPFD and FPF methods also require storing the time, state estimate, and covariance values between $t_{k-l_{max}}$ and t_{k-1}, where l_{max} represents the maximum expected lag.

For Bl1, on the other hand, it is sufficient to store the time and state covariance values between $t_{k-l_{max}}$ and t_{k-1}, i.e. the previous state estimate values are not required to be stored in Bl1. The reason for this can be explained as follows. In Bl1, past quantities are stored for the calculation of the covariance of the equivalent measurement. Since the previous estimates are not required in the calculation of this covariance value, they do not need to be stored. This is the reason for subtracting the nl_{max} term in (3.7).

In summary, the total number of scalars needed to be stored for each algorithm is given below:

$$\text{FPFD}: \left[\frac{l_{max}+1}{2}\right](n^2+3n+2), \tag{3.6}$$

$$\text{B}l1: \left[\frac{l_{max}+1}{2}\right](n^2+3n+2) - nl_{max}, \tag{3.7}$$

$$\text{FPF Naive}: \left[\frac{l_{max}+1}{2}\right](n^2+3n+2), \tag{3.8}$$

$$\text{FPF CI}: \left[\frac{l_{max}+1}{2}\right](n^2+3n+2), \tag{3.9}$$

$$\text{FPF LEA}: \left[\frac{l_{max}+1}{2}\right](n^2+3n+2). \tag{3.10}$$

Table 3.1 presents the data storage requirements for different l_{max} values when the number of state variables in x_k is 6.

l_{max}	**1**	**2**	**3**	**4**
FPFD	56	84	112	140
Algorithm Bl1	50	72	94	116
FPF Naive	56	84	112	140
FPF CI	56	84	112	140
FPF LEA	56	84	112	140

Table 3.1: Number of scalars required to be stored in terms of l_{max} for a state vector with 6 elements.

As can be seen from the table, the storage requirements all depend on the value of l_{max}, and the difference in storage requirements between the prediction-based algorithms and Bl1 grows as the maximum delay gets larger.

3.1.3 Computational Cost

In this subsection, a computation-wise comparison of the proposed methods and the existing solutions is presented. The computational costs of the algorithms FPFD, Bl1, and FPF in terms of the number of operations performed per OOSM are shown below in Table 3.2 where N_e is the number of function evaluations required by the solution of the optimization problem in FPF CI.

In the forward prediction step of the FPFD, two KF time updates and a single KF measurement update are performed. In subsequent steps, two more measurement updates are applied, assuming that the decorrelation and fusion processes have the same computational complexity of a single KF measurement update. In Bl1, along with a single measurement update for the incorporation of the OOSM, a total of two time updates are performed, one for the calculation of $P(k|k-l)$ which is required to obtain the equivalent measurement and one for the state retrodiction. Since FPF Naive is the simplified version

Operations	# of Meas. Updates	# of Time Updates	# of Diags.
FPFD	3	3	-
Bl1	1	2	-
FPF Naive	2	2	-
FPF LEA	1	2	2
FPF CI	$N_e + 1$	2	-

Table 3.2: Number of operations required to be performed for any l.

of FPFD in which the decorrelation step is not performed, the computational cost of FPF is less than that of FPFD by one time update for the calculation of the information to be decorrelated in the forward prediction step and one measurement update for decorrelation step. The computational cost of the FPF CI consists of two time updates and a single measurement update carried out for forward prediction, and N_e measurement updates required for evaluating the cost functions while solving the CI optimization problem. In the FPF LEA, two time updates and one measurement update for the forward prediction as well as two diagonalizations (Diags.) performed within the LEA fusion make up the computational cost.

3.2 Proposed Solution for Jump Markov Linear Systems

The encountered challenge in applying FPFD to JMLSs is that it involves a decorrelation process corresponding to the division of two densities, which cannot be performed in a straightforward manner since the division of JMLSs' posterior distributions (i.e., Gaussian mixtures) does not have an analytical expression as for the Kalman Filter. In [20] it is discussed that even though the ratio of two Gaussian mixtures is again a mixture, the components turn out to be non-Gaussian, which leads to the development of approximate solutions to work with IMMs. For example [20] approximates these components with Gaussians by importance sampling. Another approximate solution is proposed in [31], in which the decorrelation is not applied on the Gaussian mixtures but

on the joint posterior of the continuous base-state and discrete mode-state provided by the IMM filter. However, even in this approach, since IMM involves several approximations, some implementation issues arise, and one of them concerns the non-invertibility of the decorrelation information matrix [23]. In order to avoid these challenging and problematic issues an extension of the forward prediction-based OOSM processing solution proposed in the previous section for jump Markov linear systems is presented next.

3.2.1 Algorithm

The algorithm of the proposed solution for jump Markov linear systems which consists of two parts, i.e., forward prediction and fusion is given as follows.

FPFIMM Algorithm

- **Given:** $\{\mu^i_{b|b}, \hat{x}^i_{b|b}, P^i_{b|b}\}_{i=1}^{N_r}, \{\mu^i_{k|k}, \hat{x}^i_{k|k}, P^i_{k|k}\}_{i=1}^{N_r}, z_\tau$.
- **Initialization:**

$$\bar{\mu}^i_{b|b} = \mu^i_{b|b}, \qquad i = 1, ..., N_r, \tag{3.11a}$$

$$\hat{\bar{x}}^i_{b|b} = \hat{x}^i_{b|b}, \qquad i = 1, ..., N_r, \tag{3.11b}$$

$$\bar{P}^i_{b|b} = P^i_{b|b}, \qquad i = 1, ..., N_r. \tag{3.11c}$$

- **Forward Prediction:**

$$\left[\{\bar{\mu}^i_{\tau|b}, \hat{\bar{x}}^i_{\tau|b}, \bar{P}^i_{\tau|b}\}_{i=1}^{N_r}\right] = \text{IMMPrediction}\left(\{\bar{\mu}^i_{b|b}, \hat{\bar{x}}^i_{b|b}, \bar{P}^i_{b|b}\}_{i=1}^{N_r}\right), \tag{3.12a}$$

$$\left[\{\bar{\mu}^i_{\tau|b,\tau}, \hat{\bar{x}}^i_{\tau|b,\tau}, \bar{P}^i_{\tau|b,\tau}\}_{i=1}^{N_r}\right] = \text{IMMUpdate}\left(\{\bar{\mu}^i_{\tau|b}, \hat{\bar{x}}^i_{\tau|b}, \bar{P}^i_{\tau|b}\}_{i=1}^{N_r}, z_\tau\right), \tag{3.12b}$$

$$\left[\{\bar{\mu}^i_{k|b,\tau}, \hat{\bar{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}\}_{i=1}^{N_r}\right] = \text{IMMPrediction}\left(\{\bar{\mu}^i_{\tau|b,\tau}, \hat{\bar{x}}^i_{\tau|b,\tau}, \bar{P}^i_{\tau|b,\tau}\}_{i=1}^{N_r}\right). \tag{3.12c}$$

- **Fusion:**
  - ∗ **Mode Matched Fusion:**

    $for \quad i = 1 : N_r$

$if \quad Fusion\ Method\ is\ Naive$

$$[\hat{x}^i_{k|k,\tau}, P^i_{k|k,\tau}] = \text{Naive}(\hat{\bar{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}, \hat{x}^i_{k|k}, P^i_{k|k}). \tag{3.13}$$

$elseif \quad Fusion\ Method\ is\ CI$

$$[\hat{x}^i_{k|k,\tau}, P^i_{k|k,\tau}, w^*] = \text{CI}(\hat{\bar{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}, \hat{x}^i_{k|k}, P^i_{k|k}). \tag{3.14}$$

$elseif \quad Fusion\ Method\ is\ LEA$

$$[\hat{x}^i_{k|k,\tau}, P^i_{k|k,\tau}] = \text{LEA}(\hat{\bar{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}, \hat{x}^i_{k|k}, P^i_{k|k}). \tag{3.15}$$

end

end

* **Mode Probability Fusion:**

$if \quad Fusion\ Method\ is\ Naive\ or\ LEA$

$$[\mu^i_{k|k,\tau}] = \text{ModeProbabilityUpdateNaive} \left(\{\bar{\mu}^i_{k|b,\tau}, \hat{\bar{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}\}_{i=1}^{N_r}, \{\mu^i_{k|k}, \hat{x}^i_{k|k}, P^i_{k|k}\}_{i=1}^{N_r}\right). \tag{3.16}$$

$elseif \quad Fusion\ Method\ is\ CI$

$$[\mu^i_{k|k,\tau}] = \text{ModeProbabilityUpdateCI} \left(w^*, \{\bar{\mu}^i_{k|b,\tau}, \hat{\bar{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}\}_{i=1}^{N_r}, \{\mu^i_{k|k}, \hat{x}^i_{k|k}, P^i_{k|k}\}_{i=1}^{N_r}\right). \tag{3.17}$$

end

In the forward prediction part of the algorithm, first, the auxiliary IMM filter used to carry the information coming from the OOSM is initialized with the summary statistics of the principal filter at time t_b. Then, forward prediction is performed as in the FPF by applying the IMM filter counterparts (3.2) of the prediction and update equations of KF. As a result, summary statistics of the OOSM track $\{\bar{\mu}^i_{k|b,\tau}, \hat{\bar{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}\}_{i=1}^{N_r}$ are obtained.

In the fusion part of the algorithm, the information contained in the forward predicted auxiliary filter is incorporated into the summary statistics of the principal (original) filter in two steps i.e., mode matched estimate and covariance fusion step and mode probability fusion step. In the former, each mode of the

primary filter is combined with the corresponding identical mode of its duplicated version by one of (3.13)-(3.15), depending on the chosen fusion method.

As a result, mode matched fused estimate and covariance statistics $\{\hat{\bar{x}}^i_{k|k,\tau}, \bar{P}^i_{k|k,\tau}\}_{i=1}^{N_r}$ are obtained. In the next and final step, depending on the method chosen in the previous step, the mode probabilities of the principal filter are combined with those of the auxiliary filter by one of (3.16)-(3.17), formulated and defined in the following derivations of the fusion methods.

3.2.1.1 Naive Fusion applied in FPFIMM

Just as in [31] decorrelation is performed over joint densities, in FPFIMM naive fusion is applied to joint densities $p(x_k, r_k|Z^k)$ and $\bar{p}(x_k, r_k|Z^b, z_\tau)$, which are scaled Gaussian distributions given as follows.

$$p(x_k, r_k|Z^k) = \mu^{r_k}_{k|k}\mathcal{N}(x_k, \hat{x}^{r_k}_{k|k}, P^{r_k}_{k|k}), \tag{3.18a}$$

$$\bar{p}(x_k, r_k|Z^b, z_\tau) = \mu^{r_k}_{k|b,\tau}\mathcal{N}(x_k, \bar{\hat{x}}^{r_k}_{k|b,\tau}, \bar{P}^{r_k}_{k|b,\tau}). \tag{3.18b}$$

Notice here that, FPFIMM naive fusion can be applied to these two joint densities because the auxiliary IMM filter utilizes the identical motion models as the principal filter.

Naive fusion of these two joint distributions gives the following fused joint distribution.

$$p(x_k, r_k|Z^k, z_\tau) \propto p(x_k, r_k|Z^k)\bar{p}(x_k, r_k|Z^b, z_\tau), \tag{3.19a}$$

$$= \mu^{r_k}_{k|k}\bar{\mu}^{r_k}_{k|b,\tau}\mathcal{N}(x_k, \hat{x}^{r_k}_{k|k}, P^{r_k}_{k|k})\mathcal{N}(x_k, \bar{\hat{x}}^{r_k}_{k|b,\tau}, \bar{P}^{r_k}_{k|b,\tau}). \tag{3.19b}$$

By applying the well-known multiplication rule of the Gaussian algebra, (3.19b) can be written as a scaled Gaussian, which results in

$$p(x_k, r_k|Z^k, z_\tau) \propto \mu_{k|k}^{r_k} \bar{\mu}_{k|b,\tau}^{r_k} \mathcal{N}(\hat{x}_{k|k}^{r_k}, \bar{\hat{x}}_{k|b,\tau}^{r_k}, P_{k|k}^{r_k} + \bar{P}_{k|b,\tau}^{r_k}) \mathcal{N}(x_k, \hat{x}_{k|k,\tau}^{r_k}, P_{k|k,\tau}^{r_k}). \tag{3.20}$$

The fused mode probabilities are proportional to the scaling coefficient in (3.20), as expressed below

$$\mu_{k|k,\tau}^{r_k} \propto \mu_{k|k}^{r_k} \bar{\mu}_{k|b,\tau}^{r_k} \mathcal{N}(\hat{x}_{k|k}^{r_k}, \bar{\hat{x}}_{k|b,\tau}^{r_k}, P_{k|k}^{r_k} + \bar{P}_{k|b,\tau}^{r_k}). \tag{3.21}$$

Thus the fused mode probabilities $\mu_{k|k,\tau}^{r_k}$ can be found after normalization as follows

$$\mu_{k|k,\tau}^{r_k} = \frac{\mu_{k|k}^{r_k} \bar{\mu}_{k|b,\tau}^{r_k} \mathcal{N}(\hat{x}_{k|k}^{r_k}, \bar{\hat{x}}_{k|b,\tau}^{r_k}, P_{k|k}^{r_k} + \bar{P}_{k|b,\tau}^{r_k})}{\sum_{r_k=1}^{N_r} \mu_{k|k}^{r_k} \bar{\mu}_{k|b,\tau}^{r_k} \mathcal{N}(\hat{x}_{k|k}^{r_k}, \bar{\hat{x}}_{k|b,\tau}^{r_k}, P_{k|k}^{r_k} + \bar{P}_{k|b,\tau}^{r_k})}. \tag{3.22}$$

As a result of combining the mode probabilities, the complete updated summary statistics $\{\bar{\mu}_{k|k,\tau}^i, \bar{\hat{x}}_{k|k,\tau}^i, \bar{P}_{k|k,\tau}^i\}_{i=1}^{N_r}$ are obtained. In the following sections, this method is referred to as FPFIMM Naive.

The function of the above-described mode probability update (3.16) used in the algorithm can now be defined as follows.

– $[\tilde{\mu}^i]$ = **ModeProbabilityUpdateNaive**$\big(\{\mu^i, \hat{x}^i, P^i\}_{i=1}^{N_r}, \{\bar{\mu}^i, \bar{\hat{x}}^i, \bar{P}^i\}_{i=1}^{N_r}\big)$: This is the functional representation of the above described mode probability update (3.21) which takes the summary statistics of the principal and auxiliary IMM filters $\{\mu_{k|k}^i, \hat{x}_{k|k}^i, P_{k|k}^i\}_{i=1}^{N_r}$ and $\{\bar{\mu}_{k|b,\tau}^i, \bar{\hat{x}}_{k|b,\tau}^i, \bar{P}_{k|b,\tau}^i\}_{i=1}^{N_r}$ respectively and returns the updated/fused mode probabilities $\mu_{k|k,\tau}^{r_k}$.

3.2.1.2 CI Fusion applied in FPFIMM

In FPFIMM, Chernoff fusion, an extension of CI, is applied on the joint densities $p(x_k, r_k|Z^k)$ and $\bar{p}(x_k, r_k|Z^b, z_\tau)$, which yields the fused information given as follows.

$$p(x_k, r_k|Z^k, z_\tau) \propto \left(p(x_k, r_k|Z^k)\right)^w \left(\bar{p}(x_k, r_k|Z^b, z_\tau)\right)^{1-w} \tag{3.23a}$$

$$= (\mu_{k|k}^{r_k})^w (\bar{\mu}_{k|b,\tau}^{r_k})^{1-w} \left(\mathcal{N}(x_k, \hat{x}_{k|k}^{r_k}, P_{k|k}^{r_k})\right)^w \left(\mathcal{N}(x_k, \bar{\hat{x}}_{k|b,\tau}^{r_k}, \bar{P}_{k|b,\tau}^{r_k})\right)^{1-w} \tag{3.23b}$$

Using the fact that, the wth power of the Gaussian distribution $\mathcal{N}(x, \hat{x}, P)$ turns out to be as the scaled Gaussian as below:

$$(\mathcal{N}(x, \hat{x}, P))^w = c(w)\mathcal{N}(x; \hat{x}, w^{-1}P), \tag{3.24}$$

where

$$c(w) \triangleq \frac{(|2\pi w^{-1}P|)^{0.5}}{(|2\pi P|)^{w/2}}, \tag{3.25}$$

(3.23) can be written as follows

$$p(x_k, r_k|Z^k, z_\tau) \propto (\mu_{k|k}^{r_k})^w (\bar{\mu}_{k|b,\tau}^{r_k})^{1-w} c_{k|k}^{r_k}(w) \bar{c}_{k|b,\tau}^{r_k}(w) \mathcal{N}(x_k, \hat{x}_{k|k}^{r_k}, w^{-1}P_{k|k}^{r_k}) \times \mathcal{N}(x_k, \bar{\hat{x}}_{k|b,\tau}^{r_k}, (1-w)^{-1}\bar{P}_{k|b,\tau}^{r_k}). \tag{3.26}$$

By applying the multiplication rule, (3.26) can be written as a scaled Gaussian, which results in

$$p(x_k, r_k|Z^k, z_\tau) \propto (\mu_{k|k}^{r_k})^w (\bar{\mu}_{k|b,\tau}^{r_k})^{1-w} c_{k|k}^{r_k}(w) \bar{c}_{k|b,\tau}^{r_k}(w) \times \mathcal{N}\left(\hat{x}_{k|k,b}^{r_k}, \bar{\hat{x}}_{k|b,\tau}^{r_k}, \frac{\bar{P}_{k|k}^{r_k}}{w} + \frac{\bar{P}_{k|b,\tau}^{r_k}}{1-w}\right) \mathcal{N}(x_k, \hat{x}_{k|k,\tau}^{r_k}, P_{k|k,\tau}^{r_k}), \tag{3.27}$$

where mode-conditioned state estimates $\hat{x}_{k|k,\tau}^{r_k}$ and covariances $P_{k|k,\tau}^{r_k}$ are given by CI function in (3.14) described at the end of the section on correlation independent fusion methods in Chapter 2.

The fused mode probabilities $\mu^{r_k}_{k|k,\tau}$ are proportional to the scaling coefficient in (3.27), as expressed below

$$\mu^{r_k}_{k|k,\tau} \propto (\mu^{r_k}_{k|k})^{w}(\bar{\mu}^{r_k}_{k|b,\tau})^{1-w} c^{r_k}_{k|k}(w)\bar{c}^{r_k}_{k|b,\tau}(w)\mathcal{N}\left(\hat{x}^{r_k}_{k|k}, \bar{\hat{x}}^{r_k}_{k|b,\tau}, \frac{P^{r_k}_{k|k}}{w} + \frac{\bar{P}^{r_k}_{k|b,\tau}}{1-w}\right), \tag{3.28}$$

and hence can be found after normalization over different modes as follows

$$\mu^{r_k}_{k|k,\tau} = \frac{(\mu^{r_k}_{k|k})^{w}(\bar{\mu}^{r_k}_{k|b,\tau})^{1-w} c^{r_k}_{k|k}(w)\bar{c}^{r_k}_{k|b,\tau}(w)\mathcal{N}\left(\hat{x}^{r_k}_{k|k}, \bar{\hat{x}}^{r_k}_{k|b,\tau}, \frac{P^{r_k}_{k|k}}{w} + \frac{\bar{P}^{r_k}_{k|b,\tau}}{1-w}\right)}{\sum_{r_k=1}^{N_r}(\mu^{r_k}_{k|k})^{w}(\bar{\mu}^{r_k}_{k|b,\tau})^{1-w} c^{r_k}_{k|k}(w)\bar{c}^{r_k}_{k|b,\tau}(w)\mathcal{N}\left(\hat{x}^{r_k}_{k|k}, \bar{\hat{x}}^{r_k}_{k|b,\tau}, \frac{P^{r_k}_{k|k}}{w} + \frac{\bar{P}^{r_k}_{k|b,\tau}}{1-w}\right)}. \tag{3.29}$$

The mode probabilities of the fused estimate are obtained by substituting the optimal value w^* given by the CI function in (3.14) for w in (3.29).

As a result of combining the mode probabilities, the complete updated summary statistics $\{\bar{\mu}^i_{k|k,\tau}, \bar{\hat{x}}^i_{k|k,\tau}, \bar{P}^i_{k|k,\tau}\}_{i=1}^{N_r}$ are obtained. In the following sections, this method is referred to as FPFIMM CI.

The function of the above-described mode probability update (3.29) used in the algorithm can now be defined as follows.

– $[\tilde{\mu}^i]$ = **ModeProbabilityUpdateCI**$\big(w, \{\mu^i, \hat{x}^i, P^i\}_{i=1}^{N_r}, \{\bar{\mu}^i, \bar{\hat{x}}^i, \bar{P}^i\}_{i=1}^{N_r}\big)$:
This is the functional representation of the above described mode probability update (3.21) which takes the optimum value w^*, summary statistics of the principal and auxiliary IMM filters $\{\mu^i_{k|k}, \hat{x}^i_{k|k}, P^i_{k|k}\}_{i=1}^{N_r}$ and $\{\bar{\mu}^i_{k|b,\tau}, \bar{\hat{x}}^i_{k|b,\tau}, \bar{P}^i_{k|b,\tau}\}_{i=1}^{N_r}$ respectively and returns the updated/fused mode probabilities $\mu^{r_k}_{k|k,\tau}$.

3.2.1.3 LEA Fusion applied in FPFIMM

Since LEA is not a method that can be applied to distributions like Chernoff and Naive fusion, its derivation cannot be given in the same manner as was done for the methods mentioned above. Nonetheless, by considering the architecture that enables mode-by-mode fusion as revealed by the above derivations, the following solution can be proposed. In this solution, mode-matched estimate and covariance fusion is performed by LEA in (3.15), while the fused mode probabilities are calculated in (3.16) by the mode probability update function derived for naive fusion.

In the following sections, this method is referred to as FPFIMM LEA.

3.2.2 Data Storage Requirement

In this subsection, the storage requirements of the proposed methods and the algorithm Bl1IMM are analyzed and then compared in terms of the number of scalars they need to hold in memory.

To begin with, all the methods necessitate storing the last update time t_k along with the summary statistics $\{\mu^i_{k|k}, \hat{x}^i_{k|k}, P^i_{k|k}\}_{i=1}^{N_r}$ obtained at that time. In addition to these current quantities, all the methods require to keep the summary statistics obtained from $t_{k-l_{max}}$ to t_{k-1} and the respective update times, which shows that storage requirements depend on the value of l_{max}.

$$\text{All Methods} : (l_{max}+1)\left[N_r\left(\frac{n^2+3n+2}{2}\right)+1\right]. \tag{3.30}$$

$$\tag{3.31}$$

Table 3.3 presents the data storage requirements for different l_{max} values when the number of state variables in x_k is 4.

It can be noticed from the table that when Bl1 is adapted to IMM, the method loses its storage advantage over forward prediction-based methods.

l_{max}	**1**	**2**	**3**	**4**
BlIMM1	62	93	124	155
FPFIMM Naive	62	93	124	155
FPFIMM CI	62	93	124	155
FPFIMM LEA	62	93	124	155

Table 3.3: Number of scalars required to be stored in terms of l_{max} when the number of state variables in x_k is 4 and number of IMM filter models is 2.

3.2.3 Computational Cost

In this subsection, computation-wise comparison of the proposed FPFIMM methods and the existing solutions is presented.

Table 3.4 below shows the computational costs of Bl1IMM and FPFIMM methods in terms of the number of operations they perform per OOSM:

Ops.	**# of Meas. Upds.**	**# of Time Upds.**	**# of Mixs.**	**# of Diags.**
Bl1IMM	N_r	$2N_r$	1	-
FPFIMM Naive	$2N_r$	$2N_r$	2	-
FPFIMM LEA	N_r	$2N_r$	2	$2N_r$
FPFIMM CI	$N_r(N_e+1)$	$2N_r$	2	-

Table 3.4: Number of operations required to be performed in IMM application for any l_{max}.

where computational costs of the mode probability update operation are not given since they do not bring a significant computational burden.

Since KF measurement update and time update operations are applied by N_r filters in the JMLS case, the number of operations performed is N_r times the values in Table 3.2. In addition to these operations, Bl1IMM performs an IMM mixing to compute $P^i_{k|k-l}$ for the calculation of equivalent measurement covariance.

Moreover, two IMM mixings performed in the forward prediction step are added to the computational cost of FPFIMM methods and since fusion is performed for each mode, the number of diagonalization operations of the FPFIMM LEA increases by a factor of N_r compared to FPF LEA.

3.3 Advantages and Disadvantages

3.3.1 Advantages

1. Forward prediction-based FPF and FPFIMM methods are intuitive compared to the retrodiction-based Bl1, as the natural flow of time is forward. Therefore, unlike Bl1 and Bl1IMM, they do not require the state transition matrix to be invertible.
2. By combining information in a conservative manner, FPF can be expected to bring an improvement in estimation performance against FPFD or Bl1 in some cases, as it does not suffer from the previously discussed process noise dependence problem of other methods.
3. FPFIMM method requires no modification in the IMM filter implementation when one of the models is non-linear such as the coordinated turn model. In such a case, only the IMM prediction update operations carried out in the forward prediction step of the proposed method are expected to undergo changes. Since the standard KF estimating equations (2.52) cannot be applied due to the nonlinearities, one modification is required in the mode-matched prediction update step. If there are differences in dimensionality and type of the model states, modification is also required in the IMM mixing step.

3.3.2 Disadvantages

1. As can be seen from Table 3.2, the computational cost of all FPF methods is higher than that of Bl1, which also applies to the comparison of FPFIMM and Bl1IMM. The FPF naive has fewer operations to perform than the FPFD, but this lower computational load comes at the cost of

some performance. While the computational cost of the FPF LEA is expected to be comparable to the FPFD, the FPF CI has the potential to be computationally more expensive than the FPFD depending on N_e due to the measurement update operation load brought by the optimization step. These comparisons will be supported by the average CPU time results of the Monte Carlo simulations as imprecise approximations of the computational complexity of the algorithms in the next section.

2. As the property of conservativeness mentioned in advantage 2. may also lead to some independent information not being used in the fusion, FPF is expected to yield degraded estimation performance against FPFD or Bl1 in some cases.

CHAPTER 4

NUMERICAL RESULTS

In this chapter, the performance of the FPF and FPFIMM algorithms is tested under various simulation scenarios and compared with existing solutions based on the numerical results obtained.

4.1 Performance Testing of FPF Methods

The performance of the proposed FPF methods will be analyzed for two cases in this section. In the first case, the measurement model is linear, while in the second case, where a more practical scenario is examined, the nonlinear measurement model is considered.

4.1.1 2-D Linear Measurement Model Example

In this example, as a kinematic model (2.1), discretized continuous white noise acceleration model commonly used in tracking is considered[32]. The state vector x_k composed of two positions denoted as p and two velocities denoted as v one for each coordinate is defined as follows:

$$x_k \triangleq \begin{bmatrix} p^x & v^x & p^y & v^y \end{bmatrix}^T, \tag{4.1}$$

where superscripts x and y denote the dimensions.

Starting with the state transition matrix, the model parameters are given below:

$$F_{k,k-1} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.2}$$

where T (taken as 1 sec in the upcoming scenarios) stands for the sampling interval. The covariance of the accumulated continuous white noise acceleration is written as follows:

$$Q_{k,k-1} = q \begin{bmatrix} T^3/3 & T^2/2 & 0 & 0 \\ T^2/2 & T & 0 & 0 \\ 0 & 0 & T^3/3 & T^2/2 \\ 0 & 0 & T^2/2 & T \end{bmatrix}, \tag{4.3}$$

where q denotes the process noise intensity which is modeled to be the same for each coordinate. Since only the positions are measured in the following examples, measurement matrices H_k and H_τ are written as follows

$$H_k = H_\tau = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{4.4}$$

and the measurement noise covariances R_k and R_τ are given as follows

$$R_k = \begin{bmatrix} \sigma_{p^x}^2 & 0 \\ 0 & \sigma_{p^y}^2 \end{bmatrix}, \qquad R_\tau = \begin{bmatrix} \rho_{p^x}^2 & 0 \\ 0 & \rho_{p^y}^2 \end{bmatrix}, \tag{4.5}$$

where σ_{p^x},σ_{p^y} are the standard deviations for the in-sequence position measurements and ρ_{p^x},ρ_{p^y} are the standard deviations for the OOSMs.

4.1.1.1 Single-Lag Scenario

Here we consider a scenario where 5 OOSMs with a single lag arrive at random time instants alongside sequential measurements collected at each sampling interval. These delayed (OOSM) measurements are obtained immediately after t_{k-1} i.e., exactly one lag case. The considered scenario uses above described discretized continuous white noise acceleration model with three different values for the process noise intensity (q=25 $\mathrm{m}^2/\mathrm{s}^3$,100$\mathrm{m}^2/\mathrm{s}^3$,400$\mathrm{m}^2/\mathrm{s}^3$) and the measurement models where σ_{p^x} = ρ_{p^y} = 100 m and σ_{p^y} = ρ_{p^x} = 25 m.

Based on the above-described scenario, the FPF methods are compared with the in-sequence measurements reprocessing method (In-seq), OOSM ignoring, and the existing solutions FPFD and Bl1. The in-sequence measurement reprocessing method which provides the optimal solution is the simple approach which reprocesses all the past measurements chronologically starting from the OOSM time t_τ, and the ignore approach simply discards the OOSM. Related to the methods using CI, we found from simulation studies that there is no significant improvement in the performance of the FPFIMM CI method when N_e is chosen large, while there is a significant increase in the computational cost, i.e. the improvement in the performance is not worth the increase in the computational cost. Therefore, for this and all the following scenarios and cases, N_e is chosen as 3 to make these methods feasible in practice.

The performance results of the algorithms are obtained through the simulations which take 50 time steps. The Kalman filters of each method have identical parameters, which are matched to the true target parameters given above. The KFs are initialized with the state estimate and covariance given below.

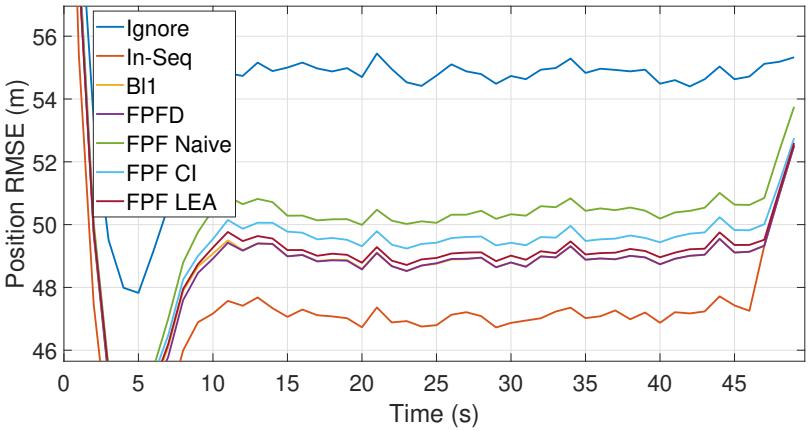
$$\hat{x}_{0|0} = \begin{bmatrix} 5000\mathrm{m} & 20\mathrm{m/s} & 5000\mathrm{m} & 20\mathrm{m/s} \end{bmatrix}^T, \tag{4.6}$$

$$P_{0|0} = \mathrm{diag}\begin{bmatrix} 100^2\mathrm{m}^2 & 5^2\mathrm{m}^2/\mathrm{s}^2 & 100^2\mathrm{m}^2 & 5^2\mathrm{m}^2/\mathrm{s}^2 \end{bmatrix}. \tag{4.7}$$

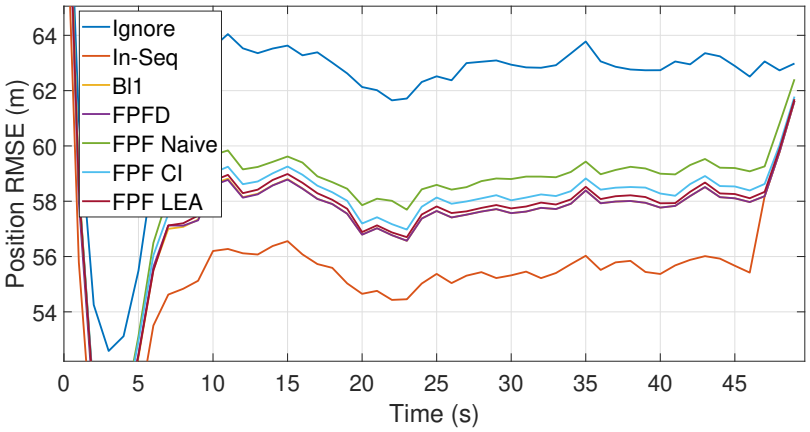
The true initial state of the target is realized from a Gaussian random variable distributed as $\mathcal{N}(x; \hat{x}_{0|0}, P_{0|0})$ at the beginning of each Monte Carlo (MC) run.

A total of 10000 Monte Carlo runs are made by changing the target state trajectory, measurement noise realizations, and time instants of OOSM arrivals. For different process noise intensities, the root mean square (RMS) position errors, RMS velocity errors, and the mean normalized estimation error squared (NEES) are shown in Fig. 4.1-4.4. Notice that Fig 4.4 is a zoomed version of Fig 4.3. As for NEES, the closer the value of the average NEES is to the number of state variables in x_k (4 in our case), the better the consistency properties of the filter, i.e., the covariance matrix computed by the filter better represents the true estimation error. On a more evaluative level, a method can be said to be consistent if the NEES results of the method are within the confidence region, bounded by the upper (99.5% probability) and lower (0.5% probability) thresholds, denoted as γ_U, γ_L, which will be shown later in the respective graphs.

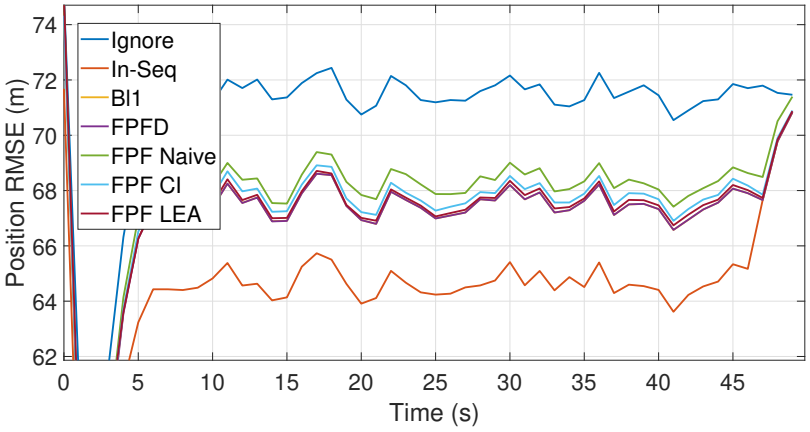
As can be seen from the figures, the solutions with the lowest RMSE are FPFD and Bl1 as expected. In terms of RMS position and velocity errors, both FPFD and Bl1 have very similar performances. Outside of these methods, the FPF LEA stands out as the best-performing method. The reason for this is that the OOSM solutions have the characteristics of fusion methods as can be deduced from Fig. 4.3. In other words, FPF Naive has larger NEES because of the overconfident behavior of the Naive fusion, which leads to inconsistent processing of subsequent measurements. This is why naive yields the worst RMSE performance among all. Furthermore, due to the overly conservative nature of the fusion algorithm CI, FPF CI has lower NEES, leading to less confident processing of future data and higher RMSE than FPF LEA. Also notice that as process noise intensity q increases, RMS position and velocity errors of the OOSM methods approach those of Ignore, which implies that all methods have some form of ignoring capability. This is due to increased process noise reducing the useful information content of the OOSM. All in all, among the FPF methods for one-lag problems, it seems reasonable to prefer FPF LEA since FPF Naive is not consistent and FPF CI has larger RMSE results due to sacrificing some more information than FPF LEA.



(a) $q = 25m^2/s^3$.

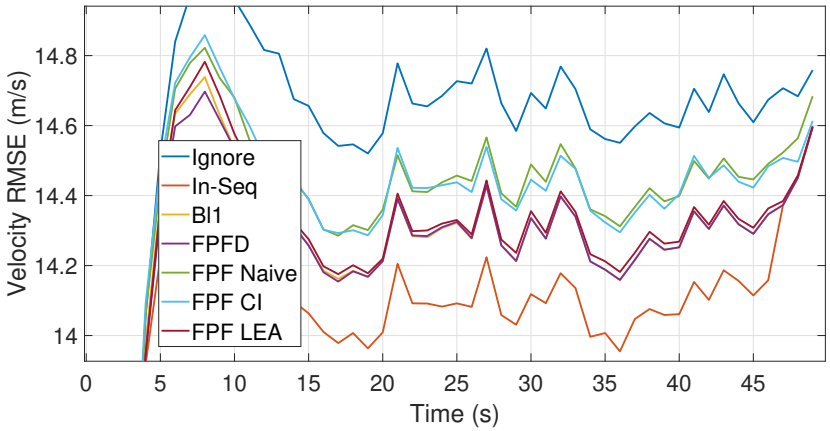


(b) $q = 100m^2/s^3$.

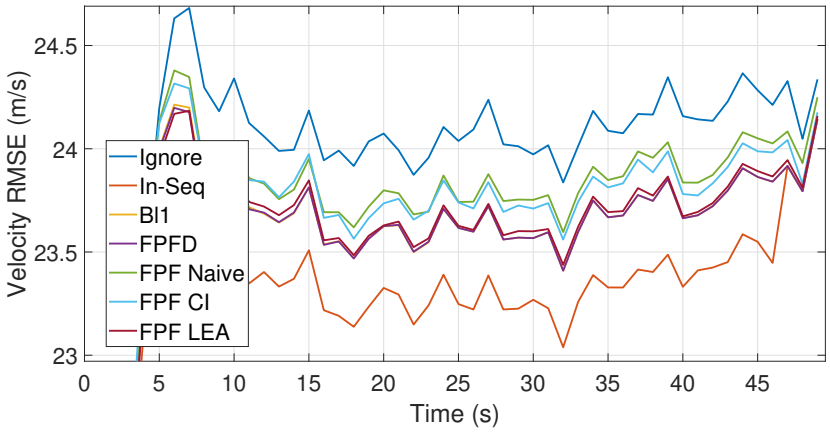


(c) $q = 400m^2/s^3$.

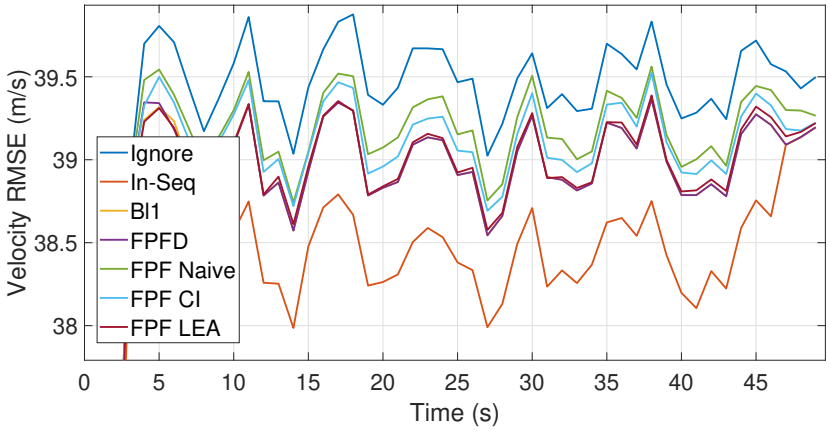
Figure 4.1: Position RMSE comparison of FPF methods and existing solutions for different process noise intensities.



(a) $q = 25m^2/s^3$.

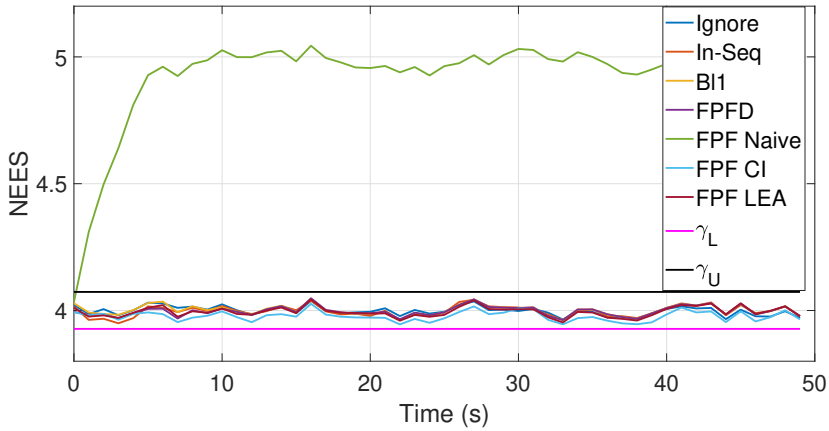


(b) $q = 100m^2/s^3$.

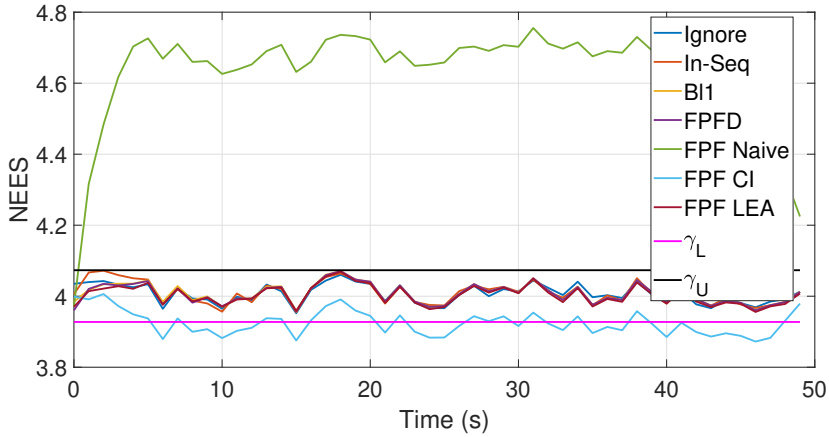


(c) $q = 400m^2/s^3$.

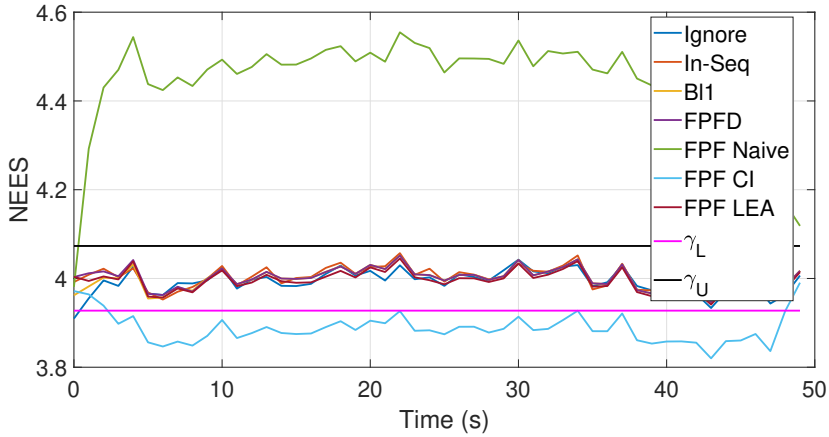
Figure 4.2: Velocity RMSE comparison of different FPF methods and existing solutions for different process noise intensities.



(a) $q = 25m^2/s^3$.

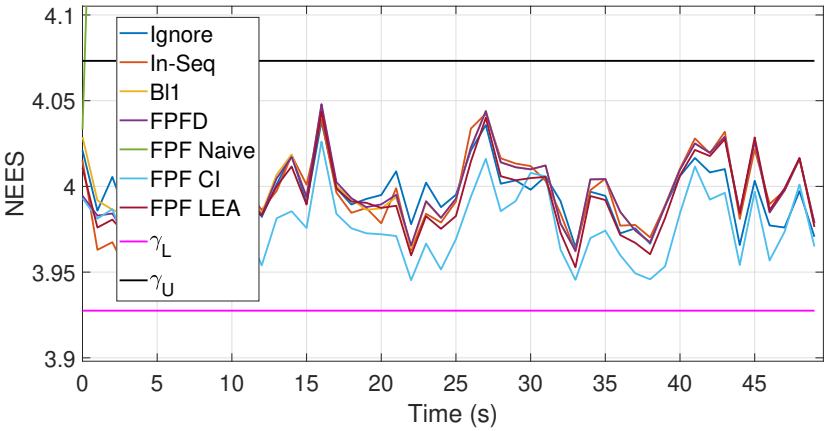


(b) $q = 100m^2/s^3$.

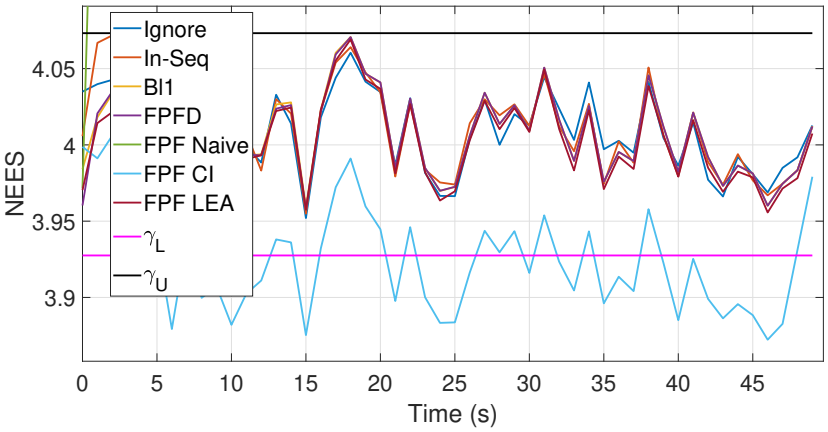


(c) $q = 400m^2/s^3$.

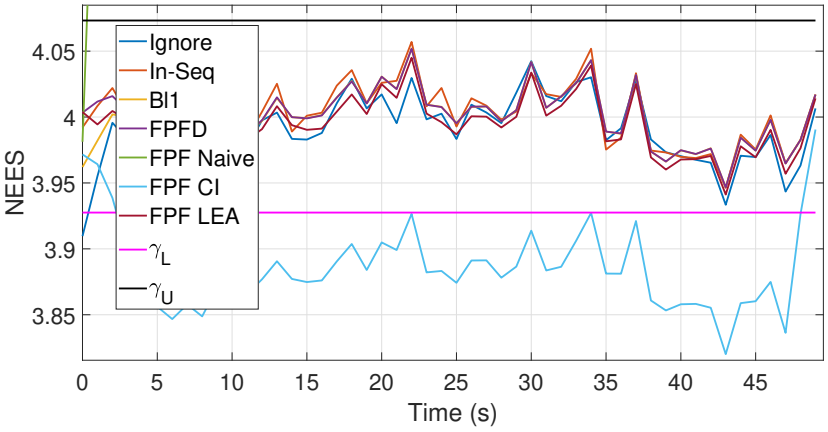
Figure 4.3: NEES comparison of FPF methods and existing solutions for different process noise intensities.



(a) $q = 25m^2/s^3$.



(b) $q = 100m^2/s^3$.



(c) $q = 400m^2/s^3$.

Figure 4.4: NEES comparison of FPF methods and existing solutions for different process noise intensities (Zoomed version of Fig. 4.3).

4.1.1.2 Multi-Lag Scenario

This scenario is identical to the one-lag scenario, except that previously the methods were compared for different process noise intensities, whereas now the methods will be compared for different lags (l = 1, 2, 3).

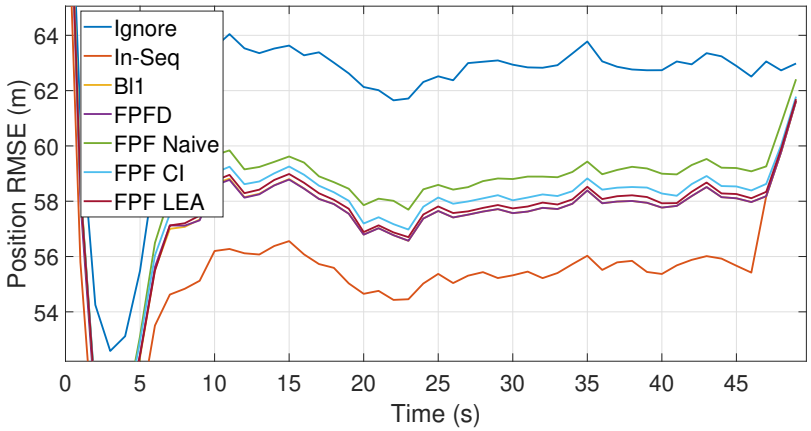
The considered scenario uses discretized continuous white noise acceleration model with the process noise intensity q=100 m^2/s^3 and the measurement models where σ_{p^x} = ρ_{p^y} = 100 m and σ_{p^y} = ρ_{p^x} = 25 m.

The performance results of the algorithms are obtained through the simulations which take 50 time steps. A total of 10000 Monte Carlo runs are made by changing the target state trajectory, measurement noise realizations, and time instants of OOSM generation in each run. For different lags, the RMS position errors, RMS velocity errors, and the mean NEES are shown in Fig. 4.5-4.8. Notice that Fig. 4.8 is a zoomed version of Fig. 4.7.

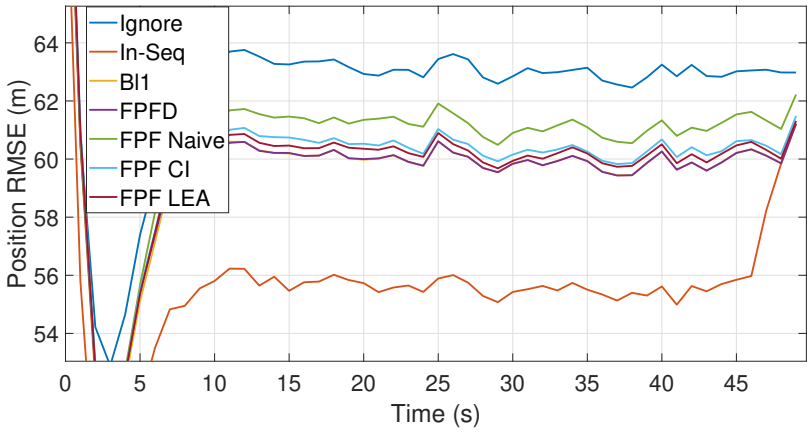
As can be seen from the figures, much of the previous discussion on the numerical results of single-lag scenarios can also be applied to the multi-lag case. For example, FPFD and Bl1 are the best algorithms in terms of RMS position and velocity errors. In addition, FPF LEA outperforms the other FPF methods. Also notice that as the number of lag increases, RMSEs of the OOSM methods approach those of Ignore. In fact, in the case with 3 lag, FPF CI gives the same RMSE results as Ignore in Figures 4.5c and 4.6c, which proves the OOSM ignoring capability of FPF CI. This is due to increased process noise diminishing the information content of measurement. However, ignoring OOSM means sacrificing information; hence, FPF CI performs as badly as FPF Naive.

The performance comparison of all these approaches in terms of time-averaged RMS position and velocity errors for different lags ($l = 1, 2, 3, 4, 5$) is given in Fig. 4.9-4.10.

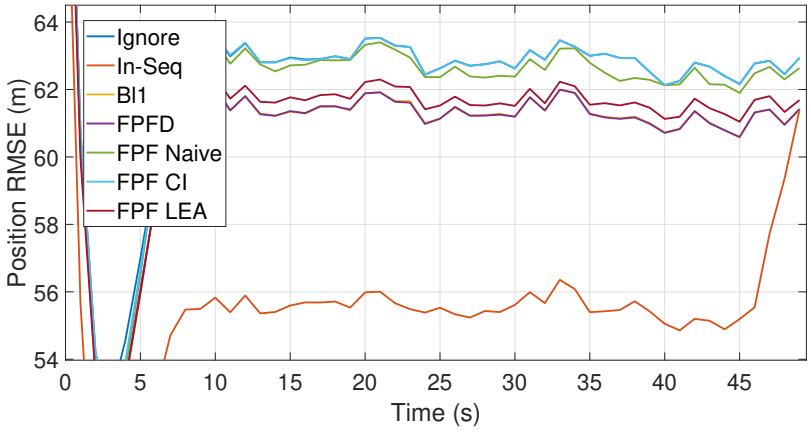
Table 4.1 shows the total CPU times of the algorithms Bl1, FPFD, FPF Naive, FPF CI, and FPF LEA for 1000 runs (processing 1000 OOSMs). Since CPU times are only an approximation of the computational cost of the algorithms,



(a) 1 lag OOSMs.

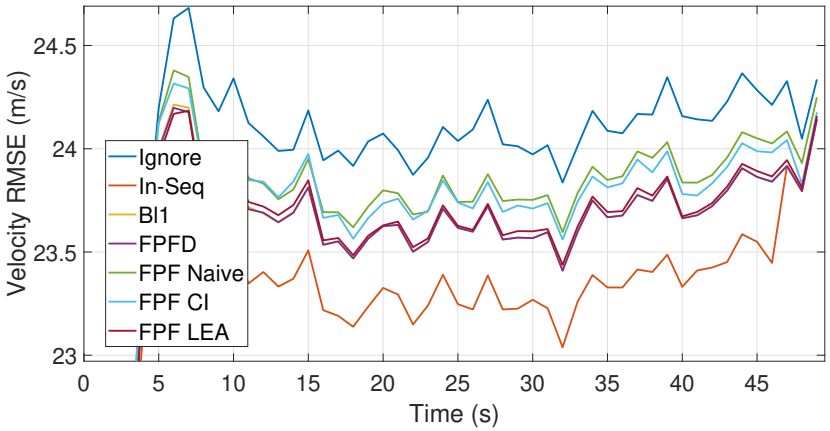


(b) 2 lag OOSMs.

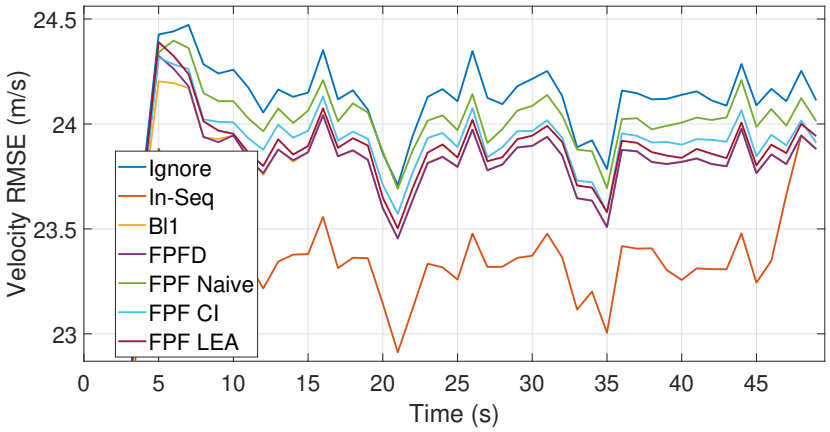


(c) 3 lag OOSMs.

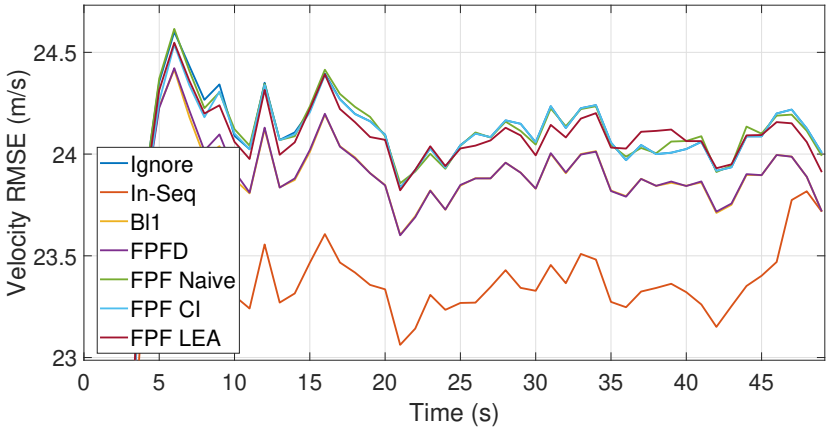
Figure 4.5: Position RMSE comparison of FPF methods and existing solutions for different OOSM lags.



(a) 1 lag OOSMs.

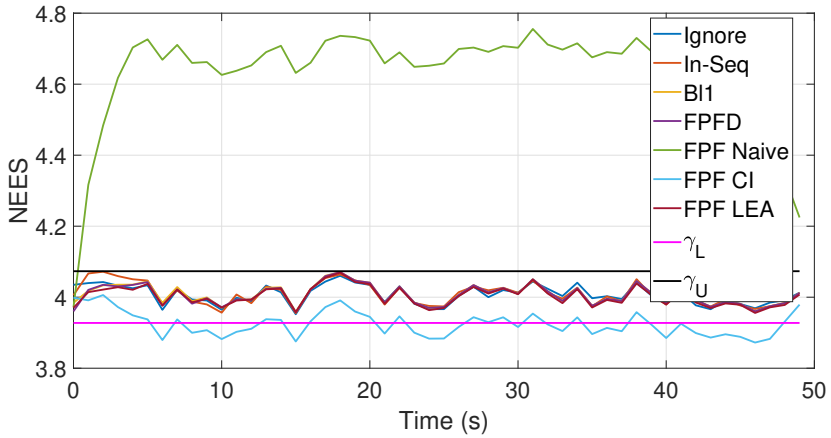


(b) 2 lag OOSMs.

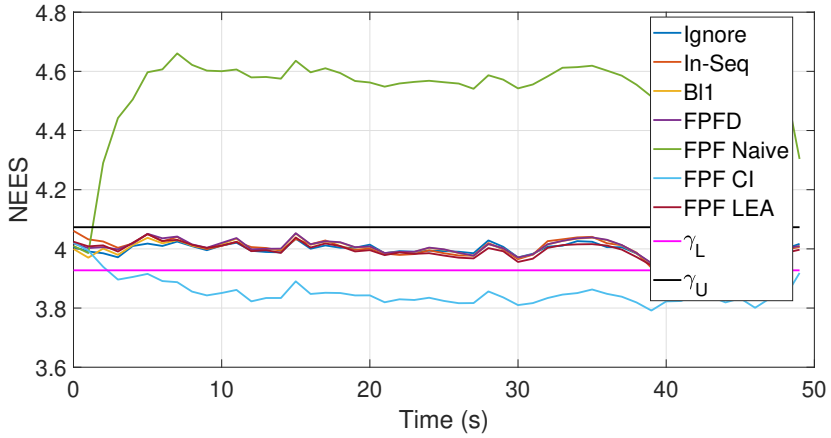


(c) 3 lag OOSMs.

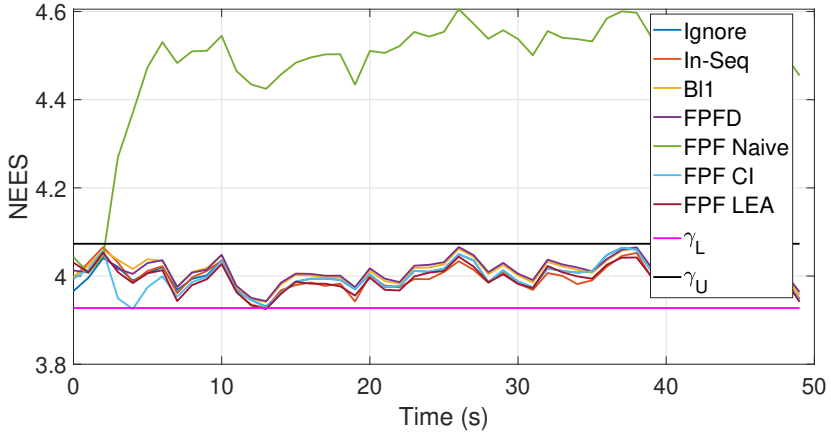
Figure 4.6: Velocity RMSE comparison of different FPF methods and existing solutions for different OOSM lags.



(a) 1 lag OOSMs.

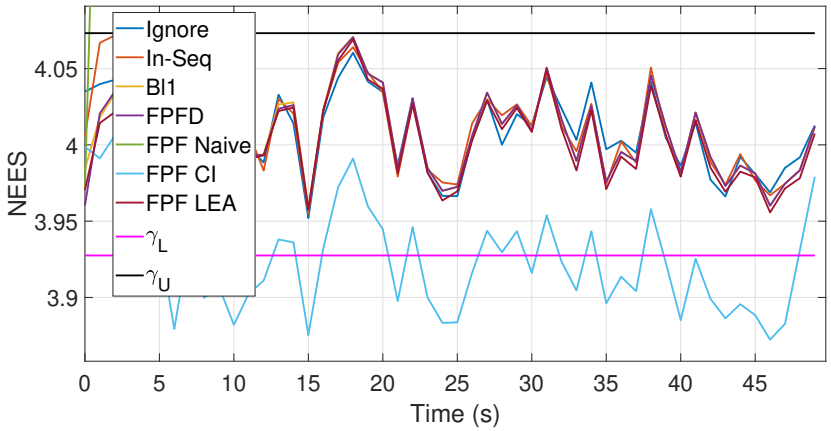


(b) 2 lag OOSMs.

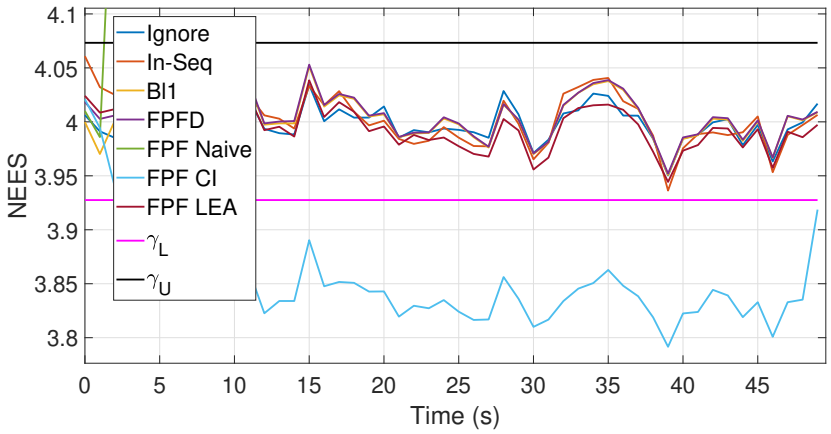


(c) 3 lag OOSMs.

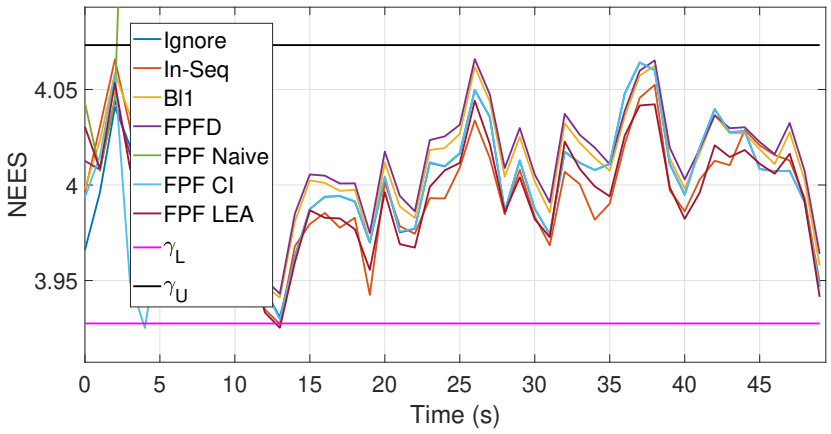
Figure 4.7: NEES comparison of different FPF methods and existing solutions for different OOSM lags.



(a) 1 lag OOSMs.



(b) 2 lag OOSMs.



(c) 3 lag OOSMs.

Figure 4.8: NEES comparison of different FPF methods and existing solutions for different OOSM lags (Zoomed version of Fig. 4.7).

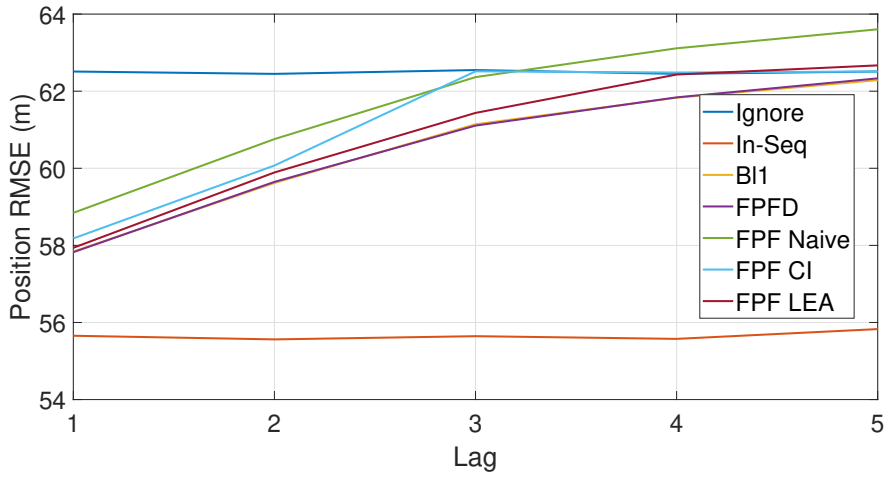


Figure 4.9: Time-averaged RMS position errors for different lags.

they are given here to support the results in Table 3.2. As can be seen from above, in accordance with Table 3.2, all FPF methods have a higher computational complexity than Bl1, and FPF naive has a lower computational load than FPFD. It is also shown that the computational cost of the FPF LEA is comparable to the FPFD as expected. Moreover, the FPF CI turns out to be practically feasible, since its cost is close to other methods for a small N_e, which in this case is 3. Finally, from the fact that the CPU times in the table do not differ much between lags, it can be concluded that the number of lags has little effect on computational complexity.

4.1.2 2-D Nonlinear Measurement Model Example

In this example, using the nonlinear measurement model instead of (2.2), FPF algorithms are tested with measurement covariances whose orientations change in state space. In this way, the performance of the fusion algorithms used in FPF is exploited to a larger extent. This example, therefore, differs from the previous one by considering a nonlinear measurement model used in practice.

The nonlinear counterpart of in-sequence (2.2) and out-of-sequence (2.5) measurement models are given as follows:

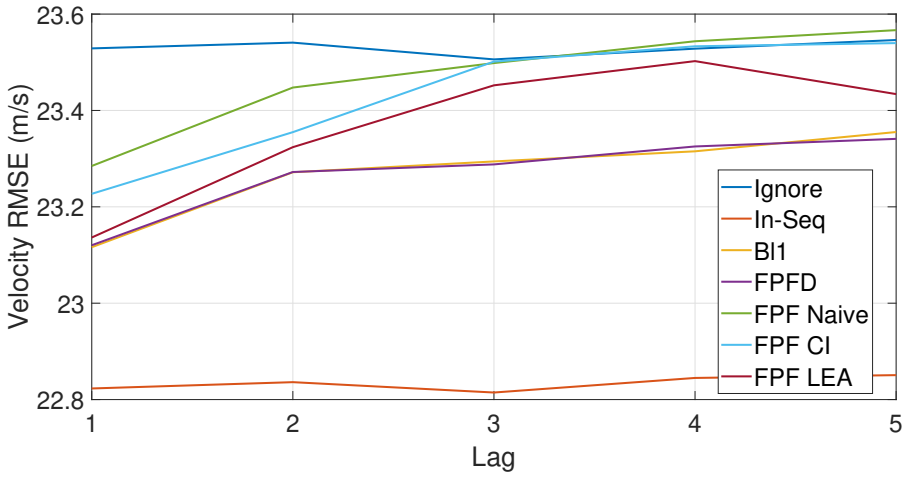


Figure 4.10: Time-averaged RMS velocity errors for different lags.

$$z_k = h(x_k) + v_k, \tag{4.8a}$$

$$z_\tau = h(x_\tau) + v_\tau, \tag{4.8b}$$

where the measurement function $h(.)$ is defined as below:

$$h(x_k) = \begin{bmatrix} f_R(x_k) \\ f_\theta(x_k) \end{bmatrix} = \begin{bmatrix} (p^x)^2 + (p^y)^2 \\ \tan^{-1}(p^y/p^x) \end{bmatrix}. \tag{4.9}$$

Here subscripts R and θ denote the range and azimuth (bearing) measurements, typical radar outputs for applications such as Ground Moving Target Indication (GMTI).

The covariances of measurement noises v_k and v_τ are given as follows:

$$R_k = \begin{bmatrix} 100^2\mathrm{m}^2 & 0 \\ 0 & 0.01^2\mathrm{rad}^2 \end{bmatrix}, \quad R_\tau = \begin{bmatrix} 1000^2\mathrm{m}^2 & 0 \\ 0 & 0.001^2\mathrm{rad}^2 \end{bmatrix}, \tag{4.10}$$

where σ_R, σ_θ are range and the bearing standard deviations for in sequence measurements and ρ_R, ρ_θ are the standard deviations for the OOSMs. The

considered scenario uses discretized continuous white noise acceleration model with the process noise intensity q=10 m^2/s^3 and the measurement models with $\sigma_R = 100$m, $\sigma_\theta = 0.01$rad and $\rho_R = 1000$m = $\rho_\theta = 0.001$rad.

The performance results of the algorithms are obtained through the simulations which take 50 time steps of 2.5 seconds. The Kalman filters of each method have identical parameters, which are matched to the true target parameters given above. The KFs are initialized with the state estimate and covariance given below.

$$\hat{x}_{0|0} = \begin{bmatrix} 50000\text{m} & 20\text{m/s} & 50000\text{m} & 20\text{m/s} \end{bmatrix}^T, \tag{4.11}$$

$$P_{0|0} = \text{diag}\begin{bmatrix} 100^2\text{m}^2 & 5^2\text{m}^2/\text{s}^2 & 100^2\text{m}^2 & 5^2\text{m}^2/\text{s}^2 \end{bmatrix}. \tag{4.12}$$

The true initial state of the target is realized from a Gaussian random variable distributed as $\mathcal{N}(x; \hat{x}_{0|0}, P_{0|0})$ at the beginning of each MC run. A total of 10000 MC runs are made by changing the target state trajectory and measurement noise realizations. Furthermore, in each run, 5 OOSMs with random lags between 1 and 5 are generated at random time instants to evaluate the overall performance of the algorithms, i.e., averaging the performance over lags from 1 to 5. The performance of the following approaches is obtained in the simulations: Ignoring OOSM, In-Seq, Bl1, FPFD, FPF Naive, FPF CI, FPF LEA. The RMS position errors, RMS velocity errors, and the mean NEES are shown in Fig. 4.11-4.14. Notice that Fig. 4.14 is a zoomed version of Fig 4.13.

As shown in the figures, the performance of the algorithms relative to each other is similar to the results in the previous example. The nonlinearity in the measurement model does not change the performance ranking of the algorithms. Among the FPF methods, the FPF LEA seems to be preferable in this case as well since it overall yields the smallest RMSEs and is the most consistent according to Fig 4.13.

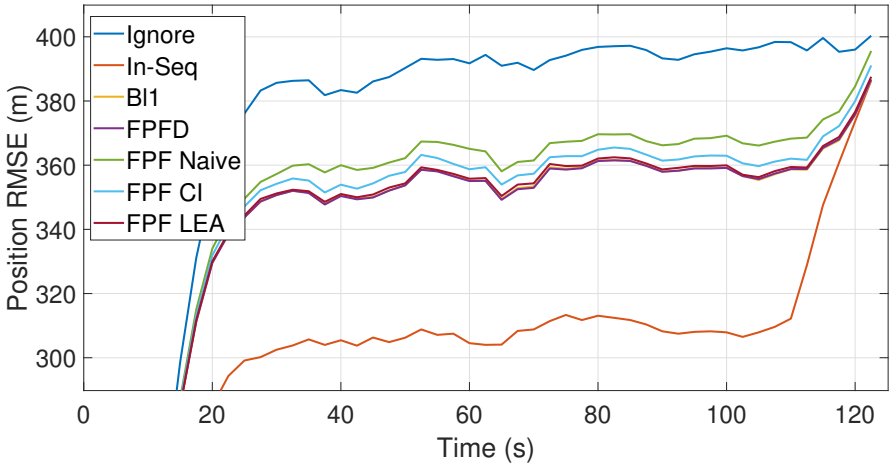


Figure 4.11: Position RMSE Values.

4.2 Performance Testing of FPFIMM Methods

The performance of the proposed FPFIMM methods will be tested for two 2-D target tracking scenarios in this section. In the first scenario, the target makes a maneuver in a way that is matched to our filter model i.e., IMM filter parameters are the true target parameters, while in a more practical second scenario, the target makes a maneuver and there is a mismatch between the true target motion and filter parameters.

4.2.1 Model Match Case

In this case, the true state data will be generated using a JMLS (2.44). The number of models used in JMLS is two, i.e., $r_k \in \{1, 2\}$, and the mode state x_k which consists of Cartesian x,y positions and velocities is defined as below.

$$x_k \triangleq \begin{bmatrix} p_k^x & v_k^x & p_k^y & v_k^y \end{bmatrix}^T. \tag{4.13}$$

Discretized continuous white noise acceleration models with different process noise covariance values given below are used in JMLS.

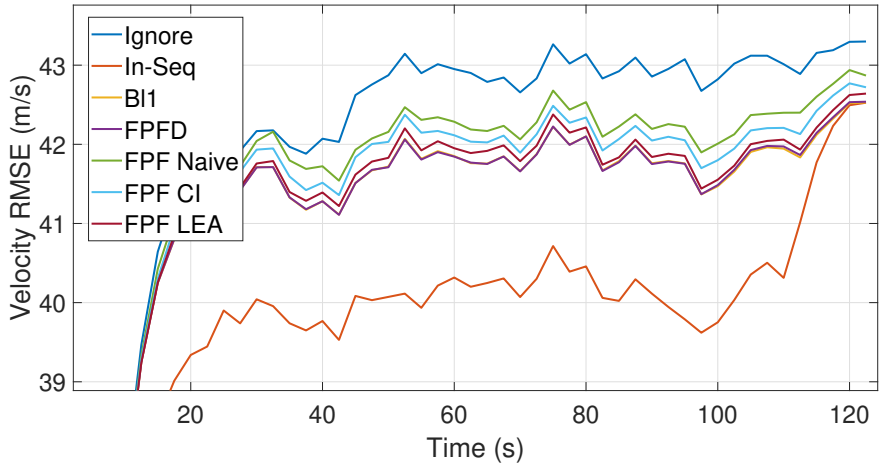


Figure 4.12: Velocity RMSE Values.

$$Q^i_{k,k-1} = q^i \begin{bmatrix} T^3/3 & T^2/2 & 0 & 0 \\ T^2/2 & T & 0 & 0 \\ 0 & 0 & T^3/3 & T^2/2 \\ 0 & 0 & T^2/2 & T \end{bmatrix}, \tag{4.14}$$

where q^i is the process noise intensity of the i'th mode. In our case, $q^1 = 1m^2/s^3$ and $q^2 = 100m^2/s^3$.

Mode-dependent state transition matrices $F^i_{k,k-1}$, are the same for each model and are given as follows.

$$F^1_{k,k-1} = F^2_{k,k-1} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.15}$$

The measurement equation for each model is nonlinear as in the above described 2-D nonlinear measurement model example and given as follows.

$$z^i_k = h^i(x_k) + v^i_k, \tag{4.16}$$

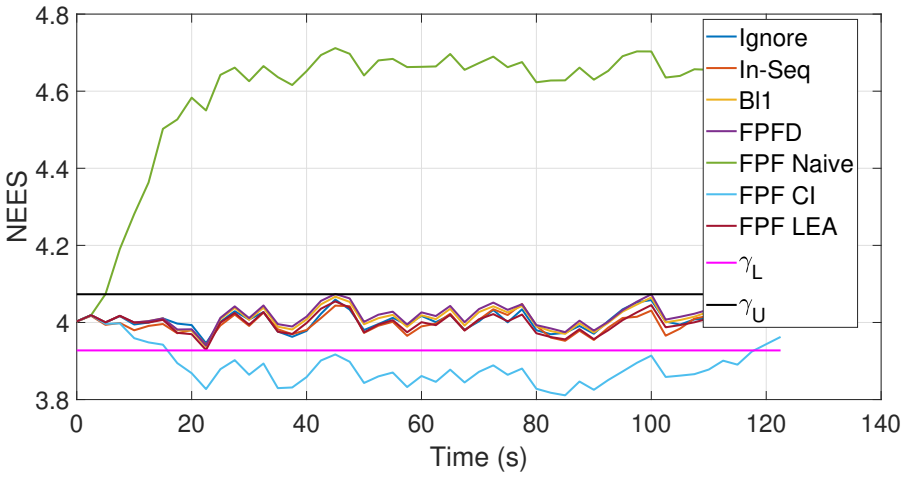


Figure 4.13: NEES Values.

where the measurement function $h^i(.)$ is the same as $h(.)$ defined in (4.8) for all i. In addition, the measurement noise covariances of each mode are also the same and are given as

$$E[v_k^i(v_k^i)^T] \triangleq R_k^i = \begin{bmatrix} (\sigma^R)^2 & 0 \\ 0 & (\sigma^\theta)^2 \end{bmatrix}, \tag{4.17}$$

where σ^R = 100m and σ^θ = 0.01rad are the range and bearing standard deviations for each model, respectively. The standard deviations of OOSMs, on the other hand, are ρ^R = 1000m, ρ^θ = 0.001rad.

The transition probability matrix $\Pi_{k,k-1}$ for the JMLS is given below

$$\Pi_{k,k-1} = \begin{bmatrix} 0.95 & 0.05 \\ 0.05 & 0.95 \end{bmatrix}^{\mathrm{T}}, \tag{4.18}$$

where T denotes the sampling interval.

The FPFIMM methods are compared with the In-seq, Ignore, and Bl1IMM. The performance results of the algorithms are obtained through the simulations which take 100 time steps of 1 second. The IMM filters of each method have

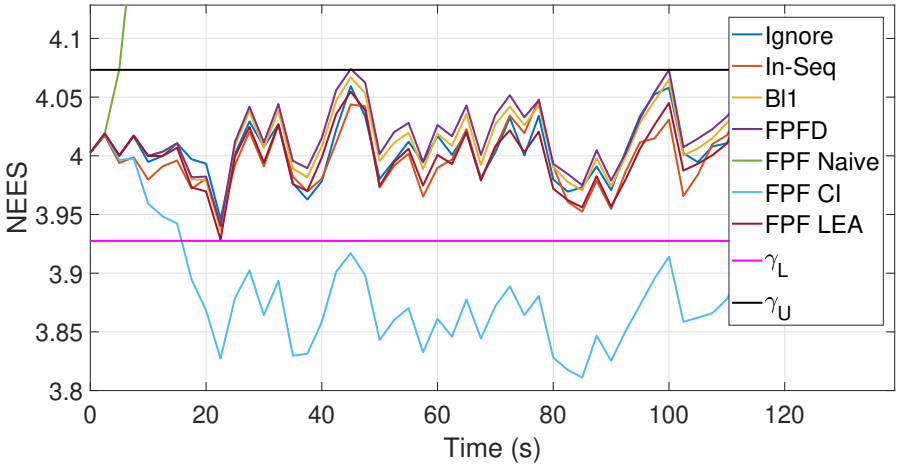


Figure 4.14: NEES Values (Zoomed version of Fig. 4.13).

identical parameters, which are matched to the true target parameters given above. The mode-conditioned estimates and covariances of the IMM filters are initialized as in (4.12) and (4.13) respectively. The initial mode probabilities are chosen to be equal in all IMM filters. The true initial base state of the target is generated as in FPF scenarios and the mode state is initiated at $r_0 = 1$.

A total of 1000 Monte Carlo runs are made by changing the target state trajectory, measurement noise realizations, and time instants of OOSM arrivals. Here the number of OOSMs generated within each run is 10. These OOSMs arrive at random time instants alongside sequential measurements collected at each sampling interval.

For this scenario, the performance of the following algorithms is obtained in the simulations: Ignore, In-Seq, Bl1IMM, IMMFPF Naive, IMMFPF CI, and IMMFPF LEA. For two different lags ($l = 1, 3$), the RMS position errors, RMS velocity errors, and the mean NEES are shown in Fig. 4.15-4.22. Notice that Fig. 4.19 and 4.20 are zoomed versions of Fig 4.21 and 4.22 respectively.

It can be seen from the RMSE plots that Bl1IMM yields the best RMS error performance. We also see that, unlike previous results, the FPFIMM LEA has an RMSE between FPF Naive and FPF CI methods, and among the FPF methods, the one that performs close to optimal is no longer LEA but CI. The reason for

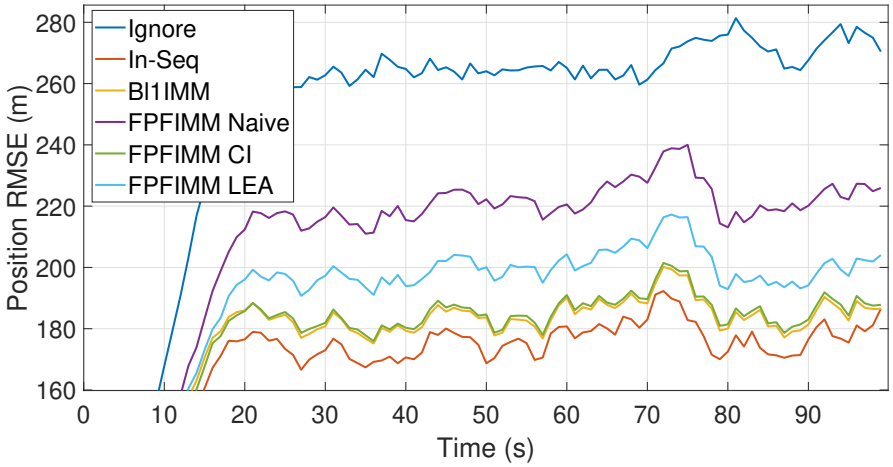


Figure 4.15: Position RMSE for 1 lag OOSMs.

this is FPFIMM LEA performs mode probability update with FPFIMM Naive's update function. This is why we see in the NEES results of FPFIMM LEA the properties of Naive fusion, not LEA. This inconsistency in FPFIMM LEA leads to poorer performance, making FPFIMM CI the best among FPFIMM methods in terms of RMS position and velocity errors. On the other hand, the FPFIMM CI shows similar characteristics to the FPF CI, by being underconfident. Due to this, it has a slightly worse performance than Bl1IMM which is neither overconfident nor underconfident according to Fig. 4.21-4.22.

4.2.2 Model-Mismatch Case

In this case, we consider the fixed true target trajectory shown in Fig. 4.23, where the target maneuvers in the middle of the scenario. The models and the parameters used in the model-match case are kept the same as in the model-match case. Thus, in this scenario, there is a mismatch between the true target dynamics and the filter parameters.

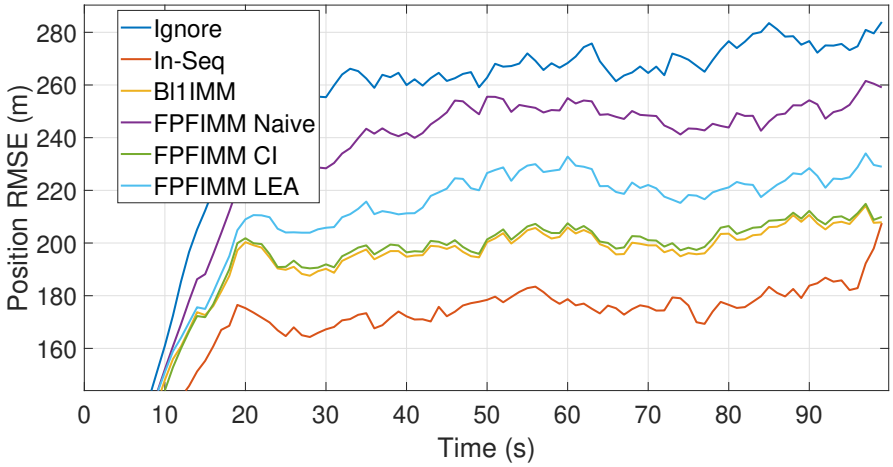


Figure 4.16: Position RMSE for 3 lag OOSMs.

4.2.2.1 Effect of the OOSM Lag

In this example, for performance evaluation, 1000 Monte Carlo runs are made by changing the measurement noise realizations and time instants of OOSM arrivals. Here the number of OOSMs generated within each run is 10. These OOSMs arrive at random time instants alongside sequential measurements collected at each sampling interval. For two different lags (l = 1, 3), the RMS position errors, RMS velocity errors, and the mean NEES are shown in Fig. 4.24-4.29.

In the non-maneuvering parts of the scenario, the best performance is achieved by FPFIMM LEA since it is overconfident in its estimate. This is in effect equivalent to the IMM filter using a lower process noise covariance which models the constant velocity motion of the target better. This overconfidence which yields larger NEESs can be seen in Fig. 4.28 and 4.29. Moreover, the performance of FPFIMM CI is slightly worse than that of the Bl1IMM as in the model match case.

In the maneuvering parts of the scenario, after In-Seq, the second best performance is achieved by Bl1IMM which is followed by FPFIMM CI. The FPFIMM LEA performs closer to the FPFIMM CI in this case than in the model match case because of the overconfidence issue discussed in the non-

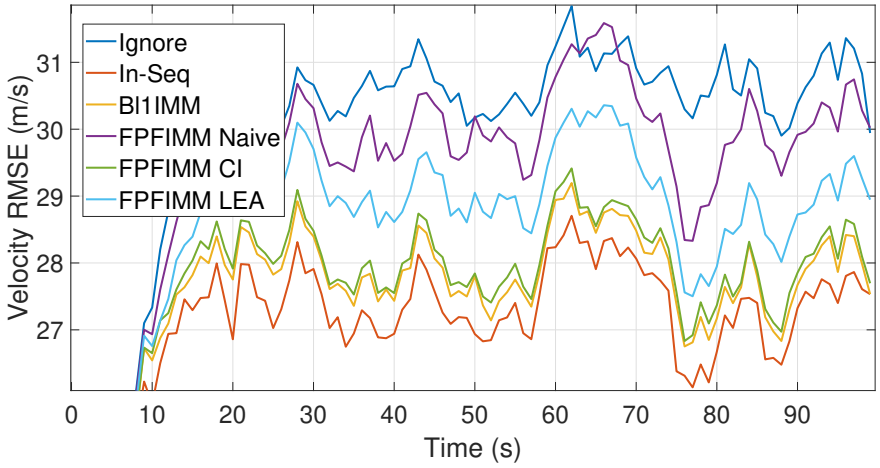


Figure 4.17: Velocity RMSE for 1 lag OOSMs.

maneuvering part. As can be seen in Figures 4.28 and 4.29, the NEESs of FPFIMM LEA are again between those of FPFIMM Naive and FPFIMM CI and closer to the value 4, which indicates consistency.

4.2.2.2 Effect of the OOSM Time

In this example, the arrival times of the OOSMs are changed from 40th second to 120th second. For each time instant, 2000 Monte Carlo runs are made by changing the measurement noise realizations and in each MC run, a single OOSM arrives at the relevant time. For this example, the performance of the following algorithms is obtained in the simulations: Ignore, In-Seq, Bl1IMM, IMMFPF Naive, IMMFPF CI, and IMMFPF LEA. For three different lags (l = 1, 2, 3), the RMS position errors of these methods are presented for three different lags (l = 1, 2, 3) in Fig. 4.30-4.31. Notice that Fig. 4.31 is a zoomed version of Fig 4.30.

Notice that, in the previous simulations, Position RMSEs are higher because there are some MC runs where no OOSM arrives at a particular time. On the other hand, in this example, the RMSEs are low as we consider the case where it is certain that in each MC run an OOSM arrives for a particular time.

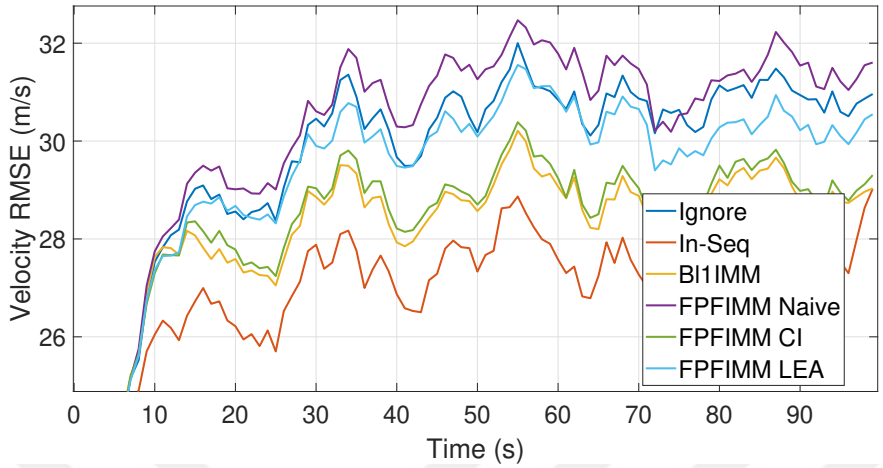


Figure 4.18: Velocity RMSE for 3 lag OOSMs.

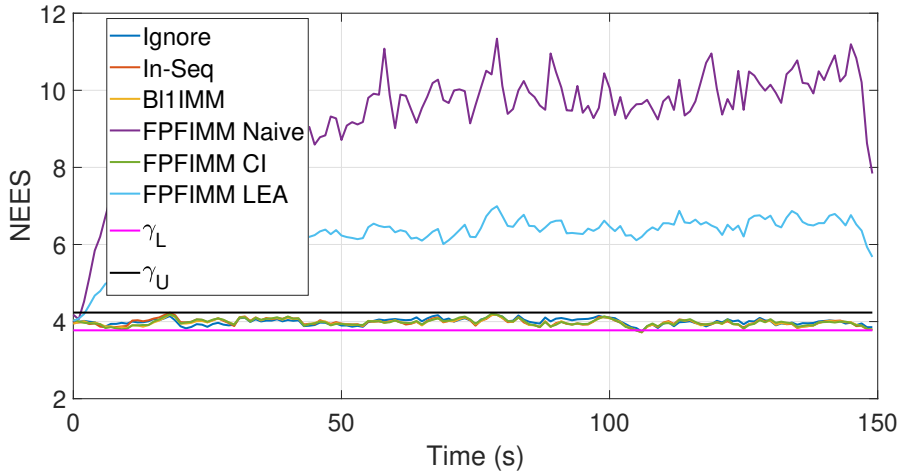


Figure 4.19: Mean NEES for 1 lag OOSMs.

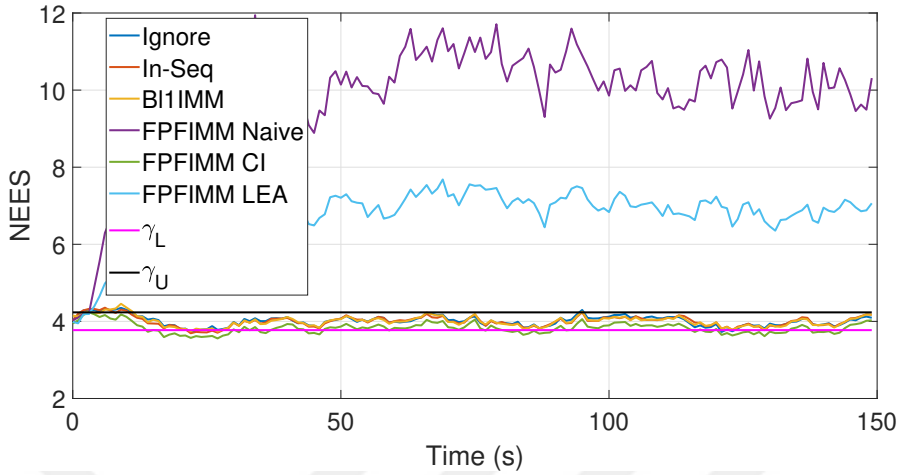


Figure 4.20: Mean NEES for 3 lag OOSMs.

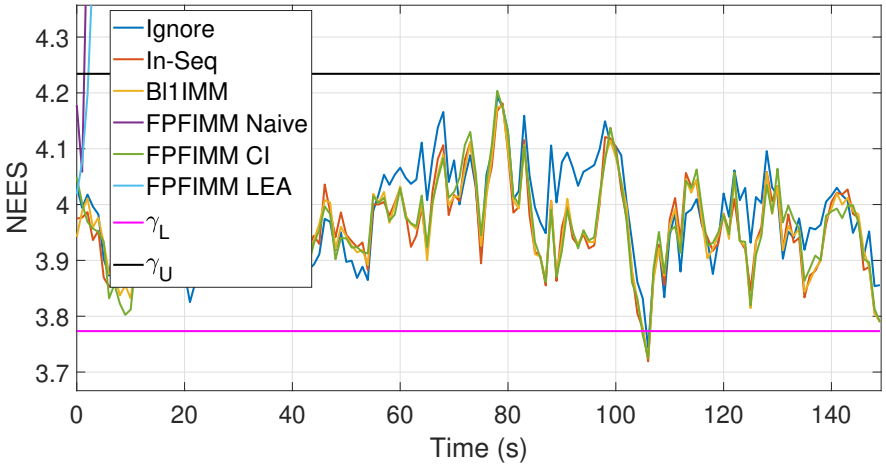


Figure 4.21: Mean NEES for 1 lag OOSMs (Zoomed version of Fig. 4.19).

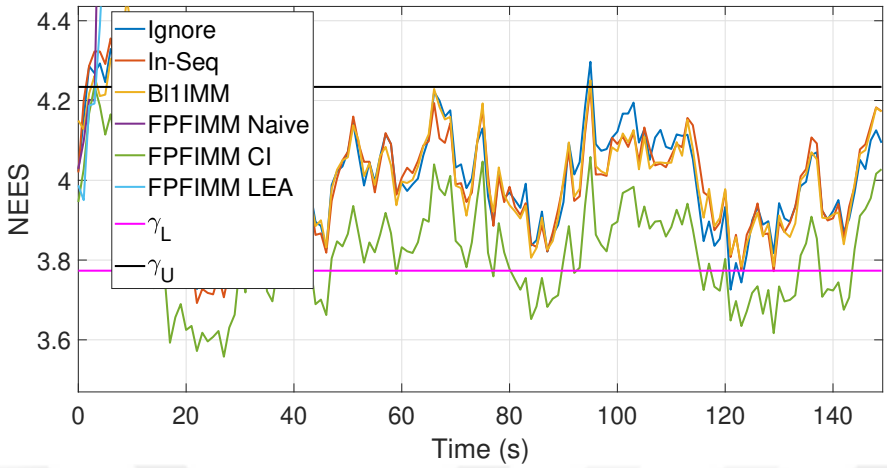


Figure 4.22: Mean NEES for 3 lag OOSMs (Zoomed version of Fig. 4.20).

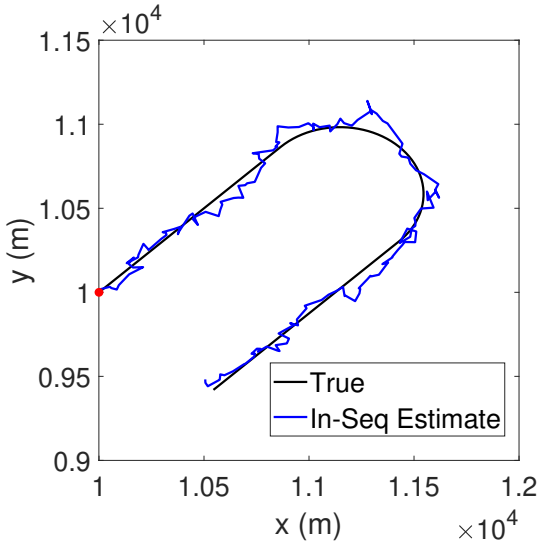


Figure 4.23: True and In-Seq estimate trajectories of the target in the model-mismatch scenario. The red dot denotes the starting point of the target trajectory.

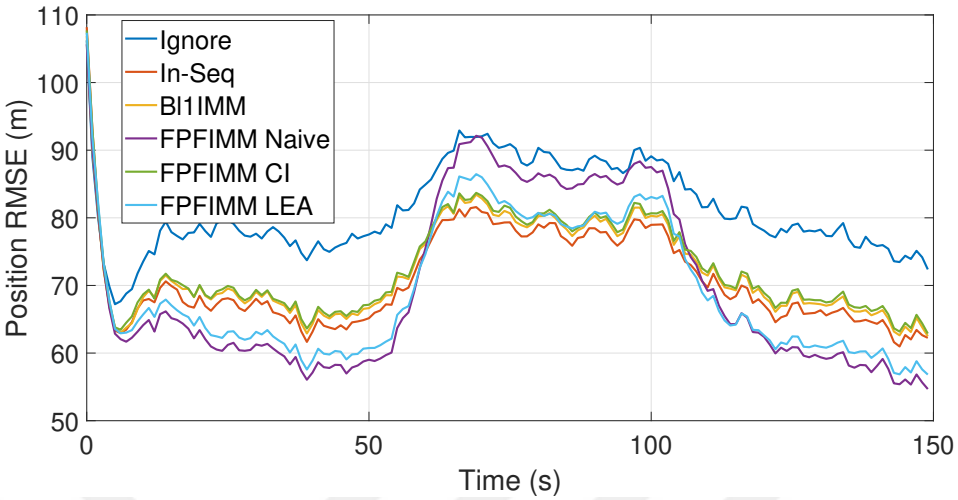


Figure 4.24: Position RMSE for 1 lag OOSMs.

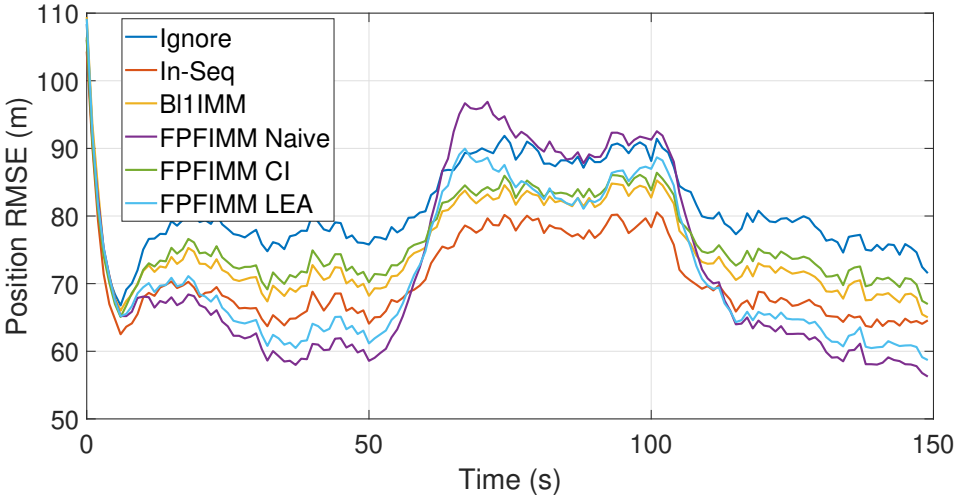


Figure 4.25: Position RMSE for 3 lag OOSMs.

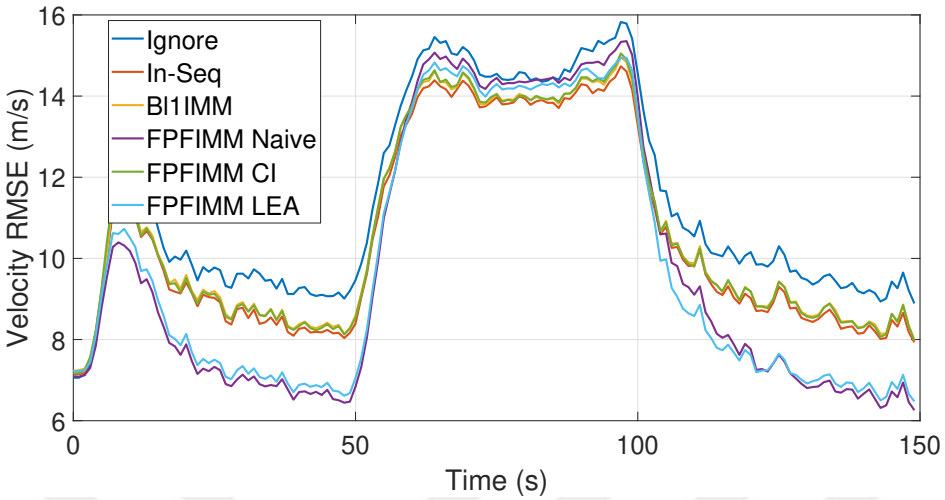


Figure 4.26: Velocity RMSE for 1 lag OOSMs.

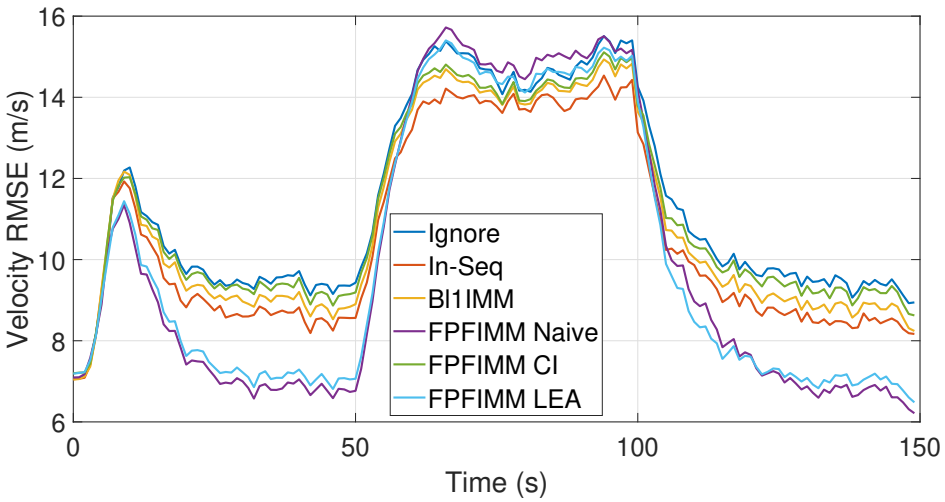


Figure 4.27: Velocity RMSE for 3 lag OOSMs.

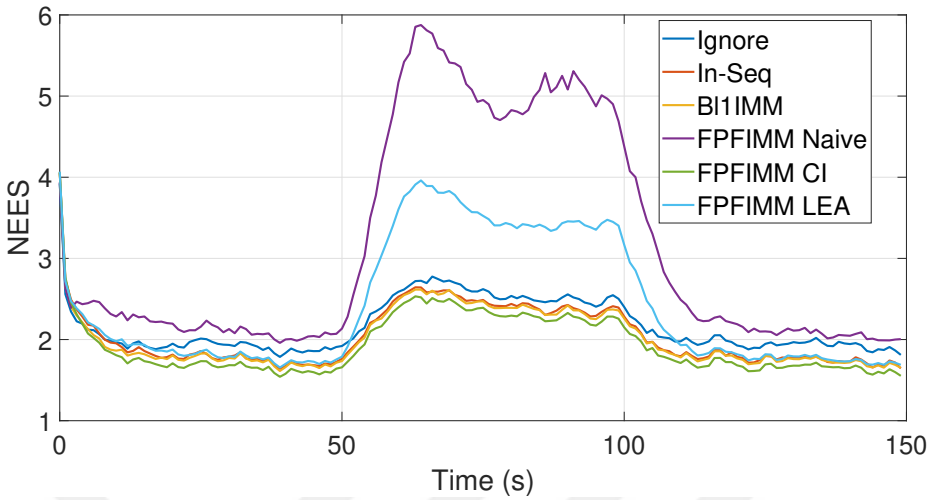


Figure 4.28: Mean NEES for 1 lag OOSMs.

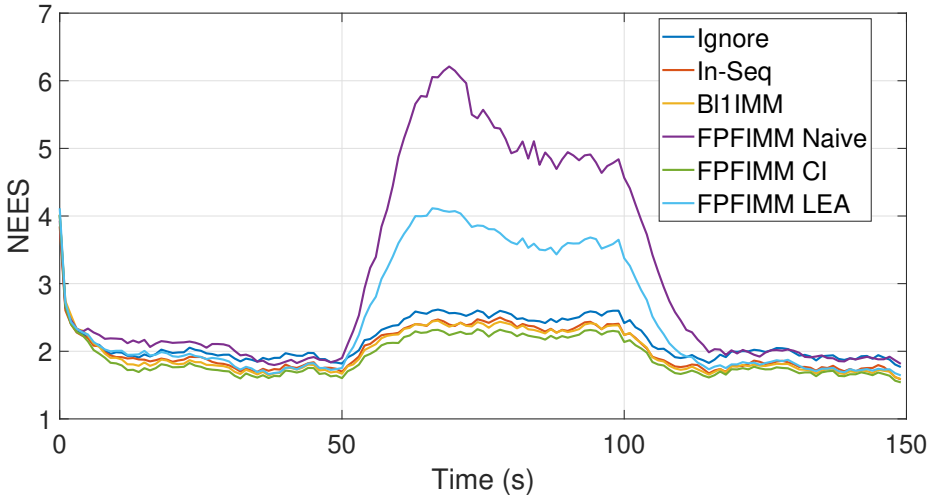
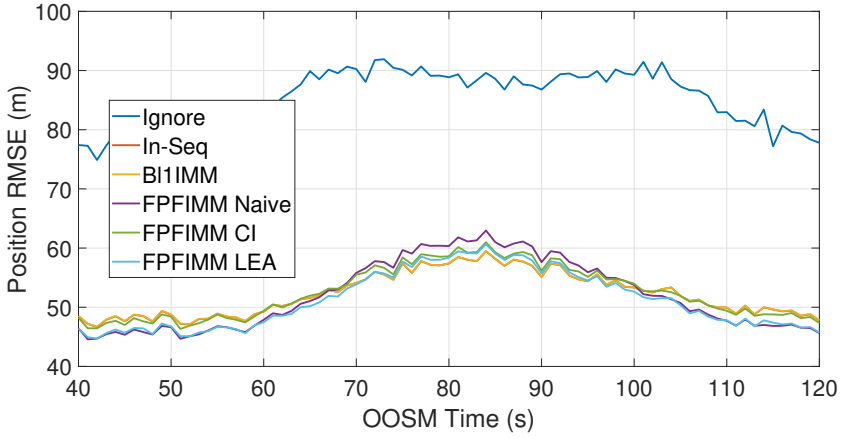
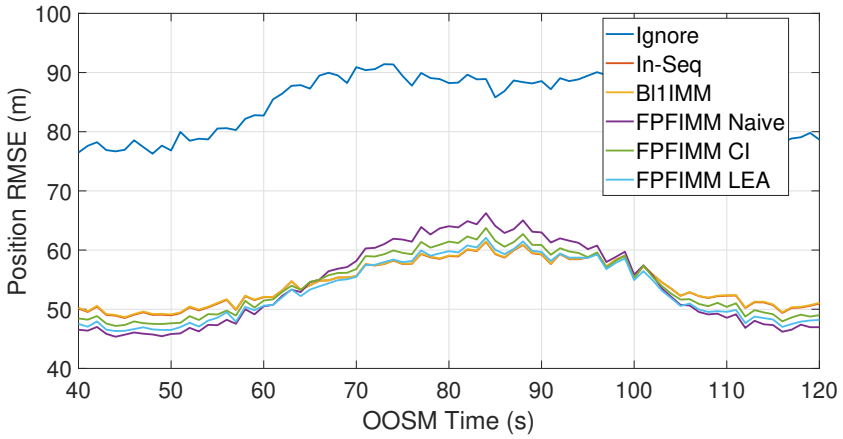


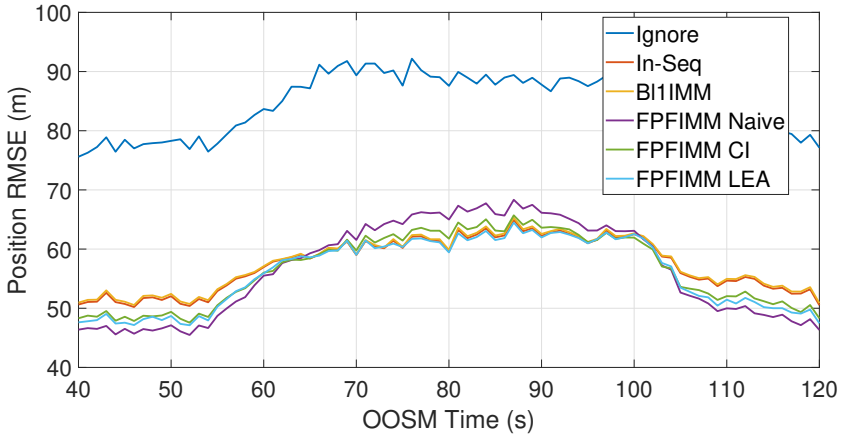
Figure 4.29: Mean NEES for 3 lag OOSMs.



(a) 1 lag OOSMs.

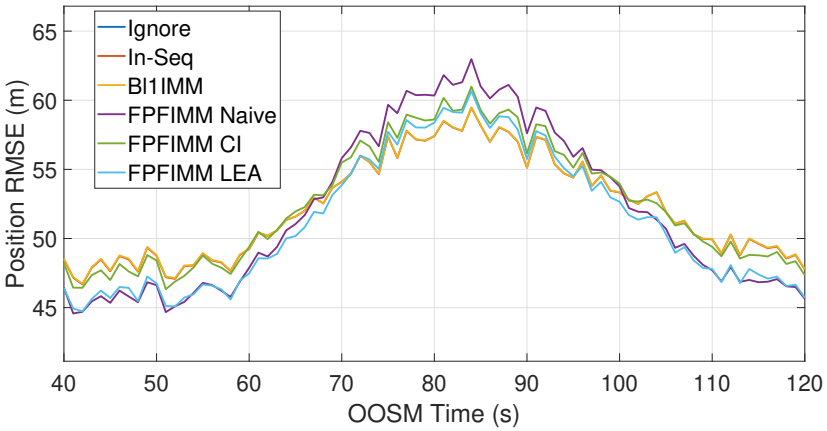


(b) 2 lag OOSMs.

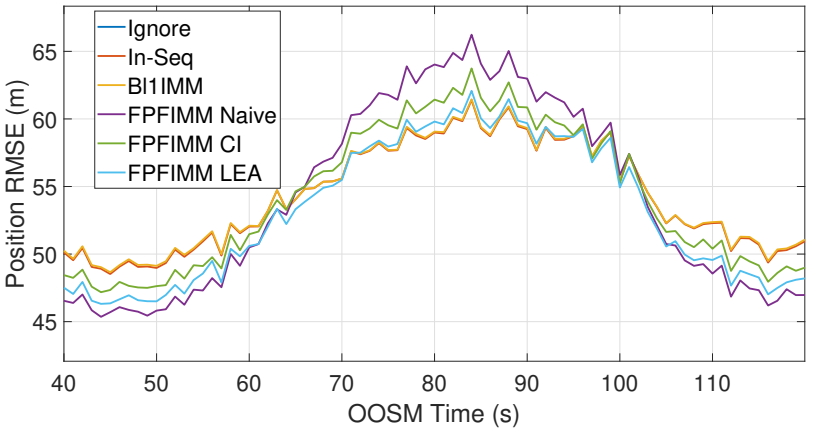


(c) 3 lag OOSMs.

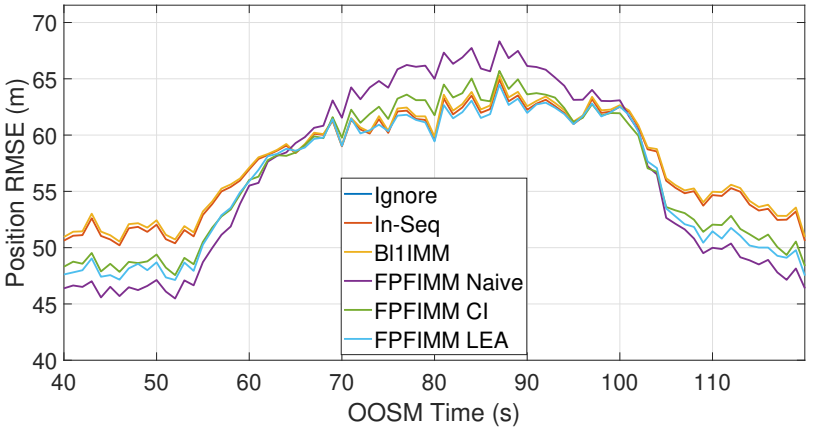
Figure 4.30: Position RMSEs with respect to OOSM times for different OOSM lags.



(a) 1 lag OOSMs.



(b) 2 lag OOSMs.



(c) 3 lag OOSMs.

Figure 4.31: Position RMSEs with respect to OOSM times for different OOSM lags (Zoomed version of Fig. 4.30).

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this work, by adopting the forward prediction and fusion ideas of FPFD which is a forward prediction-based OOSM processing method for linear Gaussian systems, we propose OOSM processing methods called Forward Prediction and Fusion (FPF) for linear Gaussian systems and its extension called FPFIMM for jump Markov linear systems. Due to the decorrelation challenge of Gaussian mixtures, which are the posterior distributions of JMLSs, direct adaptation of FPFD to these systems is not possible, and thus there was no forward prediction-based OOSM processing method applied to JMLSs in the literature. We believe that, with FPFIMM, which overcomes the decorrelation challenge encountered in FPFD by eliminating the separate decorrelation step and using a fusion algorithm where it is handled internally, a valuable contribution to the literature has been made. However, the proposed methods have higher RMS errors than the existing methods and cannot compensate for this performance degradation by either lower computation cost or storage requirement. Therefore, it can be argued that FPF and FPFIMM do not bring any improvements over the existing methods. The only advantage of the proposed methods can be considered as the fact that they are forward prediction based and therefore do not require the state transition matrix to be invertible, unlike the backward prediction-based Bl1 and Bl1IMM. The best performing FPF method, FPF LEA, shows an acceptable performance degradation compared to FPFD and Bl1 methods and has computational complexity and data storage requirements close to these methods, making it still a preferable OOSM processing alternative in practice. When it comes to the FPFIMM methods on the other hand, the best performing one turns out to be FPFIMM CI, rather

than FPFIMM LEA which is the extension of FPF LEA for IMM filter. This is due to the fact that the FPFIMM LEA method performs the mode probability updates in the same way as the overconfident FPFIMM Naive method and therefore yields poorer solutions in general. By being underconfident unlike other FPFIMM methods, FPFIMM CI shows an acceptable performance drop compared to Bl1IMM, one of the practically efficient algorithms in the literature. Moreover, it also has comparable computational complexity and data storage requirements to Bl1IMM, which makes it a preferable alternative. For all these reasons, FPFIMM CI, which performs the best among the forward prediction-based methods but is worse than Bl1 in general, may be applicable in practice as long as keeping the number of function evaluation points for the optimization step in the CI algorithm small does not lead to a significant decrease in performance. All in all, considering the inability of the proposed methods to outperform the existing ones in the literature, we can draw a conclusion that being conservative and sacrificing the independent information from the OOSM in order to avoid double counting due to common process noise leads to a performance loss rather than an improvement in general.

As mentioned already, the mode probability update in FPFIMM LEA is performed in a heuristic manner with the update function of FPFIMM Naive, yielding results worse than that of the Bl1IMM. In order to improve the performance of FPFIMMM LEA, our future plan is to work on developing a mode probability update function so that the method adopts the characteristics of one of the underconfident fusion algorithms, LEA or CI, instead of Naive as in the current version.

REFERENCES

[1] Y. Bar-Shalom and X.-R. Li, *Multitarget-multisensor tracking: principles and techniques*, vol. 19. YBs Storrs, CT, 1995.

[2] Y. Bar-Shalom, "Update with out-of-sequence measurements in tracking: Exact solution," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, pp. 769–778, 7 2002.

[3] Y. Bar-Shalom, H. Chen, and M. Mallick, "One-step solution for the multistep out-of-sequence-measurement problem in tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, pp. 27–37, 1 2004.

[4] R. D. Hilton, D. A. Martin, and W. D. Blair, "Tracking with time-delayed data in multisensor systems," tech. rep., Naval surface warfare center dahlgren div va, 1993.

[5] S. C. Thomopoulos and L. Zhang, "Decentralized filtering with random sampling and delay," *Information Sciences*, vol. 81, no. 1-2, pp. 117–131, 1994.

[6] Y. Bar-Shalom and G. Marcus, "Tracking with measurements of uncertain origin and random arrival times," *IEEE Transactions on Automatic Control*, vol. 25, no. 4, pp. 802–807, 1980.

[7] T. Larsen, N. Andersen, O. Ravn, and N. Poulsen, "Incorporation of time delayed measurements in a discrete-time Kalman filter," in *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)*, vol. 4, pp. 3972–3977 vol.4, 1998.

[8] M. Mallick, S. Coraluppi, and C. Carthel, "Advances in asynchronous and decentralized estimation," in *2001 IEEE Aerospace Conference Proceedings (Cat. No.01TH8542)*, vol. 4, pp. 4/1873–4/1888 vol.4, 2001.

[9] Y. Bar-Shalom, M. Mallick, H. Chen, and R. Washburn, "One-step solution for the general out-of-sequence-measurement problem in tracking," in *Proceedings, IEEE Aerospace Conference*, vol. 4, pp. 4–4, 2002.

[10] F. Rheaume and A. Benaskeur, "Out-of-sequence measurements filtering using forward prediction," Online 2007.

[11] S. Challa and J. Legg, "Track-to-track fusion of out-of-sequence tracks," in *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002. (IEEE Cat.No.02EX5997)*, vol. 2, pp. 919–926 vol.2, 2002.

[12] S. Challa, R. J. Evans, X. Wang, and J. Legg, "A fixed-lag smoothing solution to out-of-sequence information fusion problems *," 2002.

[13] K. Zhang, X. R. Li, and Y. Zhu, "Optimal update with out-of-sequence measurements for distributed filtering," *Proceedings of the 5th International Conference on Information Fusion, FUSION 2002*, vol. 2, pp. 1519–1526, 2002.

[14] K. Zhang, X. Li, and Y. Zhu, "Optimal update with out-of-sequence measurements," *IEEE Transactions on Signal Processing*, vol. 53, no. 6, pp. 1992–2004, 2005.

[15] H. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with markovian switching coefficients," *IEEE Transactions on Automatic Control*, vol. 33, no. 8, pp. 780–783, 1988.

[16] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan, "Interacting multiple model methods in target tracking: a survey," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 1, pp. 103–123, 1998.

[17] X. Wang and S. Challa, "Augmented state IMM-PDA for OOSM solution to maneuvering target tracking in clutter," in *2003 Proceedings of the International Conference on Radar (IEEE Cat. No.03EX695)*, pp. 479–485, 2003.

[18] Y. Bar-Shalom and H. Chen, "IMM estimator with out-of-sequence measurements," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, pp. 90–98, 1 2005.

[19] O. E. Drummond, "Track and tracklet fusion filtering," *Signal and Data Processing of Small Targets 2002*, vol. 4728, pp. 176–195, 8 2002.

[20] K. C. Chang, C. Y. Chong, and S. Mori, "Analytical and computational

evaluation of scalable distributed fusion algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, pp. 2022–2034, 10 2010.

[21] M. E. L. Ii, C.-Y. Chong, I. Kadar, M. G. Alford, V. Vannicola, and S. Thomopoulos, "Distributed fusion architectures and algorithms for target tracking," *Proceedings of the IEEE*, vol. 85, pp. 95–107, 1 1997.

[22] C. Y. Chong, S. Mori, W. H. Barker, and K. C. Chang, "Architectures and algorithms for track association and fusion," *IEEE Aerospace and Electronic Systems Magazine*, vol. 15, pp. 5–13, 1 2000.

[23] D. Acar and U. Orguner, "Decorrelation of previously communicated information for an interacting multiple model filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, pp. 404–422, 2 2021.

[24] J. Nygards, V. Deleskog, and G. Hendeby, "Safe fusion compared to established distributed fusion methods," *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, vol. 0, pp. 265–271, 7 2016.

[25] Y. Bar-Shalom, "On the track-to-track correlation problem," *IEEE Transactions on Automatic Control*, vol. 26, pp. 571–572, 1981.

[26] Y. Bar-Shalom and L. Campo, "The effect of the common process noise on the two-sensor fused-track covariance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-22, pp. 803–805, 1986.

[27] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," *Proceedings of the American Control Conference*, pp. 2369–2373, 6 1997.

[28] J. Sijs, M. Lazar, and P. P. Bosch, "State fusion with unknown correlation: Ellipsoidal intersection," *Proceedings of the 2010 American Control Conference, ACC 2010*, pp. 3992–3997, 2010.

[29] A. R. Benaskeur, "Consistent fusion of correlated data sources," *IECON Proceedings (Industrial Electronics Conference)*, vol. 4, pp. 2652–2656, 2002.

[30] M. Günay, U. Orguner, and M. Demirekler, "Chernoff fusion of Gaussian mixtures for distributed maneuvering target tracking," *18th International Conference on Information Fusion*, pp. 870–877, 7 2015.

[31] D. Acar and U. Orguner, “Information decorrelation for an interacting multiple model filter,” *2018 21st International Conference on Information Fusion, FUSION 2018*, pp. 1527–1534, 9 2018.

[32] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications To Tracking and Navigation*. 2001.