Sigma-Point Kalman Smoothing: Algorithms and Analysis with Applications to Indoor Tracking

Anindya Sankar Paul
M.S., Wright State University, USA, 2003
B.Eng., Sikkim Manipal University, India, 2001

Presented to the Department of Biomedical Engineering
and the Oregon Health & Science University
School of Medicine
in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy
in
Electrical Engineering

September 2010

The dissertation "Sigma-Point Kalman Smoothing: Algorithms and Analysis with Applications to Indoor Tracking" by Anindya Sankar Paul has been examined and approved by the following Examination Committee:

Dr. Eric A. Wan
Associate Professor
Dept. of Biomedical Engineering
Thesis Research Adviser

Dr. Misha Pavel
Professor
Dept. of Biomedical Engineering

Dr. Deniz Erdogmus
Assistant Professor
Dept. of Electrical and Computer Engineering
Northeastern University

Dr. James McNames
Associate Professor
Dept. of Electrical and Computer Engineering
Portland State University
External Committee member

Dedication

To my parents

without their support and encouragement I could not have started this journey.

and

To my wife Shreemoyee

without her this journey would not have fulfilled.

Acknowledgements

During my study at Oregon Health & Science University (OHSU) I received a lot of help from the faculty and colleagues. This dissertation would not be possible without their contributions. First and foremost I would like to express my sincere gratitude to my thesis advisor, Dr. Eric A. Wan, for his patience, guidance and continuous support over the duration of my stay in OHSU. He never failed to encourage me whenever I got stuck in my research and his invaluable suggestions were always helpful to solve difficult research problems. He helped to improve my communication skills which enabled me to effectively project my research work. I am extremely grateful for the knowledge he imparted to me during all these years which made me a better researcher. I am fortunate to have him as my mentor.

I would like to thank the members of my thesis committee, Dr. Misha Pavel, Dr. Deniz Erdogmus, and Dr. James McNames for taking the time to review this dissertation and helping me with their comments. I had the privilege of interacting with them in various projects. It was a good learning experience for me and I thoroughly enjoyed working with them. Special thanks also goes to John Hunt who helped me in understanding the hardware parts of the projects. It was an interesting experience to work with the function generators and digital oscilloscopes under his guidance. I am thankful to Dr. Tamara Hayes and her research assistants, Ann Tsay, Eric Earl, and Jon Yeargers for project support, use of laboratory facilities and data collection. I benefited immensely from the discussions related to my thesis with Dr. Rudolph van der Merwe, Houwu Bai and Dr. Zhengdong Lu. I want to thank them all for sharing their insights with me. I also thank Eric Morley for reviewing this thesis and providing me suggestions.

This work was supported in part by Intel as part of the Behavioral and Intervention Commons (BAIC).

I am indebted to my previous advisor, Dr. Arnab K. Shaw, for developing my interest on signal processing and emphasizing the importance of perseverance in research. Without his encouragement and support, I would not have the motivation to pursue a PhD degree.

Finally, I wish to thank my family for always believing in me and supporting me unconditionally. My parents made lots of sacrifices to help me with my education, for which I will always be grateful. I also would like to thank my wife Shreemoyee for proofreading this thesis, verifying the citations and references and most importantly giving me company on this incredible endeavor.

Contents

List of Tables

List of Figures

Abstract

Sigma-Point Kalman Smoothing: Algorithms and Analysis with Applications to Indoor Tracking

Anindya Sankar Paul

Supervising Professor: Dr. Eric A. Wan

The sigma-point Kalman filter (SPKF) is proved to be a more accurate alternative to the extended Kalman filter (EKF) in numerous nonlinear state estimation related applications. However, nonlinear smoothing algorithms based on the sigma-point Kalman filtering technique are not well established. We have derived new fixed-interval and fixed-lag smoothing algorithms using the sigma-point methodology and extended these nonlinear smoothers to a common family of algorithms, called *sigma-point Kalman smoothers* (SPKS). While the fixed-interval SPKS (FI-SPKS) operates on a fixed set of observations, the fixed-lag SPKS (FL-SPKS) sequentially operates on the buffered blocks of measurements as they become available. Both the FI-SPKS and FL-SPKS make use of the forward-backward (FB) and Rauch-Tung-Striebel (RTS) approaches to perform smoothing. In the FB method, a standard SPKF is used as a forward filter. The backward filter requires the use of the inverse dynamics of the forward filter. While smoothers based on the EKF simply invert the linearized dynamics, with the SPKF the forward nonlinear dynamics are never analytically linearized. Thus the backward nonlinear dynamics are not well defined. In this work, we make use of the relationship between the SPKF and weighted statistical linear regression (WSLR) to pseudo-linearize the nonlinear dynamics. The independent forward and backward estimates are then statistically combined to generate the smoothed results. The WSLR linearized dynamics are also incorporated in the RTS method to derive the backward smoothing gain which operates on the forward SPKF estimates to produce the smoothed states. We have applied the proposed SPKS to the challenging areas of probabilistic inference, such as indoor localization and multiharmonic frequency tracking, and evaluated the performance by comparing to the state-of-the-art tracking engines. Furthermore, we have successfully extended the theoretical understanding of the SPKF by analyzing its estimation-error bounds for the discrete-time nonlinear dynamical system. We have derived the mean-square error lower bound using the well-known Cramér-Rao theory. The upper error bound, which is also termed as the "stability bound", exponentially converges to the steady state if certain conditions on the state dynamics and system noises are satisfied. The theoretical derivations are experimentally verified using practical examples.

Chapter 1

Introduction

1.1 Overview

This chapter summarizes our work on the problem of probabilistic Bayesian inference that deals with estimating a set of hidden variables (referred as states) in an optimal and consistent fashion using incoming noisy measurements. We start with a brief description of the nonlinear state space model and later present the standard state estimation algorithms for the nonlinear system. In this context, we describe the sigma-point Kalman filter (SPKF) based estimator which is the core of the algorithmic development presented in this thesis. Finally, a summary of the objective and contributions of our work are provided.

1.2 Probabilistic Inference and Dynamic State Space Model

The problem of estimating the hidden states from a set of noisy sensor observations is a widely used method in airborne navigation [1, 2], robot localization and tracking [3], simultaneous localization and mapping [4], driving an unmanned aerial vehicle (UAV) [5], aerospace engineering [6] and many other similar areas. All the above applications concentrate on how a mobile robot/vehicle determines its own pose (position, velocity and orientation) relative to the environment by observing certain characteristics of its surroundings. Recently, the use of state estimation extends to many other engineering fields such as adaptive signal de-noising [7–9], robust and H_∞ control [10, 11], system identification and learning [12, 13] and modeling/parameter estimation for biomedical signals [14–16]. The general state estimation problem can often be cast in terms of estimating the state of a discrete-time dynamic state space system (DSSM),

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) \tag{1.1}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{n}_k\right), \tag{1.2}$$

where the random variable (RV) $\boldsymbol{x}_k$ represents the unobserved state of the system and $\boldsymbol{z}_k$ is the sensor observations. Both the process noise $\boldsymbol{v}_k$ and observation noise $\boldsymbol{n}_k$ are assumed Gaussian with zero mean and covariances $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ respectively. Note that we are not assuming additivity of the noise sources. The nonlinear state transition function $f_k(.)$ and observation function $h_k(.)$ are assumed to be known. The state transition/process model f_k, along with the prior distribution of the system state and the statistics of the process noise, defines how the dynamic system evolves over time. In contrast, the observation model h_k describes how the evolved state, together with the observation noise, relates to the sensor observations.

We use boldface notations to denote vectors and matrices, normal face are for scalers. Matrices are labeled in upper cases, whereas lower cases are reserved for vectors and scalers. Unless otherwise stated, these notational conventions are followed throughout the dissertation.

1.3 Recursive Bayesian Estimation and Kalman Filtering

The optimal estimate of the state $\boldsymbol{x}_k$ in the minimum-mean-square-error (MMSE) sense is given by the *conditional mean*:

$$\begin{aligned}\hat{\boldsymbol{x}}_k &= \mathrm{E}\left[\boldsymbol{x}_k | \boldsymbol{z}_{1:k}\right] \\ &= \int \boldsymbol{x}_k p\left(\boldsymbol{x}_k | \boldsymbol{z}_{1:k}\right) d\boldsymbol{x}_k,\end{aligned} \tag{1.3}$$

where the vector $\boldsymbol{z}_{1:k}$ denotes all the observations until time k.

$$\boldsymbol{z}_{1:k} = \begin{bmatrix} \boldsymbol{z}_1 & \boldsymbol{z}_2 & \ldots & \boldsymbol{z}_k \end{bmatrix}. \tag{1.4}$$

From Equation (1.3), the posterior probability $p\left(\boldsymbol{x}_k | \boldsymbol{z}_{1:k}\right)$ of the state $\boldsymbol{x}_k$ given observations $\boldsymbol{z}_{1:k}$ provides the complete solution of the state estimation problem at time k. By applying Bayes rule and making use of the conditional independence of observations, we can recursively update the state posterior density as new observations arrive [17]:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = Cp(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}), \tag{1.5}$$

where C is the normalizing constant. In order to gain insight of the above Equation, let's explain each term on the R.H.S. of Equation (1.5). The prior distribution $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})$ at time k is obtained by propagating the state posterior at time $k-1$, $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$, forward in time using the probabilistic state transition model $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$,

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1}. \tag{1.6}$$

The transition density $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ is given by the state transition model f_k and the process noise distribution $p(\boldsymbol{v}_k)$,

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \int \delta\left(\boldsymbol{x}_k - f_k\left(\boldsymbol{x}_{k-1}, \boldsymbol{v}_k\right)\right) p(\boldsymbol{v}_k)d\boldsymbol{v}_k, \tag{1.7}$$

where δ is the Dirac delta function. The observation likelihood $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$ is specified using the observation function h_k and the observation noise density $p(\boldsymbol{n}_k)$,

$$p(\boldsymbol{z}_k|\boldsymbol{x}_k) = \int \delta\left(\boldsymbol{z}_k - h_k\left(\boldsymbol{z}_k, \boldsymbol{n}_k\right)\right) p(\boldsymbol{n}_k)d\boldsymbol{n}_k. \tag{1.8}$$

The normalizing factor C is given by

$$C = \left(\int p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_k\right)^{-1}. \tag{1.9}$$

One has to solve all these multidimensional integrations shown in (1.6)-(1.9) analytically in order to compute the state posterior density at time k. Unfortunately the integrals are intractable except when the state space is linear and all the above distributions are Gaussian. For the linear-Gaussian systems, the famous Kalman filter [18] provides the closed-form MMSE estimate of the state $\boldsymbol{x}_k$.

1.4 Gaussian Approximate Filters For State Estimation

As the multidimensional integrals shown in Equations (1.6)-(1.9) are intractable for general nonlinear systems, statistical approximations must be used in order to solve them.

Although there are a number of categories of statistical approximation solutions available in the literature, among those Gaussian approximate solutions and sequential Monte-Carlo (SMC) implementations are the most prevalent. SMC based techniques, or *particle filters*, model the state distribution using a set of discrete points and can provide arbitrary accuracy to the solution with a sufficient number of particles. Still, the general requirement of large number of sample particles in order to converge to the true density can make this approach computationally prohibitive. In this dissertation, we primarily concentrate on the variants of Gaussian approximate solutions.

Gaussian approximate methods assume that all the underlying densities encountered in Equations (1.6)-(1.9) are Gaussian and hence use the first (mean) and second (covariance) order moments of those densities to perform the recursive state prediction and update. Of all the Gaussian approximate methods, the extended Kalman filter (EKF) has the most widespread use for state estimation problems [8, 11, 19–21]. The popularity of the EKF based estimator is contributed due to its ease of implementation and cheap computational requirement. In case of a nonlinear state space model, the EKF approximates the true expectations shown in (1.6)-(1.9) by using the first-order truncated Taylor series expansion around the current state estimate. The major inaccuracy of the EKF based approach stems from the nature of the first-order Taylor series linearization, which approximates the nonlinear dynamic model only up to the first order term and thereby neglects all the higher order terms. Furthermore, the EKF does not take into account the *uncertainty* of the prior state RV when it linearizes the process and observation models using the Taylor series expansion. Failure to consider the probabilistic spread of the RV during the linearization process often introduces large errors in the EKF calculated posterior statistics, which sometimes may lead to suboptimal performance and filter divergence. More detailed discussion on the EKF and its flaws can be found in Van der Merwe's dissertation [17].

The inaccuracies of the EKF gave birth to a new group of state estimation algorithms collectively called the *sigma-point Kalman filters (SPKF)*, which uses a *deterministic sampling* approach in order to propagate the state distribution over nonlinear systems. The first idea of *sigma-point transform*, which describes an efficient method of sampling and propagating a RV over nonlinear dynamics, was proposed by Julier and Uhlmann for state estimation in automatic control [22, 23]. They came up with a new derivativeless Kalman filter called the *Unscented Kalman Filter* (UKF), which applies the deterministic sigma-point transform method at each filter recursion. Later, Ito and Norgaard *et al.* proposed a new variant of the filter known as the *Central Difference Kalman Filter* (CDKF), which is based on the *Stirling's Interpolation Formula* and also makes use of the sigma-point approach to propagate first and second order statistics of a RV over nonlinear systems [24,25]. Wan and Van der Merwe brought all these different deterministic sampling approaches under a common umbrella of Gaussian approximate filters called the SPKF [26, 27]. Van der Merwe and Wan also derived numerically efficient and stable square-root versions of all the SPKF variants [27]. For algorithmic descriptions and detailed proofs of all these sigma-point methodologies, the reader can refer to Van der Merwe's dissertation [17]. The major advantage of the SPKF stems from the fact that it consistently outperforms the EKF over a wide range of state estimation problems at the same order of computational complexity [17, 26]. Like the EKF, the SPKF estimates only the first and second-order moments of the true distribution. The probability distribution is represented by a set of carefully chosen deterministic sample points (known as sigma points), which capture the mean and covariance of the RV. These sigma points are then propagated through the true nonlinear system, with the posterior mean and covariance calculated using simple weighted averaging (the value of weights depend on the choice of SPKF parameters, details can be obtained from chapter 3 in [17]). The SPKF captures the posterior mean and covariance accurately to the 2^{nd} order (3^{rd} order is achieved for symmetric distribution) compared to the EKF which linearizes the nonlinear systems and only achieves 1^{st} order accuracy. Furthermore, the SPKF no longer needs to calculate the Jacobians, which are sometimes hard to obtain analytically. We will now present the pseudo-codes of the EKF and SPKF in a concise manner. The reader can refer to [11, 12, 17, 21] for more elaborate discussion on these topics.

Below we demonstrate the EKF time-update and measurement-update recursions that produce the filtered estimates $\hat{\boldsymbol{x}}_k$ and the associated estimation error covariance $\boldsymbol{P}_{\boldsymbol{x}_k}$ at each time k. Note, both the time-update and measurement-update steps compute the Jacobians that are incorporated to predict and update the estimation error covariance.

Extended Kalman filter (EKF)

- *Initialization*:

$$\hat{\boldsymbol{x}}_0 = \mathrm{E}\left[\boldsymbol{x}_0\right] \tag{1.10}$$

$$\boldsymbol{P}_{\boldsymbol{x}_0} = \mathrm{E}\left[\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)^T\right] \tag{1.11}$$

$$\bar{\boldsymbol{v}}_0 = \mathrm{E}\left[\boldsymbol{v}_0\right] \tag{1.12}$$

$$\bar{\boldsymbol{n}}_0 = \mathrm{E}\left[\boldsymbol{n}_0\right] \tag{1.13}$$

$$\boldsymbol{Q}_0 = \mathrm{E}\left[\left(\boldsymbol{v}_0 - \bar{\boldsymbol{v}}_0\right)\left(\boldsymbol{v}_0 - \bar{\boldsymbol{v}}_0\right)^T\right] \tag{1.14}$$

$$\boldsymbol{R}_0 = \mathrm{E}\left[\left(\boldsymbol{n}_0 - \bar{\boldsymbol{n}}_0\right)\left(\boldsymbol{n}_0 - \bar{\boldsymbol{n}}_0\right)^T\right] \tag{1.15}$$

- *For* $k = 0, 1, 2, \ldots, N$

  1. *Time-update equations*:

     – Compute the process model Jacobians:

$$\boldsymbol{F}_{\boldsymbol{x}_k} = \nabla_{\boldsymbol{x}} f_k\left(\boldsymbol{x}, \bar{\boldsymbol{v}}_k\right)\big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_k} \tag{1.16}$$

$$\boldsymbol{G}_{\boldsymbol{v}_k} = \nabla_{\boldsymbol{v}} f_k\left(\hat{\boldsymbol{x}}_k, \boldsymbol{v}\right)\big|_{\boldsymbol{v}=\bar{\boldsymbol{v}}_k} \tag{1.17}$$

     – Compute the predicted state mean and covariance:

$$\hat{\boldsymbol{x}}_{k+1}^- = f_k\left(\hat{\boldsymbol{x}}_k, \bar{\boldsymbol{v}}_k\right) \tag{1.18}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}}^- = \boldsymbol{F}_{\boldsymbol{x}_k}\boldsymbol{P}_{\boldsymbol{x}_k}\boldsymbol{F}_{\boldsymbol{x}_k}^T + \boldsymbol{G}_{\boldsymbol{v}_k}\boldsymbol{Q}_k\boldsymbol{G}_{\boldsymbol{v}_k}^T \tag{1.19}$$

  2. *Measurement-update equations*:

– Compute the observation model Jacobians:

$$\boldsymbol{H}_{\boldsymbol{x}_k} = \nabla_{\boldsymbol{x}} h_k \left(\boldsymbol{x}, \bar{\boldsymbol{n}}_k\right)\big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k+1}^-} \tag{1.20}$$

$$\boldsymbol{G}_{\boldsymbol{n}_k} = \nabla_{\boldsymbol{n}} h_k \left(\hat{\boldsymbol{x}}_{k+1}^-, \boldsymbol{n}\right)\Big|_{\boldsymbol{n}=\bar{\boldsymbol{n}}_k} \tag{1.21}$$

– Update estimates incorporating the latest observation:

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}_{\boldsymbol{x}_{k+1}}^- \boldsymbol{H}_{\boldsymbol{x}_k}^T \left(\boldsymbol{H}_{\boldsymbol{x}_k} \boldsymbol{P}_{\boldsymbol{x}_{k+1}}^- \boldsymbol{H}_{\boldsymbol{x}_k}^T + \boldsymbol{G}_{\boldsymbol{n}_k} \boldsymbol{R}_k \boldsymbol{G}_{\boldsymbol{n}_k}^T\right)^{-1} \tag{1.22}$$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1}^- + \boldsymbol{K}_{k+1} \left[\boldsymbol{z}_{k+1} - h_k \left(\hat{\boldsymbol{x}}_{k+1}^-, \bar{\boldsymbol{n}}_k\right)\right] \tag{1.23}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}} = \left(\boldsymbol{I} - \boldsymbol{K}_{k+1} \boldsymbol{H}_{\boldsymbol{x}_k}\right) \boldsymbol{P}_{\boldsymbol{x}_{k+1}}^- . \tag{1.24}$$

The complete SPKF algorithm that updates the mean $\hat{\boldsymbol{x}}_k$ and the covariance $\boldsymbol{P}_{\boldsymbol{x}_k}$ of the state distribution using the observation $\boldsymbol{z}_k$ is presented below:

Sigma-point Kalman filter (SPKF)

- *Initialization*:

$$\hat{\boldsymbol{x}}_0 = \mathrm{E}\left[\boldsymbol{x}_0\right] \tag{1.25}$$

$$\boldsymbol{P}_{\boldsymbol{x}_0} = \mathrm{E}\left[\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)^T\right] \tag{1.26}$$

$$\begin{aligned} \hat{\boldsymbol{x}}_0^{\mathrm{a}} &= \mathrm{E}\left[\boldsymbol{x}_0^{\mathrm{a}}\right] \\ &= \begin{bmatrix} \hat{\boldsymbol{x}}_0^T & \bar{\boldsymbol{v}}_0^T & \bar{\boldsymbol{n}}_0^T \end{bmatrix}^T \end{aligned} \tag{1.27}$$

$$\begin{aligned} \boldsymbol{P}_{\boldsymbol{x}_0}^{\mathrm{a}} &= \left[\left(\boldsymbol{x}_0^{\mathrm{a}} - \hat{\boldsymbol{x}}_0^{\mathrm{a}}\right)\left(\boldsymbol{x}_0^{\mathrm{a}} - \hat{\boldsymbol{x}}_0^{\mathrm{a}}\right)^T\right] \\ &= \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{x}_0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Q}_0 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{R}_0 \end{bmatrix} \end{aligned} \tag{1.28}$$

- *For* $k = 0, 1, 2, \ldots, N$

1. *Compute sigma points*:

$$\boldsymbol{\chi}_k^{\mathrm{a}} = \begin{bmatrix} \hat{\boldsymbol{x}}_k^{\mathrm{a}} & \hat{\boldsymbol{x}}_k^{\mathrm{a}} + \sqrt{\left(\acute{M} + \lambda\right)\boldsymbol{P}_{\boldsymbol{x}_k}^{\mathrm{a}}} & \hat{\boldsymbol{x}}_k^{\mathrm{a}} - \sqrt{\left(\acute{M} + \lambda\right)\boldsymbol{P}_{\boldsymbol{x}_k}^{\mathrm{a}}} \end{bmatrix} \tag{1.29}$$

2. *Time-update equations*:

$$\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} = f_k\left(\boldsymbol{\chi}_{i,k}^{\boldsymbol{x}}, \boldsymbol{\chi}_{i,k}^{\boldsymbol{v}}\right) \quad i = 0, 1, \ldots, 2\acute{M} \tag{1.30}$$

$$\hat{\boldsymbol{x}}_{k+1}^{-} = \sum_{i=0}^{2\acute{M}} w_i^{(m)} \boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} \tag{1.31}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}}^{-} = \sum_{i=0}^{2\acute{M}} \sum_{j=0}^{2\acute{M}} w_{ij}^{(c)} \left(\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-}\right)\left(\boldsymbol{\chi}_{j,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-}\right)^T \tag{1.32}$$

3. *Measurement-update equations*:

$$\boldsymbol{\gamma}_{i,k+1|k} = h_k\left(\boldsymbol{\chi}^{\boldsymbol{x}}_{i,k+1|k}, \boldsymbol{\chi}^{\boldsymbol{n}}_{i,k}\right) \quad i = 0, 1, \ldots, 2\acute{M} \tag{1.33}$$

$$\hat{\boldsymbol{z}}^-_{k+1} = \sum_{i=0}^{2\acute{M}} w_i^{(m)} \boldsymbol{\gamma}_{i,k+1|k} \tag{1.34}$$

$$\boldsymbol{P}_{\tilde{\boldsymbol{z}}_{k+1}} = \sum_{i=0}^{2\acute{M}} \sum_{j=0}^{2\acute{M}} w_{i,j}^{(c)} \left(\boldsymbol{\gamma}_{j,k+1|k} - \hat{\boldsymbol{z}}^-_{k+1}\right)\left(\boldsymbol{\gamma}_{i,k+1|k} - \hat{\boldsymbol{z}}^-_{k+1}\right)^T \tag{1.35}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} = \sum_{i=0}^{2\acute{M}} \sum_{j=0}^{2\acute{M}} w_{i,j}^{(c)} \left(\boldsymbol{\chi}^{\boldsymbol{x}}_{j,k+1|k} - \hat{\boldsymbol{x}}^-_{k+1}\right)\left(\boldsymbol{\gamma}_{i,k+1|k} - \hat{\boldsymbol{z}}^-_{k+1}\right)^T \tag{1.36}$$

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} \boldsymbol{P}^{-1}_{\tilde{\boldsymbol{z}}_{k+1}} \tag{1.37}$$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{K}_{k+1}\left(\boldsymbol{z}_{k+1} - \hat{\boldsymbol{z}}^-_{k+1}\right) \tag{1.38}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}} = \boldsymbol{P}^-_{\boldsymbol{x}_{k+1}} - \boldsymbol{K}_{k+1} \boldsymbol{P}^-_{\tilde{\boldsymbol{z}}_{k+1}} \boldsymbol{K}^T_{k+1} \tag{1.39}$$

- *Parameters*:

$$\boldsymbol{x}^{\mathrm{a}}_k = \begin{bmatrix} \boldsymbol{x}^T_k & \boldsymbol{v}^T_k & \boldsymbol{n}^T_k \end{bmatrix}^T \tag{1.40}$$

$$\boldsymbol{\chi}^{\mathrm{a}} = \begin{bmatrix} (\boldsymbol{\chi}^{\boldsymbol{x}})^T & (\boldsymbol{\chi}^{\boldsymbol{v}})^T & (\boldsymbol{\chi}^{\boldsymbol{n}})^T \end{bmatrix}^T \tag{1.41}$$

$$\boldsymbol{P}^{\mathrm{a}}_{\boldsymbol{x}_k} = \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{x}_k} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{R}_k \end{bmatrix} \tag{1.42}$$

$$\lambda = \alpha^2\left(\acute{M} + \kappa\right) - \acute{M} \tag{1.43}$$

$$w_0^{(c)} = \frac{\lambda}{\left(\acute{M} + \lambda\right)} + \left(1 - \alpha^2 + \beta\right) \quad , i = 0 \tag{1.44}$$

$$w_0^{(m)} = \frac{\lambda}{\left(\acute{M} + \lambda\right)} \quad , i = 0 \tag{1.45}$$

$$w_i^{(c)} = \frac{1}{2\left(\acute{M} + \lambda\right)} \quad , i = 1, 2, \ldots, 2\acute{M} \tag{1.46}$$

$$w_i^{(m)} = \frac{1}{2\left(\acute{M} + \lambda\right)} \quad , i = 1, 2, \ldots, 2\acute{M}. \tag{1.47}$$

The SPKF parameter α controls the size of the sigma-point distribution and should be within $0 \leq \alpha \leq 1$ to avoid sampling non-local points when the nonlinearities are strong [17]. $\beta \geq 0$ is the weighting term which incorporates the higher order moments of the prior distribution. For a Gaussian prior, $\beta = 2$ [23]. The parameter κ is used to make sure the positive definiteness of the covariance matrices and the default lower bound of $\kappa \geq 0$ should work for most of the cases. The dimension of the state vector is M, $\acute{M}$ is the dimension of each augmented state, and N is the length of the observation sequence.

1.4.1 Alternate interpretation of the SPKF

In this section, we demonstrate that the SPKF performs an inherent linearization called the weighted statistical linear regression (*"WSLR"*) to locally linearize the nonlinear state space [17, 28]. However, unlike the first-order Taylor series based truncation, the WSLR technique takes into account both the mean and covariance of the Gaussian random variable (GRV) at the point of linearization. We will now give a brief introduction of the relationship between the SPKF and WSLR, which will be covered in more detail in section 2.3.1.

The WSLR algorithm relates to deriving a linear approximation of a nonlinear function $g(\boldsymbol{x})$, operating on a RV $\boldsymbol{x}$ with mean $\bar{\boldsymbol{x}}$ and covariance $\boldsymbol{P}_{\boldsymbol{x}}$, i.e.,

$$\boldsymbol{z} = g\left(\boldsymbol{x}\right) \cong \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} + \boldsymbol{\epsilon}, \tag{1.48}$$

where $\boldsymbol{A}$ and $\boldsymbol{b}$ are the statistical linearization parameters that are determined by minimizing the linearization error $\boldsymbol{\epsilon}$. Defining

$$J = \mathrm{E}\left[\boldsymbol{\epsilon}^T \boldsymbol{W} \boldsymbol{\epsilon}\right] \tag{1.49}$$

is the expected mean square error with sigma-point weighting matrix $\boldsymbol{W}$,

$$\begin{aligned}[\boldsymbol{A}, \boldsymbol{b}] &= \arg\min J \\ &= \arg\min \left(\mathrm{E}\left[\boldsymbol{\epsilon}^T \boldsymbol{W} \boldsymbol{\epsilon}\right]\right).\end{aligned} \tag{1.50}$$

Now setting the partial derivative of J with respect to the elements of $\boldsymbol{b}$ and $\boldsymbol{A}$ equal to zero, we obtain,

$$\boldsymbol{b} = \bar{\boldsymbol{z}} - \boldsymbol{A}\bar{\boldsymbol{x}} \tag{1.51}$$

$$\boldsymbol{A} = \boldsymbol{P}_{\boldsymbol{xz}}^T \boldsymbol{P}_{\boldsymbol{x}}^{-1}, \tag{1.52}$$

where $\bar{\boldsymbol{z}}$ is the mean of the RV $\boldsymbol{z}$ and $\boldsymbol{P}_{\boldsymbol{xz}}$ is the cross-covariance matrix of $\boldsymbol{x}$ and $\boldsymbol{z}$. The linearization error $\boldsymbol{\epsilon}$ has zero mean and covariance $\boldsymbol{P}_{\boldsymbol{\epsilon}}$

$$\boldsymbol{P}_{\boldsymbol{\epsilon}} = \boldsymbol{P}_{\boldsymbol{z}} - \boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}\boldsymbol{A}^T, \tag{1.53}$$

where $\boldsymbol{P}_{\boldsymbol{z}}$ is the covariance matrix of $\boldsymbol{z}$. The statistical linearization procedure discussed above relates to the sigma-point approach in the sense that the mean and covariance of $\boldsymbol{x}$ and $\boldsymbol{z}$ are computed using the sigma points extracted from the probability density function $p(\boldsymbol{x})$ and $p(\boldsymbol{z})$. To summarize, the WSLR provides the linear approximation of a general nonlinear function in the MMSE context that takes into account the mean and covariance of the RV it operates upon. Hence the WSLR technique is more accurate in the statistical sense compared to the first-order Taylor series based linearization approach. The SPKF leverages the benefits of the WSLR to produce superior state estimates than the EKF, which adopts the Taylor series to perform local linearization.

1.5 Objective and Contributions of this Work

Since its introduction, the SPKF is used in numerous state estimation related applications and proved to be a more accurate alternative to the EKF. However, nonlinear smoothing algorithms based on the sigma-point Kalman filtering technique are not well established. In addition, an in-depth analysis of the analytical bounds that quantify the state estimation error of the SPKF has not yet been performed. The aim of this dissertation research is to develop novel SPKF smoothing algorithms, apply the SPKF smoothers to the real-world tracking problems and analyze the SPKF performance using analytical error bounds.

1.5.1 Summary of Research Objectives

The broad research objective is categorized into the following four subsections:

1. Derive new fixed-interval and fixed-lag smoothing algorithms for a nonlinear system using the sigma-point methodology and combine these derived smoothers into a common family of algorithms, called *sigma-point Kalman smoothers (SPKS).*

2. Apply the proposed SPKS to the real-world pedestrian tracking system in order to locate and track a person in an indoor environment.

   (a) Implement the tag-based and tag-free tracking systems where the SPKS based estimator fuses multiple observations from different unobtrusive sensors to continuously estimate a person's 2D position and velocity.

   (b) Compare the estimation accuracy to the commercial location tracking engines.

3. Expand the use of SPKS to the multiharmonic frequency tracking problem.

   (a) Track the phase, frequency and amplitude of the fundamental frequency and its harmonic components.

   (b) Compare the performance to the extended Kalman smoother (EKS) based frequency tracker using simulated and real multiharmonic periodic signals.

4. Derive (theoretically) state estimation error bounds for the SPKF.

   (a) Prove the estimation error lower bound using the *Cramér-Rao theory* and experimentally verify its performance on a variety of benchmark problems.

   (b) Formulate an expression for the estimation error upper bound using the *Lyapunov function based stochastic stability theory* and demonstrate its performance.

1.5.2 Research Contributions

Most of the research objectives stated above are successfully completed over the course of this dissertation. My accomplishments include deriving the computationally efficient SPKF smoothing algorithms, expand the use of SPKS to the challenging areas of probabilistic inference, such as indoor localization and multiharmonic frequency tracking and extend the theoretical understanding of SPKF by analyzing its error behavior. In summary, the following new and substantial contributions are made to the body of SPKF based state estimation algorithm and its applications.

1. **Sigma-point Kalman smoothers (SPKS):**

    - The following new SPKS algorithms that include both the fixed-interval and fixed-lag methodologies are implemented.

        – *Fixed-Interval Sigma-Point Kalman Smoother (FI-SPKS)*: Estimates the state at each time using all the observations over a fixed time interval. There are two variants depending on the adopted forward-backward or Rauch-Tung-Striebel (RTS) methodology.

        (a) Forward-backward statistical linearized sigma-point Kalman smoother (FBSL-SPKS)

        (b) Rauch-Tung-Striebel statistical linearized sigma-point Kalman smoother (RTSSL-SPKS)

        – *Fixed-Lag Sigma-Point Kalman Smoother (FL-SPKS)*: Estimates the state which lags behind the current observation by a fixed time interval. The following variants include the state augmentation, forward-backward and RTS techniques.

        (a) State-augmented sigma-point Kalman smoother (Aug-SPKS)

        (b) Forward-backward a priori sigma-point Kalman smoother (FB-Priori-SPKS)

        (c) Forward-backward statistical linearized sigma-point Kalman smoother (FBSL-SPKS)

        (d) Rauch-Tung-Striebel statistical linearized sigma-point Kalman smoother (RTSSL-SPKS)

    - The relationship between the WSLR and SPKF is explored to derive the smoothing formulations.

    - The computational complexity and memory of all the fixed-interval and fixed-lag SPKS algorithms are evaluated. It is shown that the fixed-lag SPKS requires significantly less memory to generate smoothed estimates compared to the fixed-interval case.

- The performance of the SPKS algorithms are experimentally verified on two benchmark examples: Mackey-glass nonlinear time-series estimation and vehicle re-entry tracking.
- The superiority of our SPKS over the extended Kalman smoother (EKS) and other sigma-point smoothing approaches is established in terms of estimation accuracy and computational efficiency.

2. **Real-world application of SPKS algorithms: unobtrusive indoor pedestrian tracking**

   A novel SPKS based Bayesian inference system that tracks an user in an indoor environment using unobtrusive sensors is successfully developed. Two variants of the tracking mechanism are implemented based on the requirement of carrying a receiver tag.

   - *Tag-based indoor tracking using received signal strength indication (RSSI)*: In this system, the primary observation used for tracking is RSSI. A person carries a small body-borne device that periodically measures the RSSI at 3 or more standard Wi-Fi access points.
     - The SPKS combines a potential field based dynamic model with RSSI observations to track a person's 2D position and velocity.
     - The observation model is generated in a separate calibration phase by fitting nonlinear radial basis function (RBF) maps between known user positions and RSSI observations.
     - The SPKS augments the RSSI measurements with the infra-red (IR) motion sensors and binary foot-switches in order to improve the estimation accuracy of the tracker.
     - The estimation accuracy of our system is tested against a commercial tracking engine known as *Ekahau*, which also uses RSSI. The consistent performance improvement over the Ekahau was demonstrated.

- *Tag-free indoor tracking using wall-mounted ultrasonic transducers*: This system does not require a body-borne tag.
  - A novel SPKS based tag-free solution for indoor tracking is developed that utilizes range information from wall-mounted ultrasonic transducers to estimate a person's 2D position and velocity (first of its kind to author's knowledge).
  - Signal processing techniques are used for background subtraction and subsequently calculate the 1D range of the moving person.
  - The range data from active and passive sonar-modules provide the observations for the SPKS based tracking algorithm.
  - Two different tracking procedures are adopted:
    - (a) Range-map approach, which generates the observation model by fitting RBF maps between known calibration locations and 1D ranges
    - (b) Simultaneous localization and mapping (SLAM) approach
  - The SLAM based tracking procedure corresponds to simultaneously estimating the state of the person (position and velocity) and the parameters of the observation model. Parameters correspond to either the RBF parameters in position-range mapping (terrain aided navigation system (TANS-SLAM)) or the 2D sonar module locations (landmark aided navigation system (LANS-SLAM)).
  - The tracking performance is compared to an accurate commercial tag-based system developed by *Ubisense*, which uses ultra-wide-band (UWB) for time difference of arrival (TOA) localization.

3. **Real-world application of SPKS algorithms: multiharmonic frequency tracking**

   This research is performed in collaboration with James McNames's research group at Portland State University.

   - The SPKS based Bayesian inference algorithm is implemented for tracking the phase, frequency, and amplitude of the fundamental frequency and the harmonic components present in a periodic signal.

- The performance of the SPKS is compared with the EKS, based on three criterions: normalized mean-square-error, normalized frequency mean-square-error and square-frequency-error.
- It is demonstrated that the SPKS multiharmonic tracker is significantly more accurate, converges faster to the true solution, and robust to noise than the EKS.

4. **Estimation error bounds for discrete time SPKF**
    - An estimation error lower bound is derived using the well-known *Cramér-Rao theory.*
    - It is demonstrated that the estimation error of the SPKF is exponentially bounded in the mean-square sense by an upper bound.
    - The performance of the lower and upper bound is experimentally verified on two state estimation examples: Mackey-Glass nonlinear time-series estimation and tracking a vehicle that re-enters into the earth's atmosphere from the outer space.

1.6 Thesis Outline

The remainder of the thesis is organized as below:

- Chapter 2 is one of the core chapters in this dissertation, which presents the development of different SPKS algorithms in detail. The performance of the SPKS is demonstrated in two examples including the Mackey-Glass time series estimation and vehicle re-entry tracking.
- Chapter 3 focusses on the application of SPKS to the tag-based indoor tracking problem. The chapter starts with a survey of commercial indoor tracking systems

and research prototypes, including their architecture, sensor platforms, and positioning algorithms. We provide a detailed discussion of the SPKS based tracking system including the dynamic model, sensor observation models and how the various sensor outputs are integrated into a Kalman framework. The performance of the proposed algorithm is compared with a commercially available positioning engine developed by Ekahau Inc.

- Chapter 4 discusses a tag-free solution to unobtrusive indoor tracking using wall mounted ultrasonic sensors. The specific tracking application is covered in depth, giving details about how the range information is extracted from each ultrasonic signal and how the range estimates from multiple sonar units are incorporated into the SLAM framework to estimate the user's 2D position and velocity. Tracking results are shown for a number of trials and the estimates are compared to a commercial tag-based system developed by Ubisense.

- Chapter 5 introduces the SPKS tracking approach to the multiharmonic frequency tracking problem. It specifically discusses the dynamic and observation model used for frequency tracking and demonstrates the system accuracy on a set of simulated/real signals.

- Chapter 6 focuses on the derivation, implementation and verification of the state estimation error bounds for the SPKF. A thorough literature review is provided first and then the lower and upper error bound is derived from first principles. Experimental results that validate our proofs are also shown.

- Chapter 7 presents the summary and conclusions of our work.

1.7 Publications

A large majority of the work either has already been published at numerous conferences or submitted at the peer-reviewed literature for review. Here is a list of accepted publications that are related to my dissertation:

- Eric A. Wan and Anindya S. Paul, “A tag-free solution to unobtrusive indoor tracking using wall-mounted ultrasonic transducers,” Accepted in *2010 IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Zurich, Switzerland, September, 2010.

- Anindya S. Paul and Eric A. Wan, “RSSI based indoor localization and tracking using sigma-point Kalman smoothers,” *IEEE Journal on Special Topics in Signal Processing: Special Issue on Advanced Signal Processing for GNSS and Robust Navigation*, vol. 3, no. 5, pp. 860-873, October 2009.

- Sunghan Kim, Anindya S. Paul, Eric A. Wan and James McNames, “New multi-harmonic frequency tracking using the sigma-point Kalman smoother,” *EURASIP Journal on Advances in Signal Processing*, vol. 2010, no. 467150, pp. 1-13, February 2010.

- Sunghan Kim, Anindya S. Paul, Eric A. Wan and James McNames, “Multiharmonic Tracking Using Sigma-Point Kalman Filter,” In proceedings of *Annual International Conference of the IEEE Engineering in Medicine and Biology Society 2008 (IEEE EMBC 08)*, Vancouver, Canada, August, 2008.

- Anindya S. Paul and Eric A. Wan, “Wi-Fi Based Indoor Localization and Tracking Using Sigma-Point Kalman Filtering Methods,” In proceedings of *IEEE/ION Position Location and Navigation Symposium 2008 (PLANS 2008)*, Monterey, CA, USA, May, 2008.

- Anindya S. Paul and Eric A. Wan, “A new formulation for nonlinear forward-backward smoothing,” In proceedings of *IEEE International Conference on Acoustics, Speech and Signal Processing 2008 (ICASSP 2008)*, pp. 3621- 3624, Las Vegas, NV, USA, March-April, 2008.

- Misha Pavel, Tamara Hayes, Ishan Tsay, Deniz Erdogmus, Anindya S. Paul, Nicole Larimer, Holly Jimison and John Nutt, “Continuous Assessment of Gait Velocity in Parkinson’s Disease from Unobtrusive Measurements,” In proceedings of the *3rd*

International IEEE EMBS Conference on Neural Engineering, pp. 700-703, Kohala Coast, Hawaii, USA, May, 2007.

- Anindya S. Paul, Eric A. Wan and Alex T. Nelson, "Noise Reduction For Heart Sounds Using a Modified Minimum-Mean Squared Error Estimator with ECG Gating," In proceedings of *28th IEEE EMBS Annual International Conference*, pp. 3385- 3390, New York City, USA, August-September, 2006.

- Anindya S. Paul, "Dual Kalman Filter for Autonomous Terrain Aided Navigation in Unknown Environment," *Technical Report*, OGI School of Science and Engineering, Oregon Health and Science University, Beaverton, OR, May 2005.

- Anindya S. Paul and Eric A. Wan, "Dual Kalman Filters for Autonomous Terrain Aided Navigation in Unknown Environment," In proceedings of *IEEE International Joint Conference on Neural Networks (IJCNN)*, vol. 5, pp. 2784- 2789, Montreal, Canada, July-August, 2005.

In addition, the dissertation research and corresponding publications have contributed to external research grants, resulting in novel applications, new sensor platform for tracking, and further refinement of tracking algorithms.

Chapter 2

Sigma-Point Kalman Smoothers (SPKS): New Fixed-Interval and Fixed-Lag Smoothing Formulations for Nonlinear Systems

2.1 Overview

In this chapter, we apply the sigma-point Kalman filtering approach to derive a smoothing scheme for nonlinear state space system. We take advantage of the relationship between the SPKF and weighted statistical linear regression (WSLR) to derive the nonlinear sigma-point Kalman smoother (SPKS) equations. The performance of the SPKS algorithms are evaluated using the Mackey-Glass nonlinear time series estimation and the vehicle re-entry tracking examples. It is shown that the proposed computationally efficient SPKS algorithms easily outperform the extended Kalman filter (EKF), extended Kalman smoother (EKS) and SPKF and also perform comparably with other existing sigma-point smoothers.

This chapter is organized as follows: Section 2.2 introduces our approach considering both the fixed-interval (FI) and fixed-lag (FL) methods. Later, both the FI-SPKS and FL-SPKS methods are detailed in sections 2.3-2.4 with step-by-step derivations. Implementation details and experimental results are shown in section 2.5. Section 2.6 concludes the chapter with a discussion.

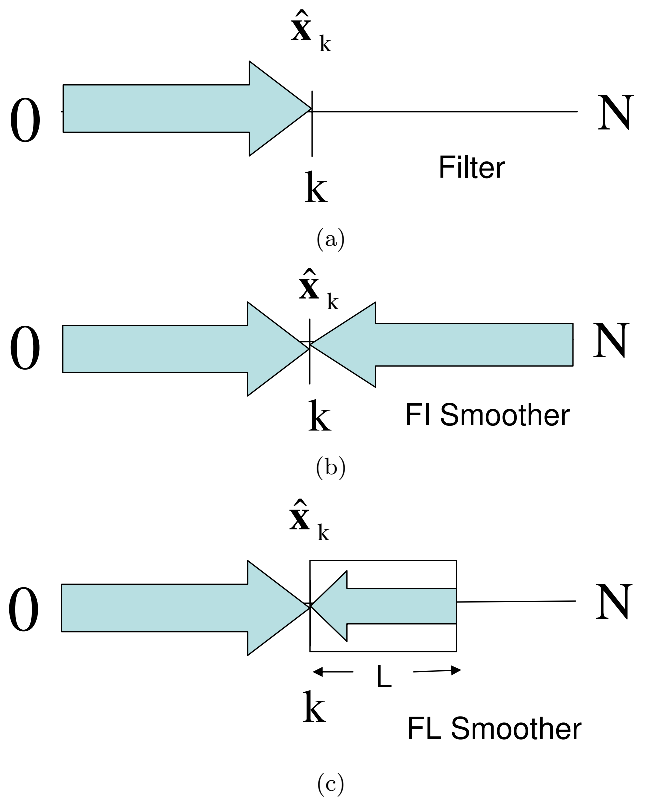


Figure 2.1: The above schematic diagrams demonstrate the methodologies of a filter and smoothers (a) Filter, (b) Fixed-Interval (FI) smoother, (c) Fixed-Lag (FL) smoother.

2.2 Introduction

2.2.1 Filter vs Smoother

The Kalman filter provides the optimal Bayesian recursive estimate in the MMSE sense for the state $\boldsymbol{x}_k$ of a linear state-space system driven by a white Gaussian noise [18]. The estimate is optimal given all noisy measurements $\boldsymbol{Z}_k = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_k]$ up to the current time index k. A filter can be represented graphically as shown in Figure 2.1(a).

In contrast, the Kalman smoother (KS) estimates the conditional expectation of the state $\boldsymbol{x}_k$ given all past and future measurements $\boldsymbol{Z}_k = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N]\,,\ 1 \le k \le N$. A filter uses only the past and current measurements up to time k to estimate $\hat{\boldsymbol{x}}_k$, whereas a smoother estimates $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ for $k \in (0, N)$ utilizing all the available measurements including the future observations. As a smoother generally deals with more measurements, it is able to achieve superior accuracy to that of a filter. The obvious downside of using a smoother is that it performs non-real-time operations, possesses higher computational complexity and takes longer processing time to compute estimates. Several common Kalman smoothing formulations are given in [11, 12, 29–31]. Based on their work, smoothers are usually divided into two main categories:

- A *fixed-interval smoother* estimates the state $\boldsymbol{x}_k$ at time k using all the measurements $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N$ over a fixed time interval N. This type of smoother, which operates within a fixed set of observations, is generally suitable for offline estimation. The operation of a fixed-interval (FI) smoother is graphically shown in Figure 2.1(b).

- A *fixed-lag smoother* estimates the system state $\boldsymbol{x}_{k-L}$ at time $k - L$ given measurements up to time k, $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_k$, where the time index k continuously moves forward as we obtain new measurements. In other words, we estimate the unknown state $\boldsymbol{x}_k$ by taking into account all past, present and L future observations. The fixed lag parameter L trades off system latency for more estimation accuracy. A schematic diagram of a fixed-lag (FL) smoother is illustrated in Figure 2.1(c).

2.2.2 Linear Smoothers

For the case of linear state space model, the fixed-interval and fixed-lag smoother formulations are derived using the Kalman filter. Forward-Backward (FB) and Rauch-Tung-Striebel (RTS) methods are perhaps the most popular ways of performing the fixed-interval smoothing [11, 12]. In the FB approach to smoothing [29], a Kalman filter is run from $k = 1$ to $k = N$ to obtain forward estimates $\hat{\boldsymbol{x}}_k^{\mathrm{f}}$ incorporating the observations $\boldsymbol{z}_j$ for $1 \leq j \leq k$. Similarly, estimates $\hat{\boldsymbol{x}}_k^{\mathrm{b}}$, which uses the observations $\boldsymbol{z}_j$ for $k \leq j \leq N$, can be obtained by operating a Kalman filter that runs backward in time from $k = N$ to $k = 1$. The backward filter dynamics is based on the inverse dynamics of the forward filter. The forward and backward estimates are then optimally combined at each k to generate the smoothed estimates $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$.

The RTS approach is another way of implementing the fixed-interval smoothing which is more computationally efficient than the FB method [30]. In the RTS form, the backward estimates are no longer needed in order to perform smoothing. In the RTS smoother, a forward Kalman filter generates state estimates $\hat{\boldsymbol{x}}_k$ and estimation error covariances $\boldsymbol{P}_k$ from time $k = 1$ to $k = N$ incorporating the measurements $\boldsymbol{z}_j$ for $1 \leq j \leq k$. In addition to the state estimates and error covariances, the forward pass also saves the intermediate results including the state prediction $\hat{\boldsymbol{x}}_k^-$ and the prediction error covariance $\boldsymbol{P}_k^-$ at each time k. A backward smoothing pass runs backward in time in a sequence from $k = N$, computing the smoothed state estimates $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ from the forward estimates and the intermediate results stored during the forward pass. Because the RTS smoother is based on applying a correction pass to the forward filtering result, not on running an independent backward filter, it can be applied in those cases when the forward state dynamics is non-invertible.

In case of fixed-lag smoothing, the *"state augmentation"* approach is the most widely used method [1, 11, 12], where an augmented state $\tilde{\boldsymbol{x}}_k$ can be formed by combining the current and L previous states

$$\tilde{\boldsymbol{x}}_k = \begin{bmatrix} \boldsymbol{x}_k & \boldsymbol{x}_{k-1} & \cdots & \boldsymbol{x}_{k-L} \end{bmatrix}^T . \tag{2.1}$$

A standard Kalman filter can be applied to estimate the full state of the augmented system, with the last element of the augmented state vector being equal to the smoothed estimate at time $k - L$ given all measurements up to time k. Although, the *state augmentation* method requires minimal effort to implement, computational complexity of this approach, which is proportional to the cube of the state dimension, can be a serious concern for certain applications.

Simon proposed a new formulation for the fixed-lag smoother to counter the high computational load, which bypasses the state augmentation by a sliding window based forward-backward approach [11]. Because no specific name was given in [11], we refer this smoother as a *fixed-lag forward-backward Kalman smoother (FL-FB-KS).* Instead of augmenting the current and L past states, the FL-FB-KS divides the data into blocks (e.g., $N = \sum N_i$) and then performs a forward-backward operation on the buffered blocks of data as they become available. The standard Kalman filter equations are applied to obtain the state estimates during the forward pass. A smoothing pass is then followed within the fixed size window, running backward in time and computing the smoothed estimate $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$.

2.2.3 Nonlinear Smoothers

The aim of our work is to derive a smoothing methodology for the nonlinear state space system using the SPKF based implementation. Although substantial work has been done on the broad topic of linear Kalman smoothing [11, 12], derivation of nonlinear smoothers is still an active area of research [16,26,31,32]. Extended Kalman smoother (EKS) remains a popular choice to estimate smoothed states using a nonlinear state space model [33, 34]. The EKS has been shown to have superior performance on a number of applications [12, 15, 31, 33], but since it makes use of the EKF as the core algorithm, it suffers from the same EKF inaccuracies, such as linearization error, filter divergence, etc., as discussed in Section 1.4. Hence in this chapter, we concentrate solely on designing a SPKS framework in order to leverage all of the benefits the SPKF exhibits over the EKF.

Sigma-Point Kalman Smoother (SPKS)

Before going into the details of our SPKS methodologies, we first describe the two variants of the SPKS that have appeared in the literature. In [17, 26], the SPKS uses a forward-backward approach. Although no specific name was given for this SPKS, in this article it will be referred as a *forward-backward nonlinear sigma-point Kalman smoother (FBNL-SPKS)* in order to differentiate from our proposed sigma-point smoothers. A standard SPKF is run in the forward direction using the nonlinear model. A second SPKF is then run in the backward direction and the two estimates are optimally combined. The backward filter dynamic model which is represented as an inverse forward dynamics is approximated by training a backward nonlinear predictor (e.g., neural network model). As the backward model needs to be fit to the data, it is both application specific and potentially time consuming. In [32], an unscented Rauch-Tung-Striebel smoother *(URTSS)* is proposed that uses a joint distribution of the current and future state in order to obtain a smoothed estimate of the current state. While this avoids the need for the inverse dynamics, the computational complexity (cube of the state dimension) increases significantly due to the doubling of the state dimension.

In this work, we have proposed the following 6 SPKS algorithms that include both the fixed-interval and fixed-lag methodologies.

- **Fixed-interval sigma-point Kalman smoother (FI-SPKS):**
  1. Forward-backward statistical linearized sigma-point Kalman smoother (FBSL-SPKS)
  2. Rauch-Tung-Striebel statistical linearized sigma-point Kalman smoother (RTSSL-SPKS)
- **Fixed-lag sigma-point Kalman smoother (FL-SPKS):**
  1. State-augmented sigma-point Kalman smoother (Aug-SPKS)
  2. Forward-backward a priori sigma-point Kalman smoother (FB-Priori-SPKS)
  3. Forward-backward statistical linearized sigma-point Kalman smoother (FBSL-SPKS)
  4. Rauch-Tung-Striebel statistical linearized sigma-point Kalman smoother (RTSSL-SPKS)

The FI-SPKS makes use of the forward-backward and RTS approaches as described in Section 2.2.2. The FBSL-SPKS, which was first presented in [35], is a forward-backward approach. In the FBSL-SPKS, a standard SPKF is used as the forward filter. The forward SPKF generates state estimate $\hat{\boldsymbol{x}}_k^{\mathrm{f}}$ at each time k using measurements up to time k. The backward filter needs an inverse dynamics of the forward filter. While smoothers based on the EKF compute the first order Taylor series based linearized dynamics, the forward non-linear dynamics are never analytically linearized with the SPKF. Hence the dynamics of the backward filter in the forward-backward approach are not well defined. Our proposed SPKS makes use of the weighted statistical linear regression (WSLR) formulation of the filter, which is straightforward, direct and computationally efficient. As will be detailed later, the WSLR is a linearization technique that takes into account the uncertainty of the prior random variable (RV) when linearizing the nonlinear model [17]. In this way, the WSLR is more accurate in the statistical sense than the first-order Taylor series based linearization which does not factor in the *probabilistic spread* at the point of linearization. By representing the forward dynamics in terms of WSLR, we are able to derive an information filter that computes states $\hat{\boldsymbol{x}}_k^{\mathrm{b}}$ by running backward in time. Estimates of the forward and backward filter are then statistically combined to generate smoothed estimates $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ in the standard manner. The complete derivation for the FBSL-SPKS is given in Section 2.3.2. The RTSSL-SPKS follows the RTS methodology in order to estimate smoothed states propagating through nonlinear dynamics. Similar to the FBSL-SPKS, the RTSSL-SPKS uses a standard SPKF as the forward filter to estimate the state $\hat{\boldsymbol{x}}_k$ and the state prediction $\hat{\boldsymbol{x}}_k^-$. Incorporating the pseudo-linearized form of SPKF, we compute a smoothing gain at each k during the backward smoothing pass that linearly combines the forward estimate, $\hat{\boldsymbol{x}}_k$, and the difference between the future smoothed estimate, $\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}}$, and the state prediction, $\hat{\boldsymbol{x}}_{k+1}^-$, in order to compute the current smoothed estimate $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$. The full derivation for the RTSSL-SPKS algorithm is presented in Section 2.3.3.

The Aug-SPKS, which adopts *state-augmentation* is the most simple and straightforward method to perform the nonlinear fixed-lag smoothing. In this method, an augmented state space system is formed at each time k by the current and L previous states. A standard SPKF is then used to estimate the augmented states using the observations up to time k with the last element of the augmented vector providing the smoothed state $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$. The Aug-SPKS algorithm is described in Section 2.4.1. The FB-Priori-SPKS is the extension of the linear fixed-lag smoother proposed by Simon [11]. Instead of augmenting the individual state vectors, the state estimates in the FB-Priori-SPKS are computed sequentially within a sliding window containing the current and L previous observations. The forward filter, which is a priori form of SPKF, operates on the statistically linearized state space to compute the state prediction $\hat{\boldsymbol{x}}_k^-$ and the prediction error covariance $\boldsymbol{P}_k^-$. A backward smoothing loop then determines the smoothed state $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ from the forward filtering results. Section 2.4.2 presents a summary of the FB-priori-SPKS algorithm. The FBSL-SPKS and RTSSL-SPKS algorithms, which we demonstrate for the fixed-interval case, is also applied for the fixed-lag category in a sliding window fashion. Instead of operating on a fixed set of N measurements, we apply the same algorithm in a windowed

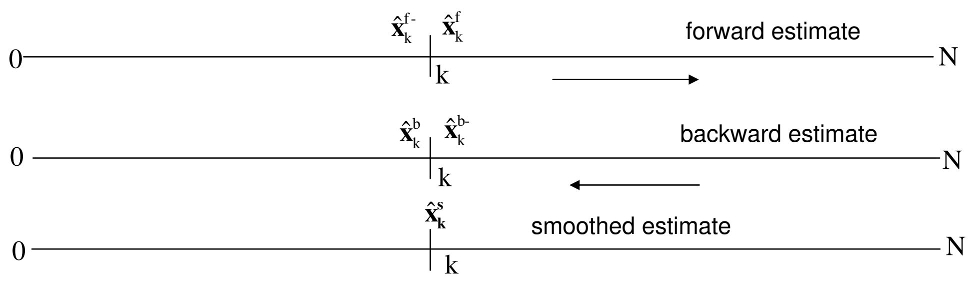


Figure 2.2: This schematic diagram explains the concept of the FBSL-SPKS methodology. The forward SPKF computes a posteriori estimate $\hat{\boldsymbol{x}}_k^{\mathrm{f}}$ and the backward information filter generates a priori estimate $\hat{\boldsymbol{x}}_k^{\mathrm{b}-}$ at each discrete time k. The forward posteriori and backward priori estimate are then statistically combined to generate the smoothed estimate $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$.

subset of L measurements from $k = j - L + 1$ to $k = j$, where the time index j constantly moves forward in time as we receive new measurements. The fixed-lag FBSL-SPKS and RTSSL-SPKS algorithms are shown in sections 2.4.3 and 2.4.4.

2.3 Fixed-Interval Sigma-Point Kalman Smoothers (FI-SPKS)

In this section, we derive the formulations for the fixed-interval smoother operating on a nonlinear state space. In the FI-SPKS framework, we develop both the FBSL-SPKS and RTSSL-SPKS algorithms to estimate states at each discrete time k using a fixed set of N measurements. Before going into the detailed algorithmic derivations for the FBSL-SPKS and RTSSL-SPKS, we first present how the relationship between the SPKF and WSLR can be applied to compute a pseudo linearized dynamics suitable for formulating the equations of the fixed-interval smoothers.

2.3.1 Relationship between the SPKF and WSLR

Consider a prior RV $\boldsymbol{x}$ which is propagated through a nonlinear function $g(\boldsymbol{x})$ to obtain a posterior RV $\boldsymbol{z}$. While $\boldsymbol{x}$ is a continuous RV observed at each time index k, for the purpose of explaining the WSLR approach, we temporarily omit the discrete time index k from the state and observation variables. Sigma points $\boldsymbol{\chi}_i, i = 0, 1, \ldots, 2M$ are selected as the prior mean $\bar{\boldsymbol{x}}$ plus and minus the columns of the square root of the prior covariance

$\boldsymbol{P_x}$

$$\boldsymbol{\chi} = \left[\begin{array}{ccc} \bar{\boldsymbol{x}} & \bar{\boldsymbol{x}} + \gamma\sqrt{\boldsymbol{P_x}} & \bar{\boldsymbol{x}} - \gamma\sqrt{\boldsymbol{P_x}} \end{array} \right], \tag{2.2}$$

where M is the RV dimension and γ is the composite scaling parameter. The sigma point set $\boldsymbol{\chi}$ completely capture the mean $\bar{\boldsymbol{x}}$ and the covariance $\boldsymbol{P_x}$ of the prior RV $\boldsymbol{x}$.

$$\bar{\boldsymbol{x}} = \sum_{i=0}^{2M} w_i \boldsymbol{\chi}_i \tag{2.3}$$

$$\boldsymbol{P_x} = \sum_{i=0}^{2M} w_i \left(\boldsymbol{\chi}_i - \bar{\boldsymbol{x}}\right)\left(\boldsymbol{\chi}_i - \bar{\boldsymbol{x}}\right)^T, \tag{2.4}$$

where w_i is the normalized scaler weight for each sigma point. Each prior sigma point is propagated through the nonlinearity to form the posterior sigma point $\boldsymbol{\gamma}_i$

$$\boldsymbol{\gamma}_i = g\left(\boldsymbol{\chi}_i\right) \quad i = 0, 1, \ldots, 2M. \tag{2.5}$$

The posterior statistics can then be approximated using weighted averaging of the posterior sigma points,

$$\hat{\boldsymbol{z}} = \sum_{i=0}^{2M} w_i \boldsymbol{\gamma}_i \tag{2.6}$$

$$\boldsymbol{P_z} = \sum_{i=0}^{2M} w_i \left(\boldsymbol{\gamma}_i - \hat{\boldsymbol{z}}\right)\left(\boldsymbol{\gamma}_i - \hat{\boldsymbol{z}}\right)^T \tag{2.7}$$

$$\boldsymbol{P_{xz}} = \sum_{i=0}^{2M} w_i \left(\boldsymbol{\chi}_i - \bar{\boldsymbol{x}}\right)\left(\boldsymbol{\gamma}_i - \hat{\boldsymbol{z}}\right)^T. \tag{2.8}$$

This deceptively simple approach captures the desired posterior statistics more accurately than using the standard Taylor series based first-order linearization techniques. The implementation is also simpler, as it avoids the need to analytically linearize the nonlinear function, and only requires direct function evaluations. The performance of the sigma-point approach in capturing the mean and covariance of a GRV which undergoes a nonlinear transformation is demonstrated in Figure 2.3. The left plot shows the mean and covariance propagation using the Monte-Carlo sampling. The center plot demonstrates the results using first-order linearization as in the EKF. The right hand plot depicts the performance of the sigma-point approach. Note, only 5 sigma points are needed to approximate the 2D distribution. The superior performance of the sigma-point approach is

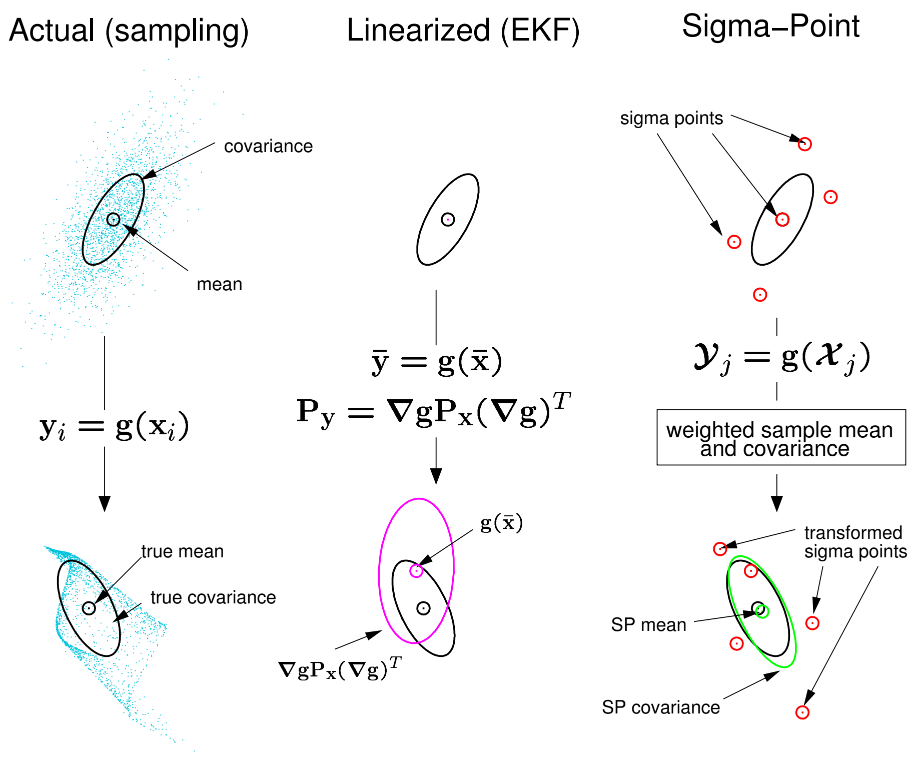

Actual (sampling)
Linearized (EKF)
Sigma−Point
covariance
mean
sigma points
$\mathbf{y}_i = \mathbf{g}(\mathbf{x}_i)$
$\bar{\mathbf{y}} = \mathbf{g}(\bar{\mathbf{x}})$
$\mathbf{P}_\mathbf{y} = \nabla\mathbf{g}\mathbf{P}_\mathbf{x}(\nabla\mathbf{g})^T$
$\mathcal{Y}_j = \mathbf{g}(\mathcal{X}_j)$
weighted sample mean and covariance
true mean
true covariance
$\mathbf{g}(\bar{\mathbf{x}})$
$\nabla\mathbf{g}\mathbf{P}_\mathbf{x}(\nabla\mathbf{g})^T$
transformed sigma points
SP mean
SP covariance

Figure 2.3: 2D example of the sigma-point approach. The accuracy of the sigma-point method in propagating the mean and covariance of the prior GRV through a nonlinear function is compared with the Monte-Carlo sampling and the EKF approaches.

clearly evident. The figure also proves that the sigma-point weighted averaging technique is an accurate approach to estimate the first and second-order statistics of a posterior GRV.

An alternate view of the sigma point approach can be found by considering the weighted statistical linearization of the nonlinear dynamics

$$\boldsymbol{z} = g\left(\boldsymbol{x}\right) \cong \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} + \boldsymbol{\epsilon}, \tag{2.9}$$

where $\boldsymbol{A}$ and $\boldsymbol{b}$ are the statistical linearization parameters and can be determined by minimizing the expected mean square error which takes into account the uncertainty of the prior RV $\boldsymbol{x}$. Define

$$J = \mathrm{E}\left[\boldsymbol{\epsilon}^T \boldsymbol{W} \boldsymbol{\epsilon}\right] \tag{2.10}$$

as the expected mean square error with sigma-point weighting matrix $\boldsymbol{W}$

$$\begin{aligned}[\boldsymbol{A},\boldsymbol{b}] =&\text{arg min } J \\ =&\text{arg min } \left(\mathrm{E}\left[\boldsymbol{\epsilon}^T\boldsymbol{W}\boldsymbol{\epsilon}\right]\right). \end{aligned} \tag{2.11}$$

Taking a partial derivative of J with respect to $\boldsymbol{b}$

$$\frac{\partial J}{\partial \boldsymbol{b}} =\mathbf{0}. \tag{2.12}$$

Now substituting J,

$$J = (g(\boldsymbol{x}) - \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^T \boldsymbol{W} (g(\boldsymbol{x}) - \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}), \tag{2.13}$$

in Equation (2.12) to obtain

$$\frac{\partial\left[(g(\boldsymbol{x}) - \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^T \boldsymbol{W} (g(\boldsymbol{x}) - \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\right]}{\partial \boldsymbol{b}} =\mathbf{0}. \tag{2.14}$$

After cross multiplication and differentiation, (2.14) simplifies to

$$\mathrm{E}\left[\boldsymbol{W}(g(\boldsymbol{x}) - \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\right] =\mathbf{0}. \tag{2.15}$$

Finally, an expression for $\boldsymbol{b}$ can be computed from (2.15)

$$\begin{aligned}\boldsymbol{b} =& \mathrm{E}\left[g(\boldsymbol{x})\right] - \boldsymbol{A}\,\mathrm{E}\left[\boldsymbol{x}\right] \\ \boldsymbol{b} =& \hat{\boldsymbol{z}} - \boldsymbol{A}\bar{\boldsymbol{x}}. \end{aligned} \tag{2.16}$$

After we obtain $\boldsymbol{b}$, a new expression for J can be formed by substituting $\boldsymbol{b}$ into Equation (2.13)

$$J = (g(\boldsymbol{x}) - \boldsymbol{A}\boldsymbol{x} - \hat{\boldsymbol{z}} + \boldsymbol{A}\bar{\boldsymbol{x}})^T \boldsymbol{W} (g(\boldsymbol{x}) - \boldsymbol{A}\boldsymbol{x} - \hat{\boldsymbol{z}} + \boldsymbol{A}\bar{\boldsymbol{x}}). \tag{2.17}$$

In order to compute $\boldsymbol{A}$, take a partial derivative of J with respect to $\boldsymbol{A}$

$$\frac{\partial J}{\partial \boldsymbol{A}} =\mathbf{0}. \tag{2.18}$$

Now substitute J from (2.17) into (2.18)

$$\frac{\partial\left[(g(\boldsymbol{x}) - \hat{\boldsymbol{z}} - \boldsymbol{A}(\boldsymbol{x} - \bar{\boldsymbol{x}}))^T \boldsymbol{W} (g(\boldsymbol{x}) - \hat{\boldsymbol{z}} - \boldsymbol{A}(\boldsymbol{x} - \bar{\boldsymbol{x}}))\right]}{\partial \boldsymbol{A}} =\mathbf{0}. \tag{2.19}$$

After cross multiplication and differentiation, Equation (2.19) simplifies to

$$E\left[\boldsymbol{W}\left[\boldsymbol{A}\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^T+\left(g\left(\boldsymbol{x}\right)-\hat{\boldsymbol{z}}\right)\tilde{\boldsymbol{x}}^T\right]\right]=\boldsymbol{0}, \tag{2.20}$$

where

$$\tilde{\boldsymbol{x}}=\left(\boldsymbol{x}-\bar{\boldsymbol{x}}\right). \tag{2.21}$$

By further simplifying Equation (2.20), an expression for $\boldsymbol{A}$ can be determined as follows

$$\begin{aligned}\boldsymbol{A}=&\mathrm{E}\left[\left(\boldsymbol{x}-\bar{\boldsymbol{x}}\right)\left(\boldsymbol{z}-\hat{\boldsymbol{z}}\right)^T\right]^T E\left[\left(\boldsymbol{x}-\bar{\boldsymbol{x}}\right)\left(\boldsymbol{x}-\bar{\boldsymbol{x}}\right)^T\right]^{-1}\\ =&\boldsymbol{P}_{\boldsymbol{xz}}^T\boldsymbol{P}_{\boldsymbol{x}}^{-1},\end{aligned} \tag{2.22}$$

where the prior mean ($\bar{\boldsymbol{x}}$) and covariance ($\boldsymbol{P}_{\boldsymbol{x}}$) are calculated in (2.3)-(2.4) from the prior sigma points. Similarly, the posterior mean ($\hat{\boldsymbol{z}}$) and covariances ($\boldsymbol{P}_{\boldsymbol{z}}$ and $\boldsymbol{P}_{\boldsymbol{xz}}$) are calculated from the posterior sigma points as shown in (2.6)-(2.8). The linearization error $\boldsymbol{\epsilon}$ has zero mean and covariance $\boldsymbol{P}_{\boldsymbol{\epsilon}}$ which can be computed as below:

$$\begin{aligned}\boldsymbol{P}_{\boldsymbol{\epsilon}}=&\mathrm{E}\left[\boldsymbol{\epsilon}^T\boldsymbol{W}\boldsymbol{\epsilon}\right]\\ =&\mathrm{E}\left[\left(g\left(\boldsymbol{x}\right)-\hat{\boldsymbol{z}}-\boldsymbol{A}\left(\boldsymbol{x}-\bar{\boldsymbol{x}}\right)\right)^T\boldsymbol{W}\left(g\left(\boldsymbol{x}\right)-\hat{\boldsymbol{z}}-\boldsymbol{A}\left(\boldsymbol{x}-\bar{\boldsymbol{x}}\right)\right)\right]\\ =&\boldsymbol{P}_{\boldsymbol{z}}-\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{xz}}-\boldsymbol{P}_{\boldsymbol{xz}}^T\boldsymbol{A}^T+\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}\boldsymbol{A}^T.\end{aligned} \tag{2.23}$$

Replacing $\boldsymbol{P}_{\boldsymbol{xz}}^T=\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}$ from (2.22)

$$\begin{aligned}\boldsymbol{P}_{\boldsymbol{\epsilon}}=&\boldsymbol{P}_{\boldsymbol{z}}-\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}\boldsymbol{A}^T-\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}\boldsymbol{A}^T+\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}\boldsymbol{A}^T\\ =&\boldsymbol{P}_{\boldsymbol{z}}-\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}\boldsymbol{A}^T.\end{aligned} \tag{2.24}$$

From the posterior error covariance $\boldsymbol{P}_{\boldsymbol{z}}$ obtained using (2.24),

$$\boldsymbol{P}_{\boldsymbol{z}}=\boldsymbol{A}\boldsymbol{P}_{\boldsymbol{x}}\boldsymbol{A}^T+\boldsymbol{P}_{\boldsymbol{\epsilon}}, \tag{2.25}$$

we observe that the covariance of the linearization error $\boldsymbol{P}_{\boldsymbol{\epsilon}}$ is added when calculating the posterior covariance $\hat{\boldsymbol{P}}_{\boldsymbol{z}}$. The addition of the linearization error to the computed posterior statistics is very important especially when there is severe nonlinearity over the uncertainty region of prior RV. First-order Taylor-series-based linearization employed by the EKF often diverges in highly nonlinear region as it only performs linearization around the mean of the RV but neglects this error term. In general, the WSLR technique is an optimal way of linearizing any nonlinear function in the MMSE sense as this approach explicitly takes into account the prior RV statistics (e.g. mean and covariance).

To form the SPKF, we consider the nonlinear state space model:

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) \tag{2.26}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{n}_k\right), \tag{2.27}$$

where $\boldsymbol{x}_k \in \mathbb{R}^M$ is the state, $\boldsymbol{z}_k \in \mathbb{R}^P$ is the observation at time index k, $\boldsymbol{v}_k$ and $\boldsymbol{n}_k$ are the Gaussian distributed process and observation noises, $f(.)$ is the nonlinear dynamic model and $h(.)$ is the nonlinear observation model function. The process and observation noise have zero mean and covariances $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$, respectively. The SPKF is then derived by recursively applying the sigma-point selection scheme shown above at every time interval to these dynamic equations (see [23, 36] for more details).

Alternatively, we may form the statistically linearized state space using the WSLR technique:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_{f,k}\boldsymbol{x}_k + \boldsymbol{b}_{f,k} + \boldsymbol{G}_{f,k}\left(\boldsymbol{v}_k + \boldsymbol{\epsilon}_{f,k}\right) \tag{2.28}$$

$$\boldsymbol{z}_k = \boldsymbol{A}_{h,k}\boldsymbol{x}_k + \boldsymbol{b}_{h,k} + \boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k}, \tag{2.29}$$

where $\boldsymbol{A}_{f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{f,k}$, $\boldsymbol{b}_{h,k}$ are the statistical linearization parameters and $\boldsymbol{\epsilon}_{f,k}$, $\boldsymbol{\epsilon}_{h,k}$ are the linearization error with mean zero and covariance $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ and $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$. All the linearization parameters can be obtained by applying Equations (2.16), (2.22) and (2.24) iteratively at each time index k. $\boldsymbol{G}_{f,k}$ is an input matrix that controls the amount of noise to be added at each state vector. Deriving the Kalman filter using the pseudo-linearized state space shown in (2.28) and (2.29) also leads to SPKF (see [17] for details). However the advantage of this statistically linearized form is that it forms the nonlinear smoothing equations which will be discussed in the following sections.

2.3.2 Forward-Backward Statistical Linearized Sigma-Point Kalman Smoother (FBSL-SPKS)

The FBSL-SPKS consists of two independent filters: a SPKF operating on a nonlinear dynamics from time $k = 1$ to $k = N$ and an information filter, which operates backward in time starting from $k = N$ on the statistically linearized dynamics. The coefficients of the WSLR linearized dynamics are derived from the prior and posterior sigma points propagated through the nonlinear system during the forward SPKF operation. The estimates of the two filters are then statistically combined to obtain the smoothed estimates. Figure 2.2 demonstrates the outline of the FBSL-SPKS algorithm. In the following, we derive the detailed algorithms used in the forward filter, backward filter and in the smoothing technique.

Forward Filter

A standard SPKF is used as the forward filter. The forward SPKF operates on the nonlinear state space demonstrated in Equations (2.26) and (2.27) in order to estimate the state $\boldsymbol{x}_k$ and the estimation error covariance $\boldsymbol{P}_{\boldsymbol{x}_k}$ using the measurements $\boldsymbol{z}_{1:k}$. For the purpose of notational convenience, the forward state estimate is denoted here as $\hat{\boldsymbol{x}}_k$ instead of $\hat{\boldsymbol{x}}_k^{\mathrm{f}}$.

Without derivation, the pseudo code for the SPKF with WSLR is shown below:

- *Initialization*:

$$\hat{\boldsymbol{x}}_0 = \mathrm{E}\left[\boldsymbol{x}_0\right] \tag{2.30}$$

$$\boldsymbol{P}_{\boldsymbol{x}_0} = \mathrm{E}\left[\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)^T\right] \tag{2.31}$$

$$\begin{aligned}\hat{\boldsymbol{x}}_0^{\mathrm{a}} &= \mathrm{E}\left[\boldsymbol{x}_0^{\mathrm{a}}\right] \\ &= \begin{bmatrix} \hat{\boldsymbol{x}}_0^T & \hat{\boldsymbol{v}}_0^T & \hat{\boldsymbol{n}}_0^T \end{bmatrix}^T \end{aligned} \tag{2.32}$$

$$\begin{aligned}\boldsymbol{P}_{\boldsymbol{x}_0}^{\mathrm{a}} &= \left[\left(\boldsymbol{x}_0^{\mathrm{a}} - \hat{\boldsymbol{x}}_0^{\mathrm{a}}\right)\left(\boldsymbol{x}_0^{\mathrm{a}} - \hat{\boldsymbol{x}}_0^{\mathrm{a}}\right)^T\right] \\ &= \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{x}_0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{R}_0 \end{bmatrix}\end{aligned} \tag{2.33}$$

For $k = 0, 1, 2, \ldots, N$

- *Calculate sigma points*:

$$\boldsymbol{\chi}_k^{\mathrm{a}} = \begin{bmatrix} \hat{\boldsymbol{x}}_k^{\mathrm{a}} & \hat{\boldsymbol{x}}_k^{\mathrm{a}} + \sqrt{\left(\acute{M} + \lambda\right)\boldsymbol{P}_{\boldsymbol{x}_k}^{\mathrm{a}}} & \hat{\boldsymbol{x}}_k^{\mathrm{a}} - \sqrt{\left(\acute{M} + \lambda\right)\boldsymbol{P}_{\boldsymbol{x}_k}^{\mathrm{a}}} \end{bmatrix} \tag{2.34}$$

- *Time-update equations*:

$$\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} = f_k\left(\boldsymbol{\chi}_{i,k}^{\boldsymbol{x}}, \boldsymbol{\chi}_{i,k}^{\boldsymbol{v}}\right) \quad i = 0, 1, \ldots, 2\acute{M} \tag{2.35}$$

$$\hat{\boldsymbol{x}}_{k+1}^{-} = \sum_{i=0}^{2\acute{M}} w_i^{(m)} \boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} \tag{2.36}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}}^{-} = \sum_{i=0}^{2\acute{M}}\sum_{j=0}^{2\acute{M}} w_{ij}^{(c)} \left(\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-}\right)\left(\boldsymbol{\chi}_{j,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-}\right)^T \tag{2.37}$$

- *Weighted Statistical Linearization of f(.)*:

$$\boldsymbol{P}_{\boldsymbol{x}_k \boldsymbol{x}_{k+1}^{-}} = \sum_{i=0}^{2\acute{M}}\sum_{j=0}^{2\acute{M}} w_{ij}^{c} \left(\boldsymbol{\chi}_{j,k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_k\right)\left(\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-1}\right)^T \tag{2.38}$$

$$\boldsymbol{A}_{f,k} = \boldsymbol{P}_{\boldsymbol{x}_k \boldsymbol{x}_{k+1}^{-}}^T \boldsymbol{P}_{\boldsymbol{x}_k}^{-1} \tag{2.39}$$

$$\boldsymbol{b}_{f,k} = \hat{\boldsymbol{x}}_{k+1}^{-} - \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k \tag{2.40}$$

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} = \boldsymbol{P}_{\boldsymbol{x}_{k+1}}^{-} - \boldsymbol{A}_{f,k}\boldsymbol{P}_{\boldsymbol{x}_k}\boldsymbol{A}_{f,k}^T \tag{2.41}$$

- *Measurement-update equations*:

$$\boldsymbol{\gamma}_{i,k+1|k} = h_k\left(\boldsymbol{\chi}^{\boldsymbol{x}}_{i,k+1|k}, \boldsymbol{\chi}^{\boldsymbol{n}}_{i,k}\right) \quad i = 0, 1, \ldots, 2\acute{M} \tag{2.42}$$

$$\hat{\boldsymbol{z}}^-_{k+1} = \sum_{i=0}^{2\acute{M}} w_i^{(m)} \boldsymbol{\gamma}_{i,k+1|k} \tag{2.43}$$

$$\boldsymbol{P}_{\tilde{\boldsymbol{z}}_{k+1}} = \sum_{i=0}^{2\acute{M}}\sum_{j=0}^{2\acute{M}} w_{i,j}^{(c)} \left(\boldsymbol{\gamma}_{j,k+1|k} - \hat{\boldsymbol{z}}^-_{k+1}\right)\left(\boldsymbol{\gamma}_{i,k+1|k} - \hat{\boldsymbol{z}}^-_{k+1}\right)^T \tag{2.44}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} = \sum_{i=0}^{2\acute{M}}\sum_{j=0}^{2\acute{M}} w_{i,j}^{(c)} \left(\boldsymbol{\chi}^{\boldsymbol{x}}_{j,k+1|k} - \hat{\boldsymbol{x}}^-_{k+1}\right)\left(\boldsymbol{\gamma}_{i,k+1|k} - \hat{\boldsymbol{z}}^-_{k+1}\right)^T \tag{2.45}$$

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} \boldsymbol{P}^{-1}_{\tilde{\boldsymbol{z}}_{k+1}} \tag{2.46}$$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{K}_{k+1}\left(\boldsymbol{z}_{k+1} - \hat{\boldsymbol{z}}^-_{k+1}\right) \tag{2.47}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}} = \boldsymbol{P}^-_{\boldsymbol{x}_{k+1}} - \boldsymbol{K}_{k+1} \boldsymbol{P}^-_{\tilde{\boldsymbol{z}}_{k+1}} \boldsymbol{K}^T_{k+1} \tag{2.48}$$

- *Weighted Statistical Linearization of h(.)*:

$$\boldsymbol{A}_{h,k} = \boldsymbol{P}^T_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} \left(\boldsymbol{P}^-_{\boldsymbol{x}_{k+1}}\right)^{-1} \tag{2.49}$$

$$\boldsymbol{b}_{h,k} = \hat{\boldsymbol{z}}^-_{k+1} - \boldsymbol{A}_{h,k} \hat{\boldsymbol{x}}^-_{k+1} \tag{2.50}$$

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} = \boldsymbol{P}_{\tilde{\boldsymbol{z}}_{k+1}} - \boldsymbol{A}_{h,k} \boldsymbol{P}^-_{\boldsymbol{x}_{k+1}} \boldsymbol{A}^T_{h,k} \tag{2.51}$$

- *Parameters*:

$$\boldsymbol{x}^{\mathrm{a}} = \begin{bmatrix} \boldsymbol{x}^T & \boldsymbol{v}^T & \boldsymbol{n}^T \end{bmatrix}^T \tag{2.52}$$

$$\boldsymbol{\chi}^{\mathrm{a}} = \begin{bmatrix} (\boldsymbol{\chi}^{\boldsymbol{x}})^T & (\boldsymbol{\chi}^{\boldsymbol{v}})^T & (\boldsymbol{\chi}^{\boldsymbol{n}})^T \end{bmatrix}^T \tag{2.53}$$

$$\boldsymbol{P}^{\mathrm{a}}_{\boldsymbol{x}_k} = \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{x}_k} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Q}_k & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{R}_k \end{bmatrix}, \tag{2.54}$$

where λ is the composite scaling parameter which is given by:

$$\lambda = \alpha^2\left(\acute{M} + \kappa\right) - \acute{M}. \tag{2.55}$$

$w_i^{(c)}$ and $w_i^{(m)}$ are the scaler sigma-point weights and they are defined as:

$$w_0^{(c)} = \frac{\lambda}{\left(\acute{M}+\lambda\right)} + \left(1-\alpha^2+\beta\right) \quad , i=0 \tag{2.56}$$

$$w_0^{(m)} = \frac{\lambda}{\left(\acute{M}+\lambda\right)} \quad , i=0 \tag{2.57}$$

$$w_i^{(c)} = \frac{1}{2\left(\acute{M}+\lambda\right)} \quad , i=1,2,\ldots,2\acute{M} \tag{2.58}$$

$$w_i^{(m)} = \frac{1}{2\left(\acute{M}+\lambda\right)} \quad , i=1,2,\ldots,2\acute{M}, \tag{2.59}$$

where $0 \leq \alpha \leq 1$, $\beta = 2$ and $\kappa = 0$ are the SPKF parameters. The dimension of the state vector is M, $\acute{M}$ is the dimension of each augmented state, $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are the process noise and the observation noise covariances at time k. Here we assumed that the length of the observation sequence is N.

As demonstrated in the pseudo code, in addition to computing the state estimate $\hat{\boldsymbol{x}}_k$ and state estimation error covariance $\boldsymbol{P}_{\boldsymbol{x}_k}$, the forward SPKF also computes the statistical linearization parameters, $\boldsymbol{A}_{f,k}$, $\boldsymbol{b}_{f,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{h,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$, at each time k in order to form a pseudo-linearized dynamics. Equations (2.28) and (2.29) display the WSLR linearized state space model. The backward filter, which we derive next, operates on this pseudo-linearized dynamics to compute its estimates.

Backward Filter

An information filter is chosen as the backward filter, which starts from time $k = N$ and then proceeds backward in time to $k = 1$. The information filter is applied on the pseudo-linearized state space formulations to estimate the state $\boldsymbol{x}_k^{\mathrm{b}}$ and the estimation error covariance $\boldsymbol{P}_k^{\mathrm{b}}$ using the measurements $\boldsymbol{z}_j$ for $k \leq j \leq N$. Due to the backward state estimation, the pseudo-linearized state space demonstrated in Equations (2.28)-(2.29) is suitably modified as

$$\boldsymbol{x}_k = \boldsymbol{A}_{f,k}^{-1}\left[\boldsymbol{x}_{k+1} - \boldsymbol{b}_{f,k} - \boldsymbol{G}_{f,k}\left(\boldsymbol{v}_k + \boldsymbol{\epsilon}_{f,k}\right)\right] \tag{2.60}$$

$$\boldsymbol{z}_k = \boldsymbol{A}_{h,k}\boldsymbol{x}_k + \boldsymbol{b}_{h,k} + \boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k} \tag{2.61}$$

From Equations (2.28) and (2.29), it is evident that the structural form of the WSLR linearized dynamical system is different than the standard linear state space shown in [12] (the statistically linearized system has different linearization coefficients and also contains additional terms, such as $\boldsymbol{b}_{f,k}$, $\boldsymbol{b}_{h,k}$, $\boldsymbol{\epsilon}_{f,k}$ and $\boldsymbol{\epsilon}_{h,k}$). Hence the standard information filter formulations derived for the linear system cannot be directly applied in this case. In the following, we derive the time-update and measurement-update equations of the backward information filter applied on the statistically linearized dynamical system from first principles. The backward filter steps make use of Lemmas 2.7.1 to 2.7.3, which are derived in Appendix 2.7.

1. *Initializations*:

   The initial conditions for the smoother are

$$\boldsymbol{S}_{N+1} = \boldsymbol{0} \tag{2.62}$$

$$\hat{\boldsymbol{y}}_{N+1} = \boldsymbol{0}, \tag{2.63}$$

   where the information matrix $\boldsymbol{S}_k = \left(\boldsymbol{P}_k^{\mathrm{b}}\right)^{-1}$ is the inverse of the state error covariance and $\hat{\boldsymbol{y}}_k = \boldsymbol{S}_k \hat{\boldsymbol{x}}_k^{\mathrm{b}}$ is defined as the information state. The estimated state and estimation error covariance of the backward filter at time k are denoted as $\hat{\boldsymbol{x}}_k^{\mathrm{b}}$ and $\boldsymbol{P}_k^{\mathrm{b}}$ respectively. The superscript “b” signifies the operation of a backward filter. For notational convenience, the estimation error covariance of the backward filter at time k is denoted here as $\boldsymbol{P}_k^{\mathrm{b}}$ instead of $\boldsymbol{P}_{\boldsymbol{x}_k}^{\mathrm{b}}$.

2. *Time-update for the information matrix*:

   The equations of the state prediction and prediction error covariance can be formed using the WSLR linearized dynamics as follows

$$\hat{\boldsymbol{x}}_k^{\mathrm{b}-} = \boldsymbol{A}_{f,k}^{-1}\left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{b}} - \boldsymbol{b}_{f,k}\right) \tag{2.64}$$

$$\boldsymbol{P}_k^{\mathrm{b}-} = \mathrm{E}\left[\left(\boldsymbol{x}_k^{\mathrm{b}} - \hat{\boldsymbol{x}}_k^{\mathrm{b}-}\right)\left(\boldsymbol{x}_k^{\mathrm{b}} - \hat{\boldsymbol{x}}_k^{\mathrm{b}-}\right)^T\right]. \tag{2.65}$$

Denoting $\boldsymbol{x}_{k+1}^{\mathrm{b}} - \hat{\boldsymbol{x}}_{k+1}^{\mathrm{b}} = \boldsymbol{e}_{k+1}^{\mathrm{b}}$ and further simplifying (2.65) with the help of Equations (2.60) and (2.64),

$$\begin{aligned}\boldsymbol{P}_k^{\mathrm{b}-} &= \mathrm{E}\left(\boldsymbol{A}_{f,k}^{-1}\boldsymbol{e}_{k+1}^{\mathrm{b}} - \boldsymbol{A}_{f,k}^{-1}\left(\boldsymbol{G}_{f,k}\boldsymbol{v}_k + \boldsymbol{G}_{f,k}\boldsymbol{\epsilon}_{f,k}\right)\right) \cdot \\ &\quad \left(\boldsymbol{A}_{f,k}^{-1}\boldsymbol{e}_{k+1}^{\mathrm{b}} - \boldsymbol{A}_{f,k}^{-1}\left(\boldsymbol{G}_{f,k}\boldsymbol{v}_k + \boldsymbol{G}_{f,k}\boldsymbol{\epsilon}_{f,k}\right)\right)^T \qquad (2.66)\\ &= \boldsymbol{A}_{f,k}^{-1}\boldsymbol{P}_{k+1}^{\mathrm{b}}\boldsymbol{A}_{f,k}^{-T} + \boldsymbol{A}_{f,k}^{-1}\boldsymbol{G}_{f,k}\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right)\boldsymbol{G}_{f,k}^T\boldsymbol{A}_{f,k}^{-T}. \qquad (2.67)\end{aligned}$$

Equation (2.67) is derived by assuming $\mathrm{E}\left[\boldsymbol{\epsilon}_{f,k}\right] = \mathbf{0}$ and $\mathrm{E}\left[\boldsymbol{v}_k\right] = \mathbf{0}$. Applying the matrix inversion Lemma (Refer to Lemma 2.7.1) on (2.67) and later simplifying we obtain,

$$\begin{aligned}\boldsymbol{S}_k^- &= \boldsymbol{A}_{f,k}^T\boldsymbol{S}_{k+1}\boldsymbol{A}_{f,k} - \\ &\boldsymbol{A}_{f,k}^T\boldsymbol{S}_{k+1}\boldsymbol{G}_{f,k}\left[\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right)^{-1} + \boldsymbol{G}_{f,k}^T\boldsymbol{S}_{k+1}\boldsymbol{G}_{f,k}\right]^{-1}\boldsymbol{G}_{f,k}^T\boldsymbol{S}_{k+1}\boldsymbol{A}_{f,k}. \qquad (2.68)\end{aligned}$$

Lets define $\boldsymbol{K}_{\mathrm{G},k}$ as the backward gain matrix,

$$\boldsymbol{K}_{\mathrm{G},k} = \boldsymbol{S}_{k+1}\boldsymbol{G}_{f,k}\left[\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right)^{-1} + \boldsymbol{G}_{f,k}^T\boldsymbol{S}_{k+1}\boldsymbol{G}_{f,k}\right]^{-1}. \qquad (2.69)$$

Substitute the above expression for $\boldsymbol{K}_{\mathrm{G},k}$ into Equation (2.68) in order to obtain the final form of $\boldsymbol{S}_k^-$,

$$\boldsymbol{S}_k^- = \boldsymbol{A}_{f,k}^T\left(\boldsymbol{I} - \boldsymbol{K}_{\mathrm{G},k}\boldsymbol{G}_{f,k}^T\right)\boldsymbol{S}_{k+1}\boldsymbol{A}_{f,k}. \qquad (2.70)$$

Note the presence of linearization error term $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ in the R.H.S of (2.68). This term governs the severity of the nonlinearity into the covariance prediction and it does not appear in the standard information filter formulation for linear systems. [11,12]. The more severe the nonlinearity is over the uncertainty region of the state, the higher will be the linearization error covariance matrices.

3. *Time-update for the information state*:

From (2.64)

$$\begin{aligned}\hat{\boldsymbol{x}}_k^{\mathrm{b}-} &= \boldsymbol{A}_{f,k}^{-1}\left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{b}} - \boldsymbol{b}_{f,k}\right) \qquad (2.71)\\ \hat{\boldsymbol{y}}_k^- &= \boldsymbol{S}_k^-\hat{\boldsymbol{x}}_k^{\mathrm{b}-} \\ &= \boldsymbol{S}_k^-\boldsymbol{A}_{f,k}^{-1}\left(\boldsymbol{S}_{k+1}^{-1}\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{b}_{f,k}\right). \qquad (2.72)\end{aligned}$$

Now substituting $\boldsymbol{S}_k^-$ from Equation (2.70) into (2.72),

$$\hat{\boldsymbol{y}}_k^- = \boldsymbol{A}_{f,k}^T \left(\boldsymbol{I} - \boldsymbol{K}_{\mathrm{G},k}\boldsymbol{G}_{f,k}^T\right)\left(\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{S}_{k+1}\boldsymbol{b}_{f,k}\right). \tag{2.73}$$

Note that the correction term $\boldsymbol{S}_{k+1}\boldsymbol{b}_{f,k}$ is subtracted out from the previous information state in Equation (2.73). This term is not present in the time-update Equation of a standard information filter derived for linear systems.

4. *Measurement-update for the information matrix*:

   We start the derivation using the Lemma 2.7.2, which provides us an expression for $\boldsymbol{P}_k^{\mathrm{b}}$

$$\boldsymbol{P}_k^{\mathrm{b}} = \left(\boldsymbol{I} - \boldsymbol{K}_k^{\mathrm{b}}\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k^{\mathrm{b}\text{-}}\left(\boldsymbol{I} - \boldsymbol{K}_k^{\mathrm{b}}\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{K}_k^{\mathrm{b}}\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\left(\boldsymbol{K}_k^{\mathrm{b}}\right)^T, \tag{2.74}$$

where $\boldsymbol{K}_k^{\mathrm{b}}$ is the backward Kalman gain and is also derived in Lemma 2.7.2

$$\boldsymbol{K}_k^{\mathrm{b}} = \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\right)^{-1}. \tag{2.75}$$

Now assume

$$\boldsymbol{L}_k = \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}\text{-}}\boldsymbol{A}_{h,k}^T\right), \tag{2.76}$$

and simplify $\boldsymbol{P}_k^{\mathrm{b}}$

$$\begin{aligned}
\boldsymbol{P}_k^{\mathrm{b}} &= \left[\boldsymbol{I} - \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\right]\boldsymbol{P}_k^{\mathrm{b}-}\left[\boldsymbol{I} - \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\right]^T + \boldsymbol{K}_k\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\boldsymbol{K}_k^T \\
&= \boldsymbol{P}_k^{\mathrm{b}-} - \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-} - \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-} + \\
&\boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-} + \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-}.
\end{aligned} \tag{2.77}$$

The above Equation is further simplified by combining the last two terms in (2.77) and using $\boldsymbol{L}_k$ from (2.76),

$$\begin{aligned}
\boldsymbol{P}_k^{\mathrm{b}} &= \boldsymbol{P}_k^{\mathrm{b}-} - 2\boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-} + \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{L}_k\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-} \\
&= \boldsymbol{P}_k^{\mathrm{b}-} - \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_k^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-} \\
\left(\boldsymbol{P}_k^{\mathrm{b}}\right)^{-1} &= \left[\boldsymbol{P}_k^{\mathrm{b}-} - \boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-}\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{\mathrm{b}-}\right]^{-1}.
\end{aligned} \tag{2.78}$$

Using the matrix inversion Lemma on (2.78), the updated information error covariance $\boldsymbol{S}_k$ is formed as follows:

$$\begin{aligned}\left(\boldsymbol{P}_k^{\mathrm{b}}\right)^{-1} &= \left(\boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1} + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \boldsymbol{A}_{h,k} \\ \boldsymbol{S}_k &= \boldsymbol{S}_k^- + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \boldsymbol{A}_{h,k}. \end{aligned} \tag{2.79}$$

The presence of linearization error covariance $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ at the R.H.S of the Equation (2.79) is responsible for possessing a different update Equation than a standard linear information filter.

5. *Measurement-update for the information state*:

   The standard Kalman update Equation incorporating the latest measurement $\boldsymbol{z}_k$ is applied in order to compute the new information state $\hat{\boldsymbol{x}}_k^{\mathrm{b}}$,

$$\hat{\boldsymbol{x}}_k^{\mathrm{b}} = \hat{\boldsymbol{x}}_k^{\mathrm{b}-} + \boldsymbol{K}_k^{\mathrm{b}} \left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^{\mathrm{b}-} - \boldsymbol{b}_{h,k}\right). \tag{2.80}$$

   Replace $\boldsymbol{K}_k^{\mathrm{b}}$ with Equation (2.75), multiply both sides of the Equation with $\boldsymbol{S}_k$ and then substitute $\boldsymbol{S}_k$ from (2.79),

$$\begin{aligned}\hat{\boldsymbol{y}}_k &= \hat{\boldsymbol{y}}_k^- + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \hat{\boldsymbol{x}}_k^{\mathrm{b}-} + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^{\mathrm{b}-} - \boldsymbol{b}_{h,k}\right) \\ &= \hat{\boldsymbol{y}}_k^- + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \left(\boldsymbol{z}_k - \boldsymbol{b}_{h,k}\right), \end{aligned} \tag{2.81}$$

   where $\boldsymbol{z}_k$ is the true measurement at time k. The information state $\hat{\boldsymbol{y}}_k$ is formed at the L.H.S of Equation (2.81) by the multiplication of the backward state $\hat{\boldsymbol{x}}_k^{\mathrm{b}}$ and the corresponding information matrix $\boldsymbol{S}_k$,

$$\hat{\boldsymbol{y}}_k = \boldsymbol{S}_k \hat{\boldsymbol{x}}_k^{\mathrm{b}}. \tag{2.82}$$

   Note how the statistical linearization parameters $\boldsymbol{b}_{h,k}$ and $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ play a part in the update of the information state.

Smoothing

The SPKF is run in the forward direction on the interval $[0, N]$ to compute the forward posterior estimates $\hat{\boldsymbol{x}}_k$ and the estimation error covariances $\boldsymbol{P}_k$. The information filter is then run backwards to form the prior backward estimates $\hat{\boldsymbol{y}}_k^-$ and the corresponding estimation error covariances $\boldsymbol{S}_k^-$. Note that a posterior estimate at time k utilizes all observations up to time k, whereas a prior estimate is the state prediction at time k based on observations until time $k-1$. The smoother statistically combines the forward and backward estimates and their respective error covariances to determine the smoothed estimate $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$ and the error covariance $\boldsymbol{P}_k^{\mathrm{s}}$, where the superscript "s" signifies the smoothed output.

The smoothed estimate $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$ is obtained by linearly combining the forward and the backward estimates

$$\hat{\boldsymbol{x}}_k^{\mathrm{s}} = \boldsymbol{K}_k^{\mathrm{f}} \hat{\boldsymbol{x}}_k + \left(\boldsymbol{I} - \boldsymbol{K}_k^{\mathrm{f}}\right) \hat{\boldsymbol{x}}_k^{\mathrm{b}-}, \tag{2.83}$$

where $\boldsymbol{K}_k^{\mathrm{f}}$ is the linear combination coefficient to be determined. To make the smoothed estimate $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$ unbiased, $\hat{\boldsymbol{x}}_k$ and $\hat{\boldsymbol{x}}_k^{\mathrm{b}-}$ are combined with coefficients that sums equal to identity. To determine $\boldsymbol{K}_k^{\mathrm{f}}$, we first compute the estimation error covariance $\boldsymbol{P}_k^{\mathrm{s}}$ between the true $\boldsymbol{x}_k^{\mathrm{s}}$ and the estimate $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$ and then minimize the trace of $\boldsymbol{P}_k^{\mathrm{s}}$ with respect to $\boldsymbol{K}_k^{\mathrm{f}}$. The smoothed error covariance $\boldsymbol{P}_k^{\mathrm{s}}$ can be defined as

$$\boldsymbol{P}_k^{\mathrm{s}} = \mathrm{E}\left[\left(\boldsymbol{x}_k^{\mathrm{s}} - \hat{\boldsymbol{x}}_k^{\mathrm{s}}\right)\left(\boldsymbol{x}_k^{\mathrm{s}} - \hat{\boldsymbol{x}}_k^{\mathrm{s}}\right)^T\right]. \tag{2.84}$$

Replacing $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$ from (2.83) and then further simplifying we obtain

$$\boldsymbol{P}_k^{\mathrm{s}} = \boldsymbol{K}_k^{\mathrm{f}}\left(\boldsymbol{P}_k + \boldsymbol{P}_k^{\mathrm{b}-}\right)\left(\boldsymbol{K}_k^{\mathrm{f}}\right)^T + \boldsymbol{P}_k^{\mathrm{b}-} - \boldsymbol{K}_k^{\mathrm{f}}\boldsymbol{P}_k^{\mathrm{b}-} - \boldsymbol{P}_k^{\mathrm{b}-}\left(\boldsymbol{K}_k^{\mathrm{f}}\right)^T. \tag{2.85}$$

Differentiating $\boldsymbol{P}_k^{\mathrm{s}}$ with respect to $\boldsymbol{K}_k^{\mathrm{f}}$ and then setting equal to zero obtains

$$\frac{\partial \boldsymbol{P}_k^{\mathrm{s}}}{\partial \boldsymbol{K}_k^{\mathrm{f}}} = \boldsymbol{0} \tag{2.86}$$

$$= 2\left[\boldsymbol{K}_k^{\mathrm{f}}\left(\boldsymbol{P}_k + \boldsymbol{P}_k^{\mathrm{b}-}\right) - \boldsymbol{P}_k^{\mathrm{b}-}\right]. \tag{2.87}$$

Solving the above, we obtain an expression for $\boldsymbol{K}_k^{\mathrm{f}}$

$$\boldsymbol{K}_k^{\mathrm{f}} = \boldsymbol{P}_k^{\mathrm{b}-}\left(\boldsymbol{P}_k + \boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}. \tag{2.88}$$

- *Smoothed Error Covariance*:

  We can put the newly obtained value of the coefficient $\boldsymbol{K}_k^{\mathrm{f}}$ into Equation (2.85) to derive an Equation for $\boldsymbol{P}_k^{\mathrm{s}}$, which is a function of the forward and backward estimation error covariances (for details, please refer to [11])

$$\boldsymbol{P}_k^{\mathrm{s}} = \left[\boldsymbol{P}_k^{-1} + \left(\boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}\right]^{-1} \tag{2.89}$$

$$= \left[\boldsymbol{P}_k^{-1} + \boldsymbol{S}_k^{-}\right]^{-1}. \tag{2.90}$$

The matrix inversion Lemma can be applied to write the above in the following alternative form

$$\boldsymbol{P}_k^{\mathrm{s}} = \boldsymbol{P}_k - \boldsymbol{P}_k\boldsymbol{S}_k^{-}\left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^{-}\right)^{-1}\boldsymbol{P}_k. \tag{2.91}$$

As all covariance matrices are positive definite, Equation (2.91) suggests that the smoothed covariance $\boldsymbol{P}_k^{\mathrm{s}}$ is smaller than or equal to the forward SPKF covariance $\boldsymbol{P}_k$.

- *Smoothed Estimate*:
  An Equation for $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$ can be obtained by substituting the value of $\boldsymbol{K}_k^{\mathrm{f}}$ into Equation (2.83)

$$\hat{\boldsymbol{x}}_k^{\mathrm{s}} = \boldsymbol{P}_k^{\mathrm{b}-}\left(\boldsymbol{P}_k + \boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}\hat{\boldsymbol{x}}_k + \boldsymbol{P}_k\left(\boldsymbol{P}_k + \boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}\hat{\boldsymbol{x}}_k^{\mathrm{b}-}. \tag{2.92}$$

We can further simplify the above Equation using the matrix inversion Lemma

$$(\boldsymbol{A} + \boldsymbol{B})^{-1} = \boldsymbol{B}^{-1}\left(\boldsymbol{A}\boldsymbol{B}^{-1} + \boldsymbol{I}\right)^{-1}, \tag{2.93}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are the invertible square matrices in order to obtain the final Equation for the smoothed state $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$,

$$\hat{\boldsymbol{x}}_k^{\mathrm{s}} = \boldsymbol{P}_k^{\mathrm{s}}\left[\boldsymbol{P}_k^{-1}\hat{\boldsymbol{x}}_k + \left(\boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}\hat{\boldsymbol{x}}_k^{\mathrm{b}-}\right] \tag{2.94}$$

$$= \left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^{-}\right)^{-1}\hat{\boldsymbol{x}}_k + \boldsymbol{P}_k^{\mathrm{s}}\hat{\boldsymbol{y}}_k^{-}. \tag{2.95}$$

Note that both the smoothed state $\hat{\boldsymbol{x}}_k^{\mathrm{s}}$ and the smoothed error covariance $\boldsymbol{P}_k^{\mathrm{s}}$ are computed by incorporating all available measurements $\boldsymbol{z}_{0:N}$.

FBSL-SPKS algorithm

The final equations of FBSL-SPKS algorithm including the forward filter, the backward filter and the smoother are demonstrated below:

- *Forward filter*:

  The SPKF algorithm with WSLR as defined in Section 2.3.2 is used as a forward filter, which starts from $k = 1$ and proceeds to $k = N$. In addition to computing the estimates $\hat{\boldsymbol{x}}_k$ and the estimation error covariances $\boldsymbol{P}_k$, statistically linearized parameters $\boldsymbol{A}_{f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{f,k}$, $\boldsymbol{b}_{h,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ and $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ are also generated at each k by applying the WSLR technique on the nonlinear dynamics.

- *Backward filter*:

  The backward filter is an information filter, which operates backward in time from $k = N$ to $k = 1$ computing the information estimate $\hat{\boldsymbol{y}}_k$ and the information error covariance $\boldsymbol{S}_k$.
  For $k = N, N-1, N-2, \ldots, 1$

  – *Time update*:

$$\boldsymbol{S}_k^- = \boldsymbol{A}_{f,k}^T \left(\boldsymbol{I} - \boldsymbol{K}_k^{\mathrm{b}} \boldsymbol{G}_{f,k}^T\right) \boldsymbol{S}_{k+1} \boldsymbol{A}_{f,k} \tag{2.96}$$

$$\hat{\boldsymbol{y}}_k^- = \boldsymbol{A}_{f,k}^T \left(\boldsymbol{I} - \boldsymbol{K}_k^{\mathrm{b}} \boldsymbol{G}_{f,k}^T\right) \left(\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{S}_{k+1} \boldsymbol{b}_{f,k}\right) \tag{2.97}$$

  – *Measurement Update*:

$$\boldsymbol{S}_k = \boldsymbol{S}_k^- + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \boldsymbol{A}_{h,k} \tag{2.98}$$

$$\hat{\boldsymbol{y}}_k = \hat{\boldsymbol{y}}_k^- + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \left(\boldsymbol{z}_k - \boldsymbol{b}_{h,k}\right) \tag{2.99}$$

- *Smoother*:

  The smoother combines the estimates obtained from the forward and backward filters in order to compute the smoothed estimates.

$$\boldsymbol{P}_k^{\mathrm{S}} = \left[\boldsymbol{P}_k^{-1} + \boldsymbol{S}_k^{-}\right]^{-1} \tag{2.100}$$

$$\hat{\boldsymbol{x}}_k^{\mathrm{S}} = \left(\boldsymbol{I} + \boldsymbol{P}_k \boldsymbol{S}_k^{-}\right)^{-1} \hat{\boldsymbol{x}}_k + \boldsymbol{P}_k^{\mathrm{S}} \hat{\boldsymbol{y}}_k^{-} \tag{2.101}$$

Note, the smoothing step employs a minimum variance estimator, which linearly combines the forward and backward estimates with constant coefficient matrices in order to estimate the smoothed state at each k. We have found in practice that the state estimates generated by the FBSL-SPKS are clearly superior to those calculated by the standard SPKF. However the performance advantage of the FBSL-SPKS comes at the price of higher computational requirement. Each forward and backward filter employed by the FBSL-SPKS exhibits $O\left(NM^3\right)$ order of computation, where M is the state dimension and N is number of observations. In addition, the computational complexity of combining the forward and the backward estimates to obtain smoothed states is also $O\left(NM^3\right)$. Another disadvantage of the FBSL-SPKS is that it demands significantly higher memory as the entire forward estimation results, $\hat{\boldsymbol{x}}_k$ with dimension $M \times 1$ and forward error covariances $\boldsymbol{P}_k$ with dimension $M \times M$, need to be saved for the future smoothing step. The implementation of the FBSL-SPKS also assumes that an inverse of the statistical linearization parameter of the process model, $\boldsymbol{A}_{f,k}$, exists and can be easily computed. In the next section, we describe the RTSSL-SPKS, which is another variant of the fixed-interval SPKS.

2.3.3 Rauch-Tung-Striebel Statistical Linearized Sigma-Point Kalman Smoother (RTSSL-SPKS)

Unlike a forward-backward smoother, there is no separate forward and backward filter in the RTS smoother formulation. As the computation of full backward estimates is no longer needed, the RTS based algorithm is more computationally efficient and easier to implement than the forward-backward smoother. In the RTSSL-SPKS, a standard SPKF is used as a forward filter. A backward smoothing pass, which proceeds from $k = N$ to $k = 1$, is then applied on the forward estimation results to obtain the smoothed estimates. Figure 2.4 demonstrates the RTSSL-SPKS algorithm. The objective of the smoothing

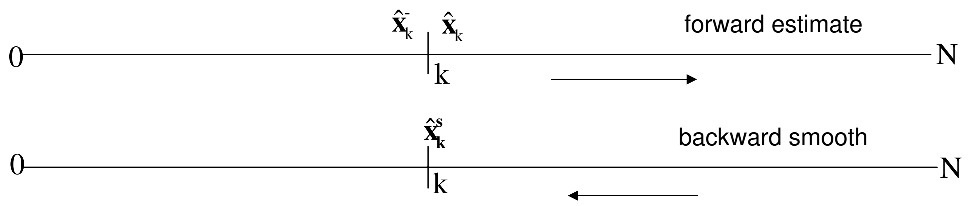


Figure 2.4: This schematic diagram explains the concept of the RTSSL-SPKS methodology. The forward SPKF generates a state estimate $\hat{\boldsymbol{x}}_k$ at each time k. The backward smoothing step applies a correction on the forward estimates to derive the smoothed estimates $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$.

pass is to employ a corrective measure on the forward estimation results based on the calculation of a smoothing gain $\boldsymbol{D}_k$. The computation of $\boldsymbol{D}_k$, which is a function of the prior and posterior estimation-error covariances, requires a linearized form of the state space. For a nonlinear system, EKF solves this problem using the Taylor series based first-order linearization. In this dissertation, we make use of the WSLR form of the nonlinear state space to perform the backward smoothing. As the statistically linearized state space defined in (2.28) and (2.29) is different from the standard linear state space used by the Kalman filter, we must derive the RTSSL-SPKS steps from first principles.

Forward Filter

The standard SPKF algorithm (refer to Section 2.3.2) is used as a forward filter, which generates state estimates $\hat{\boldsymbol{x}}_k$ and estimation error covariances $\boldsymbol{P}_k$ from time $k = 1$ to $k = N$ incorporating the measurements $\boldsymbol{z}_j$ for $1 \leq j \leq k$. In addition to computing the state estimates and error covariances, the forward pass also saves the intermediate results including the state prediction $\hat{\boldsymbol{x}}_k^-$ and the prediction error covariance $\boldsymbol{P}_k^-$ at each time k. Similar to the forward SPKF employed by the FBSL-SPKS algorithm, the forward filter in the RTSSL-SPKS algorithm also computes the statistical linearization parameters, $\boldsymbol{A}_{f,k}$, $\boldsymbol{b}_{f,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{h,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$, at each k in order to form the pseudo-linearized dynamics, which makes the backward smoothing step feasible.

Backward RTS Smoothing

Similar to the FBSL-SPKS smoother equations given in (2.90) and (2.95), the smoothed state $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ and the smoothed covariance $\boldsymbol{P}_k^{\mathrm{S}}$ for the RTSSL-SPKS algorithm are derived by making use of the statistically linearized state space. However as opposed to the FBSL-SPKS, which employs independent forward and backward filters for smoothing, the RTSSL-SPKS does not require to directly compute the backward estimates in order to perform smoothing. As the structural form of the WSLR linearized dynamics is different than the standard linear state space, the standard RTS equations derived for the linear system cannot be directly applied in this case. In the following, we derive the smoothed covariance and the smoothed state for the RTSSL-SPKS algorithm from first principles that involves a backward correction on the forward filtering result. The backward smoothing steps make use of Lemmas 2.7.3 to 2.7.11, which are derived in Appendix 2.7.

1. *Initializations*:

   The initial smoothed state and the estimation error covariance is equal to the forward filter estimate computed at the final time interval $k = N$. The initial conditions for the smoother can be shown as

$$\boldsymbol{P}_N^{\mathrm{S}} = \boldsymbol{P}_N \tag{2.102}$$

$$\hat{\boldsymbol{x}}_N^{\mathrm{S}} = \hat{\boldsymbol{x}}_N. \tag{2.103}$$

   The recursive RTS smoother is then run in the backward direction for $k = N, N-1, N-2, \ldots, 1$.

2. *RTS smoothing for estimation error covariance*:

   Below we derive the recursive Equation for the RTSSL-SPKS smoothed covariance $\boldsymbol{P}_k^{\mathrm{S}}$ at each k. Here we start with the smoothed error covariance given in Equation (2.90) and express the estimation error covariance of the backward filter $\boldsymbol{P}_k^{\mathrm{b}}$ in terms of the smoothed error covariance $\boldsymbol{P}_k^{\mathrm{S}}$ and the forward prediction error covariance $\boldsymbol{P}_k^{-}$. In that way we can avoid designing a separate backward Kalman filter and can estimate $\boldsymbol{P}_k^{\mathrm{s}}$ using only the forward filtering results. From Equation (2.67)

$$\begin{aligned}\left(\boldsymbol{P}_k+\boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1} &= \left[\boldsymbol{P}_k+\boldsymbol{A}_{f,k}^{-1}\boldsymbol{P}_{k+1}^{\mathrm{b}}\boldsymbol{A}_{f,k}^{-T}+\boldsymbol{A}_{f,k}^{-1}\boldsymbol{G}_{f,k}\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}+\boldsymbol{Q}_k\right)\boldsymbol{G}_{f,k}^{T}\boldsymbol{A}_{f,k}^{-T}\right]^{-1}\\ &= \boldsymbol{A}_{f,k}^{T}\left[\boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{f,k}^{T}+\boldsymbol{G}_{f,k}\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}+\boldsymbol{Q}_k\right)\boldsymbol{G}_{f,k}^{T}+\boldsymbol{P}_{k+1}^{\mathrm{b}}\right]^{-1}\boldsymbol{A}_{f,k}.\end{aligned} \tag{2.104}$$

Introducing

$$\boldsymbol{P}_{k+1}^{-}=\boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{f,k}^{T}+\boldsymbol{G}_{f,k}\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}+\boldsymbol{Q}_k\right)\boldsymbol{G}_{f,k}^{T}, \tag{2.105}$$

and also defining $\boldsymbol{P}_{k+1}^{\mathrm{b}}$ from Lemma 2.7.4,

$$\boldsymbol{P}_{k+1}^{\mathrm{b}}=\left[\left(\boldsymbol{P}_{k+1}^{\mathrm{s}}\right)^{-1}-\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\right]^{-1}, \tag{2.106}$$

we further simplify the Equation (2.104),

$$\begin{aligned}&\left(\boldsymbol{P}_k+\boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}=\boldsymbol{A}_{f,k}^{T}\left[\boldsymbol{P}_{k+1}^{-}+\left[\left(\boldsymbol{P}_{k+1}^{\mathrm{s}}\right)^{-1}-\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\right]^{-1}\right]^{-1}\boldsymbol{A}_{f,k}\\ &=\boldsymbol{A}_{f,k}^{T}\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\left[\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}+\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\left[\left(\boldsymbol{P}_{k+1}^{\mathrm{s}}\right)^{-1}-\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\right]^{-1}\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\right]^{-1}\cdot\\ &\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\boldsymbol{A}_{f,k}.\end{aligned} \tag{2.107}$$

Now by applying the matrix inversion Lemma, Equation (2.107) is expressed as follows:

$$\left(\boldsymbol{P}_k+\boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}=\boldsymbol{A}_{f,k}^{T}\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\left(\boldsymbol{P}_{k+1}^{-}-\boldsymbol{P}_{k+1}^{\mathrm{s}}\right)\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}\boldsymbol{A}_{f,k}. \tag{2.108}$$

Using the above Equation, we substitute $\left(\boldsymbol{P}_k+\boldsymbol{P}_k^{\mathrm{b}-}\right)^{-1}$ from the Equation (2.91) in order to obtain the smoothed covariance $\boldsymbol{P}_k^{\mathrm{s}}$,

$$\boldsymbol{P}_k^{\mathrm{s}}=\boldsymbol{P}_k-\boldsymbol{D}_k\left(\boldsymbol{P}_{k+1}^{-}-\boldsymbol{P}_{k+1}^{\mathrm{s}}\right)\boldsymbol{D}_k^{T}, \tag{2.109}$$

where we have assumed that $\boldsymbol{D}_k$=backward smoothing gain, which is defined as

$$\boldsymbol{D}_k=\boldsymbol{P}_k\boldsymbol{A}_{f,k}^{T}\left(\boldsymbol{P}_{k+1}^{-}\right)^{-1}. \tag{2.110}$$

Note that the computation of the backward gain $\boldsymbol{D}_k$ depends on the statistical linearization parameter $\boldsymbol{A}_{f,k}$, forward estimation error covariance $\boldsymbol{P}_k$ and the future prediction error covariance $\boldsymbol{P}_{k+1}^{-}$.

3. *RTS smoothing for state estimate*:

   Now we concentrate on deriving the recursive Equation for estimating the smoothed state $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ at each k. Like the derivation of the smoothed covariance, the estimation of the smoothed state is based on expressing the FBSL-SPKS backward filter equations in terms of the previously computed smoothed results and the forward filter estimates. We start the derivation using Equation (2.64). Multiplying both sides of Equation (2.64) with $\boldsymbol{S}_k^-$,

$$\hat{\boldsymbol{y}}_k^- = \boldsymbol{S}_k^- \boldsymbol{A}_{f,k}^{-1} \left(\boldsymbol{P}_{k+1}^{\mathrm{b}} \hat{\boldsymbol{y}}_{k+1} - \boldsymbol{b}_{f,k} \right). \tag{2.111}$$

By using Lemma 2.7.9,

$$\boldsymbol{S}_k^- = \boldsymbol{A}_{f,k}^T \left[\acute{\boldsymbol{Q}}_k^{-1} - \acute{\boldsymbol{Q}}_k^{-1} \left(\boldsymbol{S}_{k+1} + \acute{\boldsymbol{Q}}_k^{-1} \right)^{-1} \acute{\boldsymbol{Q}}_k^{-1} \right] \boldsymbol{A}_{f,k}, \tag{2.112}$$

where $\acute{\boldsymbol{Q}}_k$ is defined as

$$\acute{\boldsymbol{Q}}_k = \boldsymbol{G}_{f,k} \left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k \right) \boldsymbol{G}_{f,k}^T, \tag{2.113}$$

we substitute $\boldsymbol{S}_k^-$ and then further simply the Equation (2.111)

$$\begin{aligned}
\hat{\boldsymbol{y}}_k^- &= \boldsymbol{A}_{f,k}^T \acute{\boldsymbol{Q}}_k^{-1} \left(\boldsymbol{S}_{k+1} + \acute{\boldsymbol{Q}}_k^{-1} \right)^{-1} \left(\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{S}_{k+1} \boldsymbol{b}_{f,k} \right) \\
&= \boldsymbol{A}_{f,k}^T \left(\boldsymbol{I} + \boldsymbol{S}_{k+1} \acute{\boldsymbol{Q}}_k \right)^{-1} \left(\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{S}_{k+1} \boldsymbol{b}_{f,k} \right) \\
\hat{\boldsymbol{y}}_{k+1} &= \left(\boldsymbol{I} + \boldsymbol{S}_{k+1} \acute{\boldsymbol{Q}}_k \right) \boldsymbol{A}_{f,k}^{-T} \hat{\boldsymbol{y}}_k^- + \boldsymbol{S}_{k+1} \boldsymbol{b}_{f,k}.
\end{aligned} \tag{2.114}$$

Multiplying both sides by $\boldsymbol{A}_{f,k}^{-1} \boldsymbol{P}_{k+1}^{\mathrm{b}}$ and then applying Lemma 2.7.5,

$$\boldsymbol{A}_{f,k}^{-1} \acute{\boldsymbol{Q}}_k \boldsymbol{A}_{f,k}^{-T} = \boldsymbol{A}_{f,k}^{-1} \boldsymbol{P}_{k+1}^- \boldsymbol{A}_{f,k}^{-T} - \boldsymbol{P}_k, \tag{2.115}$$

we obtain,

$$\begin{aligned}
\boldsymbol{A}_{f,k}^{-1} \boldsymbol{P}_{k+1}^{\mathrm{b}} \hat{\boldsymbol{y}}_{k+1} &= \boldsymbol{A}_{f,k}^{-1} \boldsymbol{P}_{k+1}^{\mathrm{b}} \boldsymbol{A}_{f,k}^{-T} \hat{\boldsymbol{y}}_k^- + \boldsymbol{A}_{f,k}^{-1} \boldsymbol{P}_{k+1}^- \boldsymbol{A}_{f,k}^{-T} \hat{\boldsymbol{y}}_k^- - \boldsymbol{P}_k \hat{\boldsymbol{y}}_k^- + \boldsymbol{A}_{f,k}^{-1} \boldsymbol{b}_{f,k} \\
\boldsymbol{P}_{k+1}^{\mathrm{b}} \hat{\boldsymbol{y}}_{k+1} &= \left[\left(\boldsymbol{P}_{k+1}^{\mathrm{b}} + \boldsymbol{P}_{k+1}^- \right) \boldsymbol{A}_{f,k}^{-T} - \boldsymbol{A}_{f,k} \boldsymbol{P}_k \right] \hat{\boldsymbol{y}}_k^- + \boldsymbol{b}_{f,k}.
\end{aligned} \tag{2.116}$$

We use Lemma 2.7.6,

$$\boldsymbol{P}_{k+1}^{\mathrm{b}} = \left(\boldsymbol{P}_{k+1}^- + \boldsymbol{P}_{k+1}^{\mathrm{b}} \right) \boldsymbol{S}_{k+1}^- \boldsymbol{P}_{k+1}^{\mathrm{s}}, \tag{2.117}$$

and Lemma 2.7.7,

$$\boldsymbol{P}_{k+1}^{-} + \boldsymbol{P}_{k+1}^{\mathrm{b}} = \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_{k} + \boldsymbol{P}_{k}^{\mathrm{b}-}\right)\boldsymbol{A}_{f,k}^{T}, \tag{2.118}$$

to replace $\boldsymbol{P}_{k+1}^{\mathrm{b}}$ and $\boldsymbol{P}_{k+1}^{-} + \boldsymbol{P}_{k+1}^{\mathrm{b}}$ from the L.H.S and the R.H.S of the Equation (2.116),

$$\boldsymbol{A}_{f,k}\left(\boldsymbol{P}_{k} + \boldsymbol{P}_{k}^{\mathrm{b}-}\right)\boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\boldsymbol{P}_{k+1}^{\mathrm{s}}\hat{\boldsymbol{y}}_{k+1} = \left[\boldsymbol{A}_{f,k}\left(\boldsymbol{P}_{k} + \boldsymbol{P}_{k}^{\mathrm{b}-}\right) - \boldsymbol{A}_{f,k}\boldsymbol{P}_{k}\right]\hat{\boldsymbol{y}}_{k}^{-} + \boldsymbol{b}_{f,k}. \tag{2.119}$$

Now multiplying both sides of the Equation (2.119) with $\left(\boldsymbol{P}_{k} + \boldsymbol{P}_{k}^{\mathrm{b}-}\right)^{-1}\boldsymbol{A}_{f,k}^{-1}$ and then substitute $\left(\boldsymbol{P}_{k} + \boldsymbol{P}_{k}^{\mathrm{b}-}\right)^{-1}$ from the Equation (2.108)

$$\begin{aligned}\boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\boldsymbol{P}_{k+1}^{\mathrm{s}}\hat{\boldsymbol{y}}_{k+1} = \hat{\boldsymbol{y}}_{k}^{-} - \boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\left(\boldsymbol{P}_{k+1}^{-} - \boldsymbol{P}_{k+1}^{\mathrm{s}}\right)\boldsymbol{S}_{k+1}^{-}\boldsymbol{A}_{f,k}\boldsymbol{P}_{k}\hat{\boldsymbol{y}}_{k}^{-} + \\ \boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\left(\boldsymbol{P}_{k+1}^{-} - \boldsymbol{P}_{k+1}^{\mathrm{s}}\right)\boldsymbol{S}_{k+1}^{-}\boldsymbol{b}_{f,k}.\end{aligned} \tag{2.120}$$

Equation (2.108) can be further modified as

$$\boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\left(\boldsymbol{P}_{k+1}^{\mathrm{s}} - \boldsymbol{P}_{k+1}^{-}\right)\boldsymbol{S}_{k+1}^{-}\hat{\boldsymbol{x}}_{k+1}^{-} = -\left(\boldsymbol{P}_{k} + \boldsymbol{P}_{k}^{\mathrm{b}-}\right)^{-1}\boldsymbol{A}_{f,k}^{-1}\hat{\boldsymbol{x}}_{k+1}^{-}. \tag{2.121}$$

Summing Equations (2.120) and (2.121) and later using Lemma 2.7.8

$$\hat{\boldsymbol{x}}_{k+1}^{\mathrm{s}} = \boldsymbol{P}_{k+1}^{\mathrm{s}}\boldsymbol{S}_{k+1}\hat{\boldsymbol{x}}_{k+1}^{-} - \boldsymbol{P}_{k+1}^{\mathrm{s}}\boldsymbol{A}_{h,k+1}^{T}\acute{\boldsymbol{R}}_{k+1}^{-1}\boldsymbol{A}_{h,k+1}\hat{\boldsymbol{x}}_{k+1}^{-} + \boldsymbol{P}_{k+1}^{\mathrm{s}}\hat{\boldsymbol{y}}_{k+1}, \tag{2.122}$$

to substitute $\boldsymbol{P}_{k+1}^{\mathrm{s}}\hat{\boldsymbol{y}}_{k+1}$,

$$\begin{aligned}&\boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{s}} - \hat{\boldsymbol{x}}_{k+1}^{-}\right) = \hat{\boldsymbol{y}}_{k}^{-} - \boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\left(\boldsymbol{P}_{k+1}^{-} - \boldsymbol{P}_{k+1}^{\mathrm{s}}\right)\boldsymbol{S}_{k+1}^{-}\boldsymbol{A}_{f,k}\boldsymbol{P}_{k}\hat{\boldsymbol{y}}_{k}^{-} + \\ &\left[-\left(\boldsymbol{P}_{k} + \boldsymbol{P}_{k}^{\mathrm{b}-}\right)^{-1}\boldsymbol{A}_{f,k}^{-1} + \boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\boldsymbol{P}_{k+1}^{\mathrm{s}}\boldsymbol{S}_{k+1}\right]\hat{\boldsymbol{x}}_{k+1}^{-} + \\ &\left[-\boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\boldsymbol{P}_{k+1}^{\mathrm{s}}\boldsymbol{A}_{h,k+1}^{T}\acute{\boldsymbol{R}}_{k+1}^{-1}\boldsymbol{A}_{h,k+1} - \boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\boldsymbol{P}_{k+1}^{\mathrm{s}}\boldsymbol{S}_{k+1}^{-}\right]\hat{\boldsymbol{x}}_{k+1}^{-} + \\ &\boldsymbol{A}_{f,k}^{T}\boldsymbol{S}_{k+1}^{-}\left(\boldsymbol{P}_{k+1}^{-} - \boldsymbol{P}_{k+1}^{\mathrm{s}}\right)\boldsymbol{S}_{k+1}^{-}\boldsymbol{b}_{f,k},\end{aligned} \tag{2.123}$$

where $\acute{\boldsymbol{R}}_{k+1}$ expands to

$$\acute{\boldsymbol{R}}_{k+1} = \boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}. \tag{2.124}$$

Now by applying Lemma 2.7.10,

$$\hat{\boldsymbol{x}}_k^{\mathrm{S}} - \hat{\boldsymbol{x}}_k = \boldsymbol{P}_k^{\mathrm{S}} \hat{\boldsymbol{y}}_k^- - \boldsymbol{P}_k \boldsymbol{S}_k^- \left(\boldsymbol{I} + \boldsymbol{P}_k \boldsymbol{S}_k^- \right)^{-1} \hat{\boldsymbol{x}}_k, \tag{2.125}$$

and Equations (2.123), (2.108), (2.64) and (2.79), we simplify the Equation (2.123) as follows:

$$\begin{aligned} \boldsymbol{A}_{f,k}^T \boldsymbol{S}_{k+1}^- \left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}} - \hat{\boldsymbol{x}}_{k+1}^- \right) = \hat{\boldsymbol{y}}_k^- - \boldsymbol{A}_{f,k}^T \boldsymbol{S}_{k+1}^- \left(\boldsymbol{P}_{k+1}^- - \boldsymbol{P}_{k+1}^{\mathrm{S}} \right) \boldsymbol{S}_{k+1}^- \boldsymbol{A}_{f,k} \boldsymbol{P}_k \hat{\boldsymbol{y}}_k^- - \\ \boldsymbol{S}_k^- \left(\boldsymbol{I} + \boldsymbol{P}_k \boldsymbol{S}_k^- \right)^{-1} \hat{\boldsymbol{x}}_k. \end{aligned} \tag{2.126}$$

We can form the smoothed covariance $\boldsymbol{P}_{k+1}^{\mathrm{S}}$ at time $k+1$ using the Equation (2.109)

$$\boldsymbol{P}_{k+1}^{\mathrm{S}} = \boldsymbol{P}_{k+1} - \boldsymbol{D}_{k+1} \left(\boldsymbol{P}_{k+2}^- - \boldsymbol{P}_{k+2}^{\mathrm{S}} \right) \boldsymbol{D}_{k+1}^T. \tag{2.127}$$

Substituting $\boldsymbol{P}_{k+1}^{\mathrm{S}}$ from above into the Equation (2.126)

$$\begin{aligned} & \boldsymbol{A}_{f,k}^T \boldsymbol{S}_{k+1}^- \left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}} - \hat{\boldsymbol{x}}_{k+1}^- \right) = \hat{\boldsymbol{y}}_k^- - \\ & \boldsymbol{A}_{f,k}^T \boldsymbol{S}_{k+1}^- \left(\boldsymbol{P}_{k+1}^- - \boldsymbol{P}_{k+1} + \boldsymbol{D}_{k+1} \left(\boldsymbol{P}_{k+2}^- - \boldsymbol{P}_{k+2}^{\mathrm{S}} \right) \boldsymbol{D}_{k+1}^T \right) \boldsymbol{D}_k^T \hat{\boldsymbol{y}}_k^- - \\ & \boldsymbol{S}_k^- \left(\boldsymbol{I} + \boldsymbol{P}_k \boldsymbol{S}_k^- \right)^{-1} \hat{\boldsymbol{x}}_k. \end{aligned} \tag{2.128}$$

Multiplying $\boldsymbol{P}_k$ on both sides of the Equation (2.128), we obtain

$$\begin{aligned} \boldsymbol{D}_k \left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}} - \hat{\boldsymbol{x}}_{k+1}^- \right) = \boldsymbol{P}_k \hat{\boldsymbol{y}}_k^- - \boldsymbol{D}_k \left(\boldsymbol{P}_{k+1}^- - \boldsymbol{P}_{k+1} + \boldsymbol{D}_{k+1} \left(\boldsymbol{P}_{k+2}^- - \boldsymbol{P}_{k+2}^{\mathrm{S}} \right) \boldsymbol{D}_{k+1}^T \right) \boldsymbol{D}_k^T \hat{\boldsymbol{y}}_k^- \\ - \boldsymbol{P}_k \boldsymbol{S}_k^- \left(\boldsymbol{I} + \boldsymbol{P}_k \boldsymbol{S}_k^- \right)^{-1} \hat{\boldsymbol{x}}_k \end{aligned} \tag{2.129}$$

The above Equation can further be simplified as

$$\begin{aligned} & \boldsymbol{D}_k \left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}} - \hat{\boldsymbol{x}}_{k+1}^- \right) = \\ & \left(\boldsymbol{P}_k - \boldsymbol{D}_k \left(\boldsymbol{P}_{k+1}^- - \boldsymbol{P}_{k+1} + \boldsymbol{D}_{k+1} \boldsymbol{P}_{k+2}^- \boldsymbol{D}_{k+1}^T - \boldsymbol{D}_{k+1} \boldsymbol{P}_{k+2}^{\mathrm{S}} \boldsymbol{D}_{k+1}^T \right) \boldsymbol{D}_k^T \right) \hat{\boldsymbol{y}}_k^- - \\ & \boldsymbol{P}_k \boldsymbol{S}_k^- \left(\boldsymbol{I} + \boldsymbol{P}_k \boldsymbol{S}_k^- \right)^{-1} \hat{\boldsymbol{x}}_k. \end{aligned} \tag{2.130}$$

where the backward smoothing gain $\boldsymbol{D}_k$ is defined in Equation (2.110). From (2.109),

$$\begin{aligned} \boldsymbol{P}_{k+1}^{\mathrm{S}} &= \boldsymbol{P}_{k+1} - \boldsymbol{D}_{k+1} \left(\boldsymbol{P}_{k+2}^- - \boldsymbol{P}_{k+2}^{\mathrm{S}} \right) \boldsymbol{D}_{k+1}^T \\ \boldsymbol{D}_{k+1} \boldsymbol{P}_{k+2}^{\mathrm{S}} \boldsymbol{D}_{k+1}^T &= \boldsymbol{P}_{k+1}^{\mathrm{S}} - \boldsymbol{P}_{k+1} + \boldsymbol{D}_{k+1} \boldsymbol{P}_{k+2}^- \boldsymbol{D}_{k+1}^T. \end{aligned} \tag{2.131}$$

Substitute $\boldsymbol{D}_{k+1}\boldsymbol{P}^{\mathrm{S}}_{k+2}\boldsymbol{D}^{T}_{k+1}$ from the above Equation into the Equation (2.130),

$$\begin{aligned}\boldsymbol{D}_k\left(\hat{\boldsymbol{x}}^{\mathrm{S}}_{k+1}-\hat{\boldsymbol{x}}^{-}_{k+1}\right) &= \left(\boldsymbol{P}_k-\boldsymbol{D}_k\left(\boldsymbol{P}^{-}_{k+1}-\boldsymbol{P}^{\mathrm{S}}_{k+1}\right)\boldsymbol{D}^{T}_k\right)\hat{\boldsymbol{y}}^{-}_k-\boldsymbol{P}_k\boldsymbol{S}^{-}_k\left(\boldsymbol{I}+\boldsymbol{P}_k\boldsymbol{S}^{-}_k\right)^{-1}\hat{\boldsymbol{x}}_k\\ &= \boldsymbol{P}^{\mathrm{S}}_k\hat{\boldsymbol{y}}^{-}_k-\boldsymbol{P}_k\boldsymbol{S}^{-}_k\left(\boldsymbol{I}+\boldsymbol{P}_k\boldsymbol{S}^{-}_k\right)^{-1}\hat{\boldsymbol{x}}_k.\ \text{[from (2.109)]}\end{aligned} \tag{2.132}$$

Finally, define Lemma 2.7.11,

$$\hat{\boldsymbol{x}}^{\mathrm{S}}_k-\hat{\boldsymbol{x}}_k=\boldsymbol{P}^{\mathrm{S}}_k\hat{\boldsymbol{y}}^{-}_k-\boldsymbol{P}_k\boldsymbol{S}^{-}_k\left(\boldsymbol{I}+\boldsymbol{P}_k\boldsymbol{S}^{-}_k\right)^{-1}\hat{\boldsymbol{x}}_k, \tag{2.133}$$

and compare the above Equation with (2.132) in order to obtain the smoothed state $\hat{\boldsymbol{x}}^{\mathrm{S}}_k$,

$$\begin{aligned}\boldsymbol{D}_k\left(\hat{\boldsymbol{x}}^{\mathrm{S}}_{k+1}-\hat{\boldsymbol{x}}^{-}_{k+1}\right) &= \hat{\boldsymbol{x}}^{\mathrm{S}}_k-\hat{\boldsymbol{x}}_k\\ \hat{\boldsymbol{x}}^{\mathrm{S}}_k &= \hat{\boldsymbol{x}}_k+\boldsymbol{D}_k\left(\hat{\boldsymbol{x}}^{\mathrm{S}}_{k+1}-\hat{\boldsymbol{x}}^{-}_{k+1}\right).\end{aligned} \tag{2.134}$$

As shown, the smoothed state estimate $\hat{\boldsymbol{x}}^{\mathrm{S}}_k$ is equal to the linear combination with coefficient $\boldsymbol{D}_k$ of the following:

(a) The state estimate $\hat{\boldsymbol{x}}_k$ computed during the forward SPKF pass.

(b) The difference between the smoothed state $\hat{\boldsymbol{x}}^{\mathrm{S}}_{k+1}$ estimated at time $k+1$ during the backward smoothing pass and the state prediction $\hat{\boldsymbol{x}}^{-}_{k+1}$ computed by the forward SPKF estimator.

RTSSL-SPKS algorithm

The final equations of the RTSSL-SPKS algorithm including the forward filter and the backward smoother are described below:

- *Forward filter*:

  The SPKF algorithm with WSLR as defined in Section 2.3.2 is applied for $k = 1, 2, \ldots, N$. In addition to computing the estimates $\hat{\boldsymbol{x}}_k$ and the error covariances $\boldsymbol{P}_k$, the forward pass also saves the intermediate results including the state prediction $\hat{\boldsymbol{x}}_k^-$ and the prediction error covariance $\boldsymbol{P}_k^-$ at each time k. The statistically linearized parameters $\boldsymbol{A}_{f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{f,k}$, $\boldsymbol{b}_{h,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ and $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ are also generated at each SPKF recursion using the WSLR technique on the nonlinear state space.

- *Backward smoothing*:

  The backward smoothing recursion can be used to compute the smoothed estimates at all k, starting from $k = N$ and proceeding backwards to the initial time $k = 1$.

  - *Error covariance smoothing*:

$$\boldsymbol{D}_k = \boldsymbol{P}_k \boldsymbol{A}_{f,k}^T \left(\boldsymbol{P}_{k+1}^-\right)^{-1} \tag{2.135}$$

$$\boldsymbol{P}_k^{\mathrm{S}} = \boldsymbol{P}_k - \boldsymbol{D}_k \left(\boldsymbol{P}_{k+1}^- - \boldsymbol{P}_{k+1}^{\mathrm{S}}\right) \boldsymbol{D}_k^T \tag{2.136}$$

  - *State estimate smoothing*:

$$\hat{\boldsymbol{x}}_k^{\mathrm{S}} = \hat{\boldsymbol{x}}_k + \boldsymbol{D}_k \left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}} - \hat{\boldsymbol{x}}_{k+1}^-\right) \tag{2.137}$$

Note, the smoothed error covariance $\boldsymbol{P}_k^{\mathrm{S}}$ is not necessary for the estimation of the smoothed state $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ and should only be computed if it is of sufficient interest. We have found in the experimental results that the RTSSL-SPKS and FBSL-SPKS perform equally well with negligible difference in the estimation accuracy. However, the RTSSL-SPKS algorithm is more computationally efficient than the FBSL-SPKS. The computational saving is due to RTSSL-SPKS's ability to combine the backward filtering and smoothing step to a single backward smoothing pass. Both the forward filter pass and the backward smoothing pass of the RTSSL-SPKS algorithm performs an order of $O\left(NM^3\right)$ computations in order to generate the smoothed estimates. Another benefit of implementing the RTSSL-SPKS is that it does not require the inverse of $\boldsymbol{A}_{f,k}$ in order to compute the estimates, which can be particularly significant if $\boldsymbol{A}_{f,k}$ is singular or the inverse of $\boldsymbol{A}_{f,k}$ leads to numerical problems. On the downside, the RTSSL-SPKS needs more memory than the FBSL-SPKS as it needs to save all the forward filter state predictions $\hat{\boldsymbol{x}}_k^-$ and the prediction error covariances $\boldsymbol{P}_k^-$ in addition to the state estimates $\hat{\boldsymbol{x}}_k$ and the estimation error covariances $\boldsymbol{P}_k$. The high memory requirement for the RTSSL-SPKS can pose significant burden on a system, particularly for the case of large M and/or N. To address the problem of offline estimation and higher storage requirement characterized by the fixed-interval smoothers, we develop the nonlinear fixed-lag smoothers using the sigma-point Kalman filtering methodology, which is described next.

2.4 Fixed-Lag Sigma-Point Kalman Smoothers (FL-SPKS)

Here we derive the formulations for fixed-lag smoothers operating on a nonlinear state space. For a fixed-lag smoother, we estimate the state $\boldsymbol{x}_{k-L}$ given observations up to time k, $\boldsymbol{z}_{1:k}$. In other words, there exists a fixed time lag L between the current observation and the smoothed state. The fixed-lag method provides sequential estimates of the states delayed by L measurements, and is thus more appropriate to an on-line implementation than the fixed-interval approach.

2.4.1 State-Augmented Sigma-Point Kalman Smoother (Aug-SPKS)

In the Aug-SPKS, the objective is to estimate the current state $\boldsymbol{x}_k$ of dimension M using all the past, present and L future measurements, where L is the fixed lag. Alternatively, this may be viewed as estimating the lagged state $\boldsymbol{x}_{k-L}$ given all measurements up to the current time k. The Aug-SPKS is specified by simply defining a new augmenting state space,

$$\tilde{\boldsymbol{x}}_{k+1} = \begin{bmatrix} \boldsymbol{x}_{k+1} & \boldsymbol{x}_k & \cdots & \boldsymbol{x}_{k-L+1} \end{bmatrix}^T_{ML\times 1} \tag{2.138}$$

$$= \begin{bmatrix} f_k\left(\boldsymbol{x}_k\right) & \\ \begin{bmatrix} \boldsymbol{I}_{M\times M} & 0 & \cdots & 0 \\ 0 & \boldsymbol{I}_{M\times M} & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{I}_{M\times M} & 0 \end{bmatrix}_{M(L-1)\times ML} & \tilde{\boldsymbol{x}}_k \end{bmatrix}_{ML\times 1} + \begin{bmatrix} \boldsymbol{I}_{M\times M} \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{ML\times M} \boldsymbol{v}_k \tag{2.139}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k\right) + \boldsymbol{n}_k, \tag{2.140}$$

where the augmenting state $\tilde{\boldsymbol{x}}_k$ is defined as

$$\tilde{\boldsymbol{x}}_k = \begin{bmatrix} \boldsymbol{x}_k & \boldsymbol{x}_{k-1} & \cdots & \boldsymbol{x}_{k-L} \end{bmatrix}^T. \tag{2.141}$$

The process noise $\boldsymbol{v}_k$ shown in Equation (2.139) is of dimension $M \times 1$. The standard SPKF recursions shown in Section 2.3.2 are applied directly to the augmented system. The fixed-lag estimate of the last element of the augmented state vector $\boldsymbol{x}_{k-L}$ will be equal to $\hat{\boldsymbol{x}}_{k-L}$ given measurements up to time k. As can be seen, the Aug-SPKS algorithm is simple and straightforward to implement but the increased state dimension increases the overall computational complexity of the algorithm. As the computational complexity of the SPKF varies to the cubic rate of the augmented state dimension, the computational order of the Aug-SPKS to generate each smoothed estimate becomes proportional to $O\left(M^3L^3\right)$. Note that the computational load of the Aug-SPKS further increases to $O\left(NM^3L^3\right)$ in order to generate N smoothed estimates. The increased computational demand may pose a challenge in implementing this smoother in practice, particularly in those cases involving a large state dimension M. In addition to the higher computational demand, the Aug-SPKS also requires larger memory at each k due to the generation of an augmented state with length ML and an estimation-error covariance of dimension $ML \times ML$. Still this is an improvement compared to the FI-SPKS, which requires the entire forward estimation

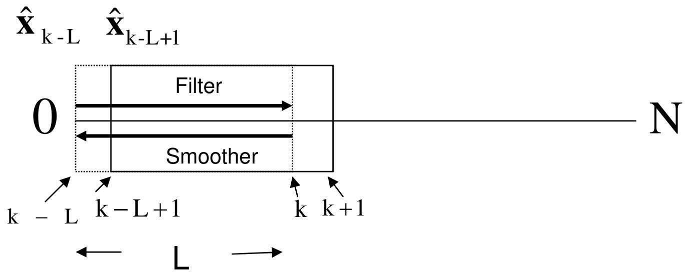


Figure 2.5: This schematic diagram explains the concept of the forward-backward fixed-lag methodology. As shown, the forward filter and backward smoother operate within a window of size L to generate state estimate $\hat{\boldsymbol{x}}_{k-L}$ delayed by L measurements. The window then slides one measurement to the right and the same forward-backward smoothing operations are repeated to obtain $\hat{\boldsymbol{x}}_{k-L+1}$.

results to be saved for the use of backward smoothing process. In the following sections, we propose three different FL-SPKS algorithms, which rely on sequential smoothing rather than state augmentation, to counter the disadvantages of Aug-SPKS.

2.4.2 Forward-Backward A Priori Sigma-Point Kalman Smoother (FB-Priori-SPKS)

The FB-Priori-SPKS algorithm is motivated from the FL-FB-KS smoother, which is originally implemented by Simon for estimating smoothed states in a linear system [11]. Unlike the Aug-SPKS, the FB-Priori-SPKS bypasses the state augmentation by estimating the states sequentially within a sliding window. The FB-Priori-SPKS uses two-pass to generate smoothed states in a given time-window: a filter estimates states $\hat{\boldsymbol{x}}_k$ for $k \in [j-L, j]$ during the forward pass and then a smoothing pass operating backward in time from $k = j$ to $k = j - L$ uses the forward pass results to obtain the final estimates. A new window is formed for $k \in [j-L+1, j+1]$ by shifting forward the previous time-window and then the same two-pass forward-backward process is repeated within the new window to obtain smoothed states. For better understanding, the above procedure is shown in Figure 2.5.

We have extended Simon's work, which formulates the fixed-lag sequential smoothing equations using the standard linear state space, to derive the FB-Priori-SPKS [11]. The equations of FB-Priori-SPKS uses a new form of SPKF. This form of the filter propagates a priori state estimate and the corresponding estimation-error covariance. We employ a priori form of the KF operating on the statistically linearized state space to estimate the smoothed states and the estimation-error covariances for the FB-Priori-SPKS. A priori form of the KF is also easier and straightforward to implement than the SPKF.

We start with the state estimation and estimation error covariance equations for the KF that is applied on the WSLR form of state space defined in Equations (2.28) and (2.29)

$$\boldsymbol{x}_k = \boldsymbol{A}_{f,k-1}\boldsymbol{x}_{k-1} + \boldsymbol{b}_{f,k-1} + \boldsymbol{G}_{f,k-1}\boldsymbol{v}_{k-1} + \boldsymbol{G}_{f,k-1}\boldsymbol{\epsilon}_{f,k-1} \tag{2.142}$$

$$\boldsymbol{z}_k = \boldsymbol{A}_{h,k}\boldsymbol{x}_k + \boldsymbol{b}_{h,k} + \boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k} \tag{2.143}$$

$$\boldsymbol{P}_k^- = \boldsymbol{A}_{f,k-1}\boldsymbol{P}_{k-1}\boldsymbol{A}_{f,k-1}^T + \boldsymbol{G}_{f,k-1}\left(\boldsymbol{Q}_{k-1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k-1}\right)\boldsymbol{G}_{f,k-1}^T \tag{2.144}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \tag{2.145}$$

$$\boldsymbol{P}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k^-\left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{K}_k\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\boldsymbol{K}_k^T \tag{2.146}$$

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{A}_{f,k-1}\hat{\boldsymbol{x}}_{k-1} + \boldsymbol{b}_{f,k-1} \tag{2.147}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^- - \boldsymbol{b}_{h,k}\right), \tag{2.148}$$

where a priori state $\hat{\boldsymbol{x}}_k^-$ and a priori covariance $\boldsymbol{P}_k^-$ is estimated by taking into account the measurements up to time $k-1$. Now, define a new Kalman gain $\boldsymbol{L}_k$, which is $\boldsymbol{A}_{f,k}$ times the standard Kalman gain $\boldsymbol{K}_k$

$$\boldsymbol{L}_k = \boldsymbol{A}_{f,k}\boldsymbol{K}_k. \tag{2.149}$$

Substitute $\boldsymbol{K}_k$ from (2.145)

$$\boldsymbol{L}_k = \boldsymbol{A}_{f,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}. \tag{2.150}$$

Next, we derive the state and the error covariance equations for a priori KF that propagates a priori state $\hat{\boldsymbol{x}}_k^-$ and covariance $\boldsymbol{P}_k^-$ in order to obtain $\hat{\boldsymbol{x}}_{k+1}^-$ and $\boldsymbol{P}_{k+1}^-$.

State propagation for a priori KF

Defining $\hat{\boldsymbol{x}}_{k+1}^-$ in terms of $\hat{\boldsymbol{x}}_k$

$$\hat{\boldsymbol{x}}_{k+1}^- = \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k + \boldsymbol{b}_{f,k}. \tag{2.151}$$

Substituting $\hat{\boldsymbol{x}}_k$ from (2.148) into (2.151)

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1}^- &= \boldsymbol{A}_{f,k}\left(\hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^- - \boldsymbol{b}_{h,k}\right)\right) + \boldsymbol{b}_{f,k} \\ &= \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k^- + \boldsymbol{L}_k\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^- - \boldsymbol{b}_{h,k}\right) + \boldsymbol{b}_{f,k}, \end{aligned} \tag{2.152}$$

where $\boldsymbol{L}_k$ is used from (2.150).

Covariance propagation for a priori KF

From Equation (2.146)

$$\boldsymbol{P}_k = \boldsymbol{P}_k^- - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\boldsymbol{P}_k^- - \boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\boldsymbol{K}_k^T + \boldsymbol{K}_k\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\boldsymbol{K}_k^T + \boldsymbol{K}_k\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\boldsymbol{K}_k^T.$$

Substituting $\boldsymbol{K}_k$ from Equation (2.145) and simplifying

$$\begin{aligned}&\boldsymbol{P}_k = \boldsymbol{P}_k^- + \\ &\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left[-\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} - \left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\right]\boldsymbol{A}_{h,k}\boldsymbol{P}_k^- + \\ &\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left[\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\right]\boldsymbol{A}_{h,k}\boldsymbol{P}_k^- + \\ &\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left[\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\right]\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-.\end{aligned} \tag{2.153}$$

The above equation can further be simplified to form an expression for $\boldsymbol{P}_k$

$$\boldsymbol{P}_k = \boldsymbol{P}_k^- - \boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-. \tag{2.154}$$

Now define $\boldsymbol{P}_{k+1}^-$ in terms of $\boldsymbol{P}_k$

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{f,k}^T + \boldsymbol{G}_{f,k}\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)\boldsymbol{G}_{f,k}^T. \tag{2.155}$$

Substitute $\boldsymbol{P}_k$ from (2.154) and after further simplifying we obtain an expression for $\boldsymbol{P}_{k+1}^-$ in terms of $\boldsymbol{P}_k^-$

$$\begin{aligned}\boldsymbol{P}_{k+1}^- &= \boldsymbol{A}_{f,k}\left[\boldsymbol{P}_k^- - \boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k^-\right]\boldsymbol{A}_{f,k}^T + \\ &\quad \boldsymbol{G}_{f,k}\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)\boldsymbol{G}_{f,k}^T \\ &= \boldsymbol{A}_{f,k}\boldsymbol{P}_k^-\left(\boldsymbol{A}_{f,k} - \boldsymbol{L}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{G}_{f,k}\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)\boldsymbol{G}_{f,k}^T.\end{aligned} \tag{2.156}$$

Note, we derive an alternate form of the KF equations that provides an *one-step* solution for propagating a priori state and a priori error covariance forward in time using the WSLR form of the nonlinear system. In this respect, it is worth mentioning that both the *one-step* a priori KF and the standard two-step KF with Equations (2.144)-(2.148) generate identical estimates of the state and estimation-error covariances.

Combining Equations (2.152) and (2.156), a priori one-step formulation of the KF can be summarized below:

$$\hat{\boldsymbol{x}}_{k+1}^{-} = \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_{k}^{-} + \boldsymbol{L}_k\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_{k}^{-} - \boldsymbol{b}_{h,k}\right) + \boldsymbol{b}_{f,k} \tag{2.157}$$

$$\boldsymbol{P}_{k+1}^{-} = \boldsymbol{A}_{f,k}\boldsymbol{P}_{k}^{-}\left(\boldsymbol{A}_{f,k} - \boldsymbol{L}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{G}_{f,k}\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)\boldsymbol{G}_{f,k}^T, \tag{2.158}$$

where the a priori gain $\boldsymbol{L}_k$ is defined as

$$\begin{aligned}\boldsymbol{L}_k &= \boldsymbol{A}_{f,k}\boldsymbol{K}_k \\ &= \boldsymbol{A}_{f,k}\boldsymbol{P}_{k}^{-}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_{k}^{-}\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}.\end{aligned} \tag{2.159}$$

The a priori KF formulations shown above are then applied on the augmented statistically linearized state space (Equations (2.28) and (2.29)) to derive the sequential two-pass smoothing algorithm. In order to maintain continuity, the derivation is skipped here and described in Appendix 2.8.

FB-Priori-SPKS algorithm

The final equations of the FB-Priori-SPKS algorithm including the forward filter and the backward smoother are demonstrated below:

- *While* $j \leq N$
- *Forward filter*:
  A priori Kalman filter defined in Equations (2.157)-(2.159) is applied on the statistically linearized state space to obtain $\boldsymbol{L}_k$, $\hat{\boldsymbol{x}}_k^-$ and $\boldsymbol{P}_k^-$ at each discrete time k.
  - *For* $k = j - L, j - L + 1, \ldots, j$

$$\boldsymbol{L}_k = \boldsymbol{A}_{f,k}\boldsymbol{P}_k^- \boldsymbol{A}_{h,k}^T \left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^- \boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \tag{2.160}$$

$$\hat{\boldsymbol{x}}_{k+1}^- = \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k^- + \boldsymbol{L}_k\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^- - \boldsymbol{b}_{h,k}\right) + \boldsymbol{b}_{f,k} \tag{2.161}$$

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{A}_{f,k}\boldsymbol{P}_k^- \left(\boldsymbol{A}_{f,k} - \boldsymbol{L}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{G}_{f,k}\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)\boldsymbol{G}_{f,k}^T, \tag{2.162}$$

  - End *For*
- *Backward smoothing*:
  - *Initialization*:

$$\hat{\boldsymbol{x}}_{k+1,k} = \hat{\boldsymbol{x}}_{k+1}^- \tag{2.163}$$

$$\boldsymbol{L}_{k,0} = \boldsymbol{L}_k \tag{2.164}$$

$$\boldsymbol{P}_k^{0,0} = \boldsymbol{P}_k^-. \tag{2.165}$$

  - *Smoothing recursions*:

$*$ *For* $i = 1, 2, \ldots, L+1$

$$\boldsymbol{L}_{k,i} = \boldsymbol{P}_k^{0,i-1}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \tag{2.166}$$

$$\boldsymbol{P}_{k+1}^{i,i} = \boldsymbol{P}_k^{i-1,i-1} - \boldsymbol{P}_k^{0,i-1}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_{k,i}^T\boldsymbol{A}_{f,k}^T \tag{2.167}$$

$$\boldsymbol{P}_{k+1}^{0,i} = \boldsymbol{P}_k^{0,i-1}\left(\boldsymbol{A}_{f,k} - \boldsymbol{L}_{k,0}\boldsymbol{A}_{h,k}\right)^T \tag{2.168}$$

$$\hat{\boldsymbol{x}}_{k+1-i,k} = \hat{\boldsymbol{x}}_{k+2-i,k} + \boldsymbol{L}_{k,i}\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^- - \boldsymbol{b}_{h,k}\right) \tag{2.169}$$

$*$ End *For*

- Increment j by one: $j = j + 1$
- End *While*

Note, the FB-Priori-SPKS algorithm proposes a set of backward recursions, which loops through for the period of entire lag L. For the first time through this loop ($i = 1$), we obtain the standard measurement update of the KF. At the end of the loop ($i = L+1$), smoothed estimates of each state with delays between 0 and L are computed using measurements up to time k.

Although we have found that both the FB-Priori-SPKS and Aug-SPKS performs comparably in a number of state estimation examples, the FB-Priori-SPKS is more computationally efficient than the Aug-SPKS. The FB-Priori-SPKS performs $O\left(LM^3\right)$ computations within each sliding window in order to generate the smoothed estimate $\hat{\boldsymbol{x}}_{k-L}$ given measurements $\boldsymbol{z}_{1:k}$. Recall that the Aug-SPKS requires a significantly higher order of computation $O\left(M^3L^3\right)$ in order to generate the smoothed estimate $\hat{\boldsymbol{x}}_{k-L}$. Although the generation of each estimate takes $O\left(LM^3\right)$ processing, the computational complexity of the FB-Priori-SPKS increases to $O\left(NLM^3\right)$ in order to compute N smoothed estimates using N sliding windows. The FB-Priori-SPKS is not only computationally superior, it also requires less memory compared to the Aug-SPKS to save each estimated state and estimation error covariance, which are of size $M \times 1$ and $M \times M$ respectively. One disadvantage of the FB-Priori-SPKS is that the backward smoothing loop calculates L

cross-estimation-error covariances $\boldsymbol{P}_k^{0,i}$ of dimension $M \times M$ within each window, which demands higher memory and also takes longer in computing the smoothed estimates. In addition, the forward estimation results of each L sized window need to be saved for the backward smoothing recursions to take place. In the next sections, we present a fixed-lag version of the FBSL-SPKS and RTSSL-SPKS methods, which are more direct and simple to implement while maintaining the same level of computational complexity and performance of the FB-Priori-SPKS.

2.4.3 Forward-Backward Statistical Linearized Sigma-Point Kalman Smoother (FBSL-SPKS)

In this section, we demonstrate how the FBSL-SPKS, derived in Section 2.3.2 for the fixed-interval case, can be extended into the fixed-lag framework. Similar to the FB-Priori-SPKS, the proposed FBSL-SPKS algorithm works in a sequential forward-backward manner within a windowed set of L measurements, from $k = j - L$ to $k = j$, where the time index k constantly moves forward. The FBSL-SPKS performs in three steps, which can be summarized as follows. A SPKF starts at $k = j - L$ and operates forward in time on the true dynamics until $k = j$ to obtain posterior state estimates $\hat{\boldsymbol{x}}_k$. An information filter starts at $k = j$ and then operates backward in time to $k = j - L$ in order to generate prior information states $\hat{\boldsymbol{y}}_k^-$. Note that the backward filter works on a statistically linearized dynamical space, the coefficients of which can be obtained from the forward SPKF recursions by using the WSLR formulations. The estimates of the forward and backward filters are then statistically combined at each time k to obtain the smoothed estimates $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$.

FBSL-SPKS algorithm

The pseudo code of the FBSL-SPKS algorithm combining the forward filter, the backward filter and the smoother is described below:

```
• While $j \leq N$
• Forward filter:
    – For $k = j - L, j - L + 1, j - L + 2, \ldots, j - 1, j$
      The SPKF algorithm as demonstrated in Section 2.3.2 is used as a forward
      filter to compute state estimates $\hat{\boldsymbol{x}}_k$ and estimation-error covariances $\boldsymbol{P}_k$ at
      each discrete time $k$. Statistically linearized parameters $\boldsymbol{A}_{f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{f,k}$, $\boldsymbol{b}_{h,k}$,
      $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ and $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ are also derived at each $k$ by applying the WSLR formulations
      on the nonlinear state space.
    – End For
• For $k = j, j - 1, j - 2, \ldots, j - L + 1, j - L$
    – Backward filter:
        * Time update:
              $\boldsymbol{S}_k^- = \boldsymbol{A}_{f,k}^T \left( \boldsymbol{I} - \boldsymbol{K}_k^{\mathrm{b}} \boldsymbol{G}_{f,k}^T \right) \boldsymbol{S}_{k+1} \boldsymbol{A}_{f,k}$        (2.170)
              $\hat{\boldsymbol{y}}_k^- = \boldsymbol{A}_{f,k}^T \left( \boldsymbol{I} - \boldsymbol{K}_k^{\mathrm{b}} \boldsymbol{G}_{f,k}^T \right) \left( \hat{\boldsymbol{y}}_{k+1} - \boldsymbol{S}_{k+1} \boldsymbol{b}_{f,k} \right)$        (2.171)
        * Measurement Update:
              $\boldsymbol{S}_k = \boldsymbol{S}_k^- + \boldsymbol{A}_{h,k}^T \left( \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} \right)^{-1} \boldsymbol{A}_{h,k}$        (2.172)
              $\hat{\boldsymbol{y}}_k = \hat{\boldsymbol{y}}_k^- + \boldsymbol{A}_{h,k}^T \left( \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} \right)^{-1} \left( \boldsymbol{z}_k - \boldsymbol{b}_{h,k} \right)$        (2.173)
```

- *Smoother*:

  The smoothed estimates are obtained by combining the forward and the backward estimates.

$$\boldsymbol{P}_k^{\mathrm{S}} = \left[(\boldsymbol{P}_k)^{-1} + \boldsymbol{S}_k^{-} \right]^{-1} \tag{2.174}$$

$$\hat{\boldsymbol{x}}_k^{\mathrm{S}} = \left(\boldsymbol{I} + \boldsymbol{P}_k \boldsymbol{S}_k^{-} \right)^{-1} \hat{\boldsymbol{x}}_k + \boldsymbol{P}_k^{\mathrm{S}} \hat{\boldsymbol{y}}_k^{-} \tag{2.175}$$

- End *For*
- Increment j by one: $j = j + 1$
- End *While*

As no state augmentation is required, the computational cost of implementing the FBSL-SPKS to generate N smoothed estimates is of the same order as the FB-Priori-SPKS, $O\left(NLM^3\right)$, where N is the number of observations, M is the state dimension and L is the lag. Like the FB-Priori-SPKS, each smoothed state and estimation error covariance generated by the FBSL-SPKS are also of size M. However, unlike the FB-Priori-SPKS, computation of cross estimation error covariances are no longer required. Comparing to the fixed-interval version of the FBSL-SPKS, the forward estimation results are still need to be saved in order to apply the backward filtering step. The major difference here is that in the fixed-lag case, the smoother works within a windowed set of L states and hence only L forward estimates are required to obtain the smoothed results. However, it inherits the fundamental drawbacks of the fixed-interval FBSL-SPKS approach: the backward filtering and the smoothing occurs in two separate steps and the backward filter needs the inverse dynamics of the forward filter.

2.4.4 Rauch-Tung-Striebel Statistical Linearized Sigma-Point Kalman Smoother (RTSSL-SPKS)

Similar to the FBSL-SPKS, the fixed-interval RTSSL-SPKS equations, detailed in Section 2.3.3, can be extended into the fixed-lag framework. The fixed-lag version of RTSSL-SPKS

SPKS	**Computation**	**Memory**
Fixed-Interval SPKS		
FBSL-SPKS	$O\left(NM^3\right)$	$\left(\hat{\boldsymbol{x}}_k, \boldsymbol{P}_k\right) \in \mathbb{R}^M$, $\forall k \in [1, N]$
RTSSL-SPKS	$O\left(NM^3\right)$	$\left(\hat{\boldsymbol{x}}_k, \boldsymbol{P}_k, \hat{\boldsymbol{x}}_k^-, \boldsymbol{P}_k^-\right) \in \mathbb{R}^M$, $\forall k \in [1, N]$
Fixed-Lag SPKS		
Aug-SPKS	$O\left(NL^3M^3\right)$	$\left(\hat{\boldsymbol{x}}_k, \boldsymbol{P}_k\right) \in \mathbb{R}^{ML}$
FB-Priori-SPKS	$O\left(NLM^3\right)$	$\left(\hat{\boldsymbol{x}}_k, \boldsymbol{P}_k, \boldsymbol{P}_k^{i,j}\ i \neq j\right) \in \mathbb{R}^M$, $\forall k \in [1, L]$
FBSL-SPKS	$O\left(NLM^3\right)$	$\left(\hat{\boldsymbol{x}}_k, \boldsymbol{P}_k\right) \in \mathbb{R}^M$, $\forall k \in [1, L]$
RTSSL-SPKS	$O\left(NLM^3\right)$	$\left(\hat{\boldsymbol{x}}_k, \boldsymbol{P}_k, \hat{\boldsymbol{x}}_k^-, \boldsymbol{P}_k^-\right) \in \mathbb{R}^M$, $\forall k \in [1, L]$

Table 2.1: Performance comparison of the proposed SPKS in terms of computation and memory.

works within a windowed set of L measurements between time $k = j - L$ and $k = j$ using a forward and a backward pass:

1. A standard SPKF is used as a forward filter, which at each k estimates a priori state $\hat{\boldsymbol{x}}_k^-$, a posteriori state $\hat{\boldsymbol{x}}_k$ and their respective error covariances by operating on the true nonlinear dynamics.

2. A backward RTS smoothing pass is then followed on the forward filtering results to generate $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ by linearly combining the forward estimates with a correction term governed by a smoothing gain. Note, the smoothing gain is a function of the statistically linearized process model, forward estimation error covariance and forward prediction covariance. The correction term is equal to the difference between the future smoothed state $\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}}$ and a priori state $\hat{\boldsymbol{x}}_{k+1}^-$ at time $k + 1$.

The time index k continuously moves forward to accommodate new measurements and the RTSSL-SPKS equations are applied on each L windowed segment to determine the smoothed estimates.

RTSSL-SPKS algorithm

The final equations of the RTSSL-SPKS algorithm including the forward filter and the backward smoother are described below:

- *While* $j \leq N$
- *Forward filter*:
  - *For* $k = j-L, j-L+1, j-L+2, \ldots, j-1, j$

    The SPKF algorithm with WSLR as defined in Section 2.3.2 is applied for $k = 1, 2, \ldots, N$. In addition to computing the estimates $\hat{\boldsymbol{x}}_k$ and the error covariances $\boldsymbol{P}_k$, the forward pass also saves the intermediate results including the state prediction $\hat{\boldsymbol{x}}_k^-$ and the prediction error covariance $\boldsymbol{P}_k^-$ at each time k. The statistically linearized parameters $\boldsymbol{A}_{f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{f,k}$, $\boldsymbol{b}_{h,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ and $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ are also generated at each SPKF recursion using the WSLR technique on the nonlinear state space.
  - End *For*
- *Backward smoothing*:
  - *For* $k = j, j-1, j-2, \ldots, j-L+1, j-L$
    - *Error covariance smoothing*:

$$\boldsymbol{D}_k = \boldsymbol{P}_k \boldsymbol{A}_{f,k}^T \left(\boldsymbol{P}_{k+1}^-\right)^{-1} \tag{2.176}$$

$$\boldsymbol{P}_k^{\mathrm{S}} = \boldsymbol{P}_k - \boldsymbol{D}_k \left(\boldsymbol{P}_{k+1}^- - \boldsymbol{P}_{k+1}^{\mathrm{S}}\right) \boldsymbol{D}_k^T \tag{2.177}$$

    - *State estimate smoothing*:

$$\hat{\boldsymbol{x}}_k^{\mathrm{S}} = \hat{\boldsymbol{x}}_k + \boldsymbol{D}_k \left(\hat{\boldsymbol{x}}_{k+1}^{\mathrm{S}} - \hat{\boldsymbol{x}}_{k+1}^-\right) \tag{2.178}$$

    – End *For*

- Increment j by one: $j = j + 1$
- End *While*

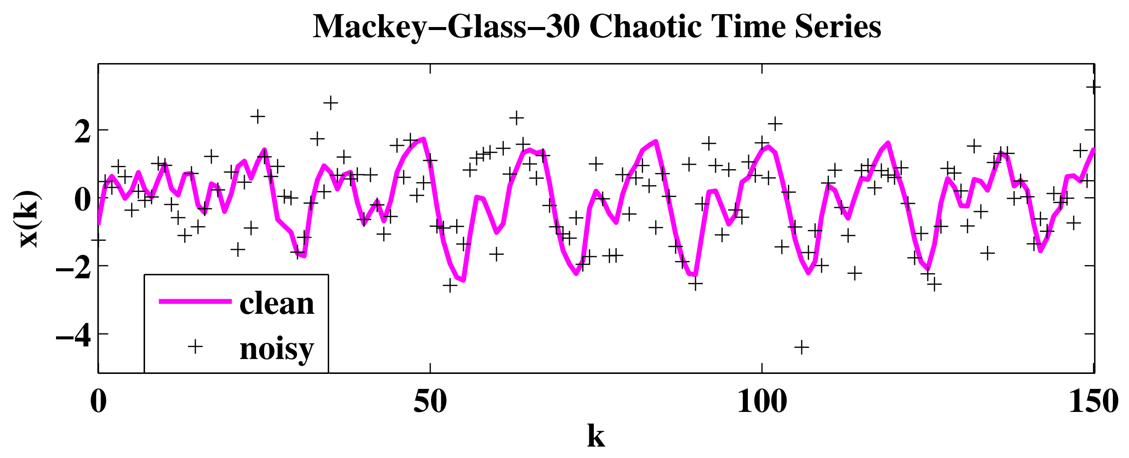
The computational complexity of RTSSL-SPKS to estimate N smoothed states is of the same order as the FB-Priori-SPKS and the FBSL-SPKS, $O\left(NLM^3\right)$, where N is the number of observations, M is the state dimension and L is the lag. In the RTSSL-SPKS, the backward filtering and the smoothing are combined to a single step and hence it avoids an additional $O\left(NLM^3\right)$ computations for performing a separate smoothing step. However, it needs to save L forward predictions with L forward estimates for the backward smoothing step which makes the RTSSL-SPKS slightly more memory intensive than the FBSL-SPKS. As only L forward estimates are required compared to N estimates to perform the backward smoothing, the fixed-lag RTSSL-SPKS requires considerably less memory than the fixed-interval case. Table 2.1 compares the computational complexity and memory of all the proposed fixed-interval and fixed-lag smoothers.

2.5 Numerical Simulations

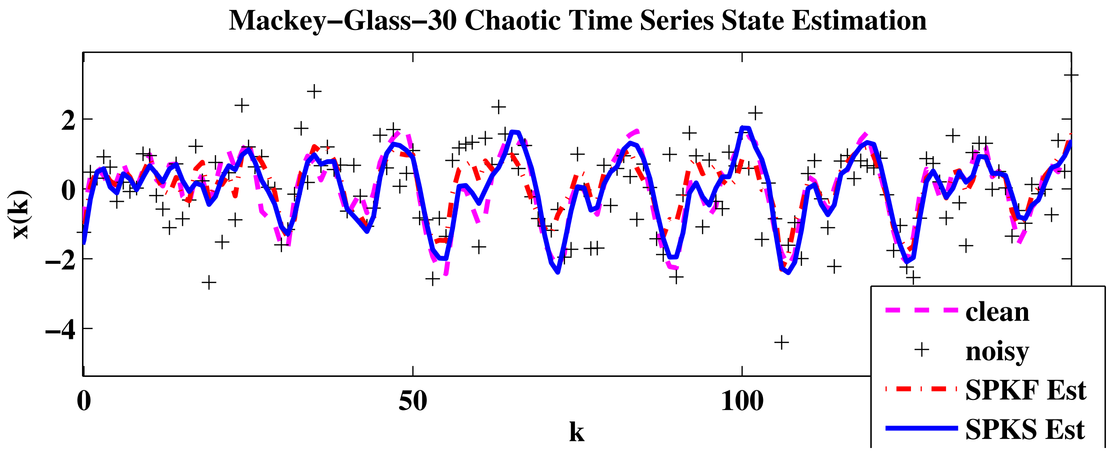
We evaluate all our fixed-interval and fixed-lag smoother algorithms in the following scenarios:

1. Estimation of an underlying clean Mackey-Glass chaotic time series corrupted by an additive white Gaussian noise.
2. Tracking a space vehicle when it re-enters into the earth's atmosphere at a high altitude and with a high speed.

These two examples were used to demonstrate the performances of previously proposed sigma point smoothers, namely the FBNL-SPKS [17, 26] and URTSS [32] and hence is chosen in this dissertation in order to facilitate comparison among our methodologies. In



(a)



(b)

Figure 2.6: Performance comparison between the FI-SPKS and the SPKF using the Mackey-Glass chaotic time series. (a): Clean and noisy time series. (b): Estimated time series.

addition to comparing various SPKS methodologies, we also evaluate our smoothers versus the EKF, EKS and SPKF.

2.5.1 Problem Description and State Space Representation

Mackey-Glass clean time series estimation

In this example, the objective is to estimate the clean Mackey-Glass-30 chaotic time series which is corrupted by an additive white Gaussian noise (SNR = 0 db). This example

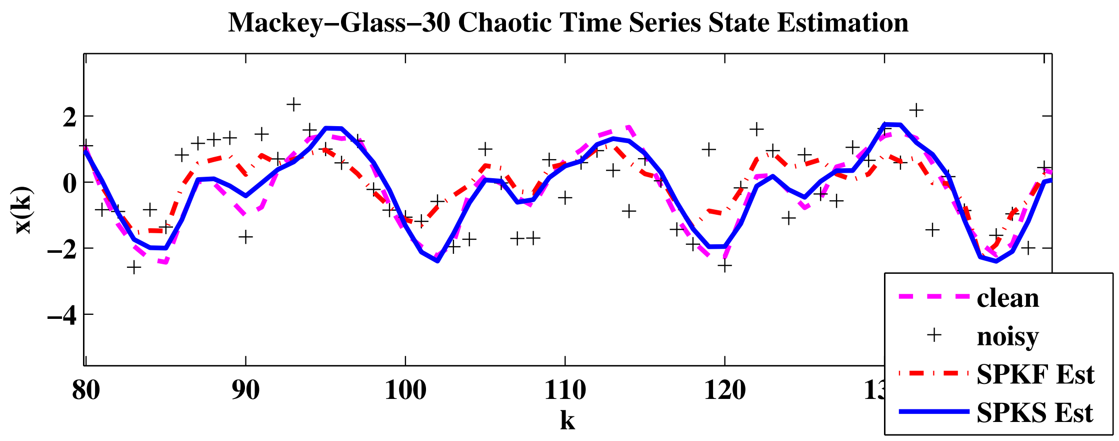


Figure 2.7: Performance comparison between the FI-SPKS and the SPKF using the Mackey-Glass chaotic time series. This plot *zooms in* on a section of the estimated time series shown in Figure 2.6(b) to demonstrate how accurately the SPKS tracks the ground truth.

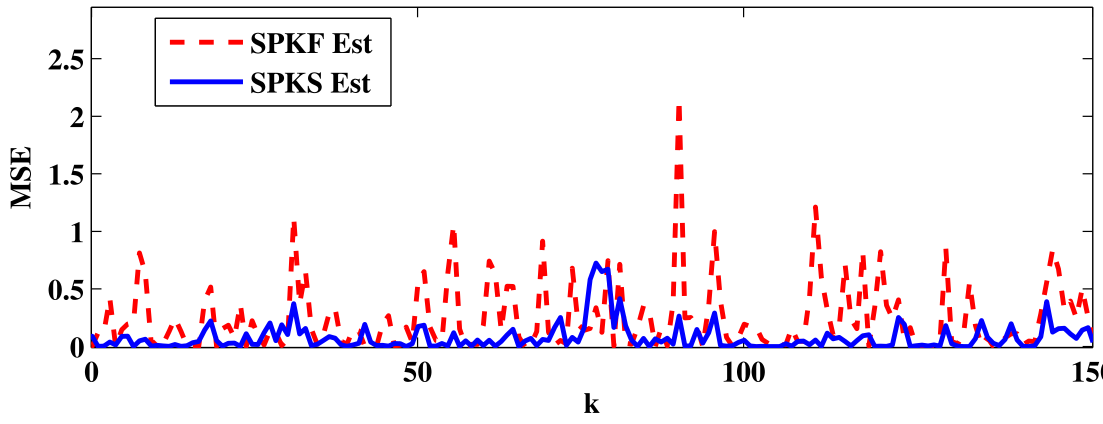
was used to demonstrate the superior performance of *FBNL-SPKS* over the traditional SPKF [26]. The Mackey-Glass time series is generated by the following continuous time differential Equation [26]

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t-3)}{1 + x(t-30)^{10}} \tag{2.179}$$

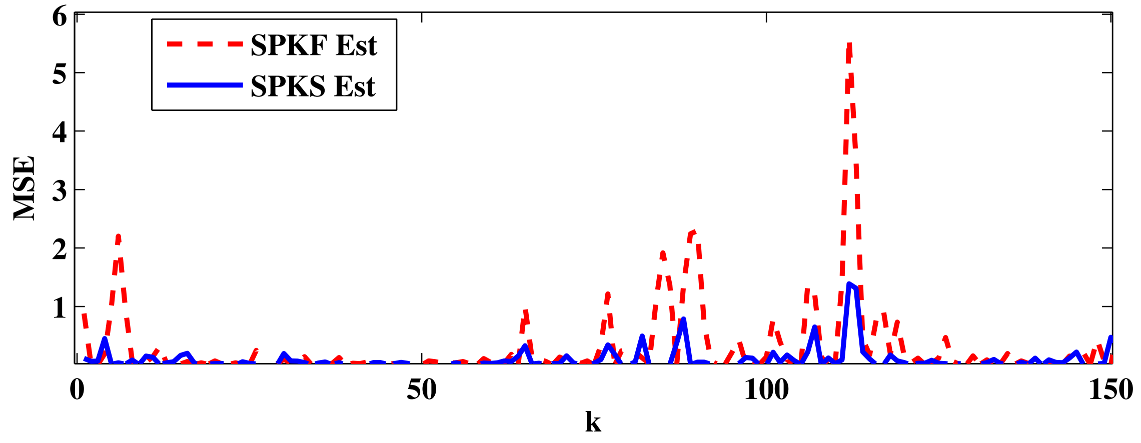
where t is the continuous time index and $x(t)$ is the time series amplitude at time t. The discrete time version of the clean time series is modeled as a nonlinear autoregressive process [17], which is shown below:

$$x_k = f\left(x_{k-1}, x_{k-2}, \ldots, x_{k-M}; \boldsymbol{w}\right) + v_k \quad \text{[in this example, } M = 6\text{]} \tag{2.180}$$

The parameterized model f is approximated by training a 6-5-1 (input-hidden-output) nodes feed-forward neural network on the clean time series. The optimum number of hidden nodes is chosen by minimizing the modeling error using a ten fold cross validation technique. $\boldsymbol{w}$ denotes the neural net weights and biases after learning. The modeling error v_k is assumed to have zero mean and covariance σ_v^2, which is computed after the convergence of the neural net weights. White Gaussian noise is added with the clean time series to obtain SNR $= 0\,$db. The noise corrupted time series z_k at each time k are



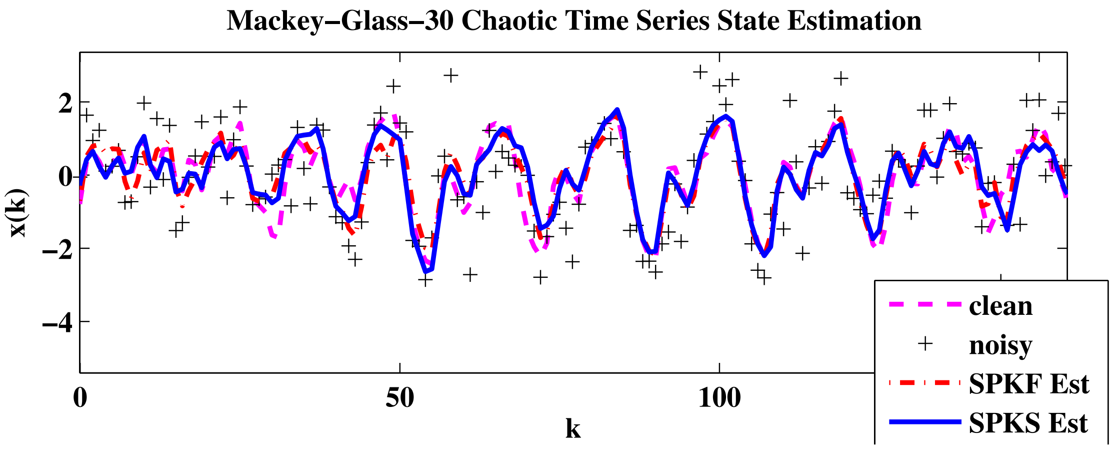
(a)



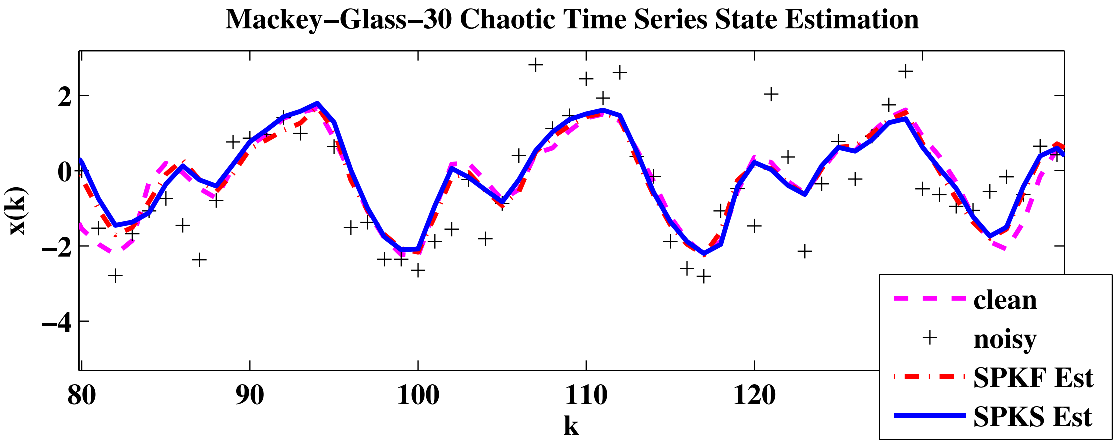
(b)

Figure 2.8: (a): Performance evaluation of the FI-SPKS in terms of MSE for the Mackey-Glass time series estimation problem. (b): This plot exemplifies a single run where the SPKF generates a large spike in the MSE, whereas the SPKS is able to keep the estimation error at the lower level. MSE is calculated between the estimates and the original clean time series by averaging over 200 randomly initialized Monte-Carlo (MC) runs. As proved from the MSE results, the RTSSL-SPKS generates more accurate estimates than the SPKF.

considered as measurements and fed into the estimator. The M element state vector is denoted as $\boldsymbol{x}_k = \begin{bmatrix} x_{k-1} & x_{k-2} & \dots & x_{k-M} \end{bmatrix}$ and n_k is the measurement noise. The state



(a)



(b)

Figure 2.9: Performance comparison between the FL-SPKS ($L = 10$) and the SPKF using the Mackey-Glass chaotic time series. (a): Estimated time series with the ground truth. (b): This plot *zooms in* on a section of the estimated time series.

space configuration of the above problem is defined as:

$$\boldsymbol{x}_{k+1} = f\left(\boldsymbol{x}_k; \boldsymbol{w}\right) + \boldsymbol{G}_{f,k} v_k \tag{2.181}$$

$$z_k = \boldsymbol{H}_k \boldsymbol{x}_k + n_k, \tag{2.182}$$

which can be expanded as

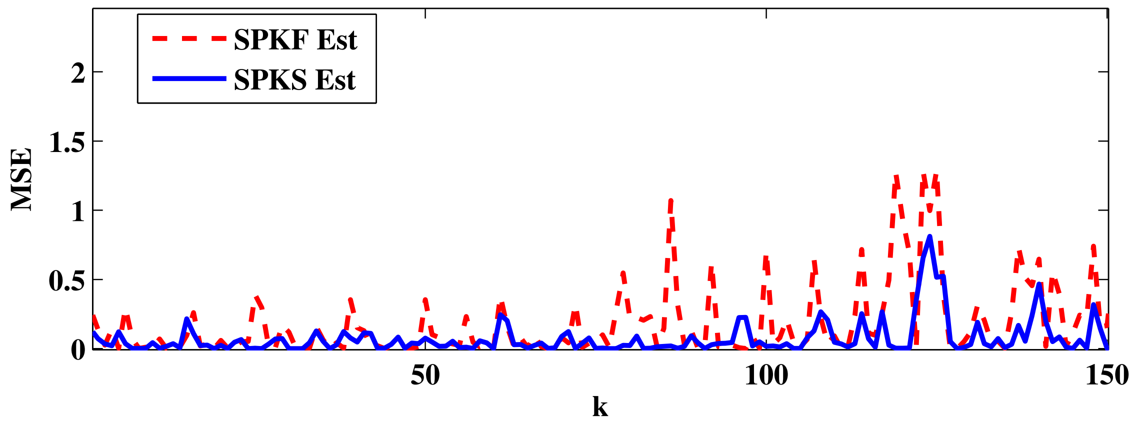


Figure 2.10: Performance evaluation of the FL-SPKS ($L = 10$) in terms of MSE for the Mackey-Glass time series estimation problem. MSE is computed between the estimates and the clean time series by averaging over 200 randomly initialized MC runs. The performance advantage of the RTSSL-SPKS over SPKF is clearly shown.

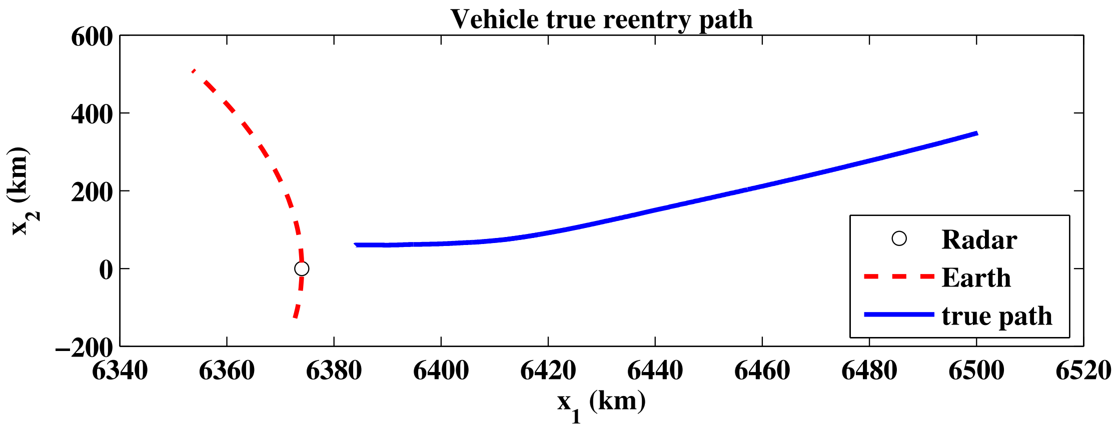
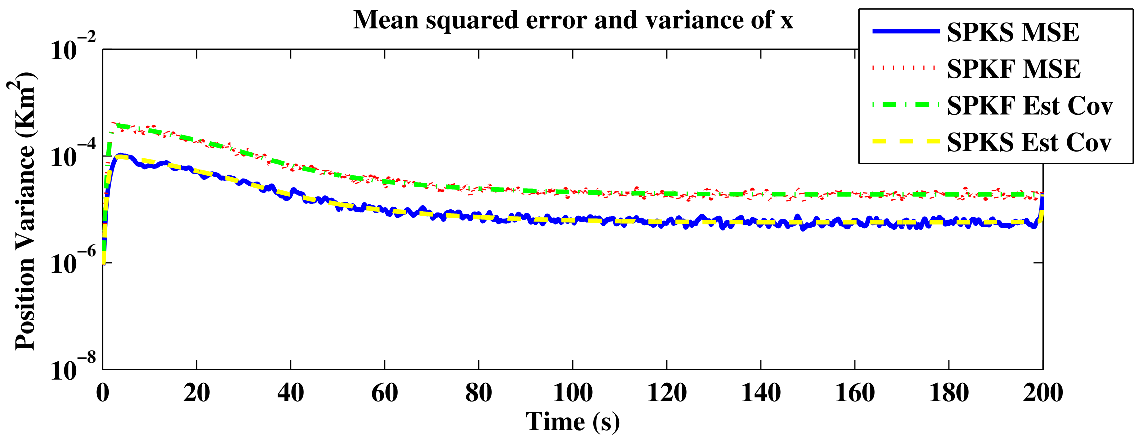


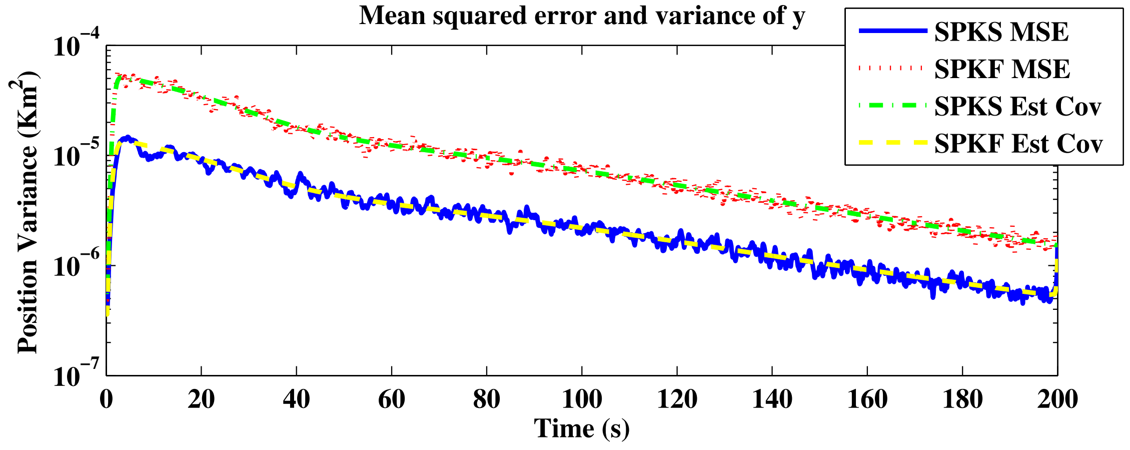
Figure 2.11: True vehicle trajectory. The solid line is the vehicle trajectory and the dashed line is the earth's surface. The radar is placed at 'o'.

- *Process Model*:

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ x_k \\ \vdots \\ x_{k-M+2} \end{bmatrix} = \begin{bmatrix} f\left(x_k, x_{k-1}, \cdots, x_{k-M+1}; \boldsymbol{w}\right) \\ \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ x_{k-1} \\ \vdots \\ x_{k-M+1} \end{bmatrix} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} v_k \qquad (2.183)$$



(a) 'x' position error



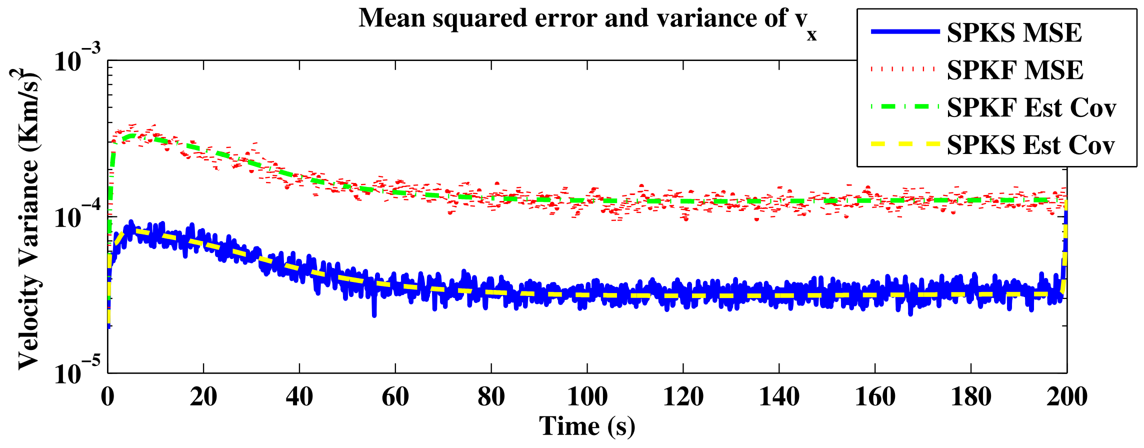
(b) 'y' position error

Figure 2.12: Performance comparison between the FI-SPKS and the SPKF on vehicle re-entry tracking example. (a) and (b): MSE and estimation-error covariances for 2D vehicle position x_k and y_k. MSE is calculated between the estimates and the ground truths by averaging over 200 randomly initialized MC runs. The MSE results clearly demonstrate that the RTSSL-SPKS is more accurate in vehicle position estimation than the standard SPKF.

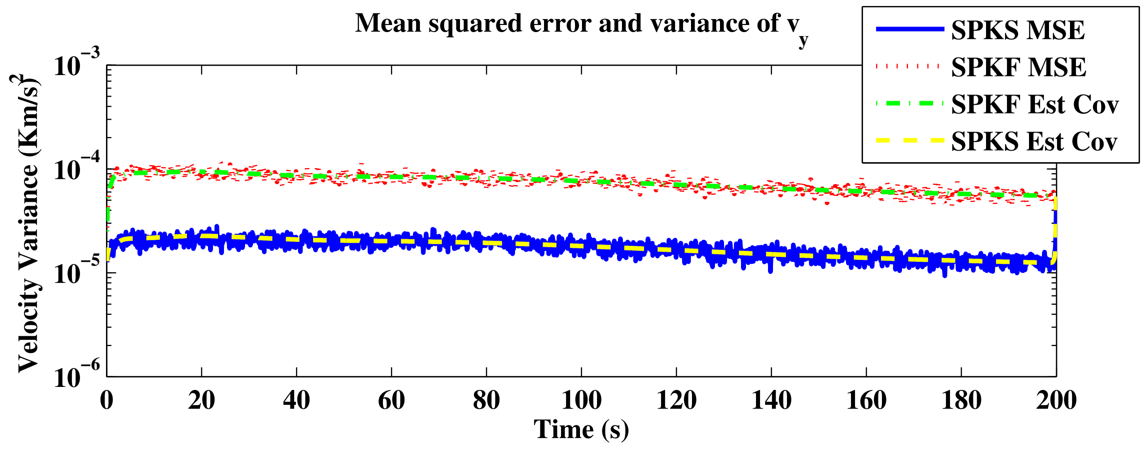
- *Observation Model*:

$$z_k = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_k + n_k. \tag{2.184}$$

Note that the process model is nonlinear while the observation model is linear.



(a) 'x' velocity error



(b) 'y' velocity error

Figure 2.13: Performance comparison between the FI-SPKS ($N = 2000$) and the SPKF on vehicle re-entry tracking example. (a) and (b): MSE and estimation-error covariances for 2D velocity of the vehicle v_{x_k} and v_{y_k}. MSE is calculated between the estimates and the ground truths by averaging over 200 randomly initialized MC runs. It is evident from the plots that the RTSSL-SPKS generates lower MSE than the SPKF for vehicle velocity estimation.

Vehicle Re-entry Tracking

In this case, the task involves to track a space vehicle that re-enters into the earth's atmosphere at a high altitude and with a significantly large velocity. A radar stationed on the earth is used to measure the range and the bearing of the vehicle. This tracking problem is regarded as particularly challenging for a state estimator because the entering vehicle is under the influence of strong nonlinear forces such as aerodynamic drag and earth's gravity [37–39]. In addition, a poor knowledge of initial vehicle state and unknown properties of the aerodynamic forces make the problem even more challenging. The aerodynamic drag is a function of the vehicle's velocity and varies exponentially with the altitude. The earth's gravitational force pulls the vehicle toward its center and its magnitude possesses a nonlinear relationship with the position of the vehicle. Julier *et al.* have successfully employed the Unscented Kalman filter (UKF) in order to estimate the vehicle's state [23]. They have also shown that the UKF can provide a significantly better estimate than the EKF.

The vehicle's state ($\boldsymbol{x}_k$) consists of its 2D position (x_k and y_k), 2D velocity (v_{x_k} and v_{y_k}) and a scaler parameter of aerodynamic drag (d_k). The vehicle state dynamics can be shown as [23]:

- *Dynamical Model*:

$$x_{k+1} = x_k + \delta T v_{x_k} \tag{2.185}$$

$$y_{k+1} = y_k + \delta T v_{y_k} \tag{2.186}$$

$$v_{x_{k+1}} = \left(1 + \delta T \boldsymbol{D}_k^{\mathrm{dr}}\right) v_{x_k} + \delta T \boldsymbol{G}_k^{\mathrm{g}} x_k + v_{p_{x,k}} \tag{2.187}$$

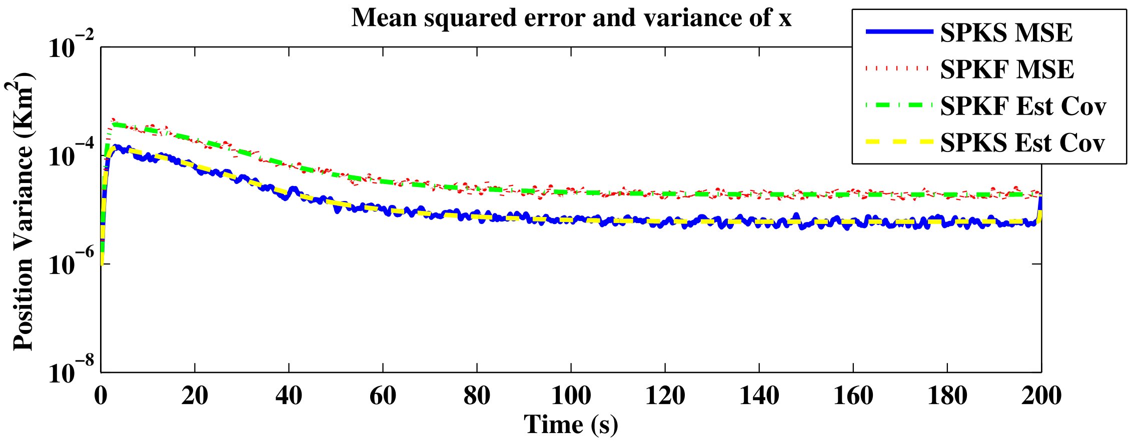
$$v_{y_{k+1}} = \left(1 + \delta T \boldsymbol{D}_k^{\mathrm{dr}}\right) v_{y_k} + \delta T \boldsymbol{G}_k^{\mathrm{g}} y_k + v_{p_{y,k}} \tag{2.188}$$

$$d_{k+1} = d_k + v_{p_{d,k}} \tag{2.189}$$

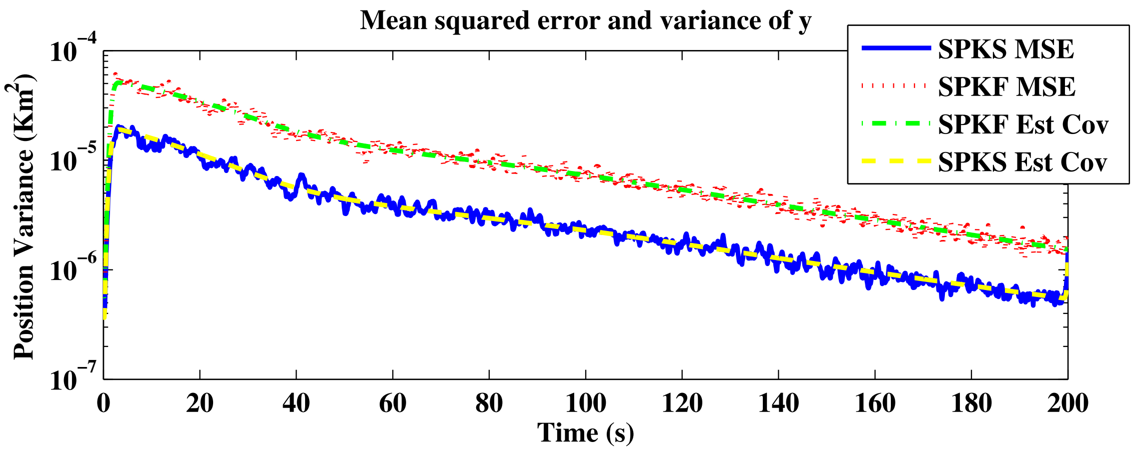
where $\boldsymbol{D}_k^{\mathrm{dr}}$ and $\boldsymbol{G}_k^{\mathrm{g}}$ are the aerodynamic drag related force term and the gravity related force term respectively at each discrete time k. All these forces are highly nonlinear and can be defined as:

$$\boldsymbol{D}_k^{\mathrm{dr}} = -\beta_k \mathrm{exp}\left[\frac{R_0 - R_k}{H_0}\right] V_k \tag{2.190}$$

$$\boldsymbol{G}_k^{\mathrm{g}} = -\frac{G m_0}{(R_0 - R_k)^3}, \tag{2.191}$$



(a) ‘x’ position error



(b) ‘y’ position error

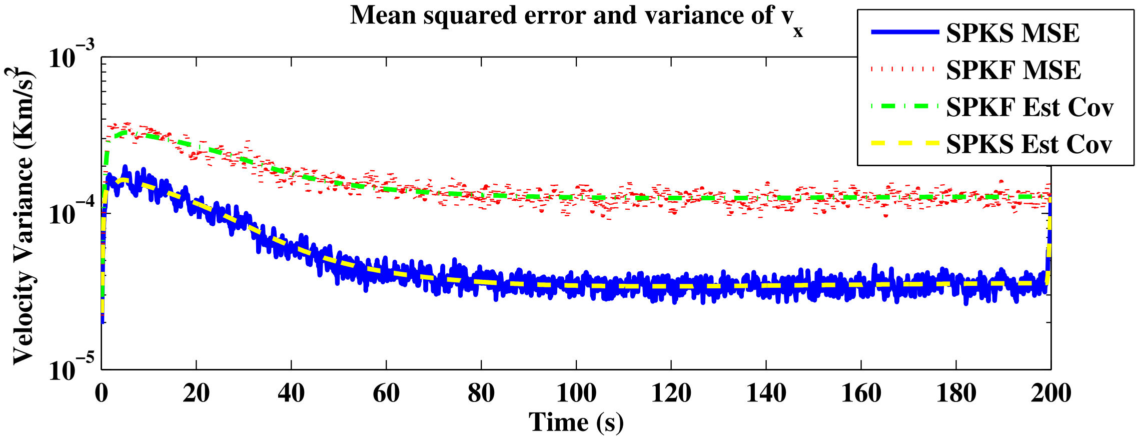
Figure 2.14: Performance comparison between the FL-SPKS (lag $L = 10$) and the SPKF on vehicle re-entry tracking example. (a) and (b): MSE and estimation-error covariances for 2D vehicle position x_k and y_k. MSE is calculated between the estimates and the ground truths by averaging over 200 randomly initialized MC runs. From the figures, the superior performance of FL-SPKS over the SPKF can be observed.

where

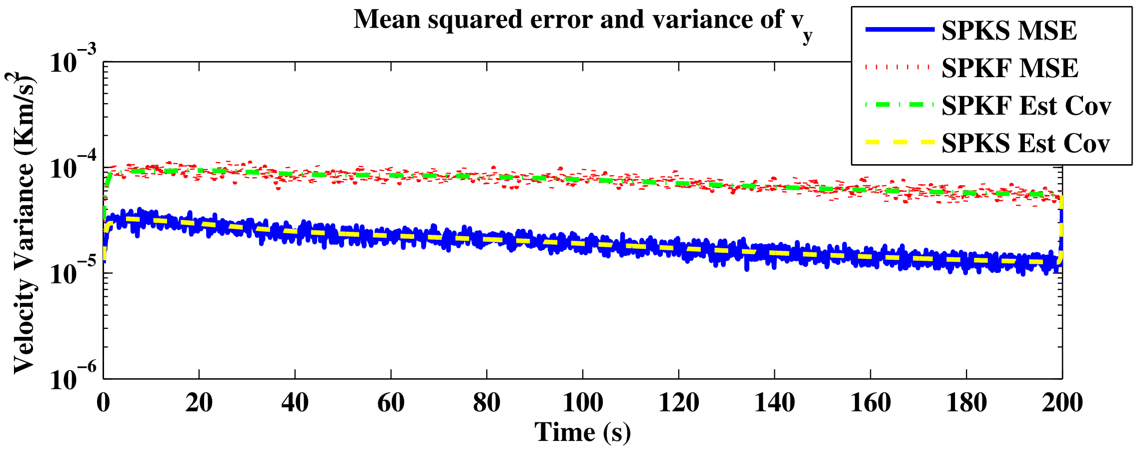
$$\beta_k = \beta_0 \exp(d_k) \tag{2.192}$$

$$\begin{aligned} R_k &= \text{Distance between the vehicle and the center of the earth} \\ &= \sqrt{x_k^2 + y_k^2} \end{aligned} \tag{2.193}$$

$$\begin{aligned} V_k &= \text{Magnitude of the vehicle's velocity} \\ &= \sqrt{v_{x_k}^2 + v_{y_k}^2}. \end{aligned} \tag{2.194}$$



(a) ‘x’ velocity error



(b) ‘y’ velocity error

Figure 2.15: Performance comparison between the FL-SPKS (lag $L = 10$) and the SPKF on vehicle re-entry tracking example. (a) and (b): MSE and estimation-error covariances for 2D velocity of the vehicle v_{x_k} and v_{y_k}. MSE is calculated between the estimates and the ground truths by averaging over 200 randomly initialized MC runs. As seen, the fixed-lag RTSSL-SPKS generates more accurate estimates than the SPKF.

The “ballistic coefficient” β_k represents the uncertainty in vehicle characteristics. The magnitudes of the typical vehicle and the earth parameters (β_0, R_0, H_0 and Gm_0), the process noise parameters ($\boldsymbol{v}_{p_k}$) and the state initialization are taken from [23]. The discrete state dynamics is formed by applying the Euler approximation on the continuous vehicle dynamics. Due to large nonlinearities, the Euler integration step δT is chosen to be small i.e. 50 ms.

- *Observation Model*:

$$r_k = \sqrt{\left(x_{1,k} - x_{\mathrm{r}}\right)^2 + \left(x_{2,k} - y_{\mathrm{r}}\right)^2} + n_{1,k} \tag{2.195}$$

$$\theta_k = \arctan\left(\frac{x_{2,k} - y_{\mathrm{r}}}{x_{1,k} - x_{\mathrm{r}}}\right) + n_{2,k}, \tag{2.196}$$

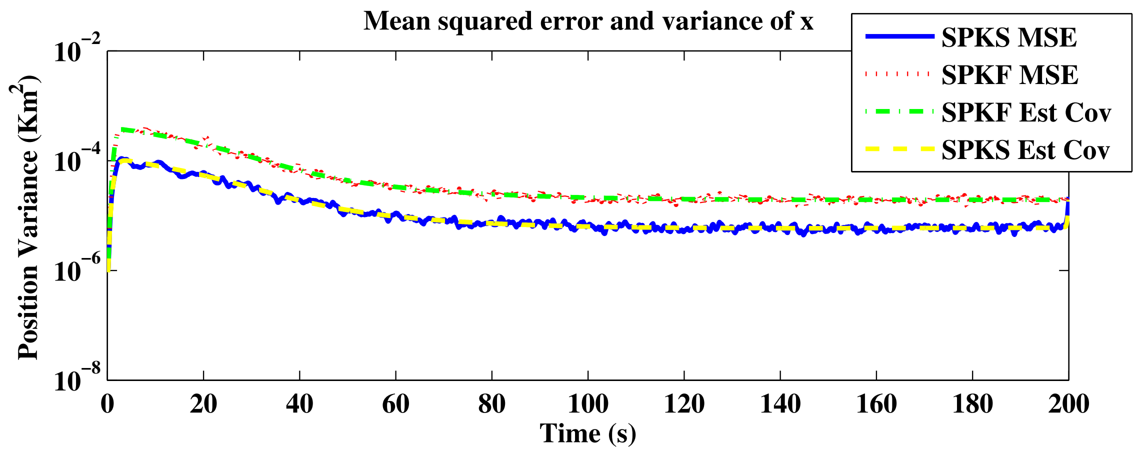
where the measurement $\boldsymbol{z}_k = \begin{bmatrix} r_k & \theta_k \end{bmatrix}$ consists of both range and bearing. A radar, which is stationed on the earth surface at x_{r} and y_{r}, observes the range and bearing at a sampling rate of 10 Hz. As can be noted, the state predictions are made at a higher rate (2 predictions per update) than the observation update. The observation noise $\boldsymbol{n}_k$ is uncorrelated zero mean white, with variances of 1 m and 17 mrd for the range and bearing respectively [23].

2.5.2 Experimental results

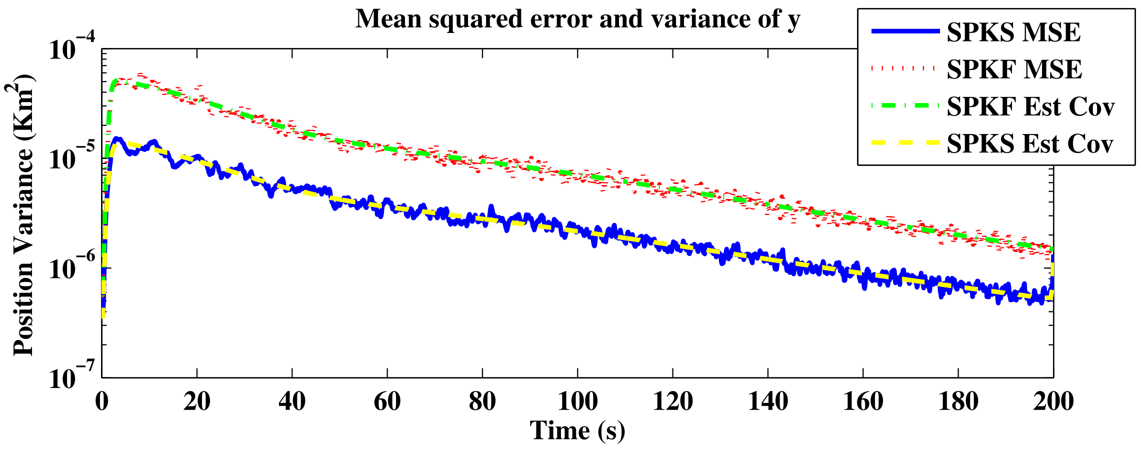
In each of the above two experiments, we show simulation results comparing our proposed SPKS approaches with the EKF, EKS, FBNL-SPKS and URTSS in terms of mean of (MSE) and standard deviation ((std)) of MSE for a *Monte-Carlo* (MC) run of *200* randomly initialized experiments. For each MC run, a different realization of both process and observation noises is generated. The MSE between the true and estimates is calculated by ensemble averaging over all *200* MC runs, which is then plotted with time. Note, the MSE is computed after the estimators have converged.

Mackey-Glass clean time series estimation

The clean and noisy Mackey-Glass time series is shown in Figure 2.6(a). Figure 2.6(b)-2.7 compares the FI-SPKS estimates with the ground truth. Figure 2.7 zooms in a section of Figure 2.6(b) in order to demonstrate how closely the FI-SPKS estimates follow the true time series. As the estimation accuracy of both the FBSL-SPKS and RTSSL-SPKS is found similar for this case, only the RTSSL-SPKS estimates are shown in the figures. Estimates obtained from the standard SPKF are also plotted on the same graph. As is clearly visible from the figures, the FI-SPKS estimates are closer to the true time series than the SPKF estimates. Figure 2.8(a) shows the MSE between the true and the



(a) 'x' position error

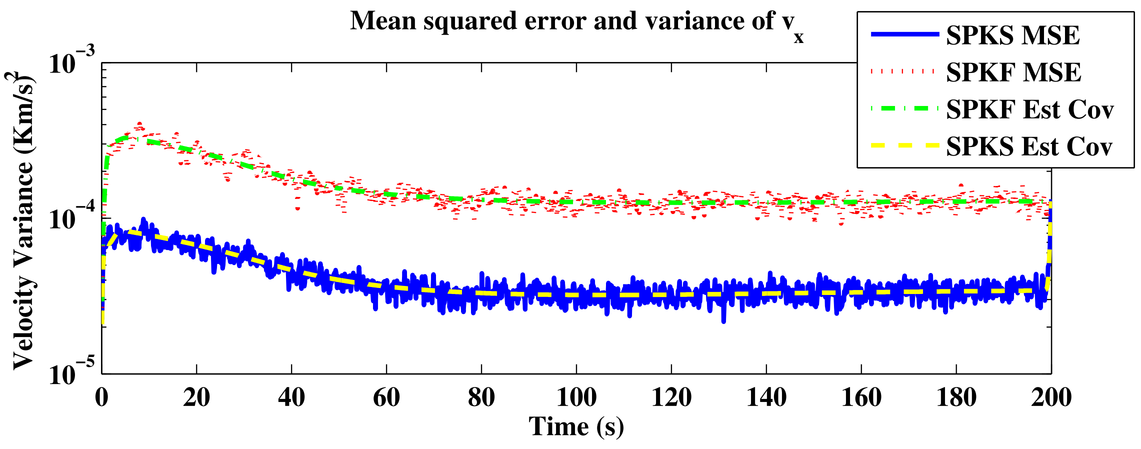


(b) 'y' position error

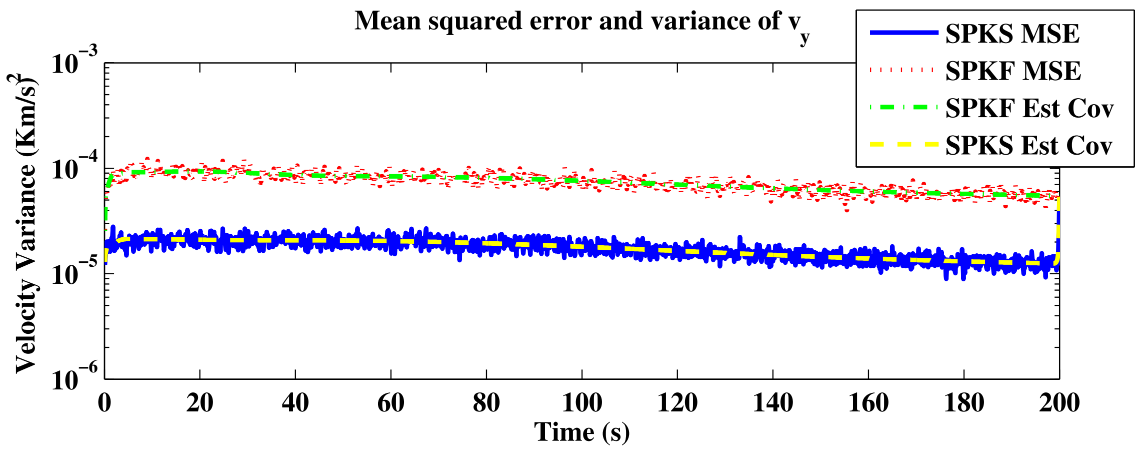
Figure 2.16: Tracking performance of the FL-SPKS algorithm for a lag $L = 40$. (a) and (b): MSE and estimation-error covariances for 2D vehicle position x_k and y_k. The above graphs demonstrate that the improvement of estimation accuracy for the FL-SPKS is related to the increasing lag L (higher L is proportional to the greater number of future measurements incorporated for the state estimation calculation). Comparing the above results with the performance of FI-SPKS, it is evident that they both generate similar level of tracking accuracy.

SPKF/RTSSL-SPKS estimated time series. It is clearly evident that the magnitude of the SPKS errors are smaller than those of the SPKFs. Figure 2.8(b) displays a specific run, where the SPKF produced a *large spike* in the MSE while the SPKS kept the estimation-error at the lower level.

Figure 2.9(a)- 2.10 demonstrate the performance accuracy of the FL-SPKS over the



(a) ‘x’ velocity error



(b) ‘y’ velocity error

Figure 2.17: Tracking performance of the FL-SPKS algorithm for a lag $L = 40$. (a) and (b): MSE and estimation-error covariances for 2D vehicle velocity v_{x_k} and v_{y_k}. As we increase the value of L, the tracking accuracy of the FL-SPKS becomes similar with that of the FI-SPKS.

SPKF. The superior accuracy of the FL-SPKS methods is clearly depicted. Note, the number of lagged states used in this example are $L = 10$. Considering that the Mackey-Glass-30 chaotic time series was sampled at every 6 s, the smoothed state estimate lags behind the current observation by 1 min. The lag $L = 10$ is chosen because we have found that the FL-SPKS obtains similar estimation accuracy with the FI-SPKS at this lag value. In other words, the performance of an offline fixed-interval smoother which works on a fixed set of predefined measurements, can be mimicked by a fixed-lag smoother with a much

Estimator	E(MSE)	std(MSE)
Filter		
EKF	1.20	0.252
SPKF	0.236	0.054
Fixed-Interval Smoother		
EKS	0.725	0.184
FBNL-SPKS	0.106	0.025
URTSS	0.099	0.024
FBSL-SPKS	0.098	0.021
RTSSL-SPKS	0.098	0.021
Fixed-Lag Smoother		
Aug-SPKS ($L = 10$)	0.120	0.025
FB-Priori-SPKS ($L = 10$)	0.121	0.024
FBSL-SPKS ($L = 10$)	0.120	0.024
RTSSL-SPKS ($L = 10$)	0.120	0.023
Aug-SPKS ($L = 30$)	0.101	0.021
FB-Priori-SPKS ($L = 30$)	0.101	0.022
FBSL-SPKS ($L = 30$)	0.101	0.021
RTSSL-SPKS ($L = 30$)	0.101	0.021

Table 2.2: Performance comparison of different estimators for the Mackey-Glass time series estimation problem. The mean of MSE (E(MSE)) and standard deviation of MSE (std(MSE)) are computed by averaging over 200 independent MC runs.

smaller set of measurements. The RTSSL-SPKS estimates are only plotted here as all the fixed-lag algorithms, including the Aug-SPKS, FB-Priori-SPKS, FBSL-SPKS and RTSSL-SPKS, perform comparably. The RTSSL-SPKS algorithm is given preference due to its superior computational efficiency and numerical advantage as described in Section 2.4.4. For example at $L = 10$, the Aug-SPKS performs $\sim$86,620 floating point operations at time k to estimate $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$. However, the fixed-lag RTSSL-SPKS algorithm requires only $\sim$2380 floating point computations in order to generate the smoothed estimate at time k.

The performance of different estimators is summarized in Table 2.2. The performance of our proposed smoothers are shown in bold. As is evident from the table, the extended Kalman smoother (EKS) not only has a worse MSE performance but the standard deviation of the MSE is also higher than the SPKS. The table further demonstrates that our proposed fixed-interval and fixed-lag smoothers perform comparably with the other existing SPKF smoothing methodologies. In contrast to the FBNL-SPKS, our proposed SPKS

Estimator	$\mathrm{E(MSE)}_x$	$\mathrm{std(MSE)}_x$	$\mathrm{E(MSE)}_{v_x}$	$\mathrm{std(MSE)}_{v_x}$
Filter				
EKF	5.95e-5	0.0023	1.58e-4	0.0034
SPKF	5.88e-5	0.0015	1.54e-4	0.0022
Fixed-Interval Smoother				
EKS	N/A	N/A	N/A	N/A
FBNL-SPKS	1.56e-5	0.0006	4.01e-5	0.0009
URTSS	1.57e-5	0.0006	4.01e-5	0.0009
FBSL-SPKS	1.56e-5	0.0006	4.00e-5	0.0009
RTSSL-SPKS	1.56e-5	0.0006	4.00e-5	0.0009
Fixed-Lag Smoother				
Aug-SPKS ($L = 10$)	1.90e-5	0.0010	5.36e-5	0.0014
FB-Priori-SPKS ($L = 10$)	1.89e-5	0.0010	5.34e-5	0.0014
FBSL-SPKS ($L = 10$)	1.89e-5	0.0009	5.34e-5	0.0014
RTSSL-SPKS ($L = 10$)	1.89e-5	0.0009	5.34e-5	0.0014
Aug-SPKS ($L = 40$)	1.63e-5	0.0007	4.03e-5	0.0011
FB-Priori-SPKS ($L = 40$)	1.63e-5	0.0007	4.03e-5	0.0011
FBSL-SPKS ($L = 40$)	1.63e-5	0.0007	4.03e-5	0.0010
RTSSL-SPKS ($L = 40$)	1.63e-5	0.0007	4.03e-5	0.0010

Table 2.3: Performance comparison of different estimators for the vehicle re-entry tracking example. The mean of MSE (E(MSE)) and standard deviation of MSE (std(MSE)) are computed by averaging over 200 independent MC runs.

methods avoid the time consuming process of learning a nonlinear backward dynamic model. Our proposed SPKS methods are more computationally and memory efficient than the URTSS, which increases each state dimension by doubling the state space. Table 2.2 also illustrates the estimation performance of the FL-SPKS for two different lag values, i.e. $L = 10$ and $L = 30$. The results demonstrate that the MSE of the FL-SPKS decreases and tends toward the FI-SPKS as we increase the lag value. However the downside is increasing computational complexity and the greater time delay between the current observation and the smoothed estimate. For example, shifting from $L = 10$ to $L = 30$ not only increases the time delay from 1 min to 3 min but also increases the number of floating point operations at time k from 2380 to 7140 for the fixed-lag RTSSL-SPKS estimator. However at $L = 30$, the computational complexity of the Aug-SPKS smoother, which performs state augmentation in order to perform smoothing, increases exponentially from 86,620 to 20,74,260 floating point operations in order to estimate $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$.

Vehicle Re-entry Tracking

Figure 2.11 displays the true vehicle re-entry path. As seen, when the vehicle moves closer to the earth's surface, its ballistic trajectory almost becomes vertical due to the increased aerodynamic drag and gravity. The performance of the SPKF and FI-SPKS in tracking the 2D position and velocity of the vehicle is depicted in Figure 2.12(a)- 2.13(b). These figures plot the estimated covariance (trace of estimation-error covariance matrix) against actual MSE between the true and estimated vehicle trajectory. Results are displayed in the same format as presented in [23] in order to demonstrate the filter consistency. While both the FBSL-SPKS and RTSSL-SPKS are applied to compute estimates, the tracking performance of the RTSSL-SPKS is only shown here. Note, the perfect alignment of MSE with its corresponding estimated covariance indicates that both the SPKS and SPKF generate consistent estimates. As is clearly visible from the figures, the FI-SPKS outperforms the SPKF for both position and velocity estimation in terms of lower MSE and lower estimated covariance.

Figure 2.14(a)- 2.15(b) demonstrate the performance accuracy of the FL-SPKS over the SPKF. The superior accuracy of the FL-SPKS methods for tracking a re-entry vehicle is clearly depicted. As before, all our proposed FL-SPKS algorithms perform comparably, and hence only the RTSSL-SPKS estimates are shown here. Note, the number of lagged states used in this example are $L = 10$. As the radar range and bearing measurements are observed at every 100 ms, the smoothed state estimate lags behind the current observation by 1 second. Comparing with the FI-SPKS estimates, the FL-SPKS generates slightly higher MSE, particularly during the initial period. The initial MSE spike for the FL-SPKS case is clearly visible in case of velocity estimation (Figure 2.15(a) and 2.15(b)). When increasing the lag from $L = 10$ to $L = 40$, (i.e. increase of time delay from 1 s to 4 s between the smoothed state and the current measurement), the FL-SPKS produces almost equal estimates with the FI-SPKS. The accuracy of the FL-SPKS estimates with state lag $L = 40$ is shown in Figure 2.16(a)-2.17(b).

Table 2.3 compares the performance of different estimators. As before, our proposed estimators are shown in bold. Note that the EKS estimates are not available. This is because the estimation-error covariance matrix became severely ill-conditioned while tracking, and hence the covariance inversion was not possible. Similar to the Mackey-Glass experiment, all the different SPKS filters perform comparably in this case. Only the results for x (x-component of position) and v_x (x-component of velocity) are shown in the table as the estimation performance for y (y-component of position) and v_y (y-component of velocity) are found to be comparable.

2.6 Discussion

In this chapter, we propose new fixed-interval and fixed-lag smoothing algorithms for the nonlinear state-space model. At the core of all the proposed smoothers lies the sigma-point Kalman filtering based Bayesian inference algorithm. Both the FI-SPKS and the FL-SPKS smoother equations are derived from the first principles and detailed step by step mathematical formulations are provided.

The FI-SPKS consists of two smoothers, namely the FBSL-SPKS and the RTSSL-SPKS, which make use of the forward-backward and the RTS methods respectively to derive its formulation. Both the FBSL-SPKS and the RTSSL-SPKS are only suitable for the offline estimation as they generally operate on a fixed set of measurements. The FBSL-SPKS consists of three components: a forward filter, a separate backward filter and a smoother. The forward filter is the SPKF, which operates on the original nonlinear state space from $k = 1$ to $k = N$ to derive the forward estimates. In addition to the state estimates, the forward filter also derives the WSLR coefficients at each k in order to form a statistically linearized state space. The backward filter is an information filter, which computes state estimates by operating from $k = N$ to $k = 1$ using the pseudo-linearized state space. The estimates of the two filters are then statistically combined to generate the smoothed estimates. The disadvantages of the FBSL-SPKS include higher computational complexity, i.e. $O\left(NM^3\right)$ for each component, and the need for greater memory as the entire forward estimation results need to be saved for the future use. In addition, the backward filter implicitly assumes that the inverse of the forward dynamical model exists. Although computationally inefficient, the FBSL-SPKS algorithm can be appealing to the practitioners because it is conceptually simpler, easier to understand and perhaps the most straightforward smoothing algorithm. Moreover the availability of the independent backward estimates using the future observations, can generate interests for implementing the FBSL-SPKS in certain applications.

The RTSSL-SPKS follows the Rauch-Tung-Striebel approach, which is made of two components: a forward filter and a backward smoother algorithm which incorporates both the backward filter and the smoother. The forward filter is the standard SPKF, which in addition to generating the state estimates and the estimation error covariances also computes the statistical linearization parameters for the nonlinear dynamic model. The backward smoothing pass linearly combines a correction term with the forward filtering results at each k to obtain the final smoothed states. Comparing to the FBSL-SPKS, we have found an almost identical estimation accuracy for the RTSSL-SPKS but the biggest advantage of the RTSSL-SPKS is that it is computationally cheaper to implement. As it avoids running an independent backward filter, its implementation saves an order of $O\left(NM^3\right)$ computations. The absence of a separate backward filter also eliminates the need for computing the inverse of the forward dynamic model, which in effect avoids the numerical problem in case the state dynamics is non-invertible. One drawback of the RTSSL-SPKS algorithm is that it needs more memory than the FBSL-SPKS as both the prior and the posterior state estimates of the forward filter are required to perform smoothing. The higher computational efficiency and the numerical advantage should make the RTSSL-SPKS as an attractive choice for the SPKS based nonlinear fixed-interval smoothing.

In practice, the demand for larger memory may pose a hinderance in implementing the FI-SPKS. Moreover, the fixed-interval method is an offline estimator, which does not collect new observations while performing the smoothing operation. In contrast, the estimated state in the FL-SPKS lags behind the current measurement by L time, where the time-delay L is an application specific constant. The Aug-SPKS algorithm is the most simplistic representation of the FL-SPKS, where an augmented state combining the current and the L past states is simultaneously estimated at each time k. The computational complexity i.e. $O\left(NM^3L^3\right)$ and the size of each state dimension i.e. ML are too prohibitive to implement this smoother in practice. In order to overcome the drawbacks of the Aug-SPKS algorithm, we have proposed three FL-SPKS algorithms, namely the FB-Priori-SPKS, FBSL-SPKS and RTSSL-SPKS, which perform a series of sequential smoothing within overlapping time-windows. All the three FL-SPKS methods reduce the computational order to $O\left(NLM^3\right)$ and also the estimated state dimension to M. The fixed-lag FBSL-SPKS and the RTSSL-SPKS both leverage the benefits and the drawbacks of their fixed-interval counterparts and we can conclude that the RTSSL-SPKS is the most efficient in terms of computation among all the FL-SPKS methods.

The performance of both our fixed-interval and fixed-lag smoothing formulations have been demonstrated in two examples, i.e. Mackey Glass time series estimation and vehicle re-entry tracking, and were also compared with the other methodologies in terms of estimation accuracy. As can be seen, the proposed SPKS algorithms clearly outperform the EKF/EKS/SPKF based approaches and also perform comparably with the existing fixed-interval SPKF based smoothers which are time consuming to learn and also bears much higher computational load.

2.7 Appendix 1

Lemma 2.7.1. *if $\boldsymbol{A} = \boldsymbol{B}^{-1} + \boldsymbol{C}\boldsymbol{D}^{-1}\boldsymbol{C}^T$, where $\boldsymbol{A}$, $\boldsymbol{D}$ are invertible square matrices and $\boldsymbol{B}$ and $\boldsymbol{C}$ matrices may or may not be square. Then by applying matrix inversion Lemma we obtain,*

$$\boldsymbol{A}^{-1} = \boldsymbol{B} - \boldsymbol{B}\boldsymbol{C}\left[\boldsymbol{D} + \boldsymbol{C}^T\boldsymbol{B}\boldsymbol{C}\right]^{-1}\boldsymbol{C}^T\boldsymbol{B} \tag{2.197}$$

Lemma 2.7.2.

- $\boldsymbol{P}_k^b = \left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k^{b-}\left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{K}_k\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\boldsymbol{K}_k^T$
- $\boldsymbol{K}_k = \boldsymbol{P}_k^{b-}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k}\boldsymbol{P}_k^{b-}\boldsymbol{A}_{h,k}^T\right)^{-1}$

Proof*:*

Let $\boldsymbol{\delta}_{x,k}$ denote as the state estimation error at time index k

$$\begin{aligned}\boldsymbol{\delta}_{x,k} &= \boldsymbol{x}_k^b - \hat{\boldsymbol{x}}_k^b \\ &= \boldsymbol{x}_k^b - \hat{\boldsymbol{x}}_k^{b-} - \boldsymbol{K}_k\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^{b-} - \boldsymbol{b}_{h,k}\right) \\ &= \left(\boldsymbol{x}_k^b - \hat{\boldsymbol{x}}_k^{b-}\right) - \boldsymbol{K}_k\left(\boldsymbol{A}_{h,k}\boldsymbol{x}_k^b + \boldsymbol{b}_{h,k} + \boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k} - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^{b-} - \boldsymbol{b}_{h,k}\right) \quad [\mathit{from}(2.29)] \\ &= \left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\left(\boldsymbol{x}_k^b - \hat{\boldsymbol{x}}_k^{b-}\right) - \boldsymbol{K}_k\left(\boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k}\right) \end{aligned} \tag{2.198}$$

The estimation error covariance can be defined as:

$$\begin{aligned}\boldsymbol{P}_k^b &= \mathrm{E}\left(\boldsymbol{\delta}_{x,k}\boldsymbol{\delta}_{x,k}^T\right) \\ &= \mathrm{E}\Big(\left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\left(\boldsymbol{x}_k^b - \hat{\boldsymbol{x}}_k^{b-}\right) - \boldsymbol{K}_k\left(\boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k}\right)\Big) \\ &\quad \Big(\left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\left(\boldsymbol{x}_k^b - \hat{\boldsymbol{x}}_k^{b-}\right) - \boldsymbol{K}_k\left(\boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k}\right)\Big)^T && (2.199) \\ &= \left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k^{b-}\left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{K}_k\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)\boldsymbol{K}_k^T && (2.200)\end{aligned}$$

Here it is assumed that expected values of $\boldsymbol{n}_k$ and $\boldsymbol{\epsilon}_{h,k}$ are zero. Now the objective is to derive the gain function $\boldsymbol{K}_k$ by minimizing the estimation error $\boldsymbol{P}_k^b$ i.e. by setting $\frac{\partial \boldsymbol{P}_k^b}{\partial \boldsymbol{K}_k} = \boldsymbol{0}$.

$$\begin{aligned}\frac{\partial \boldsymbol{P}_k^b}{\partial \boldsymbol{K}_k} &= 2\left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k^{b-}\left(-\boldsymbol{A}_{h,k}^T\right) + 2\boldsymbol{K}_k\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right) \\ &= \boldsymbol{0}\end{aligned} \tag{2.201}$$

Hence we get,

$$\begin{aligned}\boldsymbol{K}_k\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k}\boldsymbol{P}_k^{b-}\boldsymbol{A}_{h,k}^T\right) &= \boldsymbol{P}_k^{b-}\boldsymbol{A}_{h,k}^T \\ \boldsymbol{K}_k &= \boldsymbol{P}_k^{b-}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k}\boldsymbol{P}_k^{b-}\boldsymbol{A}_{h,k}^T\right)^{-1}\end{aligned} \tag{2.202}$$

Lemma 2.7.3. • $\boldsymbol{K}_k = \boldsymbol{P}_k^b \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}$

Proof*:*
Starting with Equation (2.202) *and multiplying* $\boldsymbol{I} = \boldsymbol{P}_k^b \left(\boldsymbol{P}_k^b\right)^{-1}$ *on the right hand side,*

$$\boldsymbol{K}_k = \boldsymbol{P}_k^b \left(\boldsymbol{P}_k^b\right)^{-1} \boldsymbol{P}_k^{b-} \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k} \boldsymbol{P}_k^{b-} \boldsymbol{A}_{h,k}^T\right)^{-1} \tag{2.203}$$

Substituting $\left(\boldsymbol{P}_k^b\right)^{-1}$ *from* (2.79) *and simplifying,*

$$\begin{aligned}
\boldsymbol{K}_k &= \boldsymbol{P}_k^b \left(\boldsymbol{A}_{h,k}^T + \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \boldsymbol{A}_{h,k} \boldsymbol{P}_k^{b-} \boldsymbol{A}_{h,k}^T\right) \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k} \boldsymbol{P}_k^{b-} \boldsymbol{A}_{h,k}^T\right)^{-1} \\
&= \boldsymbol{P}_k^b \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \left(\left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right) + \boldsymbol{A}_{h,k} \boldsymbol{P}_k^{b-} \boldsymbol{A}_{h,k}^T\right) \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{A}_{h,k} \boldsymbol{P}_k^{b-} \boldsymbol{A}_{h,k}^T\right)^{-1} \\
&= \boldsymbol{P}_k^b \boldsymbol{A}_{h,k}^T \left(\boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}
\end{aligned} \tag{2.204}$$

Lemma 2.7.4. • $\boldsymbol{P}_{k+1}^b = \left[\left(\boldsymbol{P}_{k+1}^s\right)^{-1} - \left(\boldsymbol{P}_{k+1}^-\right)^{-1}\right]^{-1}$

Proof*:*
Similar to (2.79) *we can also get the covariance update for the forward filter using the statistically linearized parameters,*

$$\begin{aligned}
\boldsymbol{P}_{k+1} &= \left[\left(\boldsymbol{P}_{k+1}^-\right)^{-1} + \boldsymbol{A}_{h,k+1}^T \left(\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right)^{-1} \boldsymbol{A}_{h,k+1}\right]^{-1} \\
\boldsymbol{A}_{h,k+1}^T \left(\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right)^{-1} \boldsymbol{A}_{h,k+1} &= \left(\boldsymbol{P}_{k+1}\right)^{-1} - \left(\boldsymbol{P}_{k+1}^-\right)^{-1}
\end{aligned} \tag{2.205}$$

Substituting (2.205) *into* (2.79) *and replacing* $k = k+1$*,*

$$\begin{aligned}
\boldsymbol{P}_{k+1}^b &= \left[\left(\boldsymbol{P}_{k+1}^{b-}\right)^{-1} + \left(\boldsymbol{P}_{k+1}\right)^{-1} - \left(\boldsymbol{P}_{k+1}^-\right)^{-1}\right]^{-1} \\
&= \left[\left(\boldsymbol{P}_{k+1}^s\right)^{-1} - \left(\boldsymbol{P}_{k+1}^-\right)^{-1}\right]^{-1} \quad [\textit{from } (2.90)]
\end{aligned} \tag{2.206}$$

Lemma 2.7.5. • $\boldsymbol{A}_{f,k}^{-1} \acute{\boldsymbol{Q}}_k \boldsymbol{A}_{f,k}^{-T} = \boldsymbol{A}_{f,k}^{-1} \boldsymbol{P}_{k+1}^- \boldsymbol{A}_{f,k}^{-T} - \boldsymbol{P}_k$

Proof*:*

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{A}_{f,k} \boldsymbol{P}_k \boldsymbol{A}_{f,k}^T + \boldsymbol{G}_{f,k} \left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right) \boldsymbol{G}_{f,k}^T \tag{2.207}$$

Assuming $\acute{\boldsymbol{Q}}_k = \boldsymbol{G}_{f,k} \left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right) \boldsymbol{G}_{f,k}^T$

$$\begin{aligned}
\acute{\boldsymbol{Q}}_k &= \boldsymbol{P}_{k+1}^- - \boldsymbol{A}_{f,k} \boldsymbol{P}_k \boldsymbol{A}_{f,k}^T \\
\boldsymbol{A}_{f,k}^{-1} \acute{\boldsymbol{Q}}_k \boldsymbol{A}_{f,k}^{-T} &= \boldsymbol{A}_{f,k}^{-1} \boldsymbol{P}_{k+1}^- \boldsymbol{A}_{f,k}^{-T} - \boldsymbol{P}_k
\end{aligned} \tag{2.208}$$

Lemma 2.7.6. • $\boldsymbol{P}^b_{k+1} = \left(\boldsymbol{P}^-_{k+1} + \boldsymbol{P}^b_{k+1}\right) \boldsymbol{S}^-_{k+1}\boldsymbol{P}^s_{k+1}$

Proof*:*
From (2.90)*,*

$$\begin{aligned}\boldsymbol{P}^b_{k+1} &= \boldsymbol{P}^s_{k+1}\left[\boldsymbol{I} + \boldsymbol{P}^b_{k+1}\left(\boldsymbol{P}^-_{k+1}\right)^{-1}\right]\\ &= \boldsymbol{P}^-_{k+1}\left(\boldsymbol{P}^-_{k+1}\right)^{-1}\boldsymbol{P}^s_{k+1} + \boldsymbol{P}^b_{k+1}\left(\boldsymbol{P}^-_{k+1}\right)^{-1}\boldsymbol{P}^s_{k+1}\\ &= \left(\boldsymbol{P}^-_{k+1} + \boldsymbol{P}^b_{k+1}\right)\boldsymbol{S}^-_{k+1}\boldsymbol{P}^s_{k+1} \end{aligned} \tag{2.209}$$

Lemma 2.7.7. • $\boldsymbol{P}^-_{k+1} + \boldsymbol{P}^b_{k+1} = \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_k + \boldsymbol{P}^{b-}_k\right)\boldsymbol{A}^T_{f,k}$

Proof*:*
$\boldsymbol{P}_k$ *and* $\boldsymbol{P}^{b-}_k$ *can be obtained from Lemma 2.7.5 and* (2.67) *respectively. Adding them,*

$$\begin{aligned}\boldsymbol{P}_k + \boldsymbol{P}^{b-}_k &= \boldsymbol{A}^{-1}_{f,k}\left(\boldsymbol{P}^-_{k+1} + \boldsymbol{P}^b_{k+1}\right)\boldsymbol{A}^{-T}_{f,k}\\ \boldsymbol{P}^-_{k+1} + \boldsymbol{P}^b_{k+1} &= \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_k + \boldsymbol{P}^{b-}_k\right)\boldsymbol{A}^T_{f,k}\end{aligned} \tag{2.210}$$

Lemma 2.7.8. • $\hat{\boldsymbol{x}}^s_{k+1} = \boldsymbol{P}^s_{k+1}\boldsymbol{S}_{k+1}\hat{\boldsymbol{x}}^-_{k+1} - \boldsymbol{P}^s_{k+1}\boldsymbol{A}^T_{h,k+1}\acute{\boldsymbol{R}}^{-1}_{k+1}\boldsymbol{A}_{h,k+1}\hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{P}^s_{k+1}\hat{\boldsymbol{y}}_{k+1}$

Proof*:*
Assuming $\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1} = \acute{\boldsymbol{R}}_{k+1}$ *and* $\acute{\boldsymbol{z}}_{k+1} = \boldsymbol{z}_{k+1} - \boldsymbol{b}_{h,k+1}$*,*

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{K}_{k+1}\left(\acute{\boldsymbol{z}}_{k+1} - \boldsymbol{A}_{h,k+1}\hat{\boldsymbol{x}}^-_{k+1}\right) \tag{2.211}$$

Substituting (2.211) *and* (2.81) *into the relevant portions of* (2.94)*,*

$$\begin{aligned}\hat{\boldsymbol{x}}^s_{k+1} &= \boldsymbol{P}^s_{k+1}\boldsymbol{S}_{k+1}\hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{P}^s_{k+1}\boldsymbol{S}_{k+1}\boldsymbol{K}_{k+1}\left(\acute{\boldsymbol{z}}_{k+1} - \boldsymbol{A}_{h,k+1}\hat{\boldsymbol{x}}^-_{k+1}\right)\\ &\quad + \boldsymbol{P}^s_{k+1}\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{P}^s_{k+1}\boldsymbol{A}^T_{h,k+1}\acute{\boldsymbol{R}}^{-1}_{k+1}\acute{\boldsymbol{z}}_{k+1}\end{aligned} \tag{2.212}$$

Replacing $\boldsymbol{K}_{k+1} = \boldsymbol{P}_{k+1}\boldsymbol{A}^T_{h,k+1}\acute{\boldsymbol{R}}^{-1}_{k+1}$ *into* (2.212) *and simplifying we obtain,*

$$\hat{\boldsymbol{x}}^s_{k+1} = \boldsymbol{P}^s_{k+1}\boldsymbol{S}_{k+1}\hat{\boldsymbol{x}}^-_{k+1} - \boldsymbol{P}^s_{k+1}\boldsymbol{A}^T_{h,k+1}\acute{\boldsymbol{R}}^{-1}_{k+1}\boldsymbol{A}_{h,k+1}\hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{P}^s_{k+1}\hat{\boldsymbol{y}}_{k+1} \tag{2.213}$$

Lemma 2.7.9. • $\boldsymbol{S}^-_k = \boldsymbol{A}^T_{f,k}\left[\acute{\boldsymbol{Q}}^{-1}_k - \acute{\boldsymbol{Q}}^{-1}_k\left(\boldsymbol{S}_{k+1} + \acute{\boldsymbol{Q}}^{-1}_k\right)^{-1}\acute{\boldsymbol{Q}}^{-1}_k\right]\boldsymbol{A}_{f,k}$

Proof*:*
From (2.67)*,*

$$\boldsymbol{S}^-_k = \boldsymbol{A}^T_{f,k}\left(\boldsymbol{S}^{-1}_{k+1} + \boldsymbol{G}_{f,k}\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right)\boldsymbol{G}^T_{f,k}\right)^{-1}\boldsymbol{A}_{f,k} \tag{2.214}$$

Applying matrix inversion Lemma on (2.214) *and substituting*

$$\acute{\boldsymbol{Q}}_k = \boldsymbol{G}_{f,k}\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right)\boldsymbol{G}_{f,k}^T, \tag{2.215}$$

we obtain,

$$\boldsymbol{S}_k^- = \boldsymbol{A}_{f,k}^T\left[\acute{\boldsymbol{Q}}_k^{-1} - \acute{\boldsymbol{Q}}_k^{-1}\left(\boldsymbol{S}_{k+1} + \acute{\boldsymbol{Q}}_k^{-1}\right)^{-1}\acute{\boldsymbol{Q}}_k^{-1}\right]\boldsymbol{A}_{f,k} \tag{2.216}$$

Lemma 2.7.10. • $\hat{\boldsymbol{x}}_k = \boldsymbol{A}_{f,k}^{-1}\left(\hat{\boldsymbol{x}}_{k+1}^- - \boldsymbol{b}_{f,k}\right)$

Proof*:*

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1}^- &= \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k + \boldsymbol{b}_{f,k}\\ \hat{\boldsymbol{x}}_k &= \boldsymbol{A}_{f,k}^{-1}\left(\hat{\boldsymbol{x}}_{k+1}^- - \boldsymbol{b}_{f,k}\right)\end{aligned} \tag{2.217}$$

Lemma 2.7.11. • $\hat{\boldsymbol{x}}_k^s - \hat{\boldsymbol{x}}_k = \boldsymbol{P}_k^s\hat{\boldsymbol{y}}_k^- - \boldsymbol{P}_k\boldsymbol{S}_k^-\left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^-\right)^{-1}\hat{\boldsymbol{x}}_k$

Proof*:*
Define $\hat{\boldsymbol{x}}_{k+1}^s$ *from Equation* (2.95)

$$\begin{aligned}\hat{\boldsymbol{x}}_k^s &= \left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^-\right)^{-1}\hat{\boldsymbol{x}}_k + \boldsymbol{P}_k^s\hat{\boldsymbol{y}}_k^-\\ \hat{\boldsymbol{x}}_k^s - \hat{\boldsymbol{x}}_k &= \left[\left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^-\right)^{-1} - \boldsymbol{I}\right]\hat{\boldsymbol{x}}_k + \boldsymbol{P}_k^s\hat{\boldsymbol{y}}_k^-\end{aligned}$$

Now substituting $\boldsymbol{I}$ *with* $\left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^-\right)\left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^-\right)^{-1}$ *at the R.H.S. of the above Equation,*

$$\hat{\boldsymbol{x}}_k^s - \hat{\boldsymbol{x}}_k = -\boldsymbol{P}_k\boldsymbol{S}_k^-\left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^-\right)^{-1}\hat{\boldsymbol{x}}_k + \boldsymbol{P}_k^s\hat{\boldsymbol{y}}_k^- \tag{2.218}$$

Now rearranging the terms at the R.H.S. of the above Equation

$$\hat{\boldsymbol{x}}_k^s - \hat{\boldsymbol{x}}_k = \boldsymbol{P}_k^s\hat{\boldsymbol{y}}_k^- - \boldsymbol{P}_k\boldsymbol{S}_k^-\left(\boldsymbol{I} + \boldsymbol{P}_k\boldsymbol{S}_k^-\right)^{-1}\hat{\boldsymbol{x}}_k \tag{2.219}$$

2.8 Appendix 2

2.8.1 FB-Priori-SPKS Derivations

In the following, the smoothing recursion of the FB-Priori-SPKS, where the a priori KF Equations (2.157)-(2.158) are applied on the augmented statistically linearized state space, is derived as follows:

From Equations (2.157) and (2.158), we can write the a priori KF for the augmented system of Equations (2.139) and (2.140) as follows:

$$\begin{bmatrix} \hat{\boldsymbol{x}}_{k+1}^- \\ \hat{\boldsymbol{x}}_{k,k} \\ \vdots \\ \hat{\boldsymbol{x}}_{k-L,k} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{f,k} & 0 & \cdots & 0 \\ \boldsymbol{I} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{I} & 0 \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}}_k^- \\ \hat{\boldsymbol{x}}_{k-1,k-1} \\ \vdots \\ \hat{\boldsymbol{x}}_{k-L-1,k-1} \end{bmatrix} + \tilde{\boldsymbol{L}}_k \left(\boldsymbol{z}_k - \begin{bmatrix} \boldsymbol{A}_{h,k} & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}}_k^- \\ \hat{\boldsymbol{x}}_{k-1,k-1} \\ \vdots \\ \hat{\boldsymbol{x}}_{k-L-1,k-1} \end{bmatrix} - \boldsymbol{b}_{h,k} \right) + \begin{bmatrix} \boldsymbol{b}_{f,k} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.220}$$

$$\begin{bmatrix} \boldsymbol{P}_{k+1}^{0,0} & \cdots & \left(\boldsymbol{P}_{k+1}^{0,L+1}\right)^T \\ \vdots & \ddots & \vdots \\ \boldsymbol{P}_{k+1}^{0,L+1} & \cdots & \boldsymbol{P}_{k+1}^{L+1,L+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{f,k} & 0 & \cdots & 0 \\ \boldsymbol{I} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{I} & 0 \end{bmatrix} \times$$
$$\begin{bmatrix} \boldsymbol{P}_k^{0,0} & \left(\boldsymbol{P}_k^{0,1}\right)^T & \cdots & \left(\boldsymbol{P}_k^{0,L+1}\right)^T \\ \boldsymbol{P}_k^{0,1} & \boldsymbol{P}_k^{1,1} & \cdots & \left(\boldsymbol{P}_k^{1,L+1}\right)^T \\ \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{P}_k^{0,L+1} & \boldsymbol{P}_k^{1,L+1} & \cdots & \left(\boldsymbol{P}_k^{L+1,L+1}\right)^T \end{bmatrix} \left(\begin{bmatrix} \boldsymbol{A}_{f,k}^T & \boldsymbol{I} & \cdots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{I} \\ 0 & \cdots & \cdots & 0 \end{bmatrix} - \begin{bmatrix} \boldsymbol{A}_{h,k}^T \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tilde{\boldsymbol{L}}_k^T \right) +$$
$$\begin{bmatrix} \boldsymbol{G}_{f,k}\boldsymbol{Q}_k\boldsymbol{G}_{f,k}^T & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{G}_{f,k}\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\boldsymbol{G}_{f,k}^T & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix}, \tag{2.221}$$

where the augmented Kalman gain $\tilde{\boldsymbol{L}}_k$ can be defined as

$$\tilde{\boldsymbol{L}}_k = \begin{bmatrix} \boldsymbol{L}_{k,0} \\ \boldsymbol{L}_{k,1} \\ \vdots \\ \boldsymbol{L}_{k,L+1} \end{bmatrix}. \tag{2.222}$$

The augmented $\tilde{\boldsymbol{L}}_k$ can be expanded as

$$\tilde{\boldsymbol{L}}_k = \begin{bmatrix} \boldsymbol{A}_{f,k} & 0 & \cdots & 0 \\ \boldsymbol{I} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{I} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_k^{0,0} & \left(\boldsymbol{P}_k^{0,1}\right)^T & \cdots & \left(\boldsymbol{P}_k^{0,L+1}\right)^T \\ \boldsymbol{P}_k^{0,1} & \boldsymbol{P}_k^{1,1} & \cdots & \left(\boldsymbol{P}_k^{1,L+1}\right)^T \\ \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{P}_k^{0,L+1} & \boldsymbol{P}_k^{1,L+1} & \cdots & \left(\boldsymbol{P}_k^{L+1,L+1}\right)^T \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{h,k}^T \\ 0 \\ \vdots \\ 0 \end{bmatrix} \times$$
$$\left(\begin{bmatrix} \boldsymbol{A}_{h,k} & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_k^{0,0} & \left(\boldsymbol{P}_k^{0,1}\right)^T & \cdots & \left(\boldsymbol{P}_k^{0,L+1}\right)^T \\ \boldsymbol{P}_k^{0,1} & \boldsymbol{P}_k^{1,1} & \cdots & \left(\boldsymbol{P}_k^{1,L+1}\right)^T \\ \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{P}_k^{0,L+1} & \boldsymbol{P}_k^{1,L+1} & \cdots & \left(\boldsymbol{P}_k^{L+1,L+1}\right)^T \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{h,k}^T \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} \right)^{-1} \tag{2.223}$$

After simplifying the expression for $\tilde{\boldsymbol{L}}_k$ can be reduced to

$$\tilde{\boldsymbol{L}}_k = \begin{bmatrix} \boldsymbol{A}_{f,k}\boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T \\ \boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T \\ \vdots \\ \boldsymbol{P}_k^{0,N}\boldsymbol{A}_{h,k}^T \end{bmatrix} \left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}, \tag{2.224}$$

where $\boldsymbol{P}_k^{i,i}, 0 \leq i \leq L$ and $\boldsymbol{P}_k^{i,j}, i \neq j$ are defined as the estimation-error covariances for each state and the cross-covariances between states respectively in the augmented state vector. The individual state estimates shown in (2.220) can be defined as:

$$\hat{\boldsymbol{x}}_{k-i,k} = E\left[\boldsymbol{x}_{k-i}|\boldsymbol{z}_{1:k}\right] \tag{2.225}$$

for $1 \leq i \leq L+1$. Comparing (2.222) and (2.224), we obtain

$$\boldsymbol{L}_{k,0} = \boldsymbol{A}_{f,k}\boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1} \tag{2.226}$$

$$\boldsymbol{L}_{k,i} = \boldsymbol{P}_k^{0,i-1}\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}, 1 \leq i \leq L+1 \tag{2.227}$$

Now (2.221) can be further simplified by substituting $\tilde{\boldsymbol{L}}_k$ from (2.224)

$$\begin{bmatrix} \boldsymbol{P}_{k+1}^{0,0} & \cdots & \left(\boldsymbol{P}_{k+1}^{0,L+1}\right)^T \\ \vdots & \ddots & \vdots \\ \boldsymbol{P}_{k+1}^{0,L+1} & \cdots & \boldsymbol{P}_{k+1}^{L+1,L+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{f,k}\boldsymbol{P}_k^{0,0} & \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_k^{0,1}\right)^T & \cdots & \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_k^{0,L+1}\right)^T \\ \boldsymbol{P}_k^{0,0} & \left(\boldsymbol{P}_k^{0,1}\right)^T & \cdots & \left(\boldsymbol{P}_k^{0,L+1}\right)^T \\ \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{P}_k^{0,L} & \boldsymbol{P}_k^{1,L} & \cdots & \left(\boldsymbol{P}_k^{L,L+1}\right)^T \end{bmatrix} \times$$

$$\begin{bmatrix} \boldsymbol{A}_{f,k}^T & \boldsymbol{I} & \cdots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{I} \\ 0 & \cdots & \cdots & 0 \end{bmatrix} - \begin{bmatrix} \boldsymbol{A}_{f,k}\boldsymbol{P}_k^{0,0} & \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_k^{0,1}\right)^T & \cdots & \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_k^{0,L+1}\right)^T \\ \boldsymbol{P}_k^{0,0} & \left(\boldsymbol{P}_k^{0,1}\right)^T & \cdots & \left(\boldsymbol{P}_k^{0,L+1}\right)^T \\ \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{P}_k^{0,L} & \boldsymbol{P}_k^{1,L} & \cdots & \left(\boldsymbol{P}_k^{L,L+1}\right)^T \end{bmatrix} \times$$

$$\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k^{0,0}\boldsymbol{A}_{h,k}^T + \boldsymbol{R}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right)^{-1}\boldsymbol{A}_{h,k}\begin{bmatrix} \boldsymbol{P}_k^{0,0}\boldsymbol{A}_{f,k}^T & \boldsymbol{P}_k^{0,0} & \cdots & \boldsymbol{P}_k^{0,L} \\ 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} +$$

$$\begin{bmatrix} \boldsymbol{G}_{f,k}\boldsymbol{Q}_k\boldsymbol{G}_{f,k}^T & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{G}_{f,k}\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\boldsymbol{G}_{f,k}^T & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix}. \tag{2.228}$$

Equating the first column and the diagonal elements of the left and right hand side

$$\boldsymbol{P}_{k+1}^{0,0} = \boldsymbol{A}_{f,k}\boldsymbol{P}_k^{0,0}\left(\boldsymbol{A}_{f,k} - \boldsymbol{L}_{k,0}\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{G}_{f,k}\boldsymbol{Q}_k\boldsymbol{G}_{f,k}^T + \boldsymbol{G}_{f,k}\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\boldsymbol{G}_{f,k}^T \tag{2.229}$$

$$\boldsymbol{P}_{k+1}^{0,i} = \boldsymbol{P}_k^{0,i-1}\left(\boldsymbol{A}_{f,k} - \boldsymbol{L}_{k,0}\boldsymbol{A}_{h,k}\right)^T \tag{2.230}$$

$$\boldsymbol{P}_{k+1}^{i,i} = \boldsymbol{P}_k^{i-1,i-1} - \boldsymbol{P}_k^{0,i-1}\boldsymbol{A}_{h,k}^T\boldsymbol{L}_{k,i}^T\boldsymbol{A}_{f,k}^T, \tag{2.231}$$

$$\tag{2.232}$$

for $1 \le i \le L+1$.

Similarly by expanding (2.220), we can obtain the following set of equations

$$\hat{\boldsymbol{x}}_{k+1}^- = \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k^- + \boldsymbol{L}_{k,0}\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^- - \boldsymbol{b}_{h,k}\right) + \boldsymbol{b}_{f,k} \tag{2.233}$$

$$\hat{\boldsymbol{x}}_{k+1-i,k} = \hat{\boldsymbol{x}}_{k+2-i,k} + \boldsymbol{L}_{k,i}\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k^- - \boldsymbol{b}_{h,k}\right), \tag{2.234}$$

for $1 \le i \le L+1$.

Chapter 3

A Tag-Based Approach to Unobtrusive Indoor Tracking Using RSSI

3.1 Overview

In the previous chapter, we developed a new smoothing scheme for the nonlinear system using the SPKF based approach. Suitable fixed-interval and fixed-lag sigma-point Kalman smoothing (SPKS) algorithms are derived and evaluated using the Mackey-Glass noisy time series and re-entry vehicle tracking examples. In this chapter we apply the proposed SPKS algorithms into a real world indoor tracking framework where our task is to locate and track a user in an indoor environment. Specifically, the objective is to evaluate the feasibility of building an indoor location tracking system that is cost effective for large scale deployments, can operate over existing Wi-Fi networks, and can provide flexibility to accommodate new sensor observations as they become available.

This chapter is organized as follows. Section 3.2 starts with a survey of commercial indoor tracking systems and research prototypes including their architecture, sensor platforms and positioning algorithms. It then introduces our SPKS based Bayesian inference algorithm for location tracking and summarizes the used sensor modalities. Section 3.3 discusses the Bayesian framework and details the dynamic and observation models used in our SPKS framework. Section 3.4 examines the fixed-interval and fixed-lag based SPKS algorithms implemented in our tracking system. Experimental results are given in Section 3.5, and finally discussion and conclusion are presented in Section 3.6.

3.2 Introduction And Related Work

Location and context-aware technologies play a critical role in emerging next generation mobile applications. Example goals of these applications range from tracking assets within large warehouses, monitoring people inside assisted living communities, to adapting user interfaces based on location and activity. Key to each application is the ability to accurately localize and track an individual or asset. Explicit positioning sensors based on GPS work worldwide and can sometimes achieve centimeter level accuracy. However, GPS generally requires a direct view to several satellites, resulting in limited performance for indoor environments. Development of non-GPS based solutions are thus of great interest for indoor use based on both existing signals and hardware, as well as new systems and sensor modalities. Additional design constraints pose significant challenges for development of such systems, including calibration overhead, user privacy, and the high variability of wireless channels.

3.2.1 Existing Indoor Tracking Systems: Hardware Description

A number of commercial systems and research prototypes currently exist for indoor localization. Systems typically use infra-red (IR), ultra-sound, or radio-frequency (RF) sensors [40–43]. Although they show potential for indoor tracking, each has its own limitations. The *Active Badge* System is one example of an early location-aware application [40]. The person to be tracked carries a small tag or "active badge", which emits a unique IR code every 15 seconds (s). A network of sensors pre-placed around the building pick up the periodic IR waveforms and a central "master-station" processes the data and triangulates the individual's location. Poor IR scalability and high maintenance overheads are some of the drawbacks faced by this system. The *Cricket* system places multiple "beacons" at several locations within the indoor environment which concurrently transmit RF and ultrasonic pulses. The person being tracked carries a listening device which uses time of flight (TOF) difference between RF and ultrasonic pulses in order to determine the distance to the beacons. Based on the TOF difference between multiple beacons, the closest beacon is inferred. Although Cricket improves accuracy and stability, high maintenance and calibration requirements require significant effort to use in practice. A commercially available system by *Sonitor* [44] also uses ultrasonic sensors for tracking a person. The user carries a small tag that emits its identification number via ultrasonics. Detectors scattered throughout the environment receives this information and triangulates the user. An advantage of this ultrasound based system is its immunity to interference and noise compared to RF, however the ultrasound cannot penetrate walls and is more expensive than comparable RF based positioning systems.

RF based positioning systems are one of the most popular for indoor tracking. This is due to the fact that radio signal strength (RSSI) can be obtained relatively effortlessly without the need of any specialized hardware. *RADAR* was one of the first RF signal strength based positioning system used to track people inside buildings [41]. Multiple base stations (at least 3) are placed with overlapping coverage within the area of interest. A laptop computer carried by an individual is used to collect the RSSI measurements. The system then compares the RSSI observation of the user with a set of pre-stored signal strength measurements known as "*fingerprints*" at each of the base stations to identify the user's coordinates. The major disadvantages of the fingerprinting method include the need for dense training coverage and poor extrapolation to areas not covered during training. Although RADAR employs an empirical model for RF propagation and wall attenuation, actual RF signals deviate considerably while propagating indoors due to multipath, metal reflection, and noise. Often indoor positioning systems have been designed to take advantage of public wireless local area networks (WLAN) instead of setting up proprietary RF networks. For example, *Place-Lab* uses publicly available 802.11 access points with receivers built into the users devices for positioning [45,46]. The system compares the observed RSSI with a pre-stored "*radio map*" to determine the users position. Although it has the advantage of limited calibration requirements, reported accuracy is lower than existing positioning systems. Another example of WLAN based tracking includes the *"Horus WLAN location determination system"* [47]. A laptop computer carried by the user collects RSSI which is then compared to known RSSI fingerprints in order to perform localization. A commercially available product by *Ekahau* provides a complete tag and software solution using RSSI with the 802.11 protocol [48]. Multiple

802.11 access points are placed at predefined locations in the environment. The user carries a small tag that measures the RSSI at periodic intervals. The system compares the observed RSSI with a pre-stored set of RSSI collected during a separate training phase to compute the user's current location. The system is relatively inexpensive and energy efficient, but its accuracy is quite limited in certain scenarios as seen in the experimental section of this paper. Finally, there are systems which explore the use of "*angle of arrival (AOA)*" and "*time difference of arrival (TOA)*" in order to perform user localization. For example, *Ubisense* is a tag based localization engine which uses ultra-wideband (UWB) radio technology to detect a mobile Ubisense tag [49]. Instead of depending on RSSI, it focuses on multiple proprietary access points act as sensors that independently determine the *AOA* of the UWB signal [50] and the *TOA* between a pair of sensors in order to perform positioning. While expensive and with significant calibration challenges, the Ubisense system can provide high tracking accuracy to within several centimeters.

3.2.2 Existing Indoor Tracking Systems: Algorithmic Description

Researchers have adopted a wide number of signal processing and pattern recognition based algorithms in order to perform user localization. As RSSI is widely accepted as the "*feature of choice*" for indoor positioning [43, 51], in this section we will describe the location estimation algorithms which use RSSI as its primary input. Radio propagation inside an indoor environment is extremely chaotic due to the presence of large number of obstructions/reflecting surfaces and hence learning the position-RSSI relationship can be a challenging task. Elnahrawy *et al.* discuss about the potential barriers of using signal strengths to perform localization and show that none of the present algorithms has an huge performance advantage over the others [52]. Below we will summarize a plethora of indoor tracking algorithms which have found extensive uses in numerous applications.

Range and proximity based algorithms

Range and *proximity* based algorithms for user localization are simple, straightforward and easy to implement. From a set of observed RSSI, the range based method triangulates the person's position based on a distance calculation from multiple access points [53, 54].

Unfortunately the computed distance from each RSSI can be erroneous due to the high variation of radio signal propagation in an indoor environment. Instead of depending on signal strength, a method has been proposed which computes the mapping between the transmission power of the access points and the range of the RF signal [55]. The user position can then be inferred using trilateration by exploiting the relationship between radio range and transmission power. Proximity based localization connects a user to an access point from which it receives the maximum signal power and hence the user location is assumed to be the same as the location of that corresponding access point [56, 57]. Hightower *et al.* estimates the user location by computing the centroid of k access points which generates the highest signal strength [43]. Although the "*strongest base station algorithm*", discussed above, is very simple and computationally fast, it can generate a very coarse grained location estimates. The potential solution to improve the granularity is to implement a dense grid of access points inside the indoor location, which effectively increases the hardware and computational requirement.

Fingerprinting method

The fingerprinting based "*scene analysis*" approach is one of the most widely used technologies seen in the literature [41,47,51,58]. It consists of an offline *training* phase and an online *tracking* phase. Fingerprints are generated during the training/calibration phase where RSSI data is collected at a set of marked training locations. During the tracking phase, the collected RSSI observation is compared with known *fingerprints* and the corresponding fingerprints that are the most similar with the observed ones are chosen. The user position can then be inferred as the location of the fingerprint which best matches with the observation or can be computed as a centroid of k nearest reference fingerprints. The most challenging aspect of the fingerprinting based method is to formulate a distance calculation that can measure similarity between the observed RSSI and the known RSSI fingerprints. Euclidean distance based calculation is used in order to measure the minimum distance between the observed RSSI and the mean of the fingerprints collected at each training point [59]. RADAR uses a *k-nearest-neighbors* method in order to find the closest match between fingerprints and the RSSI observation [41]. Recently, research efforts have concentrated on developing a better distance measure that can take into account the variability of the RSSI training vectors. These methods estimate a probability density for the training RSSI and then compute the likelihood/a posteriori estimates during the tracking phase using the observed RSSI and the estimated densities [47,60,61]. User localization is performed using a maximum likelihood (ML) or a maximum-a-posteriori (MAP) estimate of position. Kernel based nonlinear distance calculations have also appeared in the literature for RSSI fingerprinting [51,58]. Although these recent developments improve position estimates compared to simple *k-nearest-neighbors*, they often require substantially larger training sets and greater computational resources. Moreover, the calibration process is tedious, time consuming and manual, which reduce the scalability of the fingerprinting based approach. Searching through the whole fingerprint database for similarity measure between the observed and reference RSSI is a computationally intensive operation and recently efficient algorithms have been proposed to reduce the computational cost [62,63]. Youssef *et al.* performed an efficient location clustering, called "*joint clustering*" [62], and the "*MoteTrack*" system adopts a decentralized approach by distributing the overall RSSI signature database over a number of fixed nodes [63].

Signal propagation modeling

In contrast to fingerprinting, signal propagation modeling based techniques express the RF signal attenuation using a physics based theoretical "path loss" model [64,65]. These proposed path loss models consider both free-space signal attenuation and attenuation suffered due to reflection/refraction from the walls/obstacles. Instead of using fixed attenuation factors for walls/floors, Barsocchi *et al.* apply a linear least-squares technique to learn the attenuation coefficients by minimizing the actual and model predicted RSSI [66]. However, the RSSI propagation in indoor environment is noisy due to multipath, metal reflection, and interference noise [66,67] and hence the relationship between the position and RSSI is highly complex. Thus the RSSI propagation may not be adequately captured by a linear fixed invariant model.

Probabilistic Bayesian inference

A number of variants on the probabilistic Bayesian inference approaches have appeared in the literature [43,46,68–76]. The Bayesian inference is a probabilistic framework which sequentially estimates the unknown state from the noisy observations using a dynamic predictive model and an observation likelihood. The Bayesian methods can estimate a person's velocity and acceleration in addition to position, and can also provide an uncertainty measure of the estimates. In [68, 73], the authors survey the Bayesian filter implementations for location estimation using the ultrasound, infrared and laser range finders. They conclude that although the particle filters can converge to the true posterior state distribution for non-Gaussian and multimodal cases, the Kalman filter and its variants are the most efficient in terms of memory and computation. The Kalman filtering methods for real time positioning have long been popular in the robot tracking and navigation communities [12, 77, 78]. Recently the Kalman filter and their variants have also been applied to indoor people tracking. For examples, Fod *et al.* and Hsieh *et al.* describe a Kalman filter approach using multiple laser range finders [70,71]. More recently, the particle filters have been used to demonstrate encouraging performance, although at a high computational cost for real time people tracking [43, 74–76]. The particle filter based system described by J. Hightower *et al.* incorporates a random acceleration based human motion model as the dynamics of the system, while the sensor model (observation likelihood) uses a single Gaussian with fixed pre-defined parameters [43]. Letchner *et al.* introduce a sensor measurement model in the particle filter framework [75] that combines a Wi-Fi signal propagation model [79] and a fingerprinting technique for localization. The method assumes radially symmetric attenuation of wireless signals and also requires large training data for fingerprinting. In addition, several algorithms assume an empirical *path-loss* based radio signal propagation map to compute the likelihood of RSSI observation in a particle filter framework [45, 79, 80]. The performance of these algorithms, however, may degrade in practice due to the RSSI variability over time and location. Recently research efforts have been directed towards developing local RSSI likelihood models from the training data with known ground truth locations [74, 81]. Ferris *et al.* use Gaussian processes to generate an observation likelihood for wireless signal strength measurements in the particle filter [74]. However, learning the parameters using graph based hyperparameter estimation can be slow and take significant computational resources. Also related to the Bayesian inference are the use of hidden Markov modeling (HMM) approaches. The *Locadio* system uses a HMM on a graph of location nodes to infer position based on the variation of the Wi-Fi signal strength [72]. The person's motion is determined based on the variance of RSSI measurements over a sliding window. However, significant RSSI variability (even at the same location) can cause a high number of errors in determining whether the person is moving or still.

3.2.3 SPKS based Position Tracking using RSSI Measurements

We follow the sigma-point Kalman filter (SPKF) based Bayesian inference approach [26, 36, 82] for indoor localization and tracking. We use our recently proposed fixed-interval (FI-SPKS) and fixed-Lag sigma-point Kalman smoother (FL-SPKS) algorithms for tracking purposes [35]. While the FI-SPKS uses all N RSSI measurements in order to compute smoothed state estimates at each time, location estimate obtained from the FL-SPKS generally lags behind the current measurement by L time. Both the FI-SPKS and FL-SPKS estimators fuse a model of walking motion, room-wall configurations, and all available sensor observations in order to track 2D position and velocity. A random acceleration based model of human walking is used as the dynamic model of motion. This is augmented with a room-wall model involving a potential field created throughout the indoor environment in order to repel motion away from walls. Available sensors include RSSI, binary infra-red (IR) motion sensors, and binary foot-switches. Instead of using a fixed path loss based prior map for the observation model, we learn the position-RSSI relationship from the training data. Specifically, *Radial-Basis Function (RBF)* networks are used to provide a nonlinear mapping between known locations and observed RSSI values. These models are fit during a separate calibration process, and take into account the various multipath and other room specific characteristics. This chapter provides a more detailed description and analysis of our methods that was presented in [83, 84].

While our approach is generally independent of the specific hardware or sensor modality, the current system design uses RSSI sensors manufactured by Ekahau Inc. The person(s) to be tracked carry a small body-borne device that periodically measures the RSSI at 3 or more standard Wi-Fi access points placed at pre-defined locations. Due to tag based hardware limitations, the sampling rate is generally $4 - 8\,\mathrm{s}$. The low sampling rate motivates the use of the sigma-point smoother over the filter implementation, as the smoother provides superior interpolation of data using both past and future observations. A smoother requires buffering of data and a fixed latency in performing the actual estimates. While the computational complexity is increased, the sigma-point smoothers can still be implemented far more efficiently than comparable particle smoother formulations. Augmenting the RSSI measurements are IR motion sensors mounted to the walls that provide a binary "on" signal when it detects a motion in its range. Similarly, binary foot-switches indicate the location of a person when stepped on.

Experimentations were performed at several "living-laboratories" used to develop monitoring and assistive technologies for the elderly. The performance of our tracker was compared with the baseline Ekahau tracking engine. As will be shown, both the FI-SPKS and FL-SPKS based tracker provide significant improvement in position tracking accuracy.

3.3 Recursive Bayesian Estimation Framework

Recall that the problem of state estimation involves estimating the state of a discrete-time nonlinear dynamic system,

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) \tag{3.1}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{n}_k\right), \tag{3.2}$$

characterized by the process model $f(.)$ and observation model $h(.)$. In the following sections, we briefly summarize the different components of our tracking mechanism including the dynamic model, observation models and how the measurements from multiple sensors can be fused using the Kalman framework.

3.3.1 Dynamic Model

We define the state vector $\boldsymbol{x} = \begin{bmatrix} x & y & v_x & v_y \end{bmatrix}^T$, corresponding to 2D position and velocity for tracking purposes. A simple random acceleration based model [85, 86] is used for predicting walking motion. This is augmented with a room model involving a potential field created throughout the indoor environment in order to repel estimated motion away from walls.

The potential field can be created off-line using prior knowledge of wall configurations and large furniture location. Computationally this is achieved by dividing the space into 1 inch square cells. Each cell contains a binary certainty measure $C(i, j)$ that indicates whether the cell is occupied, *i.e.*, an obstacle exits within the cell. The force $\boldsymbol{F}_{i,j}(x, y)$ exerted on a person due to an occupied cell is made inversely proportional to the distance between the person's current position and the occupied cell position [87].

$$\boldsymbol{F}_{i,j}(x, y) = -\frac{F_{\mathrm{cr}} C(i, j)}{d_{i,j}^2(x, y)} \left(\frac{x - x_c^i}{d_{i,j}(x, y)} \vec{\boldsymbol{x}} + \frac{y - y_c^j}{d_{i,j}(x, y)} \vec{\boldsymbol{y}} \right), \tag{3.3}$$

where $d_{i,j}(x, y)$ is the distance between the person's current position, (x, y), and the occupied cell position, (x_c^i, y_c^j). $\vec{\boldsymbol{x}}$ and $\vec{\boldsymbol{y}}$ are the unit vectors along the x and y direction. F_{cr} is the force constant and design parameter that controls the overall strength of the repulsive force. If the force is too strong, location estimates will not be near walls or furniture. If the force is too small, tracking may result in trajectory estimates that pass through walls.

The total resultant force $\boldsymbol{F}_{\mathrm{r}}(x, y) = \begin{bmatrix} F_x(x, y) & F_y(x, y) \end{bmatrix}$ is the vectorial sum of forces exerted by all the occupied cells on the person's current cell location.

$$\boldsymbol{F}_{\mathrm{r}}(x, y) = \sum_{i,j} \boldsymbol{F}_{i,j}(x, y). \tag{3.4}$$

This repelling force function $\boldsymbol{F}_{\mathrm{r}}(x, y)$ is calculated off-line, and may be viewed as a potential field or simply a nonlinear function of the person's current position. Figure 3.1 displays the corresponding magnitude of the potential field for a simple multi-room example.

Combining the potential field and a random walk model yields the dynamic state-space

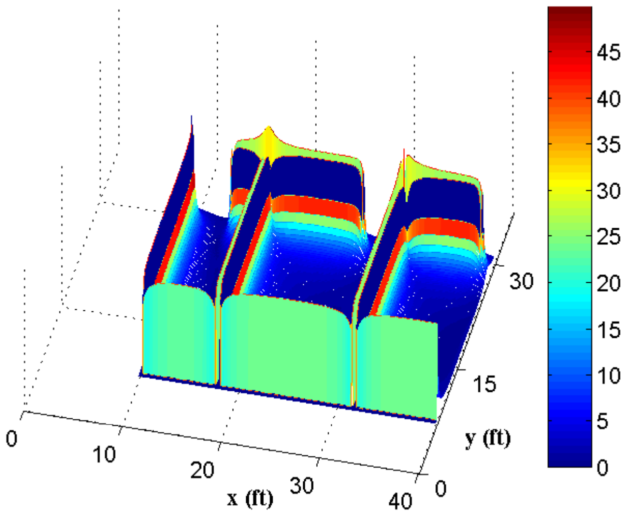


Figure 3.1: Potential field is shown in a multi-room environment. As illustrated, the magnitude of the potential force peaks at the edge of the walls and decreases exponentially.

model $f(.)$ [84],

$$x_{k+1} = x_k + \delta T v_{x_k} + \frac{\delta T^2}{2} F_{x_k}(x_k, y_k) \tag{3.5}$$

$$y_{k+1} = y_k + \delta T v_{y_k} + \frac{\delta T^2}{2} F_{y_k}(x_k, y_k) \tag{3.6}$$

$$v_{x_{k+1}} = \lambda v_{x_k} + \delta T F_{x_k}(x_k, y_k) + (1-\lambda) v_{p_{x,k}} \tag{3.7}$$

$$v_{y_{k+1}} = \lambda v_{y_k} + \delta T F_{y_k}(x_k, y_k) + (1-\lambda) v_{p_{y,k}} \tag{3.8}$$

The parameter λ smoothens the changes in velocities and also ensures that the variance of random process remains bounded. The integration time in this case is $\delta T = 1$ second. The process noise $\boldsymbol{v}_{p,k} = \begin{bmatrix} v_{p_{x,k}} & v_{p_{y,k}} \end{bmatrix}$ is modeled as a zero mean white Gaussian.

3.3.2 Observation Model

As we have three different sensor technologies, the observation model in (3.2) depends on the specific technology used.

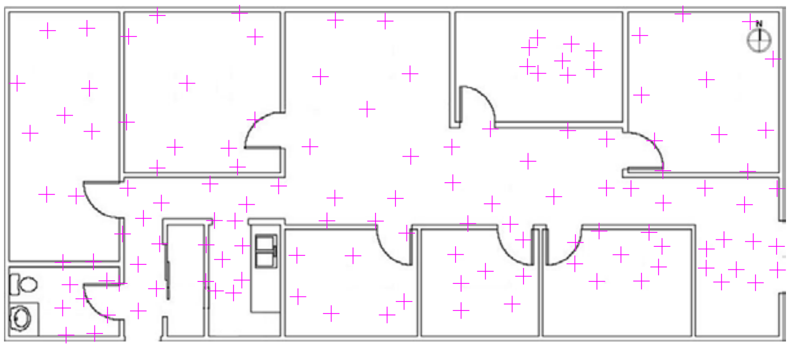


Figure 3.2: Example floor plan with calibration locations indicated by a '+'.

RSSI Observation model

A naive approach to using RSSI measurements involves comparing an observed RSSI value to a table of previously recorded RSSI values and their associated positions. This direct "table look-up" approach, however, is prone to errors due to the high variability of RSSI values. In the Bayesian framework, the observation function can be viewed as a *generative* model providing the likelihood of a RSSI observation given the current estimate of the state position. In most RSSI tracking literature, the observation likelihood is approximated with a simple fixed *a priori* distribution (*e.g.*, Gaussian distribution) [43,46]. In our method, we characterize the RSSI-position relationship and variability by fitting nonlinear mappings between position and observed RSSI values.

Data to fit the maps are first collected during a calibration phase. This involves dividing the floor plan into P rooms or sections. In each section, the vertices and center of an approximate octagonal grid are used as calibration points. See Figure 3.2 for illustration purposes. This calibration scheme was chosen to match the grid pattern used by the Ekahau positioning engine in order to allow for the direct comparisons of final performance. Note that an exact octagonal grid is not possible due to the presence of furniture, walls and other objects in the floorplan. At each calibration point, a person carrying a body borne RSSI tag spends $\boldsymbol{T}_{\mathrm{C}}$ s (generally 60 s) while RSSI data is collected. Typically, RSSI values are recorded from M (generally $3-5$) Wi-Fi access points located in the corners of the entire space to be calibrated. The person also performs a slow rotation at each point to average RSSI variability due to tag orientation. Note that if multiple tags are to be

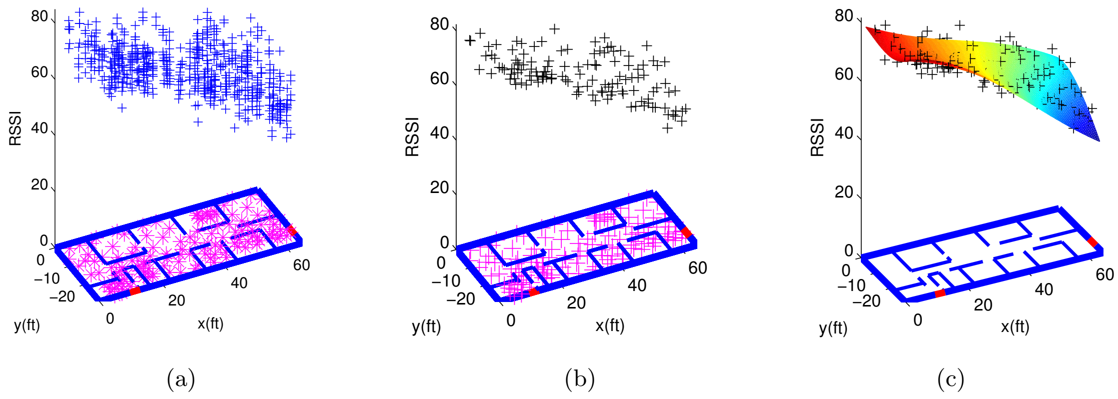


Figure 3.3: (a) Raw RSSI values recorded at each access point during calibration, (b) RSSI mean values at each calibration location, (c) RBF nonlinear maps plotted with the RSSI mean values.

calibrated simultaneously, it is advisable to physically separate the tags on the person as far as possible, as we have found that multiple tags can interfere with RSSI consistency. This process is repeated at all calibration points in the space. Figure 3.3(a) illustrates the collection of raw RSSI data at each calibration point. The number of RSSI samples collected at each calibration point are denoted as $\boldsymbol{N}_{\mathrm{r}}$ (generally $8-10$). The $\boldsymbol{N}_{\mathrm{r}}$ number of RSSI samples are then averaged to obtain a representative mean RSSI observation per calibration point as shown in Figure 3.3(b). Specific values for the amount of data collected, variability, etc., are tag specific and will be given in the experimental results section.

After RSSI data collection, a *RBF* network is used to fit a nonlinear map between known calibration locations and the mean RSSI observations as illustrated in Figure 3.3(c). A *RBF* network is a feed forward neural network consisting of a hidden layer of radial kernels and an output layer of linear neurons [88]. A Gaussian kernel is used as the radial basis. This *RBF* map represents the forward generative observation model,

$$z_{m,k} = h_m\left(x_k, y_k\right) + n_m^{\mathrm{r}}, \tag{3.9}$$

where $z_{m,k}$ is the observed RSSI from access point m, $1 \leq m \leq M$, with noise n_m^{r} assumed to be Gaussian with zero mean and standard deviation equal to the RSSI variability determined from the calibration data. The RBF observation map h_m for the m-th access point is specified by

$$h_m\left(x_k, y_k\right) = \boldsymbol{W}_m^T \boldsymbol{K}_{m,\mathrm{G}}\left(\left[\begin{array}{cc} x_k & y_k \end{array}\right]; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\right), \tag{3.10}$$

where $\boldsymbol{K}_{m,\mathrm{G}}$ is the Gaussian kernel function [88] with mean vector $\boldsymbol{\mu}_m$ and covariance matrix $\boldsymbol{\Sigma}_m$,

$$\boldsymbol{\mu}_m = \left[\begin{array}{cccc} \boldsymbol{\mu}_{m,1} & \boldsymbol{\mu}_{m,2} & \dots & \boldsymbol{\mu}_{m,\mathrm{C}} \end{array}\right]^T \tag{3.11}$$

$$\boldsymbol{\Sigma}_m = \left[\begin{array}{cccc} \boldsymbol{\Sigma}_{m,1} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Sigma}_{m,2} & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & 0 & \boldsymbol{\Sigma}_{m,\mathrm{C}} \end{array}\right], \tag{3.12}$$

where C is the number of Gaussian kernels in the hidden layer of the RBF network and $\boldsymbol{W}_m$ are the output layer linear weights,

$$\boldsymbol{W}_m = \left[\begin{array}{cccc} w_{m,0} & w_{m,1} & \dots & w_{m,\mathrm{C}-1} \end{array}\right]. \tag{3.13}$$

The parameters of each Gaussian kernel $\boldsymbol{\mu}_{m,c}$, $\boldsymbol{\Sigma}_{m,c}$ and the hidden-to-output layer weights $w_{m,c}$ are learned using a hybrid procedure that operates in two stages. The prior weight, center position and the spread parameter of each Gaussian are first obtained by modeling the known calibration locations with a Gaussian Mixture Model (*GMM*) using the Expectation Maximization (*EM*) algorithm. The output layer weights $\boldsymbol{W}_m$ are then calculated in a batch least-squares manner in order to minimize the MSE error at the output. Figure 3.3(c) illustrates a nonlinear observation map learned from the calibration data.

The observed RSSI $z_{m,k}$, RBF function h_m, and the observation noise n_m^{r} from each access point are combined to form a multi-dimensional observation model,

$$\boldsymbol{z}_k = \left[\begin{array}{cccccc} z_{1,k} & z_{2,k} & \dots & z_{m,k} & \dots & z_{\mathrm{M},k} \end{array}\right] \tag{3.14}$$

$$\boldsymbol{h} = \left[\begin{array}{cccccc} h_1 & h_2 & \dots & h_m & \dots & h_{\mathrm{M}} \end{array}\right] \tag{3.15}$$

$$\boldsymbol{n}^{\mathrm{r}} = \left[\begin{array}{cccccc} n_1^{\mathrm{r}} & n_2^{\mathrm{r}} & \dots & n_m^{\mathrm{r}} & \dots & n_{\mathrm{M}}^{\mathrm{r}} \end{array}\right] \tag{3.16}$$

where $\boldsymbol{z}_k$ is the multi-dimensional RSSI observations emanating from each access point. Similarly $\boldsymbol{h}$ and $\boldsymbol{n}^{\mathrm{r}}$ are the augmented RBF observation model and the measurement noise for access points $1 \leq m \leq \mathrm{M}$.

Once fit using calibration data, this RBF observation model may be used in the Bayesian framework for tracking. As the RBF network is trained to learn a nonlinear mapping between known calibration locations and observed RSSI values, this model takes into account room specific multi-path and non line of sight (NLOS) RSSI propagation. By learning the map, the need to specify the location of the access points is also avoided.

IR motion sensor Observation model

Infra-red (IR) motion sensors may be mounted to the walls and provide binary "on" signals when motion is detected within range. Localization using motion sensors are challenging due to their large beam width and high false alarm rate. The likelihood model for a motion sensor is modeled simply as a Gaussian distribution. The observation model is thus linear and defined as:

$$\boldsymbol{z}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{n}^{\text{ms}}, \tag{3.17}$$

where $\boldsymbol{H}$ is the observation matrix,

$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \tag{3.18}$$

and $\boldsymbol{n}^{\text{ms}}$ is the Gaussian observation noise with mean and variance associated with the IR sensor. The mean value is taken to be a position in-line with the orientation of the sensor at a distance based on the approximate sensor range. The variance is based on the beam width of the sensor. Specific values for the mean and variance are found by approximate characterization of the sensors. Filter performance is not highly sensitive to these values. Note that this simple model clearly does not take into account the specific geometry of the beam pattern, or other complicating factors such as memory and latency in the binary sensor. Incorporating a more accurate distribution would require a non-Gaussian framework (e.g., particle filters), and was not explored for this current phase of the research.

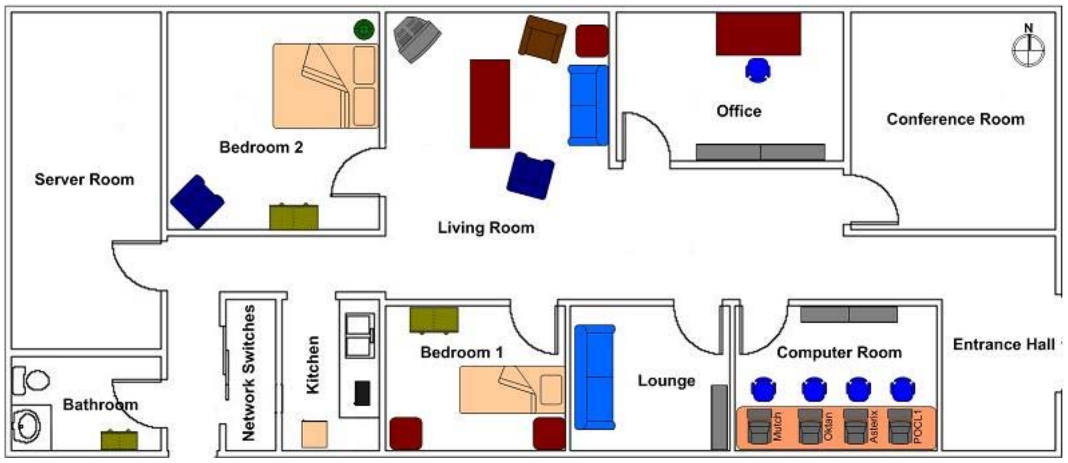


Figure 3.4: Floor layout for test Lab-I

Binary foot-switch Observation model

Similar to IR motion detectors, foot-switches may be placed on the floor to provide a binary "on" signal that indicates the location of a person. The observation likelihood may again be modeled simply as a Gaussian distribution with the corresponding observation model,

$$\boldsymbol{z}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{n}^{\mathrm{f}}, \tag{3.19}$$

where $\boldsymbol{H}$ is the observation matrix,

$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \tag{3.20}$$

and $\boldsymbol{n}^{\mathrm{f}}$ is the Gaussian observation noise for the foot-switch sensors. The mean value of the Gaussian is set to the known location of the switch. We set the variance σ_{f}^2 of the foot-switches to be approximately 10 ft in our experiments. While this is clearly larger than necessary, the aim was to simulate accuracy closer to that of the IR motion sensors during initial setup of the testing facilities. Setting the variance to a small value can provide exact localization at precise moments in time, but can also lead to "*jumps*" in trajectories at the vicinity of the foot-switches. Artificially increasing the variance helps ensure the smoothness of the estimated trajectory.

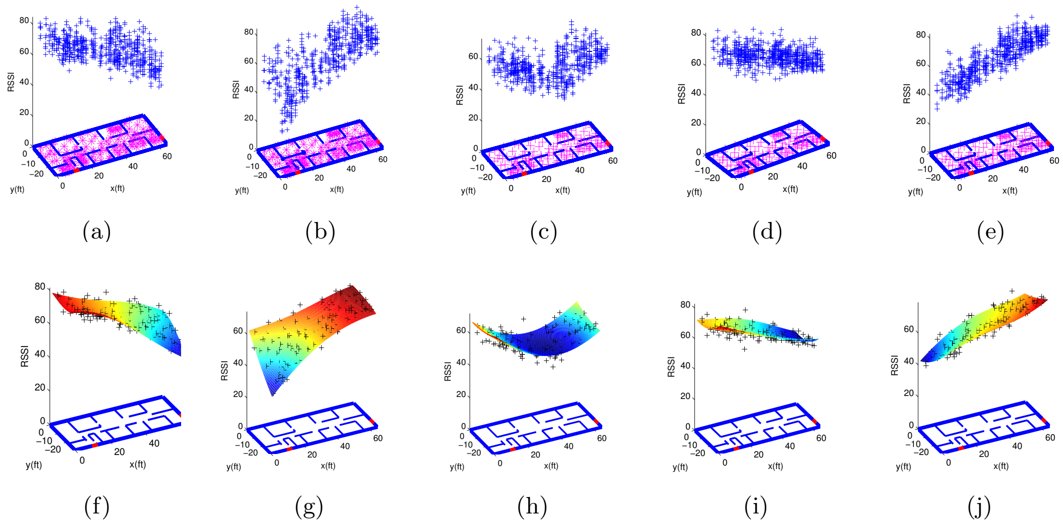


Figure 3.5: (a) to (e): Raw RSSI values from 5 access points collected during calibration at Point-of-Care *test Lab-I*, (f) to (j): Fitted RBF maps.

Multiple sensors observation model

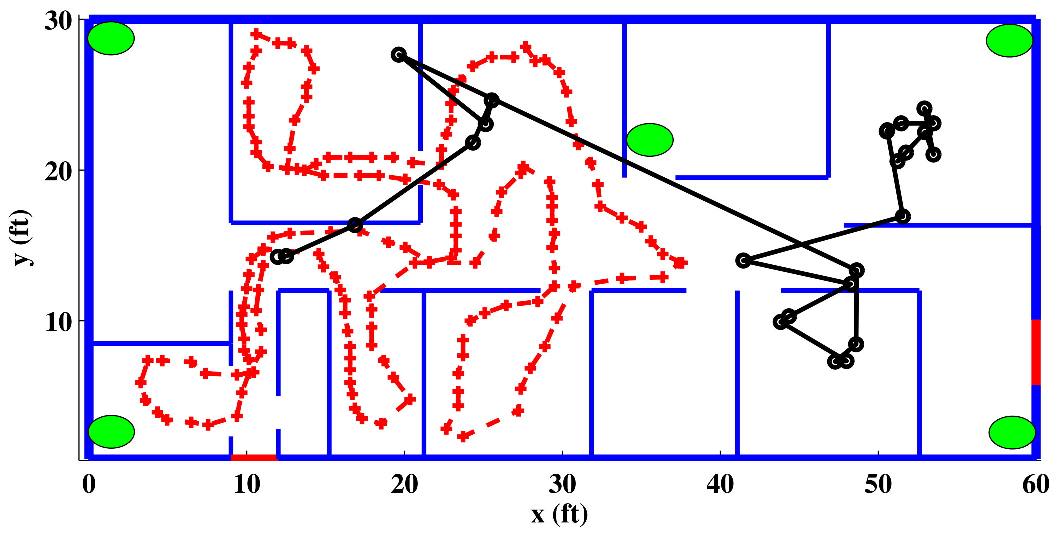
The Kalman framework allows for fusing multiple sensors of different types as available. An augmented observation vector is specified,

$$\boldsymbol{z}_k = \left[\begin{array}{ccc} \boldsymbol{z}_k^{\text{RSSI}} & \boldsymbol{z}_k^{\text{IR}} & \boldsymbol{z}_k^{\text{Foot}} \end{array} \right] \tag{3.21}$$

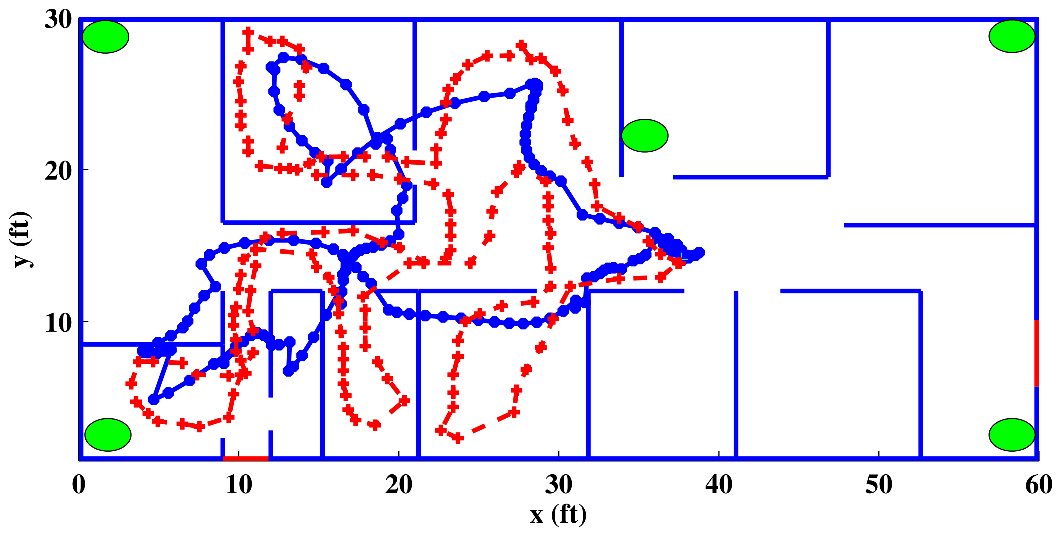
along with the corresponding observation functions. Note that the dimension of this augmented observation may change at each time step to account for varying sensor sampling rates or missing observations.

3.4 SPKS Based Location Tracker

While the SPKF may be applied directly to the tracking problem, we have found improved performance through the use of SPKS. We have investigated both of our proposed smoother variants, namely the FI-SPKS and FL-SPKS, for Wi-Fi based tracking. The FI-SPKS corresponds to a fixed interval smoothing approach whereby the final time N is fixed and smoothed estimates are found using all N measurements. For tracking purposes,



(a)



(b)

Figure 3.6: Tracking performance in test Lab-I using RSSI measurements, (a) Ekahau estimates (red: ground truth, black: estimate), (b) SPKS estimates (red: ground truth, blue: SPKS estimate). The position of the access points are shown by green circles on the floorplan. The above tracking result is shown for subject 1.

the FI-SPKS provides an off-line estimate of the position and velocity trajectories after all data up to time N has been collected. In fixed-interval SPKS (FI-SPKS) framework, we apply both the FBSL-SPKS and the RTSSL-SPKS algorithms to estimate the position and velocity of the user. We have demonstrated the detailed derivations of the FBSL-SPKS

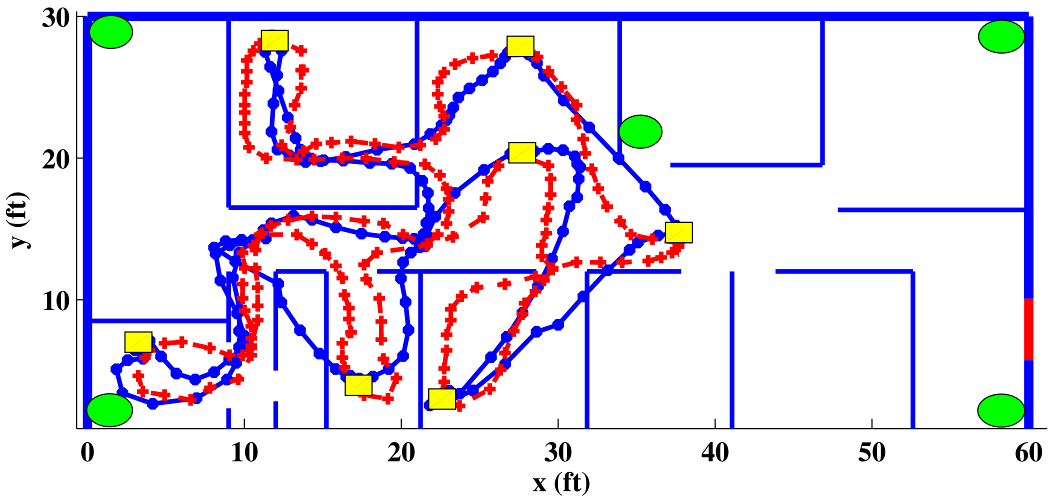
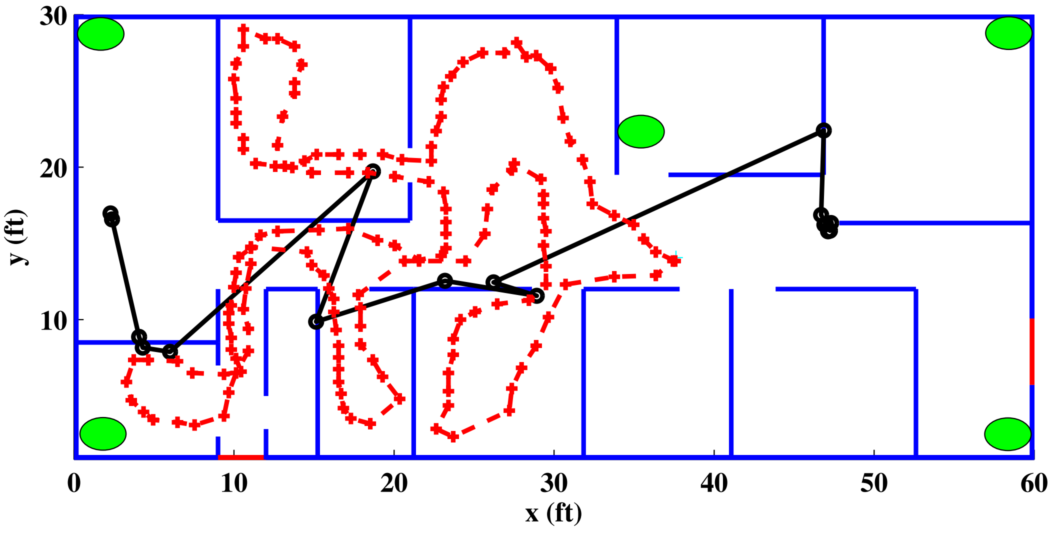


Figure 3.7: Tracking performance in test Lab-I using RSSI + foot switch observations (red: ground truth, blue: SPKS estimate). Yellow rectangular boxes indicate the position of the foot switches on the floor plan. The position of the access points are shown by green circles on the floorplan. The above tracking result is shown for subject 1.

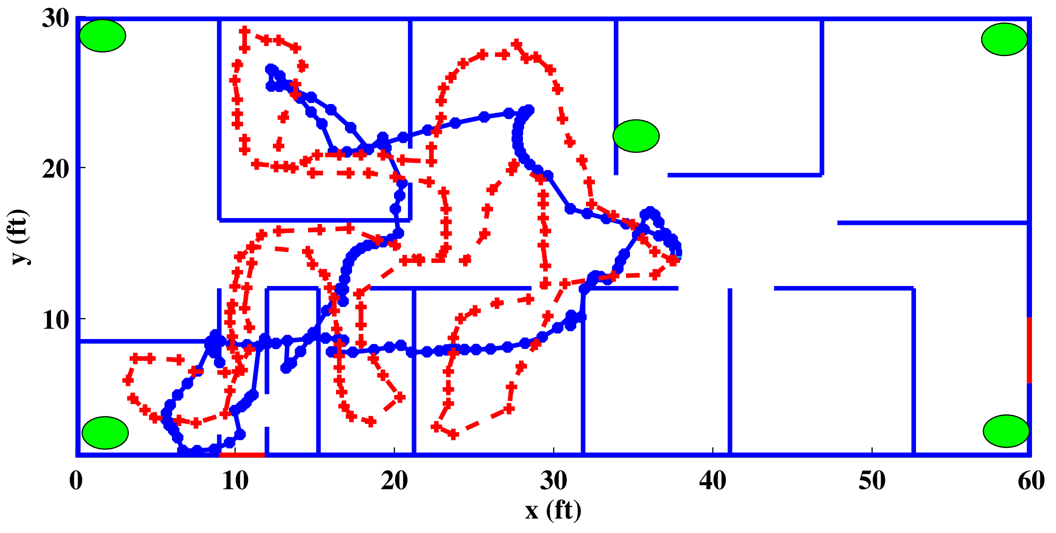
and the RTSSL-SPKS from first principle in sections 2.3.2 and 2.3.3. In the experimental result section, we have only demonstrated the performance of the RTSSL-SPKS due to its ease of implementation and low computational complexity.

We have also implemented the proposed FL-SPKS methods whereby the smoothed state estimate always lags behind the current observation by L time interval (for our case, we have used $L = 3$). For details about the different variants of the fixed-lag SPKS approaches, please refer to Section 2.4. The time difference L is a design specific constant and adds a fixed latency in computing the actual estimates. We have implemented both the FBSL-SPKS and the RTSSL-SPKS for constantly estimating user's 2D position and velocity in an indoor environment. Both the FBSL-SPKS and the RTSSL-SPKS provide pseudo real-time estimates by dividing the data into blocks (e.g., $N = \sum N_i$) and then sequentially performing the smoother operation on the buffered blocks of data as they become available. The equations of the fixed-lag FBSL-SPKS and the RTSSL-SPKS are shown in sections 2.4.3 and 2.4.4. Table 3.1 demonstrates the user-specified parameters needed to implement the SPKS based location tracker.

We had to deal with a number of problem specific issues, such as different update rates



(a)



(b)

Figure 3.8: Tracking performance in test Lab-I, (a) Ekahau estimates (red: ground truth, black: estimate), (b) SPKS estimates using RSSI measurements (red: ground truth, blue: SPKS estimate). The position of the access points are shown by green circles on the floorplan. The above tracking result is shown for subject 2.

and the time-varying observation dimensions, in order to adopt the SPKS framework in indoor location tracking problem. We will discuss below how we address these issues in our indoor positioning system.

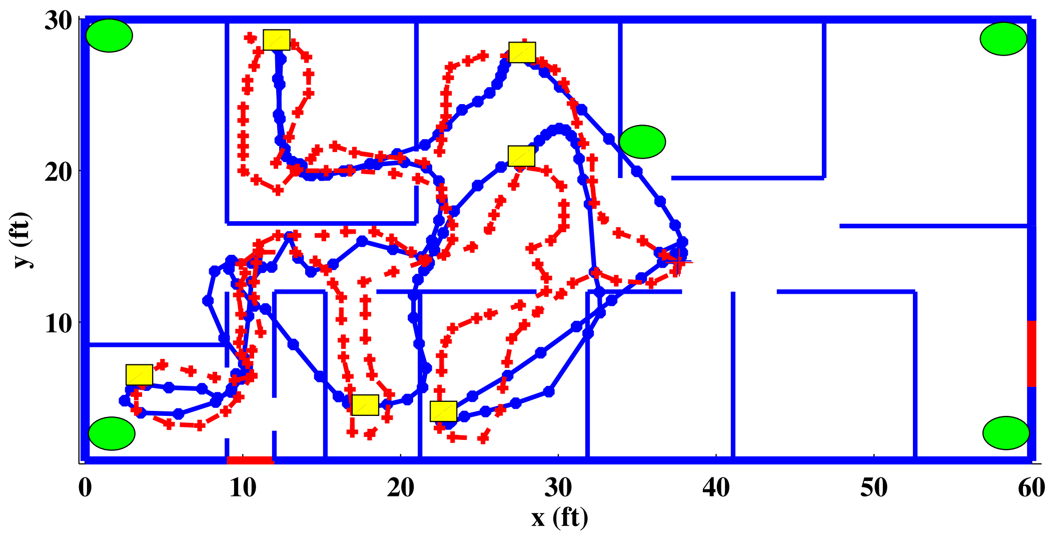


Figure 3.9: Tracking performance in test Lab-I using RSSI + foot switch observations (red: ground truth, blue: SPKS estimate). Yellow rectangular boxes indicate the position of the foot switches on the floor plan. The position of the access points are shown by green circles on the floorplan. The above tracking result is shown for subject 2.

3.4.1 Different update rates for process and observation models

The proposed location tracker uses RSSI sensors manufactured by Ekahau Inc. The Ekahau engine is a proprietary system which requires placing multiple access points at predefined locations of the house. The subject to be tracked carry a body-borne receiver tag which periodically measures the RSSI from the installed access points. Due to tag based hardware limitation, the RSSI sampling rate is between $4 - 8\,\mathrm{s}$. This low sampling rate can be a hinderance for continuous tracking and can also lead to "*jumps*" in estimated trajectories. In order to overcome this problem, we propose to operate the SPKF time-update and measurement-update steps at differing rates. In other words, every SPKF cycle does not necessarily have a measurement-update step. Recall that in filter time-update, we use the dynamic model to predict the next state $\hat{\boldsymbol{x}}_{k+1}^{-}$ from the current state $\hat{\boldsymbol{x}}_k$ and measurements up to time k, $\boldsymbol{z}_{1:k}$. While the measurement-update step utilizes the observation model for incorporating the current measurement $\boldsymbol{z}_{k+1}$ with state prediction $\hat{\boldsymbol{x}}_{k+1}^{-}$ and generates an updated state $\hat{\boldsymbol{x}}_{k+1}$. In the proposed location tracker, the SPKS based estimator performs a time-update step at every second but a measurement-update

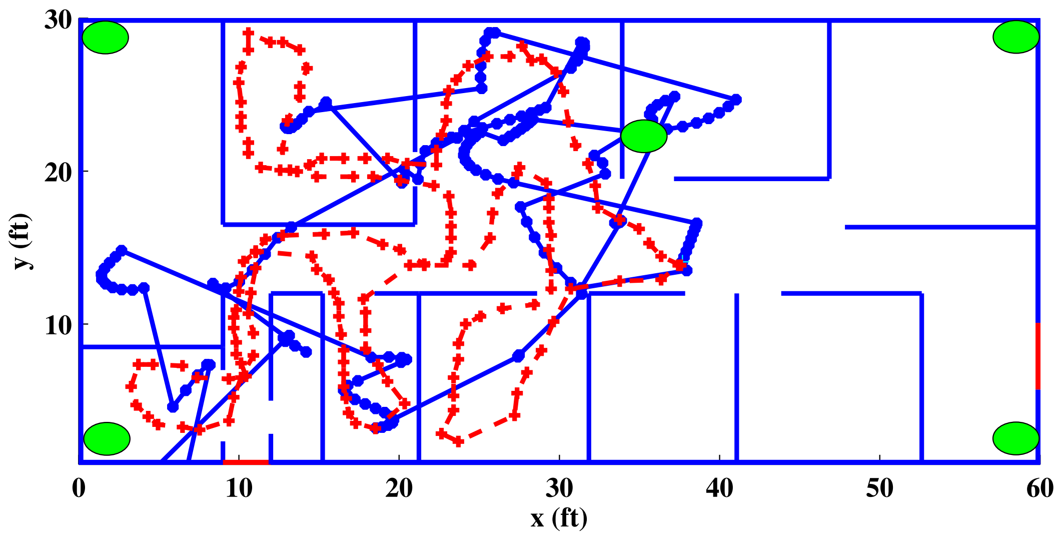
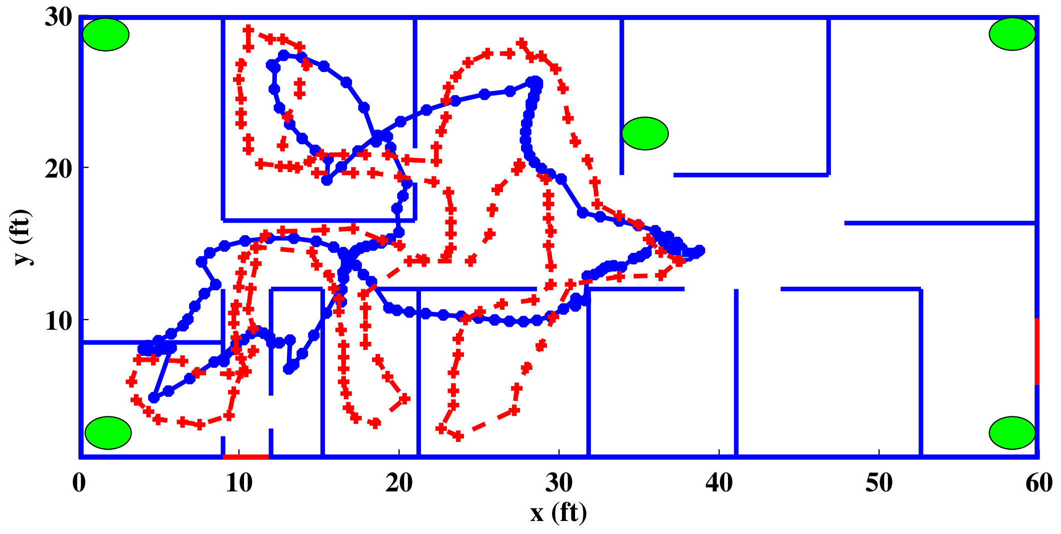


Figure 3.10: SPKF estimates using only RSSI measurements in test Lab-I. The position of the access points are shown by green circles on the floorplan.

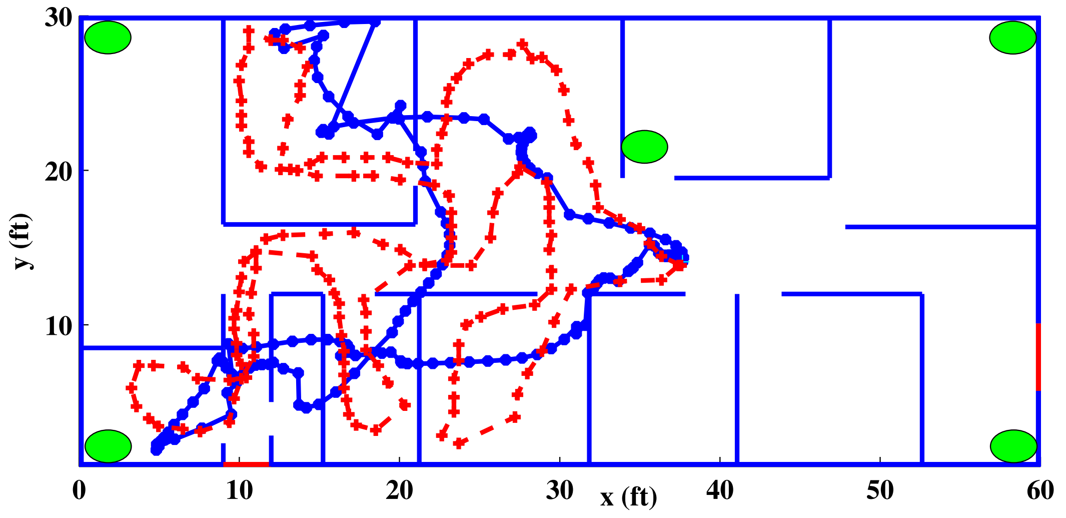
step is incorporated only when the sensor measurement is available. The fast time-update ensures the computation of state estimates at every time index even when the RSSI measurement is not available during that time. The availability of position estimates at higher rate may prove extremely useful for monitoring the elderly at their own homes.

3.4.2 Time-varying observation dimensions

As we have mentioned in Section 3.3.2, the proposed SPKS fuses multiple sensors, including RSSI, IR motion sensors and foot-switches in order to perform localization. As is evident from the experimental results (please see Section 3.5), the performance accuracy of our SPKS based location tracker has been improved by incorporating multiple sensor measurements. Since different sensors operate at different rates and have different observation dimensions, one design challenge was to derive a SPKS framework which can adapt to the time-varying observation dimensions. For example, the sampling rate of RSSI is between $4 - 8\,\text{s}$ and the dimension of RSSI measurement vector depends upon the number of RSSI sensors which actually reported the signal strength at a particular time. IR motion sensors provide binary "on" measurement only when there is a motion within its range. Binary foot-switches can only be turned "on" when pressed. In order to



(a)



(b)

Figure 3.11: (a) Fixed-interval RTSSL-SPKS estimates, (b) Fixed-lag (lag $L = 3$) RTSSL-SPKS estimates (red: ground truth, blue: SPKS estimates). The position of the access points are shown by green circles on the floorplan. The SPKS estimates are generated using only RSSI measurements.

combat the arrival of time-varying measurements, we maintain an event-log which will inform about the number of different sensor firing at each time instant. If data arrives from multiple sensors, the effective observation model used in the SPKS framework is formed by concatenating each individual sensor observation models and sigma points required

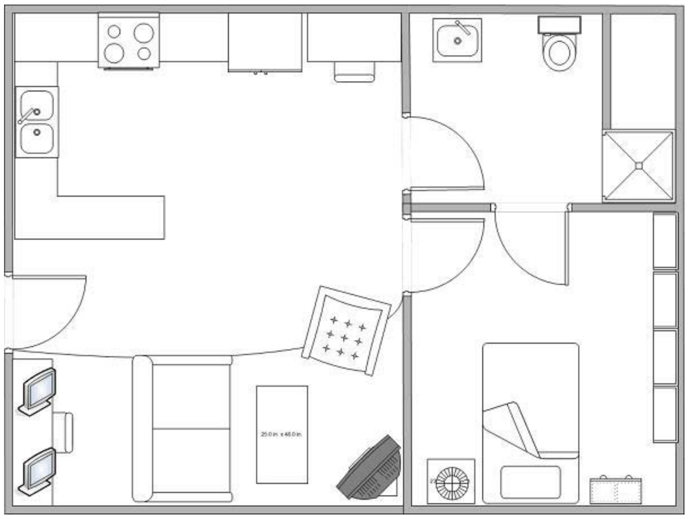


Figure 3.12: Floor layout for test Lab-II

Table 3.1: Summary of user-specified parameters

Name	**Symbol**	**Value**
Potential field force constant	$\boldsymbol{F}_{\text{cr}}$	10
Integration time	δT	1 s
Smoothing AR coefficient	λ	0.95
Number of access points	M	3-5
Measurement noise variance(foot-switch)	$\boldsymbol{n}_k^{\text{f}}$	10 ft
Measurement noise variance(constrained motion sensor)	$\boldsymbol{n}_k^{\text{ms}}$	5 ft
Measurement noise variance(unconstrained motion sensor)	$\boldsymbol{n}_k^{\text{ms}}$	room dimension
Measurement lag	L	3
Sigma-point spread	α	0.85
Sigma-point weighting term	β	2
Sigma-point parameter	κ	0
Time spent per calibration point	$\boldsymbol{T}_{\text{c}}$	1 min
Recorded RSSI per calibration point	$\boldsymbol{N}_{\text{r}}$	8-10
RSSI sampling rate	$\boldsymbol{T}_{\text{r}}$	4-8 s

for measurement-update step will be extracted from this augmented observation vector. When the event-log specifies the firing of single sensor, the effective observation model is reverted back to the corresponding sensor model. This process although very simple and straightforward needs careful logging of sensor events in order to accommodate the time-varying rate of data stream.

3.5 Experimental Results

Implementation and testing were performed at several "living-laboratories" (also called Point-of-Care labs) used to develop monitoring and assistive technologies for the elderly. A number of trials were conducted in which different subjects followed a predefined path. While walking, the subject periodically recorded the ground truth. The Ekahau real-time positional engine was also turned on during these tests. Although the exact location estimation algorithm used by the Ekahau software is not known, this still provides a commercial benchmark for evaluation of our approach. Note that the same calibration data was used for both the SPKS based tracker and the Ekahau's positioning engine. In order to prove that the accuracy of our SPKS based tracker is consistent over different locations and subjects, the experimental results displayed in this work were performed at three different sites with multiple subjects. The three test sites, described below, not only demonstrate the superior accuracy of the proposed tracker compared to Ekahau tracking engine but also expose the limitations of RSSI-based tracking in an indoor location.

3.5.1 Test Lab-I

The Point-of-Care test Lab-I is located at the "*Wallowa*" building, part of West Campus at the Oregon Health & Science University (OHSU). The test site is setup with 5 access points located at the four corners and at the center. A layout of the test Lab-I is shown in Figure 3.4. The size of the test Lab-I is 60 ft by 30 ft. In the entire environment, calibration was performed first in order to measure the RSSI variability emanating from each access point. The floor plan was divided into $P = 15$ sections. Each room was considered a section and the long corridors were divided into multiple sections. In each section 9 points were chosen to perform calibration in such a way that 8 points formed the periphery of an octagon and the remaining point was at the center of the octagon. At each calibration point, a person carrying a RSSI tag spent around one minute to collect the training RSSI data. Roughly $8 - 10$ RSSI measurements were recorded at each calibration points for a total of 1065 measurements.

As described earlier, a *RBF* network was used to fit a nonlinear map between known

Table 3.2: Performance comparison of Ekahau tracking engine with different SPKS trackers at *test Lab-I*. The E(RMSE) and std(RMSE) is computed over 20 different trials.

Estimator	E(RMSE) (ft)	std(RMSE) (ft)
Ekahau(RSSI)	28.4	12
fixed-interval SPKS(RSSI)	5.68	1.95
fixed-interval SPKS(RSSI+footswitch)	1.44	0.5
fixed-lag SPKS with $L = 3$(RSSI)	8.45	4.2
fixed-lag SPKS with $L = 3$(RSSI+footswitch)	2.73	0.62
fixed-lag SPKS with $L = 6$(RSSI)	5.85	2.2
fixed-lag SPKS with $L = 6$(RSSI+footswitch)	1.58	0.6

calibration locations and the collected RSSI values. The raw RSSI at each calibration point and the RSSI calibrated maps for 5 access point were shown in Figure 3.5(a)- 3.5(j).

We conducted several trials of moving test in which subjects walked at a normal speed following a predefined path. Two different subjects were used as the RSSI variability is observed to be subject dependent. In one trial, subject 1 took 174 s to complete the path and 25 RSSI observations were recorded during that time period. Subject 2 completed the same path in 164 s and recorded 19 RSSI observations. The sampling rate varied between $4 - 8$ s during tracking. Multi-rate filtering was implemented so that the time-update equations still provide estimates of the position and velocity at every second. Approximate ground truth was collected periodically during the walking and is also shown in the plots. Figure 3.6(a)- 3.9 compares the estimates obtained from the Ekahau engine and the SPKS tracker. From Figure 3.6(a) and 3.8(a), it can be seen that the estimates from the Ekahau tracking engine are very inaccurate and often fail to even locate the person in the correct region/room. The fixed-interval based SPKS tracker with RSSI only observations clearly tracks the person with greater accuracy (see Figure 3.6(b) and 3.8(b)).

The tracking performance of the SPKF, the fixed-interval RTSSL-SPKS and the fixed-lag RTSSL-SPKS are compared in Figure 3.10- 3.11(b). Note that while the fixed interval based RTSSL-SPKS uses all RSSI observations to obtain each smoothed estimate, the fixed-lag RTSSL-SPKS takes into account the past, present and L future measurements to provide the smoothed state at each time k. In this example, the lag between the current measurement and the estimated state is equal to $L = 3$. Considering the sampling

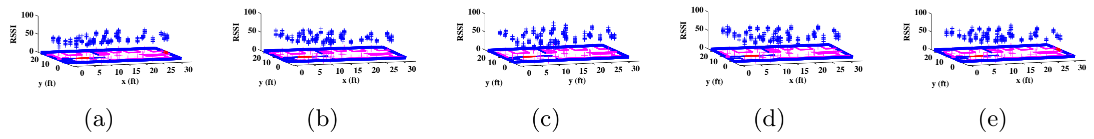


Figure 3.13: (a) to (e): Raw RSSI values from 5 access points collected during calibration at Point-of-Care *test Lab-II*.

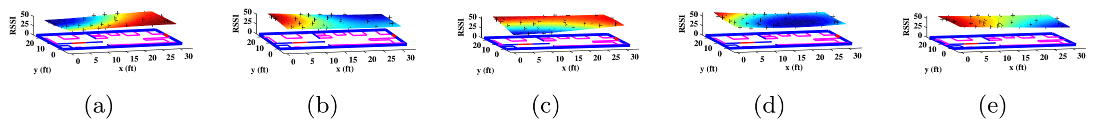


Figure 3.14: (a) to (e): Fitted RBF maps on top of the mean of RSSI collected during calibration at *test Lab-II*.

rate of $4 - 8\,\mathrm{s}$, the corresponding lag introduces a latency of $12 - 24\,\mathrm{s}$ in the location tracking framework. As shown, both the SPKS-based methods provide superior estimates as compared to the SPKF estimates. Notice how the SPKF generates large "jumps" due to poor sampling rate of the RSSI sensors. As shown in Figure 3.11(a) and 3.11(b), the SPKS is free of such problem as the smoother provides superior interpolation of data using both past and future observations. The fixed-interval based RTSSL-SPKS estimates are slightly more accurate compared to its fixed-lag counterpart. Accuracy of the FL-SPKS improves with increasing lag value and it becomes equal to that of the fixed-interval smoother for lag $L = 6$, as demonstrated in Table 3.2.

When RSSI observations are integrated with foot-switch signals, the accuracy of the SPKS based tracker improves even further (Figure 3.7 and 3.9). Note that we set the variance of the foot-switch sensors to be 10 ft in our experiments.

Table 3.2 compares the estimation accuracy of fixed-interval and fixed-lag SPKS techniques with the Ekahau tracking engine. The tracking accuracy of each estimator is demonstrated in terms of mean of root-mean-square-error (RMSE) and standard deviation (std) of RMSE for 20 different trials. We have selected three subjects and each subject performs a number of walking tests in the lab. As is clearly evident from the table, both the fixed-interval and fixed-lag SPKS significantly outperform the Ekahau tracking engine.

Table 3.3: Performance comparison of Ekahau tracking engine with different SPKS trackers at *test Lab-II*. The E(RMSE) and std(RMSE) is computed over 20 different trials by various subjects.

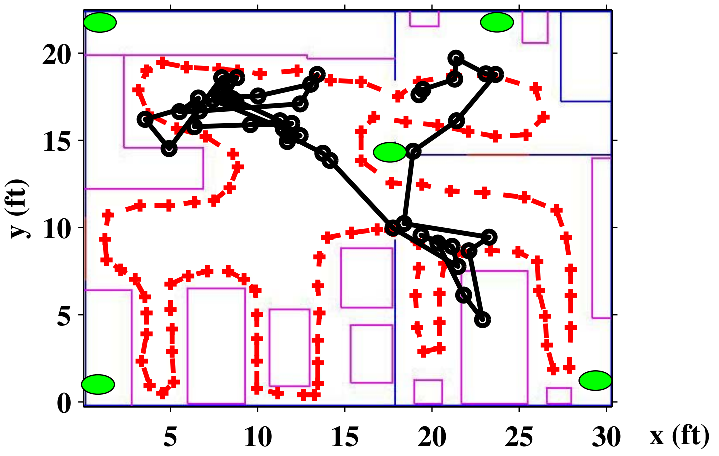
Estimator	E(RMSE) (ft)	std(RMSE) (ft)
Ekahau(RSSI)	20.22	8.2
fixed-interval SPKS(RSSI)	8.38	2.5
fixed-interval SPKS(RSSI+IR motion sensor)	6.25	2.22
fixed-lag SPKS with $L = 3$(RSSI)	11.6	3.4
fixed-lag SPKS with $L = 3$(RSSI+IR motion sensor)	9.55	2.8
fixed-lag SPKS with $L = 6$(RSSI)	8.86	2.4
fixed-lag SPKS with $L = 6$(RSSI+IR motion sensor)	7.02	2.36

3.5.2 Test Lab-II

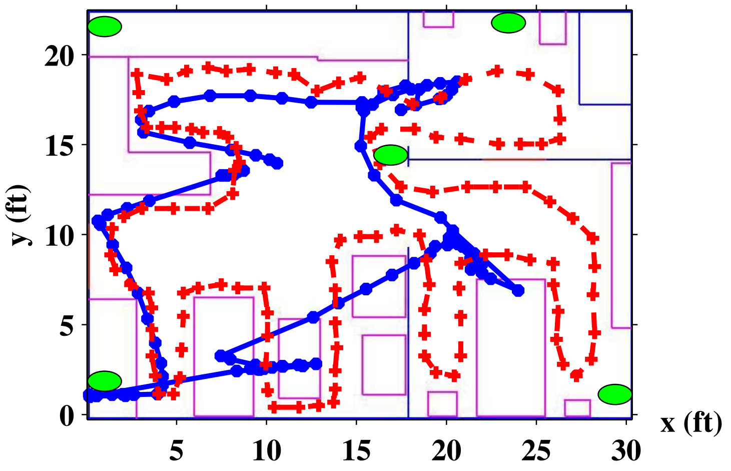
The test Lab-II is located in the "Center for Health and Healing (CHH)" at OHSU. The floor layout is shown in Figure 3.12. Similar to POCL test Lab-I, calibration was performed here for each of the 5 Wi-Fi access points. The raw RSSI collected at each calibration point and the RSSI calibrated maps for 5 access points are demonstrated in Figure 3.13(a)- 3.14(e). Notice from the raw RSSI plots, there is no significant variation of RSSI across the floor. This is due to the small size of the POCL Lab-II (30 ft by 22.5 ft). The low RSSI variability across the test Lab-II is responsible in producing the RBF maps which are almost "*flat*" and hence carry little information about one-to-one position-RSSI mapping.

The lab is also fitted with a number of IR motion sensors instead of foot switches. There are two variants of motion sensors installed in the houses based on their field-of-view. The full beam width unconstrained sensors are generally installed one per room and have variability that matches the full dimension of the room. The constrained sensors have limited beam width and are generally installed along corridors. The variability of the constrained sensors is thus significantly lower than the unconstrained ones.

In Figure 3.15(a)- 3.16, we demonstrate a walking experiment comparing the Ekahau performance to the SPKS tracker. The Ekahau estimates as observed in Figure 3.15(a) are mostly stuck in one portion of the house. The SPKS tracker performance using RSSI, with and without the IR motion sensors are depicted in Figure 3.15(b) and 3.16. Note we



(a)



(b)

Figure 3.15: Tracking performance in test Lab-II using only RSSI measurements, (a) Ekahau estimates (red: ground truth, black: Ekahau estimate), (b) SPKS estimates using RSSI measurements (red: ground truth, blue: SPKS estimate). The position of the access points are shown by green circles on the floorplan. The furniture positions are shown as magenta rectangular boxes.

only demonstrate the performance of the fixed-interval based RTSSL-SPKS in this section. While still superior to the Ekahau estimates, the small size of the POCL Lab-II and the presence of many pieces of furniture limit the performance of the SPKS tracker compared to its performance in Lab-I. The limited performance of the SPKS is expected considering how flat the observations maps are due to the low variability of RSSI across the floor. The example with test Lab-II clearly exposes one of the fundamental limitations of localization

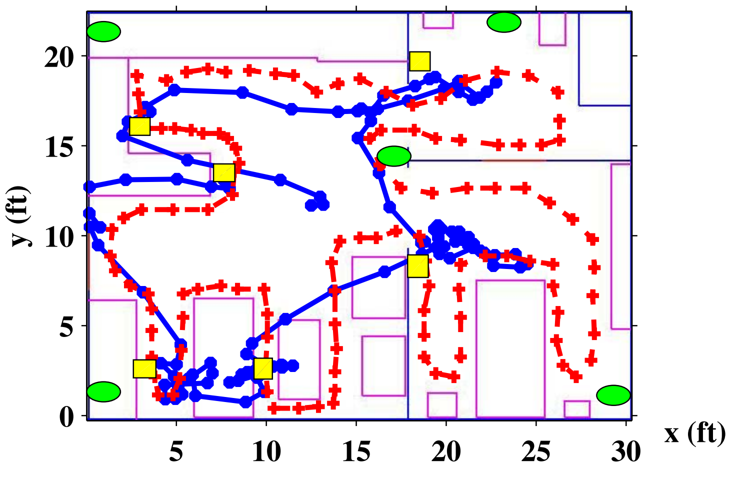


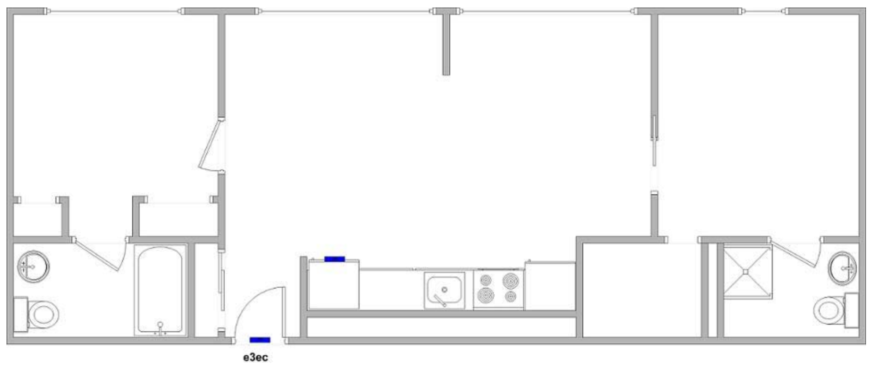
Figure 3.16: Tracking performance in test Lab-II using RSSI + motion sensor, (a) SPKS estimates using RSSI + motion sensor observations (red: ground truth, blue: SPKS estimate). Small yellow rectangular boxes indicate the location of the motion sensors on the floorplan.

using signal strength in an indoor environment. It also establishes the need of combining measurements from multiple sensors in order to improve the tracking accuracy. As shown in Figure 3.16, adding the motion sensors with RSSI improves the SPKS tracking accuracy in spite of the high false alarm and large variability of the motion sensors.

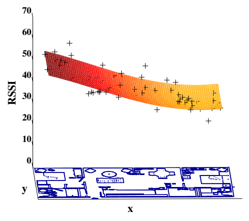
Table 3.3 compares the estimation accuracy of fixed-interval and fixed-lag SPKS techniques with Ekahau tracking engine in test Lab-II. The tracking accuracy of each estimator is demonstrated in terms of mean of RMSE and standard deviation (std) of RMSE for 20 different trials performed by various subjects. As is evident from the table, both the fixed-interval and the fixed-lag SPKS perform comparably but they clearly outperform the Ekahau over all the trials.

3.5.3 Test Lab-III

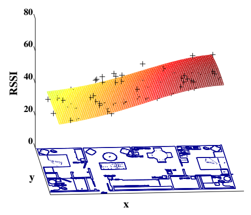
In 3.18(a)-3.18(i), we demonstrate an additional moving test at a third location, which is the real home of a person. This experiment is shown in this work to establish the superior accuracy of the SPKS based tracker compared to the commercial Ekahau positioning engine for room level localization. The size of Lab-III is 55 ft by 25 ft. The floorplan of this site is shown in Figure 3.17(a). Similar to test Lab I and II, this living lab is also



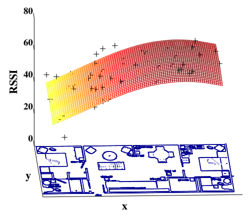
(a)



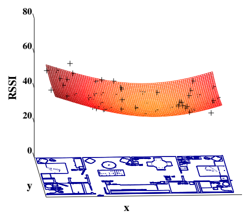
(b)



(c)



(d)



(e)

Figure 3.17: (a): Floor layout for test Lab-III. (b) to (e): Fitted RBF maps on top of the means of RSSI collected during calibration at *test Lab-III*.

equipped with 5 wireless access points placed at four corners and at the center. The entire floorplan is divided into 9 sections. In each section, an octagonal grid is formed to perform calibration. Due to a problem with the RSSI tags, a limited amount of calibration data was collected (only 212 RSSI measurements for the entire house). The RBF observation maps were learned from this small calibration data set and we would thus expect worse tracking performance corresponding to only room level localization accuracy. The fitted RBF maps are displayed in Figure 3.17(b)-3.17(e). One of the access points reported RSSI data packets only for nearby sections of the floor due to a tuning problem and hence can not be used for localization. As described, this experiment provides us an opportunity to determine the resiliency and robustness of our SPKS based tracking infrastructure in a challenging environment.

During the moving test, a subject walked on the calibration grid points at each section along a counter clockwise direction. Figure 3.18(a)-3.18(i) compare the Ekahau localization performance with that of the SPKS tracker. As seen in figures, the Ekahau tracking

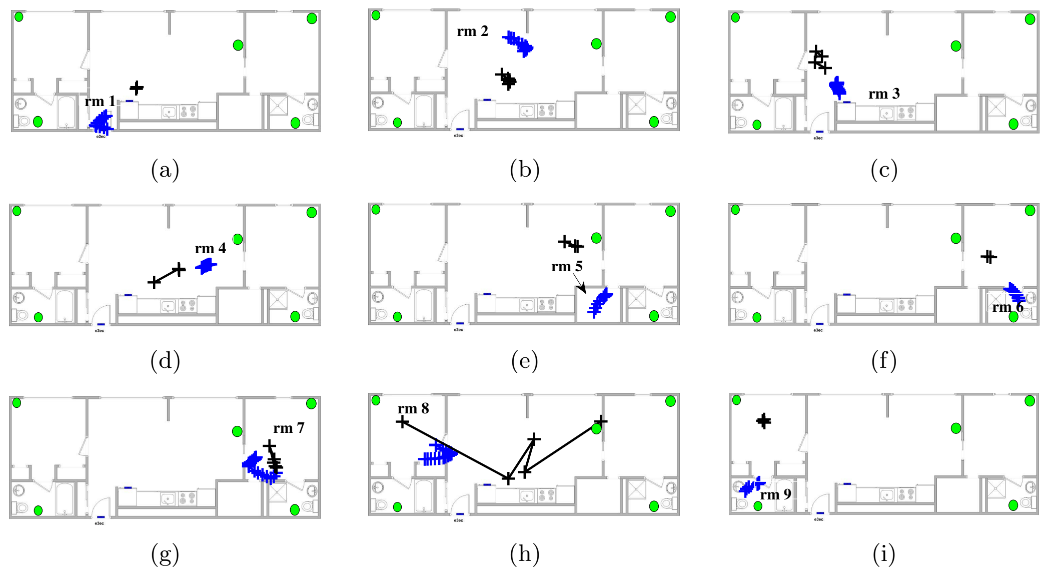


Figure 3.18: (a)-(i): Performance comparison between the Ekahau (black) and the SPKS (blue) at *test Lab-III* in terms of room-level accuracy. The position of the access points are shown by green circles on the floorplan.

engine fails to localize the person in the correct section except for a single case (as shown in Figure 3.18(g)). Most of the Ekahau estimates are randomly centered around the middle of the entire floor plan. However, the fixed-interval RTSSL-SPKS based tracker correctly localizes the person in all of the sections.

Table 3.4 summarizes the performance and superiority of the SPKS based tracker over other popular estimation techniques, including the EKF, EKS and SPKF, in terms of RMSE position error. As seen, the EKF, EKS, SPKF and SPKS all clearly outperform the Ekahau. It is also verified that the SPKS has clear performance advantage over the EKS. Table 3.5 displays the superior accuracy of the SPKS based tracker over the Ekahau in terms of room level localization. The room accuracy corresponds to the percentage of times the estimated location lies in the correct room (the section where the subject is present). Room-level accuracy is an important metric, as instead of precise localization, many real life applications operate at the level of rooms. The table verifies the ability of the SPKS tracker to operate precisely at the room-level even when the calibration data is small and some of the access points fail to operate as desired.

Table 3.4: Performance comparison of the SPKS with other estimators in terms of RMSE. Overall RMSE is calculated relative to the observed true trajectory over 50 different trials performed at various test labs and real houses. RMSE for test Lab-III is not shown as accurate ground truths were not available.

Estimator	**Overall**	**Lab-I**	**Lab-II**
Ekahau (RSSI)	23.88	28.4	20.22
EKF(RSSI)	12.32	11.73	13.41
EKS(RSSI)	9.54	9.49	10.2
SPKF(RSSI)	8.95	8.21	9.85
SPKS(RSSI)	6.45	5.68	8.38
SPKS(RSSI+IR motion sensor)	5.92	N/A	6.26
SPKS(RSSI+footswitch)	1.44	1.44	N/A

Table 3.5: Performance comparison of the SPKS with the Ekahau in terms of *room level localization*. Room level accuracy is defined as the percentage of times the estimated location falls within the desired room. Overall room level accuracy is calculated by ensemble averaging over 50 different trials performed at various test labs and real houses.

Estimator	**Overall**	**Lab-I**	**Lab-II**	**Lab-III**
Ekahau (RSSI)	58.8	54.32	62.3	60.95
SPKS(RSSI)	82.21	84.02	75.18	95.22
SPKS(RSSI+IR motion sensor)	83.2	N/A	77.08	95.34
SPKS(RSSI+footswitch)	97.78	97.78	N/A	N/A

3.6 Discussion and Future Work

A new method and system have been developed for RSSI based indoor localization and tracking. Instead of using simple fingerprinting or a fixed a priori distribution for the RSSI tags, an observation function is generated from the RSSI calibration data by fitting nonlinear maps between the known calibration locations and RSSI mean values. The RSSI maps are incorporated into a Bayesian framework that fuses all sensor measurements with a simple dynamic model of walking. The dynamic model consists of a random acceleration model augmented with repulsive forces to account for the room-wall configurations. For the Bayesian inference, we primarily consider the proposed fixed-interval and fixed-lag SPKS estimators, which are derived from first principles in Chapter 2. The SPKS trackers can provide superior interpolation using both past and future observations and hence provide considerable improvement in the tracking accuracy compared to the standard

SPKF. The SPKS tracker can accommodate a multi-rate processing where state estimates are determined at a higher rate (e.g., every second) while the RSSI observations occur at a slower update rate. Missing observations are also easily handled by the approach.

While the primary sensors are Wi-Fi tags, the approach can also incorporate multiple types of sensors. In the current implementation, both IR motion sensors and simple foot-switches were incorporated. As a predominantly software solution, the approach provides the flexibility to incorporate sensors from multiple manufacturers. Performance was evaluated in a number of “living laboratories”, where the tracking accuracy was demonstrated to be superior to the available industry positioning engine developed by the Ekahau Inc. The proposed system is currently being deployed into a number of houses in order to continuously monitor elderly for clinical purposes.

Although the SPKS based approach is capable of accurate tracking, several factors may limit the performance of the proposed approach. RSSI noise, spatial variability, and sampling rate all affect accuracy. Complexity of the position-RSSI correspondence due to the severe multipath, NLOS propagation, metal reflection, interference from other Wi-Fi devices, and RSSI noise can pose significant challenges in successful generation of the observation maps. The accuracy of the RSSI observation maps also depend on the number of RSSI measurements collected over the entire indoor environment during calibration. A sufficient number of calibration points at different locations must be selected to capture the full spatial characteristics of the environment. The octagonal positioning used in this study may or may not be sufficient. While a complex spatial variability in the RSSI makes fitting the observation maps more difficult, the complexity of the maps is precisely what makes tracking possible. A relative flat observation map is non informative. The actual shape of the RSSI map is also influenced by the type, location, and orientation of the RF access points, which must be properly located and tuned so that RSSI can be acquired over the entire house for each access point. During tracking, accuracy is further influenced by the RSSI noise and the sampling rate, which both contributes to location uncertainty. The $4 - 8\,\mathrm{s}$ sampling rate in this study was due to the hardware limitations. Clearly a faster sampling rate is always preferred. Incorporation of non-RSSI sensors may also affect accuracy. In this work, IR motion sensors and foot-switches were used. However, we only incorporated a very simple model that did not account for beam patterns, false alarms, or latency.

Future directions may include refined models of walking motion and better observation models for the RSSI based tracking method. In this context, one can investigate whether the RSSI based tracking framework can be augmented by a digital compass and accelerometer sensors which may provide an additional orientation information. The orientation information can be used either to improve the current dynamic model by changing to a more sophisticated "heading" model or to learn orientation dependent RSSI radio maps during calibration. A detailed sensor characterization test for motion sensors may also be used which will help to understand the beam pattern and correspondingly to develop a better likelihood model. In addition, self calibration based simultaneous localization and mapping (SLAM) approach can also be designed, whereby the parameters of the RSSI maps are continuously updated to account for changes in environment or to even avoid the initial off-line calibration procedure. Tag-free solutions can also be investigated that would provide an alternative to RSSI, allowing for unobtrusive localization without the use of body worn tags.

Chapter 4

A Tag-free Solution to Unobtrusive Indoor Tracking Using Wall-Mounted Ultrasonic Transducers

4.1 Overview

In the previous chapter we applied the sigma-point Kalman smoother (SPKS) to track a person in an indoor environment using RSSI observations. The RSSI based tracking framework is tag-based where a user carries a radio transceiver tag while walking. In this chapter, we propose a novel tag-free solution for indoor tracking using ultrasonic transducers or sonar units. The tracking is performed utilizing sonar-range maps learned from the calibration data. We also apply the *simultaneous localization and mapping (SLAM)* algorithm to track a person which simultaneously estimates the location of the wall-mounted sonar units with user positions.

This chapter is organized as follows. Section 4.2 discusses the use of ultrasonic sensors in tracking robots and underwater vehicles. It then summarizes the various components of the SPKS based Bayesian inference algorithm used in our tracking system. Section 4.3 describes the data acquisition hardware, which captures the analog echoes. Section 4.4 shows the signal processing steps, which compute the 1D range of the moving person. Section 4.5 describes the proposed SPKS based Bayesian inference algorithm which uses learned observation maps and range measurements from active sonar units to track a person. Section 4.6 implements two types of SLAM algorithms which simultaneously computes the observation model parameters with the user state. Section 4.7 incorporates the range observations from passive sonar units into the system and demonstrates its performance. Finally, Section 4.8 summarizes our contribution.

4.2 Introduction

There is a great demand for accurate indoor localization and tracking systems with the recent advancement of context aware technologies. The next generation mobile applications often need to accurately locate a person in order to provide information about its surroundings. As GPS is unable to provide location based information in an indoor environment, researchers are actively investigating various sensing technologies to develop an accurate indoor positioning system. In this light, a plethora of sensors such as infra-red (IR), ultra-sonic badges and radio-frequency (RF) based wireless local area network (WLAN) have been proposed and widely used in indoor localization [40–42, 48, 49, 89]. In Section 3.2, we surveyed a number of current indoor people-tracking systems and their architectures. All these different positioning systems require the user to carry a body-borne receiver tag while walking. For example, a person to be tracked has to carry a small IR tag in the *Active Badge* system that periodically transmits IR waveforms to a network of sensors [40]. In the *Cricket* system, which uses RF and ultrasonic sensors, the person carries a listening device that uses time of flight (TOF) difference between the RF and ultrasonic waves in order to locate the user [42]. For the wireless local-area network (WLAN) based tracking systems, such as *Place-Lab* and *Ekahau*, an user needs to carry a RF transceiver in order to measure the Wi-Fi signal strength from multiple access points [45, 46, 48]. Although the tag-based tracking systems can generate satisfactory tracking accuracy, in some applications (e.g., monitoring activities of daily living of seniors in independent living facilities), wearing a tag may be seen as undesirable or simply a nuisance. In addition, most of the tag based systems require extensive calibration. Despite these limitations of the tag based approach, there are even fewer options for tag-free monitoring system. Arrays of infra-red (IR) motion sensors may be employed to determine the region-level location but the multiple IR systems are prohibitively expensive and complicated to install. Mitsubishi Electric Research Laboratories, for example, has a prototype system that requires over 200 IR sensors to be installed in the ceiling of a large office building [90]. Video-based tracking draws from advances in automated surveillance and can often be very effective, though performance may degrade with complicated background clutter and other non-ideal environments [89, 91, 92]. The primary disincentive to video-based tracking, however, is privacy concerns. People don't want a video camera in their home, constantly monitoring their activities. With these concerns in mind, we propose utilizing ultrasonic range sensors for indoor tracking.

The study of well-known creatures like bats and dolphins shows that they use *time-of-flight (TOF)* ranging approach to calculate distance from obstacles. Sound navigation and ranging (*sonar*) method has traditionally been used to navigate an underwater vehicle like *submarines* [93, 94]. It has also extensively been used to determine the distance and direction of an underwater vehicle and to enable communication between vehicles [95, 96]. Although the acoustic frequency used by the sonar technique varies from low (infrasonic) to high (ultrasonic), the ultrasonic frequency based sonars are the most prevalent. In addition to underwater navigation, the ultrasonic sonars have also found applications in tracking position and orientation of indoor mobile robots [77, 97]. The robot or vehicle is generally equipped with multiple sonar transducers which transmit ultrasonic wave at periodic intervals and then waits for the corresponding *echo*. The accurate estimation of propagation time, defined as the time duration between the instant of signal transmission and its reception, is applied to compute the range of the surrounding obstacles. Each estimated range information is then used to triangulate the robot's pose [98–100]. Recently, the probabilistic Bayesian inference framework has also been employed to infer the robot's pose from multiple sonar range measurements [77, 78, 101, 102]. Ultrasonic sensors are also applied in indoor people-tracking using a tag-based approach. A commercially available tag-based tracking system by Sonitor uses ultrasonic technology for tracking a person [44]. The user carries a small tag which emits its identification number via ultrasonics. Detectors scattered throughout the environment receive this information and triangulate the user's location.

In this chapter, we present a novel tag-free solution for indoor location tracking that utilizes low-cost wall-mounted ultrasonic transducers. The sonar modules transmit an ultrasonic wave at a periodic interval and capture analog echoes, which are then digitized and analyzed in order to estimate the 1D range of the moving person. Simplistically, the range corresponds to the strongest echo received after reflection. The proposed system utilizes a number of signal processing techniques including Bandpass filtering, Hilbert transformation, and background subtraction to remove interference from other objects (*e.g.*, chairs) in the room. An adaptive threshold is then used to determine the locations of strong echoes. Finally, clustering is performed to determine several candidate range estimates. These candidate range estimates from multiple transducers provide the observations for the subsequent tracking algorithm to determine a person's 2D position and velocity.

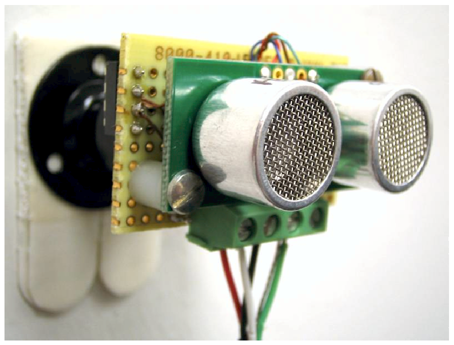
We implement a probabilistic Bayesian inference algorithm based on a fixed-lag sigma-point Kalman smoother (FL-SPKS) in order to localize and track a person. The proposed SPKS fuses a model of walking motion, room-wall configurations and all available sonar range observations in order to track a person. Initially a random acceleration-based human walk model is used as the dynamic model of motion. Later, we apply the so-called "*coordinated turn (CT) model*" [86], as it is more suitable to mimic the way a human walks. The two dimensional CT based target maneuver model relies on target *kinematics*, in contrast to the random acceleration model, which is based on random processes. The CT model also takes into account the spatial coupling between dimensions. The target maneuver model is augmented with a room-wall configuration involving a potential field created throughout the indoor environment in order to repel motion away from walls. This chapter provides a more detailed description and analysis of our methods that was presented in [103].

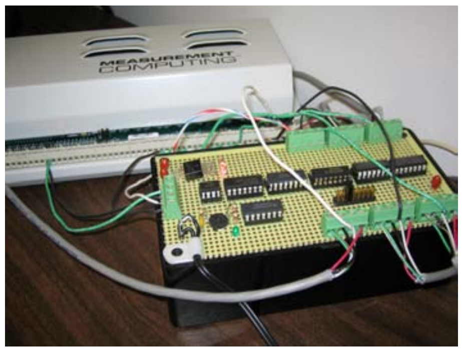
Instead of using an observation model known *a priori*, we use two different methods to estimate the parameters of the observation model. The first method is similar to what we have implemented for RSSI based indoor tracking. This method uses the *Radial-Basis Function (RBF)* network to learn a nonlinear mapping between known user locations and range measurements. Note that each ultrasonic transducer has its own observation map and these maps are fit during a separate offline calibration phase. Unfortunately the whole calibration process is tedious, time consuming and also requires significant manual effort.

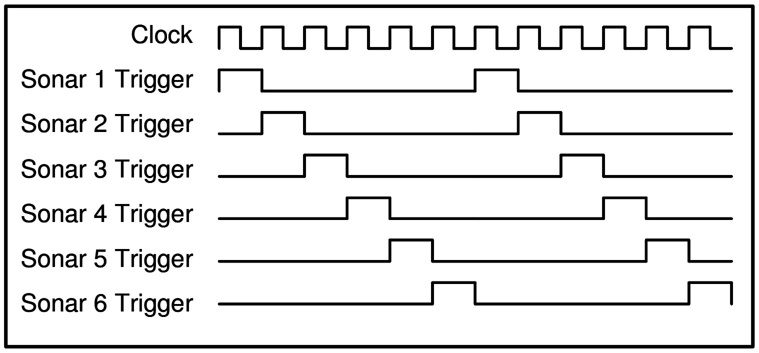
In addition, any error in measuring the ground-truth directly affects the accuracy of the learned model, which effectively degrades the state estimates. Hence our next phase of research focusses on investigating a SLAM based "*self-calibration*" technique, whereby the parameters of the observation models are continuously updated in the tracking phase to account for changes in the environment or in order to make the initial off-line calibration procedure redundant. The task is to simultaneously estimate the state of the person (2D position, velocity and turn rate) and the parameters of the observation model. Parameters correspond to the 2D sonar module locations along with a correction factor for the speed of sound to account for multipath, reflection/refraction and other measurement errors. A dual SPKS framework is introduced to tackle this problem that works by alternating between using one SPKS to localize the user given the current estimated parameters, and a second SPKS to update the estimate of the parameters given the current position of the user.

An ultrasonic sensor is in "*active*" mode when it triggers an ultrasonic signal and records the primary echo after it bounces off walls/targets. In "*passive*" mode, an ultrasonic sensor never transmits but only records the indirect reflections of an ultrasonic signal coming from an active unit. Initially we only consider the range observations recorded by the active sonar transducers for the tracking purposes. In the last phase of this work, we investigate whether we can improve the tracking accuracy by incorporating the passive range measurements with the already used active observations. Although the same SPKS tracking algorithm is employed for this task, the dimension of the observation model is increased in order to include the additional passive measurements.

Experiments were performed at a test lab where the performance of our tag-free sonar based solution was compared with the commercial tag based *Ubisense* positioning engine [49]. From the simulation results, it is clear that ultrasonic sensors can be successfully used to accurately track a person in an indoor location. The estimation accuracy exhibited by our proposed tag-free tracker is shown to be comparable with the Ubisense positioning engine. The SLAM approach, where the model parameters are learned online in conjunction with the state estimates further improves the tracking accuracy. Finally, we also demonstrate that incorporating the passive sonar ranges into the location tracking


(a)


(b)



(c)

Figure 4.1: (a) Devantech Sonar Module. (b) Custom trigger generator board. (c) Sequential triggers of ultrasonic sensors.

indeed helps to localize the person with higher accuracy.

4.3 System Hardware and Data Acquisition

The sonar module selected for tracking is a low cost unit manufactured by Devantech Inc., which transmits ultrasonic pulses at the rate of 40 kHz. The ultrasonic signal emitted from each sonar unit can travel up to a distance of 14 ft. The chosen module has two separate transducers, one for transmitting the ultrasound and the other for listening the corresponding echo. Each ultrasonic unit is mounted using a gimbal joint as shown in Figure 4.1(a) in order to help translation and rotation along any direction on the wall. The analog *echo* return is tapped from the output of the preamplifier stage lies at the back of the sonar circuit board. For testing, 6 ultrasonic units were mounted on the walls

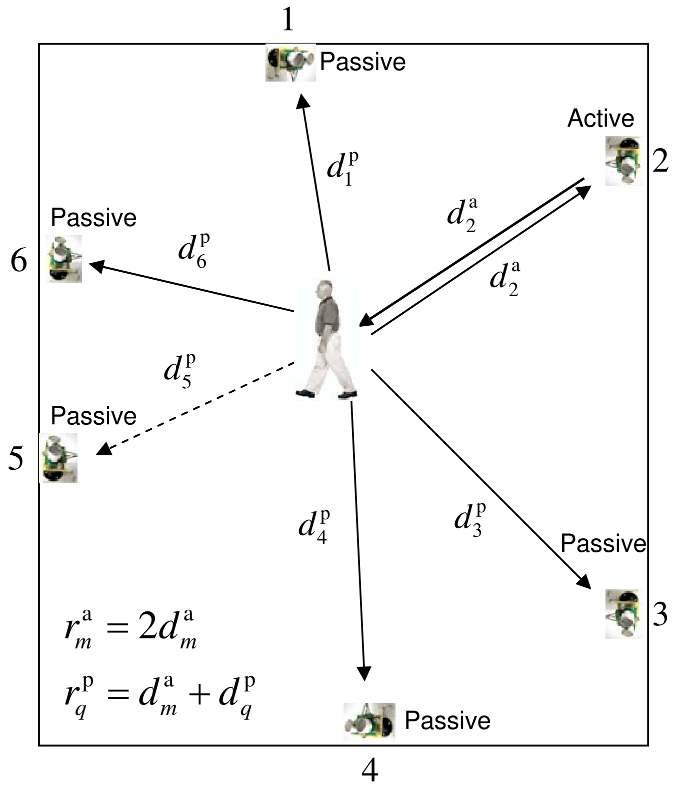


Figure 4.2: This figure illustrates the approximate location of six ultrasonic sonar modules installed in the test-area. Note that the asymmetry was due to constraints with office equipment in the room. A primary range observation r_m^{a} recorded by the current m-th active sonar unit is equal to twice the distance, d_m^{a}, between the sonar unit and the person. The range measurement r_q^{p} recorded by the p-th passive unit is equal to the sum of distances, d_m^{a} and d_q^{p}, where d_q^{p} denotes the distance between the person and the passive unit.

of a single room as illustrated in Figure 4.2.

The prototype system consists of a standard PC running MATLAB control software connected to a 8 channel, 16 bit DAQ manufactured by *Measurement Computing*. The DAQ is responsible for both controlling a custom "trigger circuit" and for digitizing all analog sonar returns. Note that the signal is digitized using the DAQ at a sampling rate of f_{S} ($f_{\mathrm{S}} = 250\,\mathrm{kHz}$) to avoid aliasing. The trigger circuit, as displayed in Figure 4.1(b), uses 555 multivibrators and flip-flops in order to fire the multiple ultrasonic sensors in a sequential manner. A positive triggering edge generated from the CPU passes through the trigger generator board which in turn fires a sonar unit. The corresponding unit becomes "active" and transmits an ultrasonic wave. The circuit board then waits for a predetermined amount of time T_f mill-seconds (ms) (in this case, $T_\mathrm{f} = 30\,\mathrm{ms}$) for the active unit to record the primary echoes. Simultaneously, the other sonar units are in "passive" mode and only record indirect reflections or shadows coming from the active unit. After T_f ms, the circuit board issues another trigger pulse to fire the next sonar unit, which goes into the active mode. This procedure runs in a sequential manner until all 6 installed ultrasonic sensors have been fired. The sequential pattern of sonar triggering with clock cycles for 6 sensor modules is displayed in Figure 4.1(c). Finally, all the collected ultrasonic reflections with the synchronization timing information are transferred to the CPU for further processing. Note that there exists a random time delay of T_d ms (the mean and standard deviation of T_d, $(T_\mathrm{d})_\mathrm{mean} = 40\,\mathrm{ms}$ and $(T_\mathrm{d})_\mathrm{std} = 50\,\mathrm{ms}$, are computed by averaging over 1000 trials) between the CPU triggering and when the actual sonar firing occurs. Exact knowledge of T_d is essential in order to synchronize the data timing and is provided by the trigger generator board. All the ultrasonic units are custom cabled with the trigger generator using an OP-AMP cable driver circuit.

4.4 Signal Processing: Range Calculation

The following signal processing steps are carried out sequentially on a received ultrasonic signal in order to determine range candidates, which provide the observations for the subsequent tracking algorithm.

4.4.1 Bandpass Filtering

A raw ultrasonic reflection profile is shown in Figure 4.3(a). The raw data is collected at the end of the preamplifier stage of a sonar module. In order to make the signal base-band, the raw data is passed through a eighth-order zero-phase Butterworth-Bandpass filter with lower and upper cutoff frequencies of f_L and f_H respectively (in this case, $f_\mathrm{L} = 37\,\mathrm{kHz}$ and $f_\mathrm{H} = 42\,\mathrm{kHz}$). The Bandpass filter also removes the DC component of the input signal. As the frequency of the emitted ultrasonic wave is about 40 kHz, the cutoff frequencies are chosen accordingly so that they can maintain the overall signal response while keeping

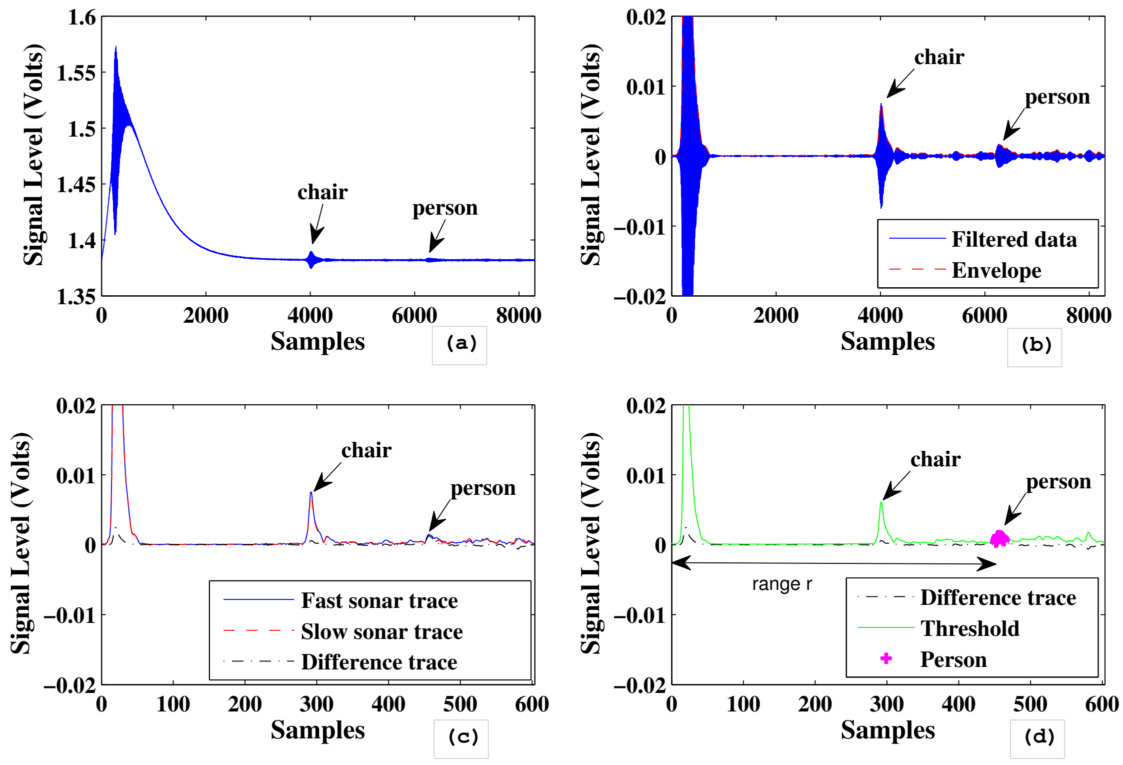


Figure 4.3: Signal processing steps performed on each reflected signal in order to identify the potential range candidates, (a) Raw analog signal, (b) Band-pass filtered signal with envelope, (c) Sonar traces with background subtraction, (d) Adaptive threshold (*magenta* shows the location where the sonar trace exceeds the threshold).

the distortion negligible. The Bandpass filtering is also the required preprocessing step for envelope detection, which needs a base-band input. Figure 4.3(b) depicts the Bandpass signal. Note that the signal consists of echoes from background objects and a moving person. In fact, the echo return from the nearest object (a chair) is larger than the echo from the person we are trying to track.

4.4.2 Envelope Detection and Downsampling

The output signal from the Bandpass filter is passed through a envelope detection block in order to extract its instantaneous amplitude. The envelope detection method works by creating a complex signal called an "*analytic signal*" of the input using a Hilbert transformer [104]. The analytic signal consists of a real part which is the original signal

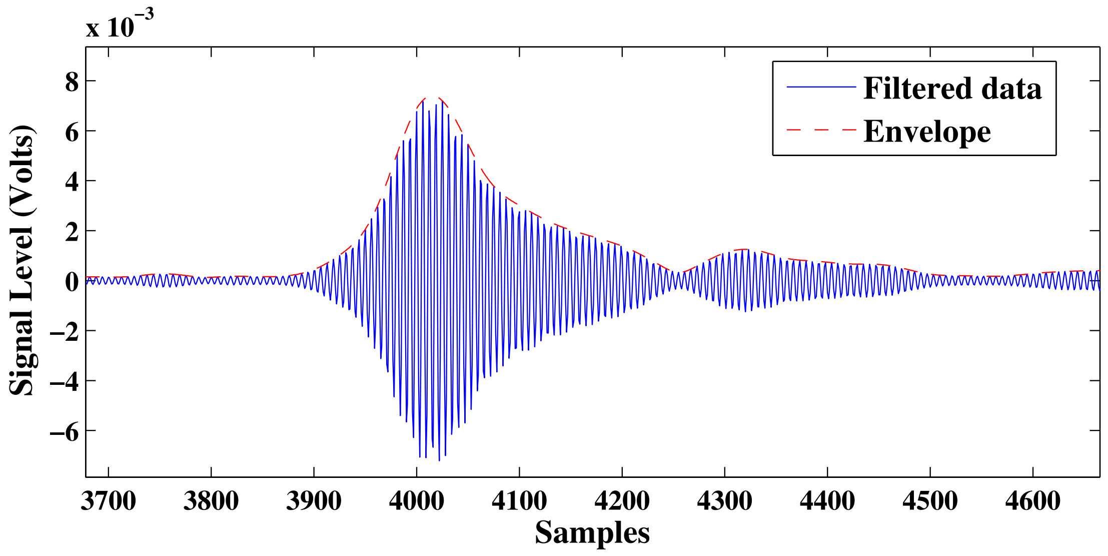


Figure 4.4: Zoomed version of the sonar envelope.

and an imaginary part which is the Hilbert transform of the original signal. An *n-point* Fast-Fourier Transform (FFT) is used to calculate the Hilbert transform. In order to match the delay caused by the Hilbert transform, the original signal is time-delayed before being added to the imaginary part of the signal. The instantaneous amplitude of the signal is then extracted by calculating the absolute value of the analytic signal. In order to reduce jitter and smooth the envelope, the result is also subjected to a low-pass filter.

In our experiment, signal downsampling is performed after the envelope detector. Recall that the sampling rate of the A/D converter while digitizing the received ultrasonic signal is chosen to be sufficiently high (250 kHz) in order to prevent aliasing. As the large number of samples add higher processing/storage requirements and are also unnecessary for future processing, a *decimation* step is followed on the extracted envelope of the sonar signal. The decimation step applies a low pass filter on the input data and then resamples the resulting smoothed signal at the rate of $(f_{\mathrm{S}})_{\mathrm{resamp}}$ kHz (we have adopted $(f_{\mathrm{S}})_{\mathrm{resamp}} = 20$ kHz). Note, the downsampled signal corresponds to a resolution of 0.685 inch/sample, which we have found sufficient for our application. Figure 4.3(b) displays the signal envelope in "*red*" with the Bandpass signal. Figure 4.4 is the zoomed version of Figure 4.3(b), which demonstrates the performance of Hilbert transform based

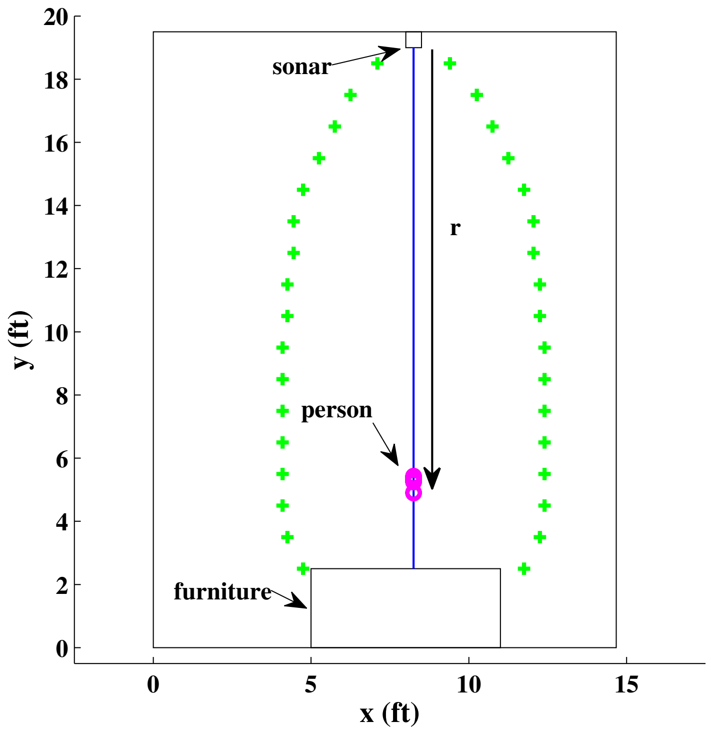


Figure 4.5: This figure maps a single range into the 2D location of a subject (*magenta*: location of the person, *green*: approximate sonar bandwidth).

envelope detection method. This final Bandpass, demodulated, and downsampled signal is referred to as the *sonar trace*, $s^v_{m,k}(n)$, where m indicates the module index, k is the time index for the 2 Hz cycle rate, n is the time index of the trace sampled at 20 kHz, and $v \in \{a, p\}$ indicates whether the trace was an "active" or "passive" return.

4.4.3 Background Subtraction

The signal envelope may consist of echoes emanating from the moving person and background static objects (e.g., chairs in the room). The challenging problem lies in computing the range of the person by removing echoes from static objects which may be closer in distance and appear larger than the echo of interest. Simplistically, the range corresponds to the timing at which the maximum energy from an echo is received. However, due to a phenomenon called *multipath interference*, the reflected signal from a target can take both direct and indirect paths to reach at the receiver. The variation of path lengths produces

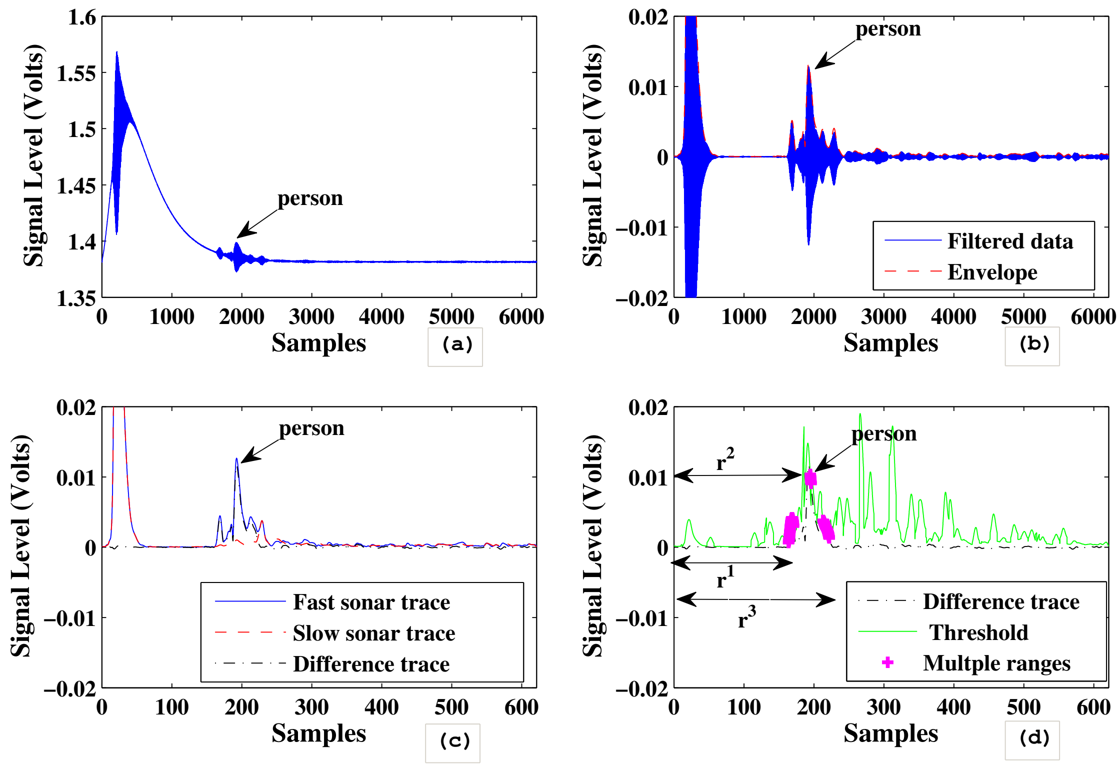


Figure 4.6: This figure depicts an example where the sonar trace exceeds the threshold at multiple locations, (a) Raw analog signal, (b) Band-pass filtered signal with envelope, (c) Sonar traces with background subtraction, (d) Adaptive threshold (*magenta* denotes multiple range candidates).

phase differences among the waves reaching at the receiver which may cause signals to interfere constructively or destructively with one another. As a result, the amplitude of the sonar envelope becomes higher and lower with time, which indicates that the strongest echo does not always correspond to the range of the person. In order to solve this problem, we apply the background subtraction by taking the difference of two time-averaged versions of the sonar trace. Specifically,

$$s^{\text{fast}}_{m,k}(n) = \lambda^{\text{f}} s^{\text{fast}}_{m,k-1}(n) + \left(1-\lambda^{\text{f}}\right) s^{v}_{m,k}(n) \tag{4.1}$$

$$s^{\text{slow}}_{m,k}(n) = \lambda^{\text{s}} s^{\text{slow}}_{m,k-1}(n) + \left(1-\lambda^{\text{s}}\right) s^{v}_{m,k}(n) \tag{4.2}$$

$$s^{\Delta}_{m,k}(n) = s^{\text{fast}}_{m,k}(n) - s^{\text{slow}}_{m,k}(n), \tag{4.3}$$

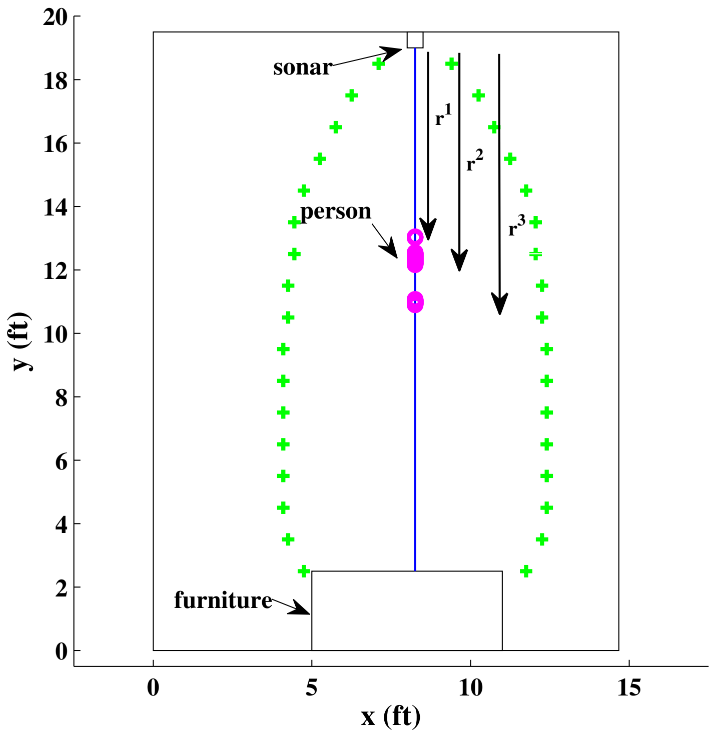


Figure 4.7: Multiple range candidates correspond to different locations of a subject in 2D (*magenta*: probable locations, *green*: approximate sonar bandwidth).

where λ^{f} and λ^{S} are chosen to provide 1 s and 15 s time constants respectively. The difference trace $s^{\Delta}_{m,k}(n)$ preserves only the strong echoes which possess the potential as range candidates. These traces are illustrated in Figure 4.3(c).

4.4.4 Adaptive Threshold Selection

The next step involves determining an adaptive threshold that can be used on the difference signal to determine the locations of strong echoes. While the background subtraction with time averaging helps remove interference, signals still fluctuate in time. The variance of this fluctuation is also affected by how close the echo is to the transceiver, as well as the size and location of the person relative to other objects. We thus apply a threshold

$t_{m,k}^{\mathrm{th}}(n)$, which adapts to the standard deviation of the difference signal.

$$t_{m,k}^{\mathrm{var}}(n) = \lambda^{\mathrm{f}} t_{m,k-1}^{\mathrm{var}}(n) + \left(1 - \lambda^{\mathrm{f}}\right) \left[s_{m,k}^{\Delta}(n)\right]^2 \tag{4.4}$$

$$t_{m,k}^{\mathrm{th}}(n) = \alpha \sqrt{t_{m,k}^{\mathrm{var}}(n)}, \tag{4.5}$$

where α (we have used $\alpha = 4$) is a threshold specific constant. Figure 4.3(d) displays the adaptive threshold in "green". The *magenta* blob demonstrates the location in the difference trace that exceeds the threshold, indicating the range, r, of a subject from the sonar unit. In the above figure, there is just one magenta blob, which maps the location of the subject in 2-D to within a circle of radius r as shown in Figure 4.5. However, there arises many cases when the signal envelope exceeds the threshold at multiple locations. Figure 4.6 displays one typical example, which has three *magenta* blobs corresponding to ranges r^1, r^2 and r^3. Figure 4.7 demonstrates how multiple ranges denote three different locations of the subject inside the room.

4.4.5 Clustering

The multiple strong echoes, which rise above the threshold are clustered and the center of each cluster is regarded as the potential range candidate. Note that we do not have any prior knowledge of the number of clusters each signal might contain and hence we have implemented the iterative self-organizing data analysis algorithm ("*ISODATA*") based clustering technique for this purpose. The *ISODATA* is an iterative version to the popular k-means algorithm [105], which can automatically select the number of clusters based on some statistical criterions. The operation of the algorithm can be summarized as follows:

1. Perform k-means clustering
2. Split any clusters into two whose elements are sufficiently dissimilar
3. Merge any two clusters whose elements are sufficiently close
4. *if* there is any change in the total number of clusters, go back to (1)
   else end.

We have selected the following parameters in order to compare the elements within a cluster and between two clusters

- σ_c^2: Maximum spread of each cluster permitted before splitting
- D_c: Minimum distance between two clusters before merging

Note that the above parameters are application specific and are chosen using a 10 fold cross validation procedure based on the criterion that maximizes the tracking accuracy. The values of σ_c^2 and D_c used in this application are shown in Table 4.1. For more details of the clustering algorithm, one can refer to [105]. The centroids of these clusters represent the final range candidates, $r_{m,k}^i$, where i is the candidate index. The set of all range candidates from all sonar modules, both in active and passive mode, are then collectively fed as observations $\boldsymbol{z}_k$ either to the observation map based Bayesian estimator (Section 4.5) or to the SLAM estimator (Section 4.6) for tracking the person.

4.5 Indoor Tracking Using Range Observation Maps

In this section, we apply the SPKS based recursive Bayesian estimation framework using range observations and a predictive model of human walking to perform tag-free indoor tracking. Note, here we have only considered range observations from active sonar units.

4.5.1 Recursive Bayesian Estimation Framework

Recall that the problem of recursive Bayesian estimation can be cast in terms of estimating the state $\boldsymbol{x}_k$ of a standard discrete-time nonlinear dynamic system,

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) \tag{4.6}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{n}_k\right). \tag{4.7}$$

In the following, we will define the different components of the above equations in the context of ultrasonic range based tracking application.

4.5.2 Dynamic Model

We consider a simple random acceleration based model [85, 86] and a CT model with unknown turn rate ω [86] as the system dynamic model $f(.)$. Both of these models are augmented with a room model involving a potential field created throughout the indoor environment in order to repel estimated motion away from walls. The potential field based random acceleration model used here is exactly the same as that used in the RSSI based tracking algorithm, which can be found in Section 3.3.1. The random acceleration model assumes that the target acceleration is an independent (i.e., "white noise") random process and the target motion is uncoupled (or weakly coupled) across different coordinates. Although this model is fairly simple to implement, a target maneuver is rarely independent with respect to time. Hence in the next step, we consider a "*CT*" model which relies on target kinematics and also takes into account spatial coupling across coordinates. This "heading" based model is also more realistic given people tend to walk forward (or backward) instead of sideways. In the following section, we provide a short description for each of these models.

Random acceleration model

The state vector at time k corresponds to person's 2D position and velocity

$$\boldsymbol{x}_k = \left[\begin{array}{cccc} x_k & y_k & v_{x_k} & v_{y_k} \end{array} \right]. \tag{4.8}$$

This model is driven from time k to $k+1$ by a white noise random process defined as follows:

$$x_{k+1} = x_k + \delta T v_{x_k} + \frac{\delta T^2}{2} F_{x_k}\left(x_k, y_k\right) \tag{4.9}$$

$$y_{k+1} = y_k + \delta T v_{y_k} + \frac{\delta T^2}{2} F_{y_k}\left(x_k, y_k\right) \tag{4.10}$$

$$v_{x_{k+1}} = \lambda v_{x_k} + \delta T F_{x_k}\left(x_k, y_k\right) + \left(1-\lambda\right) v_{p_{x,k}} \tag{4.11}$$

$$v_{y_{k+1}} = \lambda v_{y_k} + \delta T F_{y_k}\left(x_k, y_k\right) + \left(1-\lambda\right) v_{p_{y,k}}, \tag{4.12}$$

where $F_{x_k}\left(x_k, y_k\right)$ and $F_{y_k}\left(x_k, y_k\right)$ are the x and y components of the resultant potential field $\boldsymbol{F}_{\mathrm{r}}(x_k, y_k)$ at time k. For more details about the potential field, refer to Section 3.3.1.

The parameter λ smoothens the changes in velocities and also ensures that the variance of random process remains bounded. The Euler integration time $\delta T = 0.5\,\mathrm{s}$. The process noise is white Gaussian with zero mean

$$\boldsymbol{v}_{p,k} = \begin{bmatrix} v_{p_{x,k}} & v_{p_{y,k}} \end{bmatrix}. \tag{4.13}$$

Coordinated turn (CT) model with unknown turn rate

This model is based on the assumption that the target moves with nearly constant angular rate ω perturbed by random noise. The model uses target kinematics which takes into account the spatial coupling across x and y coordinates. In contrast to the random acceleration model, the CT model includes the *turn rate* ω_k as an extra component in the state to be estimated. The new state vector is defined as:

$$\boldsymbol{x}_k = \begin{bmatrix} x_k & y_k & v_{x_k} & v_{y_k} & \omega_k \end{bmatrix} \tag{4.14}$$

The discrete time version of the CT model can be described as follows:

$$x_{k+1} = x_k + \frac{\sin(\bar{\omega}_k \delta T)}{\bar{\omega}_k} v_{x_k} - \frac{1-\cos(\bar{\omega}_k \delta T)}{\bar{\omega}_k} v_{y_k} + \frac{\delta T^2}{2} F_{x_k}(x_k, y_k) + v_{p_{x,k}} \tag{4.15}$$

$$y_{k+1} = y_k + \frac{1-\cos(\bar{\omega}_k \delta T)}{\bar{\omega}_k} v_{x_k} + \frac{\sin(\bar{\omega}_k \delta T)}{\bar{\omega}_k} v_{y_k} + \frac{\delta T^2}{2} F_{y_k}(x_k, y_k) + v_{p_{y,k}} \tag{4.16}$$

$$v_{x_{k+1}} = \cos(\bar{\omega}_k \delta T) v_{x_k} - \sin(\bar{\omega}_k \delta T) v_{y_k} + \delta T F_{x_k}(x_k, y_k) + v_{p_{v_x,k}} \tag{4.17}$$

$$v_{y_{k+1}} = \cos(\bar{\omega}_k \delta T) v_{y_k} + \sin(\bar{\omega}_k \delta T) v_{x_k} + \delta T F_{y_k}(x_k, y_k) + v_{p_{v_y,k}} \tag{4.18}$$

$$\omega_{k+1} = \omega_k + v_{p_{\omega,k}}, \tag{4.19}$$

where $\bar{\omega}_k$ is the central point of ω_{k+1} and ω_k

$$\bar{\omega}_k = \frac{1}{2}(\omega_{k+1} + \omega_k). \tag{4.20}$$

Note that the CT model is nonlinear as ω_k is a state component. The process noise $\boldsymbol{v}_{p,k}$, defined as

$$\boldsymbol{v}_{p,k} = \begin{bmatrix} v_{p_{x,k}} & v_{p_{y,k}} & v_{p_{v_x,k}} & v_{p_{v_y,k}} & v_{p_{\omega,k}} \end{bmatrix}, \tag{4.21}$$

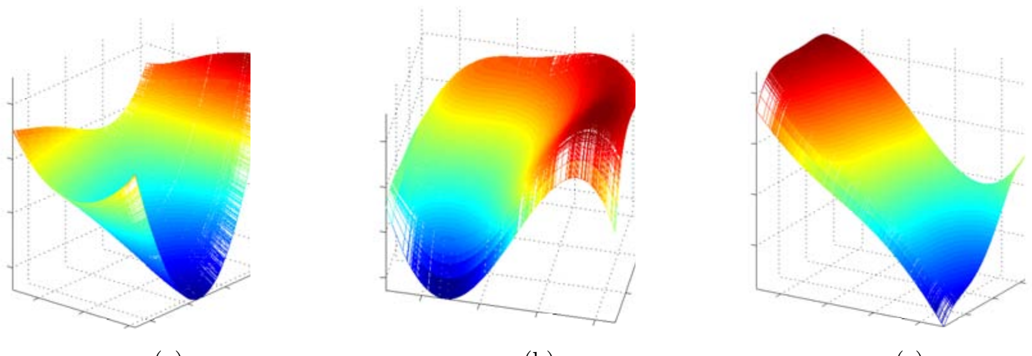
is zero mean white Gaussian and is used to model the perturbation of the trajectory from the ideal CT motion.

4.5.3 Observation Model

The observation model $h(.)$ in the recursive Bayesian estimation context can be viewed as a *generative model* which provides an observation prediction given the current state estimate. In case of tracking using ultrasonic transducers, the task of each observation model is to predict a circular arc with radius equal to person's range given person's current estimated position. In this work, we learn the position-range relationship from the training data. Similar to the observation map characterizing the RSSI-position relationship (as shown in Section 3.3.2), a nonlinear map is fit between known locations and range values extracted from each recorded signal. Note, as we have only considered active sonar measurements, the number of observation maps to be learned is $M = 6$. Although adding a training mechanism can increase the system overhead, it needs to be performed only once for a given location.

Data to fit the maps are first collected during a training phase. We have employed the *Ubisense* [49], which is an industry standard tag based tracking engine, in order to collect user's position in the calibration phase. In the training phase, a person carrying an Ubisense tag walks around the entire room for T_{tr} s. For our experiment, we have chosen $T_{\text{tr}} = 15\,\text{mins}$. While walking, the user follows a number of movement patterns namely *Lawnmower* (vertical and horizontal), *diagonal*, *circle*, *sine wave* so that the system is not biased to a particular data sequence. Typically, M number of ultrasonic transducers (we have used $M = 6$ for our experiment) are set up in the entire space to be calibrated in order to record the echoes. Figure 4.2 displays the location of 6 ultrasonic units installed in the test-area. Signal processing steps described in Section 4.4 are then applied to each recorded ultrasonic signal to obtain potential range candidates. A recursive map learning algorithm is used that iteratively learns an observation map by choosing a range candidate at time k which is closest to the map predicted range corresponding to the current user position. Before presenting the pseudo-code of the algorithm, we first describe the equations of the observation maps and its parameters.

A *RBF* network is used to fit a nonlinear map between the Ubisense estimated positions and range measurements obtained from each recorded signal. As we have already used the



(a) (b) (c)

Figure 4.8: Sonar maps fit between known locations and observed range values.

same RBF framework in Section 3.3.2 for tracking a person using RSSI, we are not going to provide much details here. However, in order to maintain continuity we summarize below the observation model equations. This *RBF* map is represented as,

$$z_{m,k} = h_m\left(x_k, y_k\right) + n_m \tag{4.22}$$

where $z_{m,k}$ is the observed range from sonar module m, $1 \leq m \leq M$, with noise n_m assumed to be Gaussian with zero mean and standard deviation determined from the calibration data. The user position at time k is denoted as

$$\boldsymbol{x}_{p_k} = \left[\begin{array}{cc} x_k & y_k \end{array}\right]. \tag{4.23}$$

The RBF observation map h_m for the m-th ultrasonic module is specified by

$$h_m\left(x_k, y_k\right) = \boldsymbol{W}_m^T \boldsymbol{K}_{m,\mathrm{G}}\left(\left[\begin{array}{cc} x_k & y_k \end{array}\right]; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\right), \tag{4.24}$$

where $\boldsymbol{K}_{m,\mathrm{G}}$ is the Gaussian kernel function [88] with mean vector $\boldsymbol{\mu}_m$ and covariance matrix $\boldsymbol{\Sigma}_m$, and $\boldsymbol{W}_m$ are the output layer linear weights. These parameters are generally learned during RBF training process with known user positions and range values. Figure 4.8 illustrates three different observation maps.

Each recorded signal corresponding to the ultrasonic transducer m provides a vector of probable range candidates $\boldsymbol{z}_{m,k}$ at time k

$$\boldsymbol{z}_{m,k} = \left[\begin{array}{ccccc} z_{m,k}^1 & z_{m,k}^2 & \ldots & z_{m,k}^i & \ldots z_{m,k}^{\mathrm{I}} \end{array}\right], \tag{4.25}$$

where $z^i_{m,k}$ is the centroid of the i-th cluster. The objective here is to devise a technique that can facilitate selection of a suitable $z_{m,k}$ from the given pool of $\boldsymbol{z}_{m,k}$, which can then be used for estimating the observation map h_m, as described before.

The pseudo-code of the recursive map learning algorithm for the m-th ultrasonic transducer is shown below. The algorithm takes the vector $\boldsymbol{z}_{m,k}$, which contains all the potential range candidates, and the user position $\boldsymbol{x}_k$ as inputs and generates an observation map h_m.

```
• Initialization
    – $j = 1$
    – For $k = 1 : N$
        ∗ Collect the potential range candidates $\boldsymbol{z}_{m,k}$.
        ∗ Select an initial $z^j_{m,k}$ from $\boldsymbol{z}_{m,k}$ based on the cluster size, highest signal
          level within a cluster, and proximity of previously calculated range.
    – End For
    – Form an augmented observation $\boldsymbol{z}_m$ and augmented position $\boldsymbol{X}_m$ as below:
```

$$\boldsymbol{z}^j_m = \begin{bmatrix} z^j_{m,1} & z^j_{m,2} & \ldots & z^j_{m,k} & \ldots & z^j_{m,\mathrm{N}} \end{bmatrix} \tag{4.26}$$

$$\boldsymbol{X}_m = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_k & y_k \\ \vdots & \vdots \\ x_N & y_N \end{bmatrix}. \tag{4.27}$$

```
• While $h^j_m \neq h^{j-1}_m$
    – Map learning
      Learn the RBF observation map $h^j_m$ for the $m$-th sensor using the correspon-
      dence between $\boldsymbol{X}_m$ and $\boldsymbol{z}^j_m$.
    – For $k = 1 : N$
```

∗ *Observation Prediction*

Use h_m^j to predict $\hat{z}_{m,k}^j$ according to the formula

$$\hat{z}_{m,k}^j = h_m^j \left(x_k, y_k\right). \tag{4.28}$$

∗ *Gating*

A suitable *"gating"* mechanism is applied on $\boldsymbol{z}_{m,k}$ using $\hat{z}_{m,k}^j$ to obtain $z_{m,k}^{j+1}$. It chooses the closest range candidate $z_{m,k}^{\text{min}}$ to the prediction $\hat{z}_{m,k}^j$. If no candidate is found to be within some radius (*e.g.*, 1 ft) then the entire set of $\boldsymbol{z}_{m,k}$ is discarded.

$$z_{m,k}^{j+1} = z_{m,k}^{\text{min}}. \tag{4.29}$$

– End *For*

– Generate a new observation vector based on the gated measurements

$$\boldsymbol{z}_m^{j+1} = \left[\begin{array}{cccccc} z_{m,1}^{j+1} & z_{m,2}^{j+1} & \dots & z_{m,k}^{j+1} & \dots & z_{m,\text{N}}^{j+1} \end{array}\right], \tag{4.30}$$

– increment j: $j = j + 1$

- End *While*

- After the convergence is achieved (generally within $5 - 6$ iterations),

$$\boldsymbol{z}_m^{\text{F}} = \boldsymbol{z}_m^j, \tag{4.31}$$

learn the final h_m using the correspondence between $\boldsymbol{z}_m^{\text{F}}$ and $\boldsymbol{X}_m$.

The above algorithm is applied for all M ultrasonic units to generate h_m, where $1 \leq m \leq$ M. We have further assumed that N number of training measurements are available to learn each observation map h_m. The final range $z_{m,k}$, RBF map h_m, and the observation noise n_m from each ultrasonic sensor are combined to form a multi-dimensional observation model,

$$\boldsymbol{z}_k = \begin{bmatrix} z_{1,k} & z_{2,k} & \dots & z_{m,k} & \dots & z_{\mathrm{M},k} \end{bmatrix} \tag{4.32}$$

$$\boldsymbol{h} = \begin{bmatrix} h_1 & h_2 & \dots & h_m & \dots & h_{\mathrm{M}} \end{bmatrix} \tag{4.33}$$

$$\boldsymbol{n} = \begin{bmatrix} n_1 & n_2 & \dots & n_m & \dots & n_{\mathrm{M}} \end{bmatrix}, \tag{4.34}$$

where $\boldsymbol{z}_k$ is the multi-dimensional range observations emanating from each sensor. Similarly $\boldsymbol{h}$ and $\boldsymbol{n}$ are the augmented RBF observation model and the measurement noise for sensors $1 \leq m \leq \mathrm{M}$.

4.5.4 SPKS Based Location Tracker

The SPKS based Bayesian inference algorithm is at the core of our tracking framework. Specifically, we implement the fixed-lag RTSSL-SPKS (details of the FL-SPKS algorithm is provided in Section 2.4), as it can operate at close to real time. In the FL-SPKS method, we obtain an estimate of the state at time $k - L$ given measurements up to and including time k, where k is the time index and constantly moves forward. The time lag L is a design specific constant and adds a fixed latency in the state estimator. The value of L used in our experiment is shown in Table 4.1, which corresponds to a lag of 2 s between the current observation and smoothed state. The fixed-lag RTSSL-SPKS divides the data into blocks (e.g., $N = \sum N_i$) and then sequentially performs the RTSSL-SPKS operation on the buffered blocks of data as they become available. The mathematical formulation of fixed-lag RTSSL-SPKS is shown in Section 2.4.4.

In order to incorporate the ultrasonic sensors into the SPKS location tracker, we need to solve a number of problem specific issues such as gating and time-varying observation dimensions. We first describe about those design issues before elaborately presenting the pseudo-code of the SPKS.

Iterative gated measurement recursion

This section deals with integrating a gating mechanism with the SPKS estimation process. The recorded ultrasonic signal for the m-th transducer at time k provides a set of range

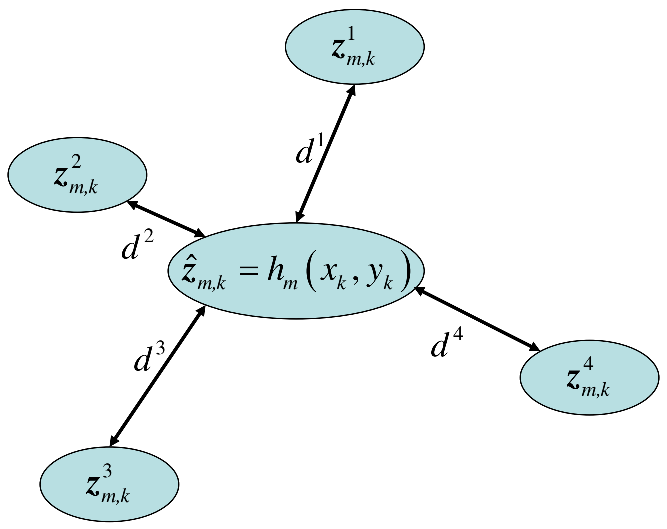


Figure 4.9: This schematic diagram demonstrates the "gating" procedure followed during the SPKS estimation process. As shown, the Euclidian distances are computed from the predicted range $\hat{z}_{m,k}$ to each of the potential candidates. The candidate $z^2_{m,k}$ is chosen as the distance d^2 between $\hat{z}_{m,k}$ and $z^2_{m,k}$ is the smallest.

candidates $\boldsymbol{z}_{m,k}$. *Gating* performs an iterated measurement update step in the forward SPKF algorithm to select a final range observation $z_{m,k}$ from the set of candidates in $\boldsymbol{z}_{m,k}$, for all $1 \leq m \leq$ M. Heuristically, the range measurement is first selected that is closest to the predicted measurement based on the predicted state $\hat{\boldsymbol{x}}_k^-$. Using this selected range, the state estimate $\hat{\boldsymbol{x}}_k$ is updated, and then the process is repeated until convergence of the state estimate.

Below we demonstrate an algorithmic outline of the gated measurement recursion at time k, which takes the time-updated state $\hat{\boldsymbol{x}}_k^-$ as input and generates an estimated state $\boldsymbol{x}_k$.

- *Initialization*

$$\hat{\boldsymbol{x}}_k^j = \hat{\boldsymbol{x}}_k^- . \tag{4.35}$$

- *While* $\boldsymbol{z}_k^j \neq \boldsymbol{z}_k^{j-1}$

  - *Observation prediction*:

$$\hat{z}_{m,k}^j = h_m\left(\hat{\boldsymbol{x}}_k^j\right), \ \ \forall m = 1 \ldots \mathrm{M} \tag{4.36}$$

  - *Gating*

$$z_{m,k}^j = z_{m,k}^{\min}, \ \ \forall m = 1 \ldots \mathrm{M} \tag{4.37}$$

    where $z_{m,k}^{\min}$ is the closest range candidate to the prediction $\hat{z}_{m,k}^j$. If no candidate is found to be within some radius (e.g., 0.5 ft) then the entire set of $\boldsymbol{z}_{m,k}$ is discarded. The gating mechanism is adopted in Figure 4.9 to avoid confusion from the four potential range candidates.

  - The gated and predicted measurements from all the sonar units are combined to form augmented observation vectors

$$\boldsymbol{z}_k^j = \left[\begin{array}{cccccc} z_{1,k}^j & z_{2,k}^j & \ldots & z_{m,k}^j & \ldots & z_{\mathrm{M},k}^j \end{array}\right] \tag{4.38}$$

$$\hat{\boldsymbol{z}}_k^j = \left[\begin{array}{cccccc} \hat{z}_{1,k}^j & \hat{z}_{2,k}^j & \ldots & \hat{z}_{m,k}^j & \ldots & \hat{z}_{\mathrm{M},k}^j \end{array}\right]. \tag{4.39}$$

  - *State Update*

$$\hat{\boldsymbol{x}}_k^{j+1} = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k^j\left(\boldsymbol{z}_k^j - \hat{\boldsymbol{z}}_k^j\right), \tag{4.40}$$

    where $\boldsymbol{K}_k^j$ is the Kalman gain at j-th iteration.

- increment j: $j = j + 1$

- End *While*
- After convergence, the final state estimate at time k is

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^j. \tag{4.41}$$

Note, the above recursion generally takes less than 5 iterations to converge.

Time-varying observation dimensions

As noted in the gated measurement recursion algorithm, the dimension of the observation vector varies over time. Hence one design challenge is to derive a SPKS based tracking framework that can adapt to the time-varying observation dimensions. In order to accomplish the task, we maintain an event-log during SPKS operations which notes down the specific sensors that generates an observation at time k. Note that the core SPKS algorithm remains unaffected by the dimension change of the observation vector. As per the event-log, an augmented observation vector, observation model and sensor noise vector are formed at each time k and used in the observation step of the SPKS estimation process. This process although very simple and straightforward needs careful logging of sensor events in order to accommodate the time-varying rate of data stream.

ISPKS algorithm

We implement a modified iterated sigma-point Kalman smoother (ISPKS), which integrates the gating mechanism and the recursive measurement update of the iterated sigma-point Kalman filter (ISPKF) in our tracking framework. ISPKF is first proposed by Sibley *et al.* who have applied it in estimating a sequence of feature measurements for a "*long range stereo*" system [106]. Unlike the standard EKF and SPKF, which uses the one-step measurement update, the ISPKF employs the *Gauss-Newton* recursion in the measurement update step that generates a maximum a posteriori (MAP) solution of the state [106, 107]. Sibley *et al.* have demonstrated that the tracking accuracy of the ISPKF is significantly higher compared to the EKF/SPKF based estimator in the "long range stereo" application. However, the measurement recursion adopted by the ISPKF is computationally intensive compared to the EKF/SPKF, which increases the computational cost from $O\left(M^3\right)$ to $O\left(JM^3\right)$ at each k, where J is the number of iterations performed.

Since the gating procedure is based on an iteration of the state estimate, we may also follow the Gauss-Newton recursive state update procedure to take advantage of this iteration. However, in practice we have found that the ISPKF provides only minor improvements in tracking performance for our application. Further work is needed to better characterize the benefits of both the ISPKF and the gating mechanism to determine whether the extra complexity is warranted.

Below we present the complete pseudo-code of the ISPKS algorithm with *gating.* The forward filter is the ISPKF, which uses the same time-update equations as the standard SPKF. However, the measurement-update step of the ISPKF consists of a state-update iteration instead of the standard one-step measurement-update followed by the SPKF. The gating mechanism as outlined in Section 4.5.4 is applied at each iteration of the measurement-update step. The iterative posterior state update Equation (4.65), which computes $\hat{\boldsymbol{x}}_k^l$, contains all the terms of the standard SPKF and an additional MAP correction term involving the Kalman gain, cross-error covariance between the state and observation, state prediction covariance and the difference between the predicted state and previous iterated state. New sigma points are extracted at each iteration from the latest estimate of the posterior state distribution with mean $\hat{\boldsymbol{x}}_k^l$ (superscript l denotes iteration) and covariance $\boldsymbol{P}_{\boldsymbol{x}_k}$, which are propagated over the observation model $\boldsymbol{h}$ to compute a new measurement prediction $\left(\hat{\boldsymbol{z}}_k^l\right)^-$. Gating is applied on the range candidates using $\left(\hat{\boldsymbol{z}}_k^l\right)^-$ and a new posterior state $\hat{\boldsymbol{x}}_k^{l+1}$ is adapted using the state update Equation (4.65). This process is repeated until convergence, which is achieved when the MSE between two consecutive posterior states $\hat{\boldsymbol{x}}_k^l$ and $\hat{\boldsymbol{x}}_k^{l+1}$ becomes less than 0.5 ft. Note, for the first time through the measurement-update loop ($l = 1$), we obtain the standard measurement update of the SPKF. After convergence of the forward ISPKF, a backward RTS smoothing pass is then followed on the forward filtering results within a L sized window to obtain the smoothed estimates.

- *Forward Filter Initialization*:

$$\hat{\boldsymbol{x}}_0 = \mathrm{E}\left[\boldsymbol{x}_0\right] \tag{4.42}$$

$$\boldsymbol{P}_{\boldsymbol{x}_0} = \mathrm{E}\left[\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)^T\right] \tag{4.43}$$

$$\hat{\boldsymbol{x}}_0^{\mathrm{a}} = \mathrm{E}\left[\boldsymbol{x}_0^{\mathrm{a}}\right] \tag{4.44}$$

$$= \left[\begin{array}{ccc} \hat{\boldsymbol{x}}_0^T & \hat{\boldsymbol{v}}_0^T & \hat{\boldsymbol{n}}_0^T \end{array}\right]^T \tag{4.45}$$

$$\boldsymbol{P}_{\boldsymbol{x}_0}^{\mathrm{a}} = \left[\left(\boldsymbol{x}_0^{\mathrm{a}} - \hat{\boldsymbol{x}}_0^{\mathrm{a}}\right)\left(\boldsymbol{x}_0^{\mathrm{a}} - \hat{\boldsymbol{x}}_0^{\mathrm{a}}\right)^T\right] \tag{4.46}$$

$$= \left[\begin{array}{ccc} \boldsymbol{P}_{\boldsymbol{x}_0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{R}_0 \end{array}\right] \tag{4.47}$$

- *While* $j \leq N$

  1. *Forward filter Recursions*:

     - For $k = j - L, j - L + 1, \ldots, j$
     - *Calculate sigma points*:

$$\boldsymbol{\chi}_k^{\mathrm{a}} = \left[\begin{array}{ccc} \hat{\boldsymbol{x}}_k^{\mathrm{a}} & \hat{\boldsymbol{x}}_k^{\mathrm{a}} + \sqrt{\left(\acute{M} + \lambda\right)\boldsymbol{P}_{\boldsymbol{x}_k}^{\mathrm{a}}} & \hat{\boldsymbol{x}}_k^{\mathrm{a}} - \sqrt{\left(\acute{M} + \lambda\right)\boldsymbol{P}_{\boldsymbol{x}_k}^{\mathrm{a}}} \end{array}\right] \tag{4.48}$$

     - *Time-update equations*:

$$\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} = f\left(\boldsymbol{\chi}_{i,k}^{\boldsymbol{x}}, \boldsymbol{\chi}_{i,k}^{\boldsymbol{v}}\right) \quad i = 0, 1, \ldots, 2\acute{M} \tag{4.49}$$

$$\hat{\boldsymbol{x}}_{k+1}^{-} = \sum_{i=0}^{2\acute{M}} w_i^{(m)} \boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} \tag{4.50}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}}^{-} = \sum_{i=0}^{2\acute{M}} \sum_{j=0}^{2\acute{M}} w_{ij}^{(c)} \left(\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-}\right)\left(\boldsymbol{\chi}_{j,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-}\right)^T \tag{4.51}$$

– *Weighted Statistical Linearization of f(.)*:

$$\boldsymbol{P}_{\boldsymbol{x}_k\boldsymbol{x}_{k+1}^-} = \sum_{i=0}^{2\acute{M}}\sum_{j=0}^{2\acute{M}} w_{ij}^c \left(\boldsymbol{\chi}_{j,k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_k\right)\left(\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} - \hat{\boldsymbol{x}}_{k+1}^{-1}\right)^T \tag{4.52}$$

$$\boldsymbol{A}_{f,k} = \boldsymbol{P}_{\boldsymbol{x}_k\boldsymbol{x}_{k+1}^-}^T \boldsymbol{P}_{\boldsymbol{x}_k}^{-1} \tag{4.53}$$

$$\boldsymbol{b}_{f,k} = \hat{\boldsymbol{x}}_{k+1}^- - \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k \tag{4.54}$$

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} = \boldsymbol{P}_{\boldsymbol{x}_{k+1}}^- - \boldsymbol{A}_{f,k}\boldsymbol{P}_{\boldsymbol{x}_k}\boldsymbol{A}_{f,k}^T \tag{4.55}$$

– *Measurement-update equations*:

$$\boldsymbol{\gamma}_{i,k+1|k} = \boldsymbol{h}\left(\boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}}, \boldsymbol{\chi}_{i,k}^{\boldsymbol{n}}\right) \quad i = 0, 1, \ldots, 2\acute{M} \tag{4.56}$$

$$\hat{\boldsymbol{z}}_{k+1}^- = \sum_{i=0}^{2\acute{M}} w_i^{(m)}\boldsymbol{\gamma}_{i,k+1|k} \tag{4.57}$$

– *Measurement-update recursions*:

The Gauss-Newton recursion followed in the measurement update step is shown below:

∗ *Initialization*:

$$\boldsymbol{\gamma}_{i,k+1}^0 = \boldsymbol{\gamma}_{i,k+1|k} \tag{4.58}$$

$$\left(\hat{\boldsymbol{z}}_{k+1}^0\right)^- = \hat{\boldsymbol{z}}_{k+1}^- \tag{4.59}$$

$$\left(\boldsymbol{\chi}_{i,k+1}^0\right)^{\boldsymbol{x}} = \boldsymbol{\chi}_{i,k+1|k}^{\boldsymbol{x}} \tag{4.60}$$

$$\hat{\boldsymbol{x}}_{k+1}^0 = \hat{\boldsymbol{x}}_{k+1}^- \tag{4.61}$$

∗ *For* $l = 1, 2, \ldots L_{\mathrm{C}}$

· Apply *gating* to determine $\boldsymbol{z}_{k+1}^l$.

· *State update*:

$$\boldsymbol{P}^l_{\tilde{\boldsymbol{z}}_{k+1}} = \sum_{i=0}^{2\acute{M}} \sum_{j=0}^{2\acute{M}} w^{(c)}_{i,j} \left(\boldsymbol{\gamma}^{l-1}_{j,k+1} - \left(\hat{\boldsymbol{z}}^{l-1}_{k+1} \right)^- \right) \cdot \left(\boldsymbol{\gamma}^{l-1}_{i,k+1} - \left(\hat{\boldsymbol{z}}^{l-1}_{k+1} \right)^- \right)^T \quad (4.62)$$

$$\boldsymbol{P}^l_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} = \sum_{i=0}^{2\acute{M}} \sum_{j=0}^{2\acute{M}} w^{(c)}_{i,j} \left(\left(\boldsymbol{\chi}^{l-1}_{j,k+1} \right)^{\boldsymbol{x}} - \hat{\boldsymbol{x}}^{l-1}_{k+1} \right) \cdot \left(\boldsymbol{\gamma}^{l-1}_{i,k+1} - \left(\hat{\boldsymbol{z}}^{l-1}_{k+1} \right)^- \right)^T \quad (4.63)$$

$$\boldsymbol{K}^l_{k+1} = \boldsymbol{P}^l_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} \left(\boldsymbol{P}^l_{\tilde{\boldsymbol{z}}_{k+1}} \right)^{-1} \quad (4.64)$$

$$\hat{\boldsymbol{x}}^l_{k+1} = \hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{K}^l_{k+1} \left(\boldsymbol{z}^l_{k+1} - \left(\hat{\boldsymbol{z}}^{l-1}_{k+1} \right)^- \right) - \boldsymbol{K}^l_{k+1} \left(\boldsymbol{P}^l_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}} \right)^T \left(\boldsymbol{P}^-_{\boldsymbol{x}_{k+1}} \right)^{-1} \left(\hat{\boldsymbol{x}}^-_{k+1} - \hat{\boldsymbol{x}}^{l-1}_{k+1} \right) \quad (4.65)$$

· *Sigma-points extraction*:

$$\left(\hat{\boldsymbol{x}}^l_{k+1} \right)^{\mathrm{a}} = \begin{bmatrix} \left(\hat{\boldsymbol{x}}^l_{k+1} \right)^T & \boldsymbol{n}^T \end{bmatrix}^T \quad (4.66)$$

$$\left(\boldsymbol{P}^l_{\boldsymbol{x}_{k+1}} \right)^{\mathrm{a}} = \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{x}_k} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_k \end{bmatrix} \quad (4.67)$$

$$\left(\boldsymbol{\chi}^l_{k+1} \right)^{\mathrm{a}} = \begin{bmatrix} \left(\hat{\boldsymbol{x}}^l_{k+1} \right)^{\mathrm{a}} \\ \left(\hat{\boldsymbol{x}}^l_{k+1} \right)^{\mathrm{a}} + \sqrt{\left(\acute{M} + \lambda \right) \left(\boldsymbol{P}^l_{\boldsymbol{x}_{k+1}} \right)^{\mathrm{a}}} \\ \left(\hat{\boldsymbol{x}}^l_{k+1} \right)^{\mathrm{a}} - \sqrt{\left(\acute{M} + \lambda \right) \left(\boldsymbol{P}^l_{\boldsymbol{x}_{k+1}} \right)^{\mathrm{a}}} \end{bmatrix} \quad (4.68)$$

· *Observation prediction*:

$$\boldsymbol{\gamma}^l_{i,k+1} = \boldsymbol{h} \left(\left(\boldsymbol{\chi}^l_{i,k+1} \right)^{\boldsymbol{x}}, \left(\boldsymbol{\chi}^l_{i,k+1} \right)^{\boldsymbol{n}} \right) \; i = 0, \ldots, 2\acute{M} \quad (4.69)$$

$$\left(\hat{\boldsymbol{z}}^l_{k+1} \right)^- = \sum_{i=0}^{2\acute{M}} w^{(m)}_i \boldsymbol{\gamma}^l_{i,k+1} \quad (4.70)$$

∗ *End For*

– Compute the final state estimate and state estimation error covariance at time $k+1$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}^{l}_{k+1} \tag{4.71}$$

$$\boldsymbol{P}_{\boldsymbol{x}_{k+1}} = \boldsymbol{P}^{-}_{\boldsymbol{x}_{k+1}} - \boldsymbol{K}^{l}_{k+1}\boldsymbol{P}^{l}_{\tilde{\boldsymbol{z}}_{k+1}}\left(\boldsymbol{K}^{l}_{k+1}\right)^{T} \tag{4.72}$$

– *Weighted Statistical Linearization of h(.)*:

$$\boldsymbol{A}_{h,k} = \boldsymbol{P}^{T}_{\boldsymbol{x}_{k+1}\boldsymbol{z}_{k+1}}\left(\boldsymbol{P}^{-}_{\boldsymbol{x}_{k+1}}\right)^{-1} \tag{4.73}$$

$$\boldsymbol{b}_{h,k} = \hat{\boldsymbol{z}}^{-}_{k+1} - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}^{-}_{k+1} \tag{4.74}$$

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} = \boldsymbol{P}_{\tilde{\boldsymbol{z}}_{k+1}} - \boldsymbol{A}_{h,k}\boldsymbol{P}^{-}_{\boldsymbol{x}_{k+1}}\boldsymbol{A}^{T}_{h,k} \tag{4.75}$$

– *End For*

2. *Backward smoothing*:

– *For* $k = j, j-1, j-2, \ldots, j-L+1, j-L$

∗ *Error covariance smoothing*:

$$\boldsymbol{D}_{k} = \boldsymbol{P}_{\boldsymbol{x}_k}\boldsymbol{A}^{T}_{f,k}\left(\boldsymbol{P}^{-}_{\boldsymbol{x}_{k+1}}\right)^{-1} \tag{4.76}$$

$$\boldsymbol{P}^{\mathrm{S}}_{k} = \boldsymbol{P}_{\boldsymbol{x}_k} - \boldsymbol{D}_{k}\left(\boldsymbol{P}^{-}_{\boldsymbol{x}_{k+1}} - \boldsymbol{P}^{\mathrm{S}}_{k+1}\right)\boldsymbol{D}^{T}_{k} \tag{4.77}$$

∗ *State estimate smoothing*:

$$\hat{\boldsymbol{x}}^{\mathrm{S}}_{k} = \hat{\boldsymbol{x}}_{k} + \boldsymbol{D}_{k}\left(\hat{\boldsymbol{x}}^{\mathrm{S}}_{k+1} - \hat{\boldsymbol{x}}^{-}_{k+1}\right) \tag{4.78}$$

– *End For*

- Increment j by one: $j = j+1$
- *End While*
- *where*:

$$\boldsymbol{x}^{\mathrm{a}} = \begin{bmatrix} \boldsymbol{x}^{T} & \boldsymbol{v}^{T} & \boldsymbol{n}^{T} \end{bmatrix}^{T} \tag{4.79}$$

$$\boldsymbol{\chi}^{\mathrm{a}} = \begin{bmatrix} (\boldsymbol{\chi}^{\boldsymbol{x}})^{T} & (\boldsymbol{\chi}^{\boldsymbol{v}})^{T} & (\boldsymbol{\chi}^{\boldsymbol{n}})^{T} \end{bmatrix}^{T} \tag{4.80}$$

$$\boldsymbol{P}^{\mathrm{a}}_{\boldsymbol{x}_k} = \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{x}_k} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Q}_{k} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{R}_{k} \end{bmatrix} \tag{4.81}$$

- *Parameters*:

  λ is the composite scaling parameter

$$\lambda = \alpha^2 \left(\acute{M} + \kappa\right) - \acute{M}, \tag{4.82}$$

and $w_i^{(c)}$ and $w_i^{(m)}$ are the scaler sigma-point weights defined as:

$$w_0^{(c)} = \frac{\lambda}{\left(\acute{M} + \lambda\right)} + \left(1 - \alpha^2 + \beta\right) \quad , i = 0 \tag{4.83}$$

$$w_0^{(m)} = \frac{\lambda}{\left(\acute{M} + \lambda\right)} \quad , i = 0 \tag{4.84}$$

$$w_i^{(c)} = \frac{1}{2\left(\acute{M} + \lambda\right)} \quad , i = 1, 2, \ldots, 2\acute{M} \tag{4.85}$$

$$w_i^{(m)} = \frac{1}{2\left(\acute{M} + \lambda\right)} \quad , i = 1, 2, \ldots, 2\acute{M} \tag{4.86}$$

where M is the dimension of each state, $\acute{M}$ is the dimension of each augmented state, $\boldsymbol{Q}_k$ is the process-noise covariance and $\boldsymbol{R}_k$ is the observation-noise covariance. The length of the observation sequence is N and L is the lag between the current measurement and the estimated state. The final smoothed state and associated error covariance are denoted as $\hat{\boldsymbol{x}}_k^{\mathrm{S}}$ and $\boldsymbol{P}_k^{\mathrm{S}}$ respectively. The values of the SPKF parameters are shown in Table 4.1. Note that in the standard ISPKF implementation L_{C} is taken as a small fixed value. In our implementation, L_{C} is determined from the MSE criterion between $\hat{\boldsymbol{x}}_k^{l}$ and $\hat{\boldsymbol{x}}_k^{l+1}$.

Table 4.1 demonstrates the user-specified parameters needed to implement the SPKS based location tracker.

4.5.5 Experimental Results

Implementation and testing were performed at a test Lab (called Sonar-Lab) located in the "*Jack-Murdock*" building at the Oregon Health & Science University (OHSU). The test Lab is 19.2 ft by 14.8 ft cluttered with various office furniture. A number of trials

Table 4.1: Summary of user-specified parameters

Name	**Symbol**	**Value**
Time constant for AR averaging (short)	T_{sc}	1 s
Time constant for AR averaging (long)	T_{lc}	15 s
Potential field force constant	$\boldsymbol{F}_{\mathrm{cr}}$	10
Euler Integration time	δT	0.5 s
Smoothing AR coefficient	λ	0.95
Number of ultrasonic sensors	M	6
Measurement lag	L	4
Sigma-Point spread	α	0.85
Sigma-Point weighting term	β	2
Sigma-Point parameter	κ	0
Time spent for training	T_{tr}	15 min
Sampling rate of the SPKS	$\boldsymbol{f}_{\mathrm{s}}$	2 Hz
Max sample spread allowed in one cluster	σ_c^2	40
Min sample distance between two clusters	D_c	25

were conducted in the test Lab in which different subjects followed random trajectories. In order to obtain a commercial benchmark for evaluation of our approach, the *Ubisense* real-time tracking engine was turned on during these tests. Four Ubisense access points and six ultrasonic transducers were set up at various locations of the test Lab.

Ubisense is a commercial tag based positioning engine which uses the *ultra-wideband (UWB)* radio technology to detect a mobile Ubisense tag. Ubisense places multiple access points at the predefined locations of the room which transmits *UWB* signals at periodic intervals. The person carries a receiver tag which independently determines the *"angle-of-arrival (AOA)"* of the UWB signal and the *"time difference of arrival (TOA)"* between a pair of sensors in order to perform positioning. The Ubisense positioning engine can provide centimeter level tracking accuracy if the person's tag is always viewed along a direct path by at least two access points. The drawback of Ubisense is it is costly, tag-based and its calibration mechanism, which includes positioning the access points at the optimal orientations relative to the tag involves several implementation challenges.

In order to perform user localization and continuous tracking, our system was first calibrated in order to generate the observation maps from the training data. During the calibration phase for the ISPKS based tracking system, a person carrying an Ubisense tag

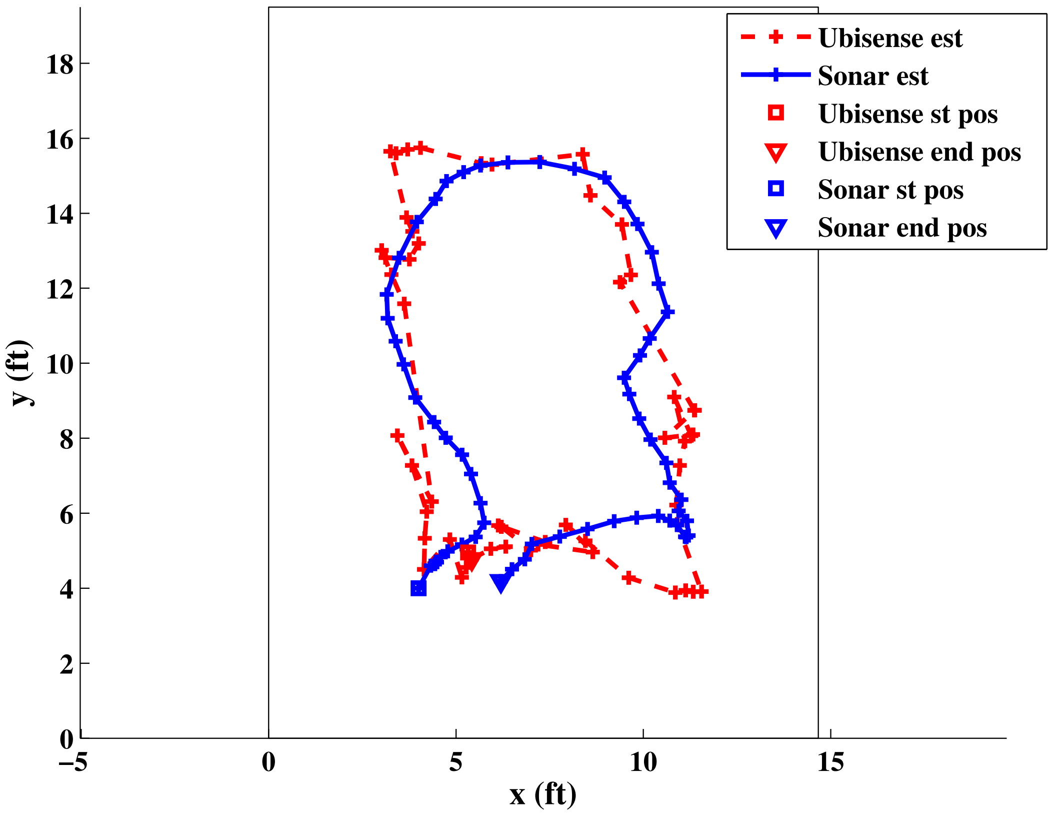


Figure 4.10: Tracking performance in Sonar-Lab (*red*: Ubisense estimates, *blue*: ISPKS estimates). The root-mean-square-error (RMSE) between the ISPKS and the Ubisense estimated positions is equal to RMSE = 1.95 ft. The above tracking result is shown for subject 1. The start and end positions for the Ubisense and sonar trajectories are marked for better visualization.

walked around the whole test location for $T_{\text{tr}} = 15\,\text{mins}$. Roughly $2-3$ measurements per second from all 6 ultrasonic sensors were recorded during training for a total of 2341 measurements. As described in Section 4.5.3, a recursive *RBF* network was used to fit a nonlinear map between the Ubisense estimated user locations and the person's ranges. Three representative range calibrated observation maps are depicted in Figure 4.8.

Our objective is to compare the tracking performance of the tag-free ISPKS estimator with the Ubisense positioning engine. In order to demonstrate that the accuracy of the ISPKS tracker is consistent over multiple subjects, we performed three trials of moving test in which three different subjects walked at a normal speed in the Sonar-Lab. Note

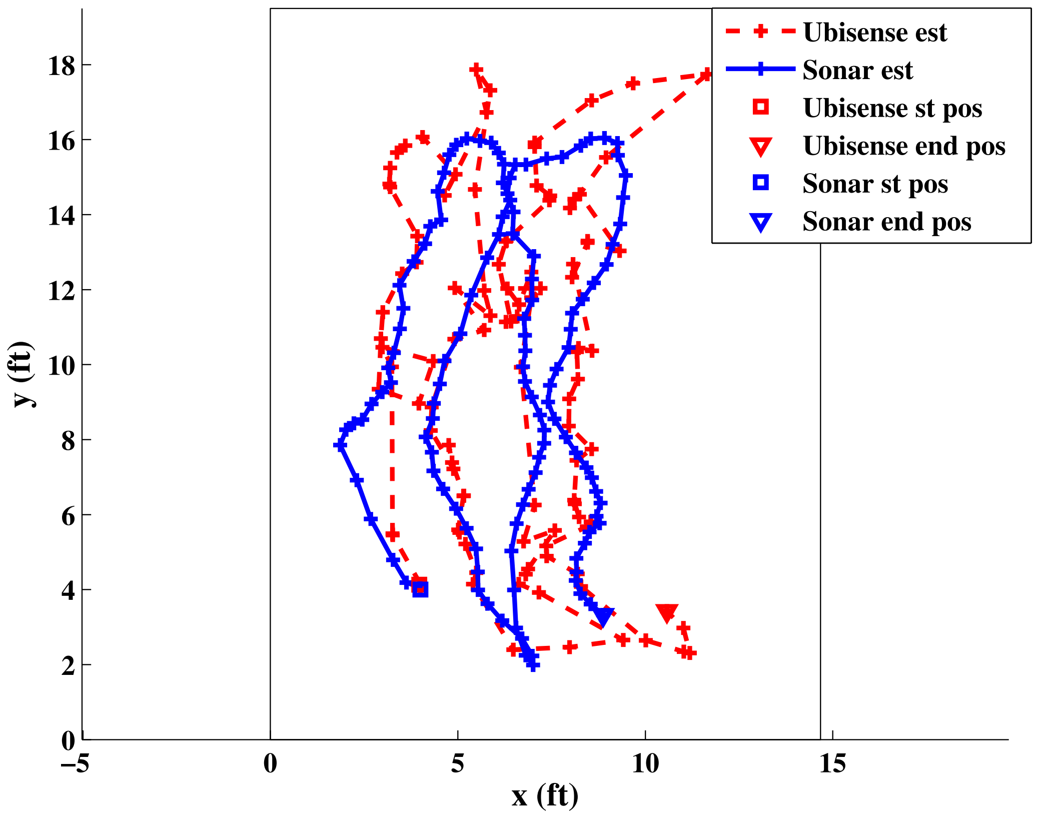


Figure 4.11: Tracking performance in Sonar-Lab (*red*: Ubisense estimates, *blue*: ISPKS estimates). The RMSE between the ISPKS and the Ubisense estimated positions is equal to RMSE $= 2.52\,\text{ft}$. The above tracking result is shown for subject 2. The start and end positions for the Ubisense and sonar trajectories are marked for better visualization.

that the subjects followed different trajectories while walking. Subject 1 took 30 s to complete the path and 78 range observations were recorded during that time period. Subject 2 completed another path in 60 s and recorded 157 range observations. Subject 3 also walked for 60 s and recorded 164 measurements. Although the sampling rate of the incoming ultrasonic sensor data was found to vary between $333 - 500\,\text{ms}$, the ISPKS estimator was operated at a fixed rate of 2 Hz during tracking. In the tracking examples, the lag between the current measurement and smoothed state was set to $L = 4$, which corresponds to a a time delay of $L\delta T = 2\,\text{s}$, but provides for smoother trajectory estimates.

Figure 4.10-4.12 compares the position estimates obtained from the Ubisense engine and the ISPKS tracker. From the experimental results, it is clearly evident that the

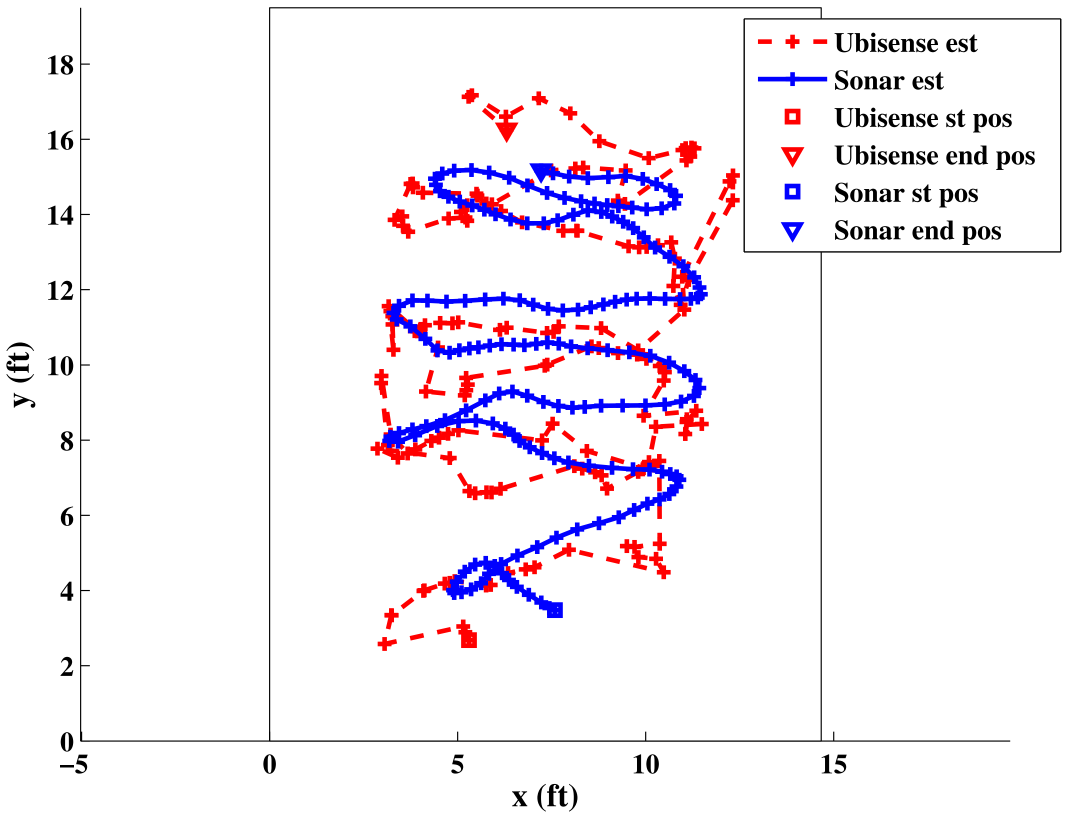
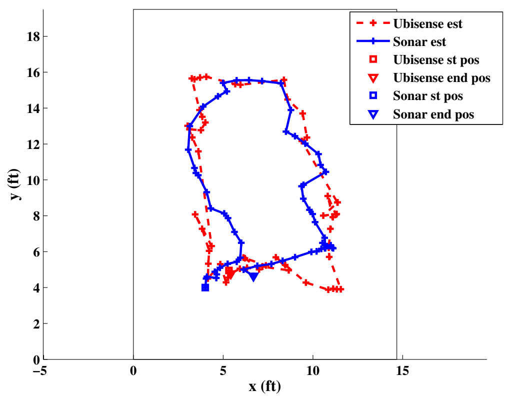
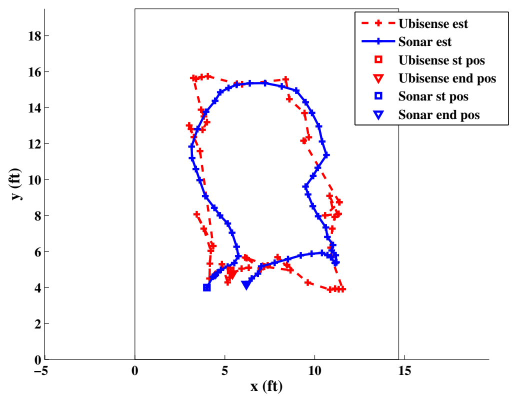


Figure 4.12: Tracking performance in Sonar-Lab (*red*: Ubisense estimates, *blue*: ISPKS estimates). The RMSE between the ISPKS and the Ubisense estimated positions is equal to RMSE = 2.84 ft. The above tracking result is shown for subject 3. The start and end positions for the Ubisense and sonar trajectories are marked for better visualization.

position estimates computed by the ISPKS is comparable with the Ubisense positioning engine. Notice in Figure 4.10-4.12 how Ubisense sometimes generates large *jumps* in its estimates. This occurs due to the failure of the Ubisense tag to always maintaining a direct line of sight with at least two of the access points. In contrast, the ISPKS tracker consistently provides smooth estimates. In addition, the estimation accuracy of the proposed ISPKS is also shown to be consistent over multiple subjects and different paths. Only the performance of the ISPKS tracker using the CT model as system dynamics is demonstrated in the figures. The CT model is chosen because it is more realistic and it takes into account the spatial coupling across 2D coordinates. Figure 4.13(a)-4.13(b) compares the performance of the CT based ISPKS with that based on a random

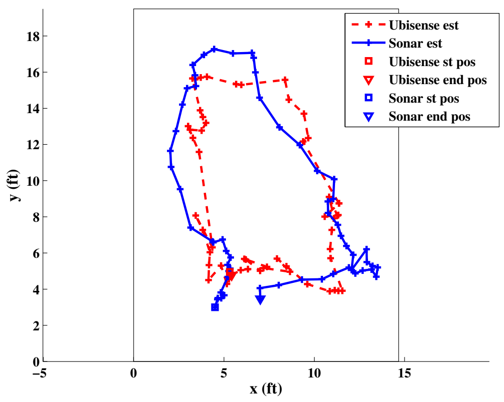


(a) RMSE=2.12 ft

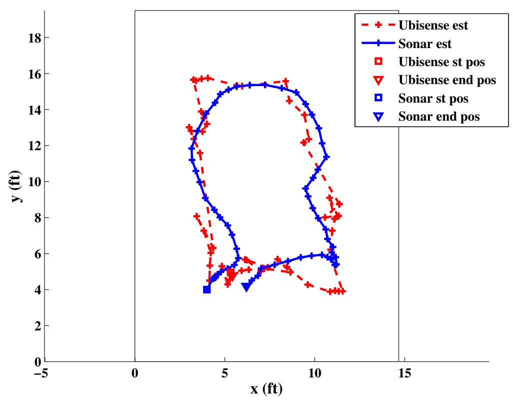


(b) RMSE=1.95 ft

Figure 4.13: Performance of the ISPKS based trackers using two different dynamic models (*red*: Ubisense estimates, *blue*: ISPKS estimates). (a) ISPKS estimates using random acceleration based model, (b) ISPKS estimates using CT model with unknown turn rate.



(a) RMSE=3.94 ft

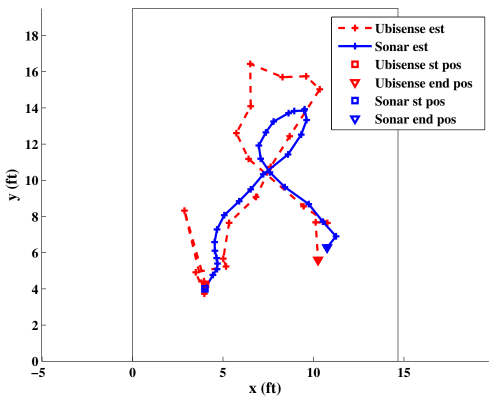


(b) RMSE=1.95 ft

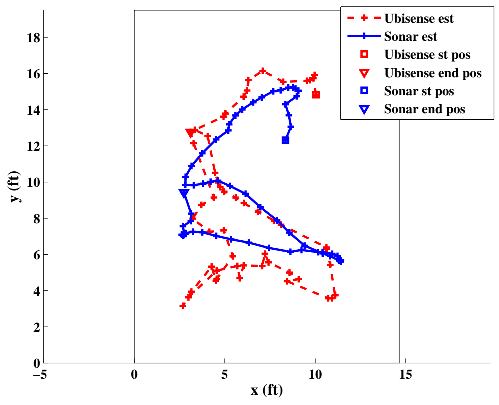
Figure 4.14: Performance of the ISPKF and ISPKS based trackers in estimating user positions(*red*: Ubisense estimates, *blue*: ISPKF/ISPKS estimates). (a) ISPKF estimates, (b) ISPKS estimates.

acceleration model. As shown, the estimates obtained from the CT based ISPKS are slightly more accurate compared to that of the random acceleration model.

The tracking performance of the ISPKF and ISPKS are compared in Figure 4.14(a)-4.14(b). Note that the ISPKF also applies the recursive *gating* technique as the ISPKS. As shown, the ISPKS estimates are superior to the ISPKF. Like Ubisense, the ISPKF also demonstrates "jumps" at some parts of the estimation trajectory. The ISPKS is free



(a) RMSE=3.32 ft



(b) RMSE=3.87 ft

Figure 4.15: Tracking Performance of the ISPKS in estimating user positions(*red*: Ubisense estimates, *blue*: ISPKS estimates). The above plots are for two trials performed by different subjects.

Table 4.2: Tracking performance of the ISPKS and ISPKF. The mean and standard deviation of RMSE is calculated by averaging over 50 different trials performed at the Sonar-Lab.

Estimator	E(RMSE) (ft)	std(RMSE) (ft)
ISPKF	4.78	2.2
ISPKS	3.08	1.75

of such problem as the smoother provides superior interpolation of data using both past and future observations. Table 4.2 tabulates the RMSE between the Ubisense and ISPKS estimates computed using 50 different trials performed at the Sonar-Lab. Note that the trials were performed by multiple subjects and each subject followed a separate random trajectory while walking.

There exists a few trials where we have found that the ISPKS tracker is not very accurate in position estimation. Figure 4.15(a) and 4.15(b) demonstrate two such cases. One likely cause of lower accuracy may be related to the inaccurate modeling of observation maps. Lack of sufficient training data, failure to identify the correct range from the range candidates are might be some of the reasons which prevent from generating accurate observation maps. Learning observation maps also involve a calibration phase, which is a time consuming and tedious process. In the following we demonstrate our recent effort in investigating the "*simultaneous localization and mapping (SLAM)*" whereby the parameters of the observation models are continuously updated in the tracking phase to account for the lack of observation maps.

4.6 Simultaneous Localization and Mapping (SLAM)

4.6.1 Introduction

SLAM defines the task of simultaneously estimating the state of a moving agent (for e.g. position, velocity and attitude) and a map of the surrounding environment given limited sensing capabilities. The SLAM technique is essential in order to safely navigate a vehicle in an environment that is constantly changing or when no a priori knowledge is available about the surroundings. The state estimation and map generation processes are interdependent, i.e. based on the map of the environment, the SLAM algorithm estimates the state of the agent while the state estimates are then further used to re-tune the maps. The simultaneous map-building feature of the SLAM algorithm has proved highly beneficial in locating an unmanned vehicle in an unsurveyed environment without any direct positioning sensor. As no a priori knowledge is needed for the environment and the initial state of the vehicle, the prospect of SLAM algorithm has attracted a great deal of attention in the mobile robotics and autonomous vehicle navigation communities [3, 4, 108–113].

There are two predominant approaches to the SLAM problem as seen in the literature. One approach, called a landmark aided navigation system (LANS) models the environment by estimating the location of the landmarks scattered throughout and based on this model estimates the agent's state [4, 112]. The noisy range from each landmark is generally observed using the radar/sonar sensors, which is then used to estimate both the state of the agent and position of each landmark. The other approach, called a terrain aided navigation system (TANS), builds an environmental map from the concurrent terrain information provided by its sensors and then generates state estimates based on it [2, 114]. Key aspects of this approach include the choice of representation for the map and the algorithms to perform the estimation or map building. For both of these approaches, the prevailing method is to use the EKF to fuse the measurements from various sensors in order to estimate the state and the map [2, 4]. Research has also been performed on comparing alternative stochastic estimation techniques. For example, a constrained hidden Markov model (HMM) was applied for the TANS approach [114]. A few research teams also tackled the SLAM problem using batch estimation techniques [115]. However, the major disadvantage in these techniques is that they cannot be operated in an online mode. The use of the UKF [22] to replace the EKF has also been proposed for unmanned aerial vehicle (UAV) navigation. In [5], an UKF was implemented to navigate a small UAV through an unsurveyed environment.

This work applies both the *TANS* and *LANS* based SLAM approaches in our ultrasonic range based tracking framework to simultaneously estimate the observation model parameters with the person's state. Although the use of SLAM is prevalent in robot and UAV navigation systems for modeling an unknown environment, we are not aware of any literature that applied the SLAM approaches in the tag-free indoor tracking case. In the TANS based approach, the RBF observation maps characterize the environment. Like the map-based tracking approach described in the previous section, we still use a separate calibration phase to learn the parameters of each observation map. However, instead of keeping the parameters fixed during the tracking phase, we constantly update the map parameters at each time k using the currently estimated position. The newly generated observation maps at time k are then used to estimate the user's state $\hat{\boldsymbol{x}}_{k+1}$ at time $k+1$. The re-estimation of the observation maps not only accounts for changes in environment but also helps to alleviate any inaccuracies involved in map-learning during the calibration phase. In the LANS based tracking method, the location of each sonar unit is simultaneously estimated with the state vector using the sonar range observations. A dual Kalman framework is introduced to tackle this problem that works by alternating between using one SPKS to localize the user given the current estimated locations of the ultrasonic transducers, and a second SPKS to update the 2D sonar module locations given the current position of the user. The dual framework is first proposed by Wan *et al.* [116]. In addition to the 2D locations of the transducers, the observation model also contains a "correction factor" to take into account various noise and modeling errors. As both the state of the person and observation model parameters are estimated online during the tracking phase, the offline calibration phase is no longer needed and hence completely eliminated from

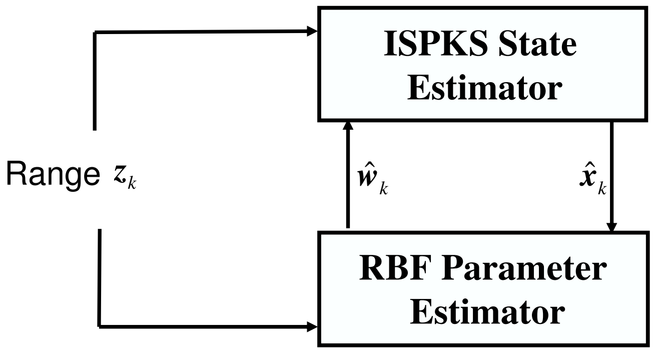


Figure 4.16: Sequential state and map estimation procedure in TANS-SLAM. An ISPKS is used for state estimation and the RBF learning algorithm is applied to estimate the observation map parameters.

the LANS framework. The "*self-calibration*" procedure saves a significant amount of time and manual effort, which could be very useful in some applications (e.g., installing the ultrasonic transducers into hundreds of homes to monitor daily living of seniors).

4.6.2 TANS Based SLAM Algorithm For Indoor Tracking

In the TANS based approach, the objective is to simultaneously estimate the parameters of each observation map with the user state using the range information from the sonar units. As the sensor observation maps represent the tracking environment, the idea here is to constantly adapt them using the state estimates in order to reflect environmental changes. A calibration phase still needs to be performed, where we learn the initial observation maps corresponding between known user positions and observed ranges. User tracking involves two estimation procedures to be performed concurrently:

1. An ISPKS tracker takes into account the previously generated observation maps $\hat{\boldsymbol{h}}_k$ and the new range measurements $\boldsymbol{z}_{k+1}$ to compute the new state estimate $\hat{\boldsymbol{x}}_{k+1}$.

2. The estimated user position in $\hat{\boldsymbol{x}}_{k+1}$ and the observed range $\boldsymbol{z}_{k+1}$ are augmented with their previous values obtained during the tracking and calibration phase. The RBF learning process is adopted that fits a nonlinear map for each sonar transducer corresponding between the augmented user position and range measurements to compute new observation maps $\hat{\boldsymbol{h}}_{k+1}$.

Note that in the above SLAM procedure, the state estimation occurs using the ISPKS, but the observation maps are learned using the two phase RBF learning algorithm described in Section 3.3.2. In this case, the ISPKS and RBF learning algorithm used only the range measurements recorded by the active ultrasonic units. The TANS-SLAM method is shown schematically in Figure 4.16. We can also use a dual framework in this case, where two ISPKS filters may be applied to estimate the state and observation maps. However, due to the large dimension of the observation map parameters, we avoid using the ISPKS for the observation maps as it can significantly increase the computational complexity of the estimation process.

In this context, we want to refer to our previous work where we have estimated the state of a vehicle (position and velocity) in an unknown environment using a low cost inertial measurement unit (IMU) and three simple terrain sensors. Each terrain sensor is simulated to provide noisy measurements of some "characteristic" (for e.g. altitude, temperature, or vision based features) of the environment at the current vehicle location. A dual Kalman framework, where two SPKF filters run concurrently to estimate the vehicle state and environment map parameters is adopted in our simulation. The details of our work with experimental results are shown at the end of this chapter in Appendix 4.9.

SLAM Framework

The state space model for dealing with the SLAM problem in the TANS based system is defined as,

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) \tag{4.87}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{w}_k, \boldsymbol{n}_k\right). \tag{4.88}$$

Note that $\boldsymbol{w}_k$ corresponds to the parameters for the observation map h_k at time k. The parameters $\boldsymbol{w}_k$ are adapted at each time k using the current estimated state and noisy sensor measurements. In the following, we will define the different components of the above equations in the context of ultrasonic range based tracking application.

Dynamic Model

The state vector is the same as shown in Section 4.5.2, which includes the person's 2D position, velocity and unknown turn rate

$$\boldsymbol{x}_k = \left[\begin{array}{ccccc} x_k & y_k & v_{x_k} & v_{y_k} & \omega_k \end{array} \right]. \tag{4.89}$$

We follow the same potential field embedded "*CT*" model as described in Section 4.5.2 for the dynamic model f_k.

Observation Model

We have used the same RBF observation model, which is covered in Section 4.5.3. Here we will provide a brief summary in order to identify the observation model parameters that we are going to learn. The observation model h_m for the m-th sensor, which maps the known 2D positions into the 1D ranges, can be defined as follows

$$z_{m,k} = h_m\left(x_k, y_k\right) + n_m, \tag{4.90}$$

where $z_{m,k}$ is the observed range from sensor m, $1 \leq m \leq M$, which is chosen from several range candidates by applying a *gating* mechanism as shown in Section 4.5.3. The 2D user position is denoted as $\left[\begin{array}{cc} x_k & y_k \end{array} \right]$. The RBF observation map h_m is comprised of several parameters

$$h_m\left(x_k, y_k\right) = \boldsymbol{W}_m^T \boldsymbol{K}_{m,\mathrm{G}}\left(\left[\begin{array}{cc} x_k & y_k \end{array} \right]; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\right), \tag{4.91}$$

where $\boldsymbol{K}_{m,\mathrm{G}}$ is denoted as the Gaussian kernel function [88] with mean vector $\boldsymbol{\mu}_m$ and covariance matrix $\boldsymbol{\Sigma}_m$ and $\boldsymbol{W}_m$ is the weights for the output layer. The parameters for the m-th sensor can be denoted as a single vector $\boldsymbol{w}^m$

$$\boldsymbol{w}^m = \left[\begin{array}{ccc} \boldsymbol{W}_m & \boldsymbol{\mu}_m & \boldsymbol{\Sigma}_m \end{array} \right]. \tag{4.92}$$

The parameter vector $\boldsymbol{w}^m$ is learned by fitting a nonlinear map between the augmented measurement $\boldsymbol{z}_m$ and the user position $\boldsymbol{X}$ collected during the calibration phase, where

$$\boldsymbol{z}_m = \begin{bmatrix} z_{m,1} & z_{m,2} & \cdots & z_{m,k} & \cdots & z_{m,N} \end{bmatrix} \tag{4.93}$$

$$\boldsymbol{X} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_k & y_k \\ \vdots & \vdots \\ x_N & y_N \end{bmatrix}. \tag{4.94}$$

The measurement vector $\boldsymbol{z}_m$, weight parameter $\boldsymbol{w}^m$, and observation noise n_m from each ultrasonic transducer are combined to form a multi-dimensional observation model,

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{z}_1 & \boldsymbol{z}_2 & \ldots & \boldsymbol{z}_m & \ldots & \boldsymbol{z}_{\mathrm{M}} \end{bmatrix} \tag{4.95}$$

$$\boldsymbol{w} = \begin{bmatrix} \boldsymbol{w}^1 & \boldsymbol{w}^2 & \ldots & \boldsymbol{w}^m & \ldots & \boldsymbol{w}^{\mathrm{M}} \end{bmatrix} \tag{4.96}$$

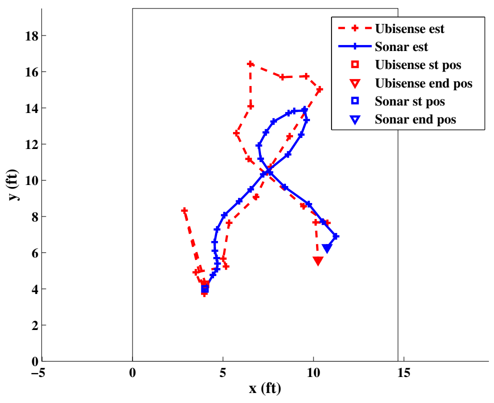
$$\boldsymbol{n} = \begin{bmatrix} \boldsymbol{n}_1 & \boldsymbol{n}_2 & \ldots & \boldsymbol{n}_m & \ldots & \boldsymbol{n}_{\mathrm{M}} \end{bmatrix}, \tag{4.97}$$

where $\boldsymbol{z}$, $\boldsymbol{w}^m$ and $\boldsymbol{n}$ are the augmented measurement vector, weight parameter vector and measurement noise for M number of ultrasonic transducers.

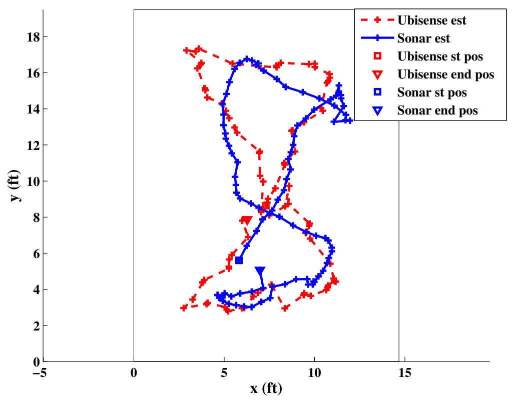
SLAM Algorithm

Our objective is to develop an estimation framework that works by alternating between learning the map parameters given the current estimated position of the user, and using the newly obtained maps to localize the user. Note, the estimator is operated at a fixed rate of $f_s = 2\,\mathrm{Hz}$. At $k = 0$, before the first measurement arrives, the map parameters are initialized as

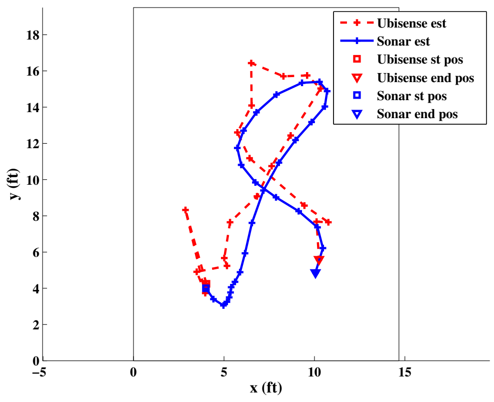
$$\hat{\boldsymbol{w}}_0 = \boldsymbol{w}, \tag{4.98}$$



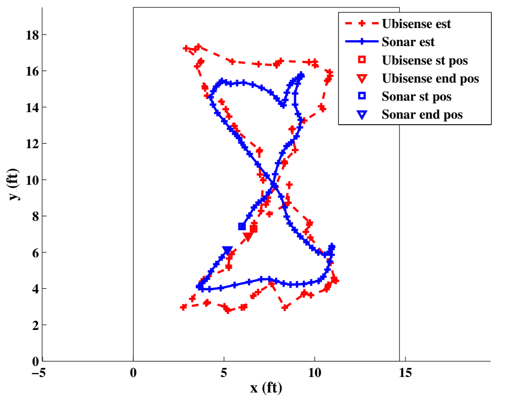
(a) Trial 1, RMSE=3.32 ft



(b) Trial 2, RMSE=3.44 ft



(c) Trial 1, RMSE=1.86 ft



(d) Trial 2, RMSE=2.82 ft

Figure 4.17: Comparison of the tracking performance between the TANS-SLAM and range-map estimation using two different trials, (a) to (b): range-map estimation results (*blue*): user positions are estimated using fixed observation maps learned during the calibration phase. (c) to (d): TANS-SLAM results (*blue*): user positions are estimated using simultaneous learning of observation map parameters. The Ubisense results *red* provide the ground truth for comparison.

where $\boldsymbol{w}$ is the augmented parameter vector that is learned during the calibration. Furthermore, the augmented measurement and user position at time $k = 0$ are denoted as

$$\boldsymbol{z}_0 = \boldsymbol{z} \tag{4.99}$$

$$\boldsymbol{X}_0 = \boldsymbol{X}, \tag{4.100}$$

where $\boldsymbol{z}$ and $\boldsymbol{X}$ are obtained in the calibration phase.

After the arrival of a new user range $z_{m,k}$ $\forall m, 1 \leq m \leq \mathrm{M}$, the ISPKS equations with gating, as shown in Section 4.5.4, are applied to estimate the user state at time k, $\hat{\boldsymbol{x}}_k$. Note that the estimation of $\hat{\boldsymbol{x}}_k$ takes into account the observation maps with parameters $\hat{\boldsymbol{w}}_{k-1}$ learned at the previous time step. The position estimate, $\hat{\boldsymbol{x}}_k$ and the range measurement $z_{m,k}$ are then augmented with $\boldsymbol{X}_{k-1}$ and $\boldsymbol{z}_{m,k-1}$ to form a new measurement and position vector $\boldsymbol{z}_{m,k}$ and $\boldsymbol{X}_k$ respectively

$$\boldsymbol{z}_{m,k} = \left[\begin{array}{cc} \boldsymbol{z}_{m,k-1} & z_{m,k} \end{array} \right] \tag{4.101}$$

$$\boldsymbol{X}_k = \left[\begin{array}{cc} \boldsymbol{X}_{k-1} & \hat{\boldsymbol{x}}_k \end{array} \right]. \tag{4.102}$$

The parameters $\hat{\boldsymbol{w}}_k^m$ for the m-th observation map are then computed by fitting a RBF map between the newly generated $\boldsymbol{z}_{m,k}$ and $\boldsymbol{X}_k$. Note that the two phase RBF learning method described in Section 3.3.2 is applied to estimate the parameter vector $\hat{\boldsymbol{w}}_k^m$. The simultaneous estimation of the user position and map parameters continues at each time step k until the arrival of the last measurement. Some points to be noted in the above algorithm:

- The map parameters $\hat{\boldsymbol{w}}_k$ are learned only when a new range measurement is incorporated in the estimation process. Hence during the ISPKS backward operation, which operates on the same set of measurements as the forward filter, the parameter vector does not alter its value.

- As demonstrated, the dimension of Equations (4.101) and (4.102) grows without bounds and hence the computational complexity of the estimation process exponentially increases with k. To combat the higher computational growth and keep the dimension of $\boldsymbol{X}_k$ and $\boldsymbol{z}_{m,k}$ fixed, we eliminate the oldest state estimate and range measurement obtained during tracking at every 10 s. However, none of the calibration data is removed as it contains more accurate user positions.

- Instead of applying the RBF learning procedure at each k, which is computationally intensive, we estimate the observation map parameters at every 3 s. Note, the Equations (4.101) and (4.102) are formed at each k but we allow a time-interval of 3 s between two consecutive RBF parameter estimation.

Table 4.3: Tracking accuracy of the TANS-SLAM method in terms of RMSE. The mean and standard deviation of RMSE is calculated by averaging over 50 different trials performed by multiple subjects in the test Lab.

Estimator	E(RMSE) (ft)	std(RMSE) (ft)
TANS-SLAM	2.64	1.12
Range-map	3.08	1.75

Experimental Results

In this section, we demonstrate two trials in which two different subjects walked at a normal speed in the Sonar-Lab. Our objective is to compare the estimation performance of the proposed TANS-SLAM approach against the range-map case described in Section 4.5. Note that the same calibration data and range observations are used for both the cases. Here we have considered only active sonar ranges to generate the results. The Ubisense tracking engine, which provides reliable ground truth was turned on for performance comparison with our tracker.

Figure 4.17(a)-4.17(d) illustrate the superiority of the TANS-SLAM approach in indoor tracking. The results of the range-map examples are presented in Figure 4.17(a)-4.17(b). In this case, the observation maps are learned during the calibration phase and are kept fixed during tracking an user. As can be seen, the estimated trajectories fail to follow the Ubisense estimates closely, which result significantly higher RMSE between the Ubisense and ISPKS estimates. On the other hand, Figure 4.17(c)-4.17(d) demonstrate the TANS-SLAM examples. As is clearly evident from the above figures, the SLAM approach improves the quality of the ISPKS outputs and henceforth reduces the RMSE between the Ubisense and ISPKS estimates. Table 4.3 demonstrates the RMSE of position estimates over 50 different trials performed in the test Lab. As shown, the SLAM approach generates slightly superior mean and standard deviation of RMSE over the range-map estimator. The figures and table verify that constantly updating the observation map parameters in conjunction with the state estimates can indeed generate superior tracking results.

Despite the promising nature of the TANS-SLAM method in indoor tracking case, there are some deficiencies to be noted:

- Although the TANS technique simultaneously estimates the map parameters with

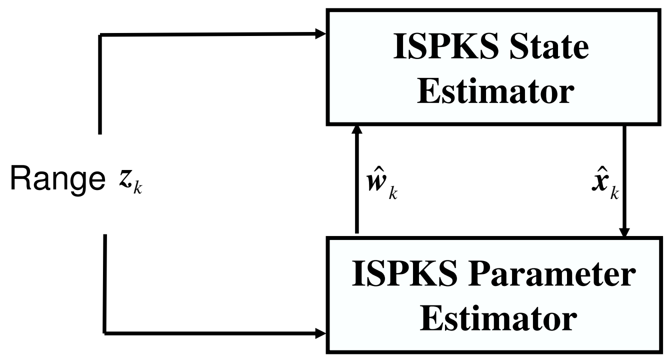


Figure 4.18: Sequential Dual Estimation Framework in LANS-SLAM. Two ISPKS operates alternately to estimate the state and observation model parameters.

the user state, it still involves an offline calibration phase. Unfortunately performing calibration is time-consuming, tedious and also requires manual effort.

- The observation maps consist of several parameters, computation of which demands higher computational load and slows down both the state and map estimation process.
- The estimation accuracy of the TANS method is not consistent over multiple trials and subjects.

Below we investigate another SLAM technique for tag-free indoor tracking using the LANS approach, which takes care of the above disadvantages.

4.6.3 LANS Based SLAM Algorithm For Indoor Tracking

The LANS-SLAM approach corresponds to simultaneously estimating the state of the person and the parameters of the observation model using range observations from the ultrasonic transducers. The parameters consist of the 2D sonar module locations along with a parameter c_k which is set to the speed of sound. However, this term also adapts as a "correction factor" to account for the effect of multipath, reflection/refraction and various noise and modeling errors. In this section, we have only considered range measurements recorded by active ultrasonic modules to track a person. The major advantage of the LANS technique lies in its self-calibrating nature, which means that the calibration phase is no longer needed. The number of observation parameters to be estimated is significantly less than that of the TANS-SLAM and thus reduces the computational complexity of the estimation process. A dual Kalman framework is proposed, which allows two ISPKS filters to run concurrently:

- One ISPKS to track the person at time k given the estimated parameters at the previous time step $k-1$.
- A second ISPKS to estimate the parameters at k given the current estimated location of the person.

A schematic diagram of the LANS-SLAM method is shown in Figure 4.18. Note that for parameter estimation, it is sufficient to run a sigma-point Kalman filter (SPKF). However, for coding purposes and duality, the smoother may be used with only minor differences in performance.

SLAM Framework

We first define a discrete time nonlinear dynamical system for estimating user's state

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_{x_k}\right) \tag{4.103}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{w}_k, \boldsymbol{n}_k\right), \tag{4.104}$$

where $\boldsymbol{w}_k$ is the parameter vector which is estimated by forming a second *dual* state-space,

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \boldsymbol{v}_{w_k} \tag{4.105}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{w}_k, \boldsymbol{n}_k\right), \tag{4.106}$$

where the parameters $\boldsymbol{w}_k$ are now considered the state, with a random walk model. The user's state $\boldsymbol{x}_k$ are now considered "parameters" in the same observation equation. The SPKS (or SPKF), may again be used to estimate the state (i.e. parameters) in an efficient manner, and can be shown corresponds to a modified-Newton optimization method [17]. In the following, we will define the different components of the above equations in the context of ultrasonic range based tracking application.

Dynamic Model

The state dynamic model, which is a CT model with wall potential field is exactly the same as followed in the TANS model.

Observation Model

The *time-of-flight* observation model for the m-th active ultrasonic unit can be defined as follows

$$z_{m,k} = h_{m,k}\left(\left[\begin{array}{cc} x_k & y_k \end{array}\right]; \left[\begin{array}{cc} \boldsymbol{x}^{\mathrm{S}}_{m,k} & c_k \end{array}\right]\right) + n_{m,k}, \tag{4.107}$$

where the observation model $h_{m,k}$ is proportional to the distance between the m-th sonar unit and the person,

$$h_{m,k} = \frac{2N_{\mathrm{S}}}{c_k}\sqrt{\left(x_k - x^{\mathrm{S}}_{m,k}\right)^2 + \left(y_k - y^{\mathrm{S}}_{m,k}\right)^2}, \tag{4.108}$$

where $z_{m,k}$ is the observed range of the user from the active sonar module m, $1 \leq m \leq M$ and $\left[\begin{array}{cc} x_k & y_k \end{array}\right]$ is the user position at time k. The 2D coordinates of the m-th ultrasonic sensor is denoted as

$$\boldsymbol{x}^{\mathrm{S}}_{m,k} = \left[\begin{array}{cc} x^{\mathrm{S}}_{m,k} & y^{\mathrm{S}}_{m,k} \end{array}\right]. \tag{4.109}$$

The speed of sound at the specific indoor location is denoted as c_k. N_{S} is a conversion factor which is used to convert the time-of-flight equations into samples. In practice, $z_{m,k}$ corresponds to a sample count, where $N_{\mathrm{S}} = 20$ is the conversion factor for the 20 kHz sample rate. The constant term 2 appeared at the right hand side of Equation (4.108) indicates that the range information extracted from each ultrasonic signal actually contains twice the distance between the user and each ultrasonic transducer. Figure 4.2 demonstrates that the range observation r^{a}_m recorded by the m-th active unit corresponds to twice the distance, d^{a}_m, between the active unit and the person (superscript "a" denotes active unit). The parameter vector $\boldsymbol{w}_k$ to be estimated contains the 2D coordinates of each ultrasonic transducer and the speed of sound

$$\boldsymbol{w}_k = \left[\begin{array}{ccccccc} c_k & \boldsymbol{x}^{\mathrm{S}}_{1,k} & \boldsymbol{x}^{\mathrm{S}}_{2,k} & \ldots & \boldsymbol{x}^{\mathrm{S}}_{m,k} & \ldots & \boldsymbol{x}^{\mathrm{S}}_{\mathrm{M},k} \end{array}\right]. \tag{4.110}$$

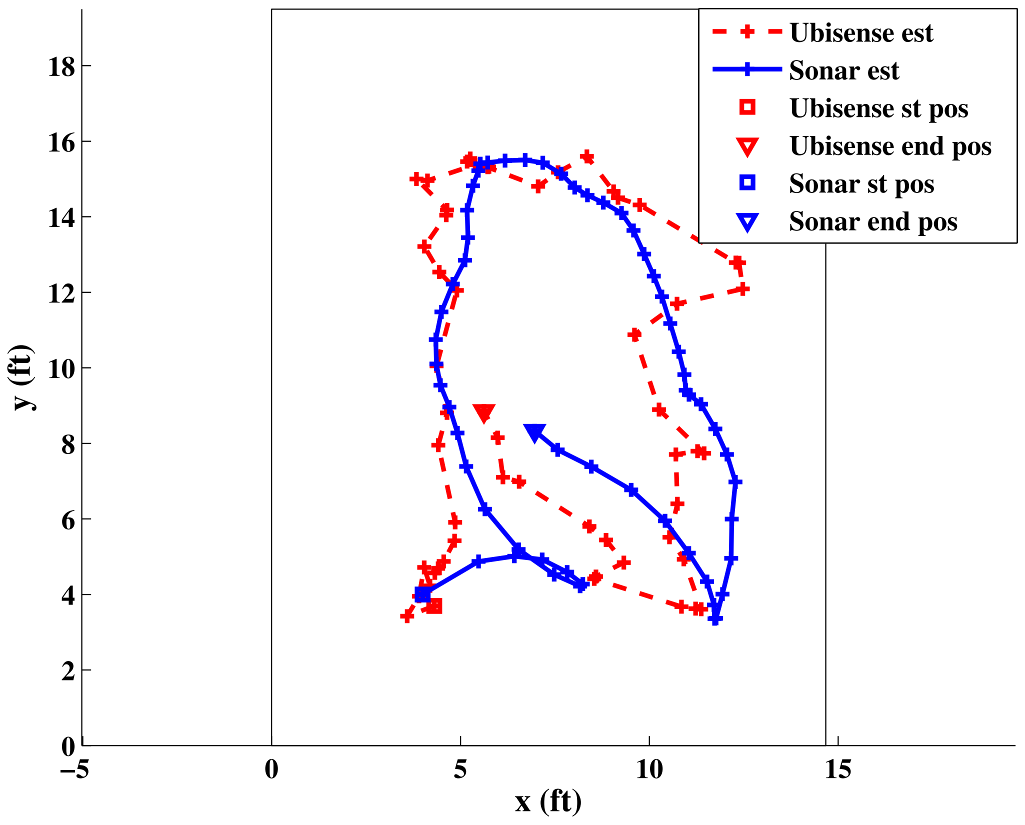


Figure 4.19: Tracking performance in Sonar-Lab using the LANS-SLAM method (*red*: Ubisense estimates, *blue*: ISPKS estimates). The RMSE between the ISPKS and the Ubisense estimated positions is equal to RMSE = 2.12 ft. The above tracking result is shown for subject 1.

The measurement $z_{m,k}$, observation function $h_{m,k}$, and observation noise $n_{m,k}$ from each ultrasonic sensor are combined to form a multi-dimensional observation model,

$$\boldsymbol{z}_k = \begin{bmatrix} z_{1,k} & z_{2,k} & \dots & z_{m,k} & \dots & z_{\mathrm{M},k} \end{bmatrix} \tag{4.111}$$

$$\boldsymbol{h}_k = \begin{bmatrix} h_{1,k} & h_{2,k} & \dots & h_{m,k} & \dots & h_{\mathrm{M},k} \end{bmatrix} \tag{4.112}$$

$$\boldsymbol{n}_k = \begin{bmatrix} n_{1,k} & n_{2,k} & \dots & n_{m,k} & \dots & n_{\mathrm{M},k} \end{bmatrix}. \tag{4.113}$$

SLAM Algorithm

The dual Kalman model shown in Section 4.6.3 is applied that works by alternating between learning the parameter vector $\hat{\boldsymbol{w}}_k$ given the current estimated position of the user $\hat{\boldsymbol{x}}_k$, and using the newly obtained $\hat{\boldsymbol{w}}_k$ to estimate $\hat{\boldsymbol{x}}_{k+1}$. Two separate ISPKS filters with gating as described in Section 4.5.4 are used for the state and parameter estimation. The incoming range observations are fed to both the filters. As before, the dual estimator is operated at a fixed rate of $f_s = 2\,\text{Hz}$. There are some points to be noted in the above dual estimation algorithm:

- Before the arrival of the first measurement, at $k = 0$, the weight vector $\boldsymbol{w}_k$ is initialized as

$$\hat{\boldsymbol{w}}_0 = \boldsymbol{w}_0, \tag{4.114}$$

  where $\boldsymbol{w}_0$ is

$$\boldsymbol{w}_0 = \begin{bmatrix} c_0 & \boldsymbol{x}^{\text{S}}_{1,0} & \boldsymbol{x}^{\text{S}}_{2,0} & \dots & \boldsymbol{x}^{\text{S}}_{m,0} & \dots & \boldsymbol{x}^{\text{S}}_{\text{M},0} \end{bmatrix}. \tag{4.115}$$

  The 2D coordinates $\boldsymbol{x}^{\text{S}}_{m,0}$ for the m-th sensor is roughly chosen to lie within $2-3\,\text{ft}$ from its actual location and $c_0 = 1128\,\text{ft/s}$ which is the speed of sound in free space. The values of $\boldsymbol{x}^{\text{S}}_{m,0}$ and c_0 are chosen accordingly for faster convergence.

- At $k = 0$, the state $\boldsymbol{x}_k$, containing the position of the person being tracked is initialized to within an approximate radius of 2 ft.

- Convergence of the parameters using the dual approach is typically achieved within 5-10 s of tracking.

Experimental Results

In this section, our objective is to demonstrate the tracking accuracy of the proposed LANS-SLAM approach using ultrasonic range observations. Note, we have only used the active range observations to generate the results. We demonstrate two different trials in which two subjects walked in the Sonar-Lab. Subject 1 took 60 s to complete the path and 154 range observations were recorded during that time period. Subject 2 completed the other path in 50 s and recorded 136 sonar ranges. The Ubisense tracking engine, which provides reliable ground truth was turned on for performance comparison with our tracker.

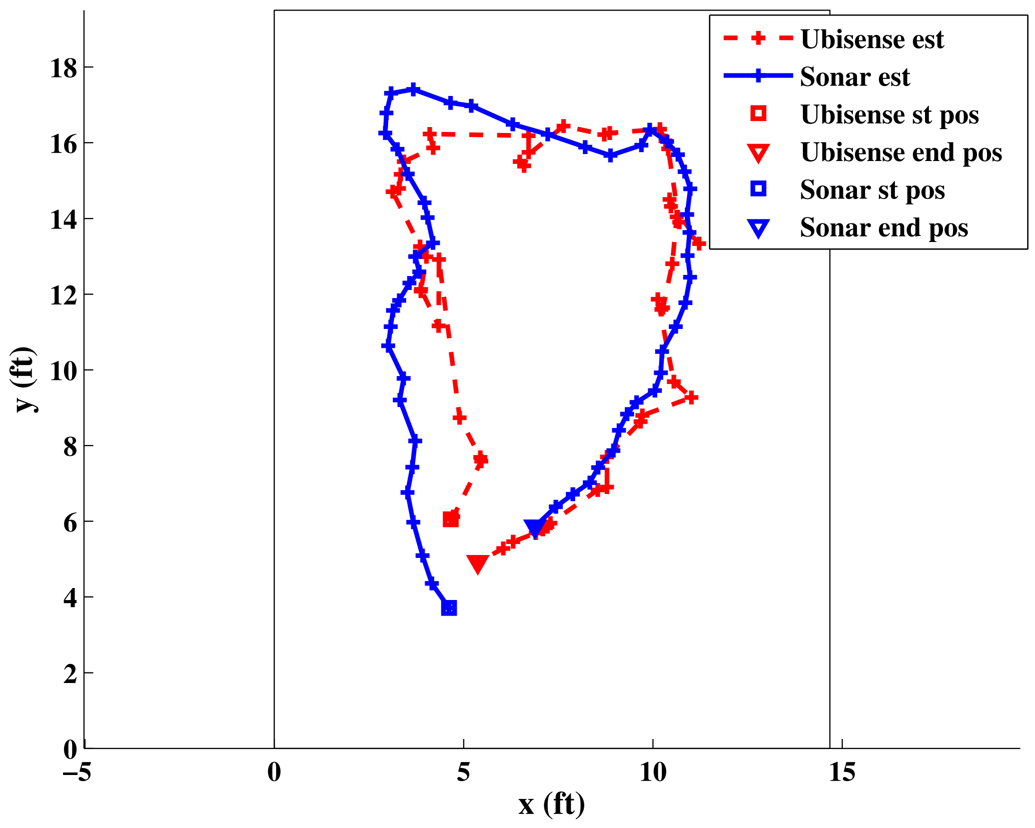
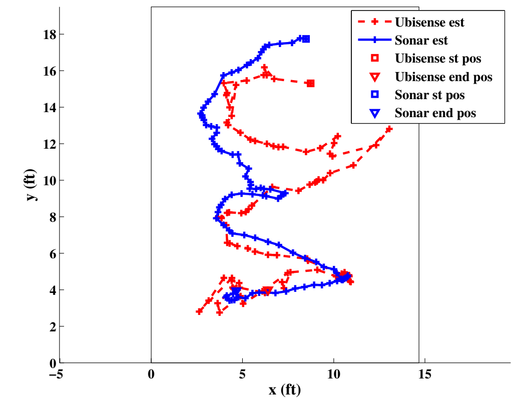
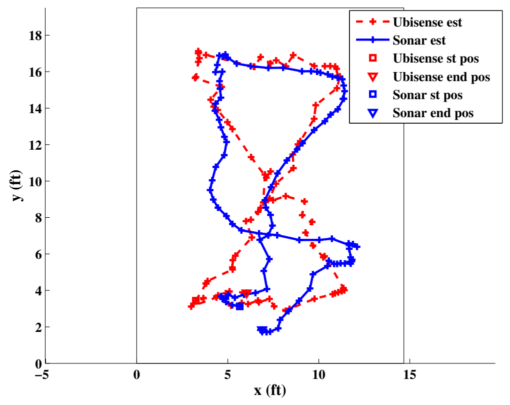


Figure 4.20: Tracking performance in Sonar-Lab using the LANS-SLAM technique (*red*: Ubisense estimates, *blue*: ISPKS estimates). The RMSE between the ISPKS and Ubisense estimated positions is equal to RMSE = 1.78 ft. The above tracking result is shown for subject 2.

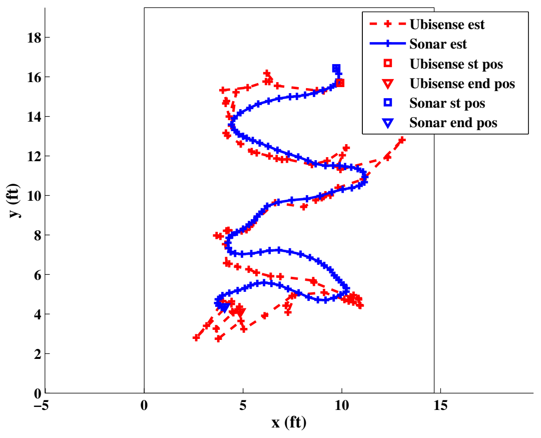
Figure 4.19-4.20 compares the position estimates obtained from the Ubisense engine and the LANS-SLAM method. As can be seen, the dual ISPKS adopted by the LANS-SLAM generates position estimates comparable to the Ubisense engine. As expected, the tracking is not accurate during the initial period because the observation model parameters require some time to converge to its true values. However, after the parameters converge, the LANS-SLAM estimated positions closely follow the Ubisense. The figures clearly illustrate that the superior accuracy of the Ubisense system, which is expensive, tag-based and requires a detailed calibration procedure, can be mimicked by our in-house, tag-free, self-learning ISPKS based tracking engine. In addition, the estimation accuracy of the LANS-SLAM approach is found to be consistent over multiple subjects and random paths.



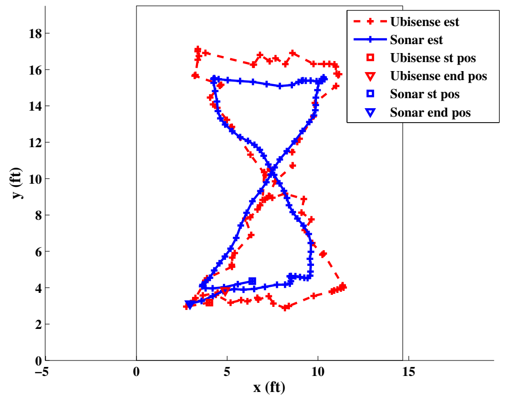
(a) Trial 1, RMSE=4.92 ft



(b) Trial 2, RMSE=3.81 ft



(c) Trial 1, RMSE=1.43 ft



(d) Trial 2, RMSE=2.3 ft

Figure 4.21: Comparison of the tracking performance between the LANS-SLAM and range-map estimation using two different trials, (a) to (b): range-map results (*blue*): user positions are estimated using fixed observation maps learned during the calibration phase. (c) to (d): LANS-SLAM results (*blue*): user positions are estimated using a dual ISPKS. The Ubisense results *red* provide the ground truth for comparison.

Figure 4.21(a)-4.21(d) compares the estimation accuracy of the *LANS-SLAM* approach against the range-map estimation case described in Section 4.5. The same incoming range measurements are used by both of these techniques to generate the position estimates. Figure 4.21(a)-4.21(b) present the range-map estimation results using the learned observation maps. Note, the parameters of each observation map are learned in a separate calibration phase and are kept fixed during user tracking. In these cases, the estimated

Table 4.4: Tracking accuracy of the LANS-SLAM in terms of RMSE. The mean and standard deviation of RMSE is calculated by averaging over 50 different trials performed by multiple subjects in the test Lab.

Estimator	E(RMSE) (ft)	std(RMSE) (ft)
LANS-SLAM	2.02	1.04
TANS-SLAM	2.64	1.12
Range-map	3.08	1.75

trajectories fail to follow the Ubisense estimates closely, which result significantly higher RMSE between the Ubisense and ISPKS estimates. On the other hand, Figure 4.21(c)-4.21(d) demonstrate the tracking performance of the LANS-SLAM algorithm. As is clearly evident from the figures, the LANS-SLAM method improves the quality of the estimated positions which reduces the RMSE. Table 4.4 tabulates the RMSE of position estimates for 50 different trials performed by multiple subjects in the Sonar-Lab. As seen, the LANS-SLAM algorithm generates tracking results with superior mean and standard deviation of RMSE over the range-map based estimator. Note that the LANS-SLAM produces better tracking accuracy than the TANS-SLAM method. The above figures and table suitably verify that the dual LANS-SLAM method indeed converges and after convergence it can also track a person with superior accuracy.

4.7 Indoor Tracking Using Active and Passive Ultrasonic Transducers

In this section our objective is to evaluate how incorporating the range measurements from "*passive*" ultrasonic sensors affect the estimator performance. Recall that the specific sonar-module that emits an ultrasonic signal and records the primary echo is referred as the *"active"* unit. The other sonar-modules at that time act as "*passive*" units and record indirect reflections or shadows coming from the active unit.

Figure 4.22 demonstrates the ultrasonic reflections recorded by two sensors when they alternately emit an ultrasonic signal. First, the ultrasonic sensor 1 transmits an ultrasonic burst and listens its primary reflection, which is shown in Figure 4.22 (a). Notice, the strongest echo comes from the person's body. The sonar unit 2 records the indirect

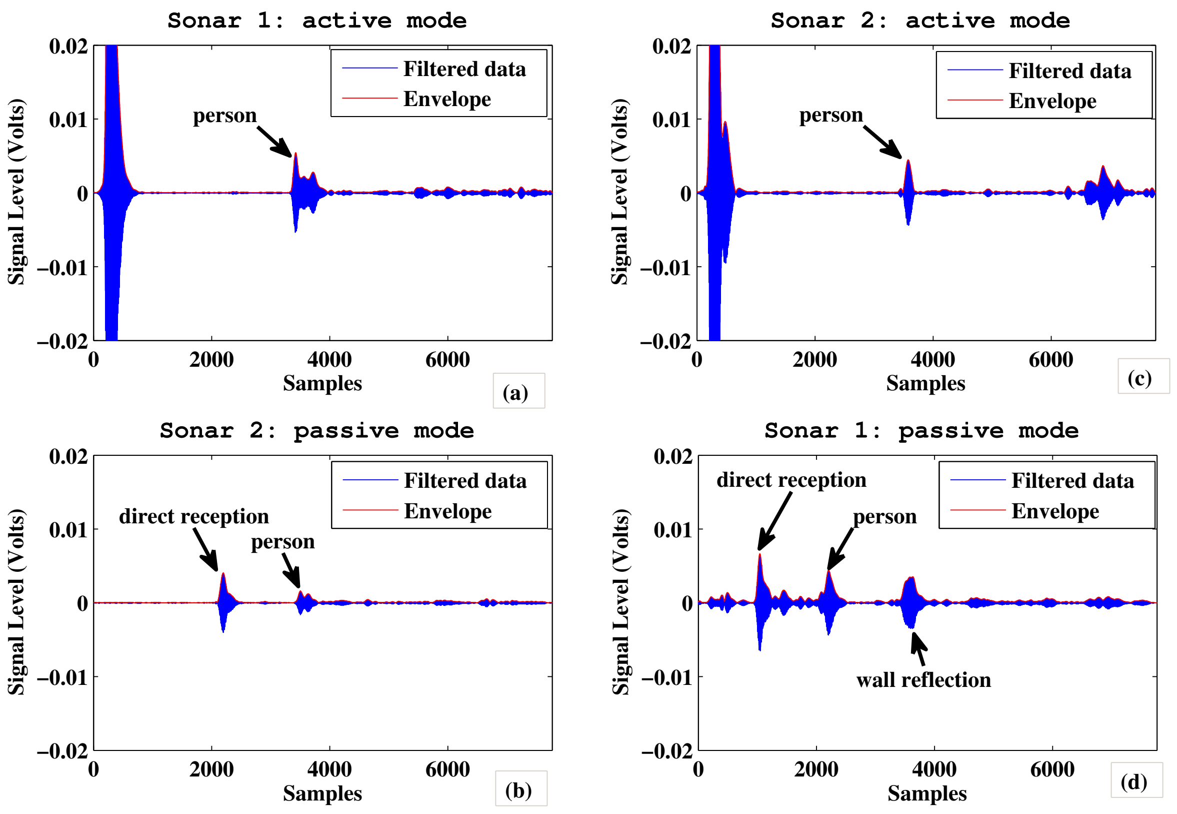


Figure 4.22: Ultrasonic signals for active and passive transducers. We show the outputs of two sonar units when they alternately are in active and passive mode. (a) and (b): Ultrasonic signals recorded by sensor 1 and 2 when sensor 1 is in active (*transmit and listen*) and sensor 2 is in passive (*only listen*) mode. (c) and (d): Ultrasonic signals recorded by sensor 1 (passive) and sensor 2 (active).

reflection demonstrated in Figure 4.22 (b). We turn our attention to two signal peaks in this figure: the strongest one is the result of direct signal propagation from the sensor 1 transmitter to sensor 2 receiver; the other is the reflection from the person's body. Figure 4.22 (c) and Figure 4.22 (d) demonstrate the ultrasonic reflections recorded by the sonar units 2 and 1 respectively, when the sensor 2 is in active and sensor 1 is in passive mode. As shown in Figure 4.22 (d), the passive ultrasonic signal possesses two other strong echoes beside the body reflection, which may result from direct reception and wall reflection. In this respect, we want to mention that the amplitude of the echo due to direct signal propagation from the active transmitter unit to the passive receiver unit remains constant except when a person/object blocks the direct path.

This concept can easily be generalized to an arbitrary number M of ultrasonic sensors. Each individual sensor takes its turn periodically to transmit an ultrasonic signal whereas all M sensors listen the returning echo. Previously we only incorporated the primary echo received by the active unit and neglected the other $M-1$ passive observations. In other words, only $M=6$ sonar observations are used at each k in the tracking algorithm. In this section, we include all active and passive measurements in the estimation framework. Hence for $M=6$, there are a total of $M^2=36$ recorded sonar returns (6 active and 30 passive) at each k. This in effect increases the dimension of the observation vector and correspondingly causes delay in generating the estimates.

4.7.1 State Space Model

We use the dual estimation framework shown in Section 4.6.3, where the observation model parameters $\boldsymbol{w}_k$ are simultaneously estimated with the state $\boldsymbol{x}_k$. We follow the LANS-SLAM approach as it is *self-calibrating*, contains less number of parameters to estimate and produces the best tracking performance.

Dynamic Model

We apply the potential field embedded CT model (see Section 4.5.2) as the dynamic model.

Observation Model

The observation models for active and passive sensors follow the *time-of-flight* equations. The observation equation for the m-th active sonar unit with 2D coordinates $\begin{bmatrix} x^{\mathrm{S}}_{m,k} & y^{\mathrm{S}}_{m,k} \end{bmatrix}$ can be shown as

$$z^{\mathrm{a}}_{m,k} = h^{\mathrm{a}}_{m,k}\left(\begin{bmatrix} x_k & y_k \end{bmatrix}; \begin{bmatrix} x^{\mathrm{S}}_{m,k} & y^{\mathrm{S}}_{m,k} & c_k \end{bmatrix}\right) + n^{\mathrm{a}}_{m,k}, \ \text{for } 1 \le m \le \mathrm{M} \tag{4.116}$$

where the observation model $h^{\mathrm{a}}_{m,k}$ (superscript “a” denotes active unit) for the m-th active unit is the same as shown in Equation (4.108)

$$h^{\mathrm{a}}_{m,k} = \frac{2N_{\mathrm{S}}}{c_k}\sqrt{\left(x_k - x^{\mathrm{S}}_{m,k}\right)^2 + \left(y_k - y^{\mathrm{S}}_{m,k}\right)^2}. \tag{4.117}$$

Similarly, we demonstrate the observation equation for the q-th passive unit when the m-th sensor is active

$$z^{\mathrm{p}}_{q,k} = h^{\mathrm{p}}_{q,k}\left(\left[\begin{array}{cc} x_k & y_k \end{array}\right];\left[\begin{array}{ccccc} x^{\mathrm{s}}_{m,k} & y^{\mathrm{s}}_{m,k} & x^{\mathrm{s}}_{q,k} & y^{\mathrm{s}}_{q,k} & c_k \end{array}\right]\right) + n^{\mathrm{p}}_{q,k},\ \text{for } 1 \leq q \leq \text{M-1}, \forall q \neq m \tag{4.118}$$

where the observation model $h^{\mathrm{p}}_{q,k}$ (superscript "p" denotes passive unit) defines the distance from the m-th active unit to the subject, plus the distance from the subject to the q-th passive unit,

$$h^{\mathrm{p}}_{q,k} = \frac{N_{\mathrm{S}}}{c_k}\left(\sqrt{\left(x_k - x^{\mathrm{s}}_{m,k}\right)^2 + \left(y_k - y^{\mathrm{s}}_{m,k}\right)^2} + \sqrt{\left(x_k - x^{\mathrm{s}}_{q,k}\right)^2 + \left(y_k - y^{\mathrm{s}}_{q,k}\right)^2}\right). \tag{4.119}$$

The q-th sensor is located at $\left[\begin{array}{cc} x^{\mathrm{s}}_{q,k} & y^{\mathrm{s}}_{q,k} \end{array}\right]$. The range observation at time k, $z^{\mathrm{a}}_{m,k}$ is obtained from the recording of the m-th active sensor. The passive sensor q generates a range observation $z^{\mathrm{p}}_{q,k}$ from an indirect reflection coming from the m-th active unit. The parameter vector $\boldsymbol{w}_k$ to be estimated contains the 2D coordinates of each ultrasonic transducer and the speed of sound

$$\boldsymbol{w}_k = \left[\begin{array}{ccccccc} c_k & \boldsymbol{x}^{\mathrm{s}}_{1,k} & \boldsymbol{x}^{\mathrm{s}}_{2,k} & \ldots & \boldsymbol{x}^{\mathrm{s}}_{m,k} & \ldots & \boldsymbol{x}^{\mathrm{s}}_{\mathrm{M},k} \end{array}\right]. \tag{4.120}$$

Figure 4.2 demonstrates that the range observation r^{a}_m recorded by the m-th active sonar unit corresponds to twice the distance, d^{a}_m, between the active unit and the person. Whereas the passive range observation r^{p}_q corresponds to the sum of distances d^{a}_m and d^{p}_q, where d^{p}_q is the distance from the person to the q-th passive unit.

In addition to the active and passive ultrasonic units, we also include a state constraint in the observation model corresponding to a "shadow" condition. As shown in Figure 4.22, there is a signal return due to the direct path between an active and a passive unit. A sudden drop or change in the variation of the direct path return indicates that the subject has walked directly between the m-th active and q-th passive unit. For example as shown in Fig. 4.2, the subject is blocking the direct path between the active unit and the 5-th passive unit. Hence the corresponding passive unit should record a sudden drop of signal strength in its direct return from the active unit. If this condition is detected, an additional observation equation is included, which constrains the 2D position of the subject to be on a straight line between the m-th and q-th ultrasonic units.

$$z^{\mathrm{c}}_{q,k} = \boldsymbol{h}^{\mathrm{c}}_{q,k} \begin{bmatrix} x_k & y_k \end{bmatrix}, \tag{4.121}$$

where $z^{\mathrm{c}}_{q,k}$ is the pseudo observation

$$z^{\mathrm{c}}_{q,k} = \frac{N_{\mathrm{S}}}{c_k} \left[\left(y^{\mathrm{s}}_{m,k} - y^{\mathrm{s}}_{q,k} \right) x^{\mathrm{s}}_{q,k} + \left(x^{\mathrm{s}}_{q,k} - x^{\mathrm{s}}_{m,k} \right) y^{\mathrm{s}}_{q,k} \right]. \tag{4.122}$$

The constraint observation model $\boldsymbol{h}^{\mathrm{c}}_{q,k}$ is defined as

$$\boldsymbol{h}^{\mathrm{c}}_{q,k} = \frac{N_{\mathrm{S}}}{c_k} \begin{bmatrix} \left(y^{\mathrm{s}}_{m,k} - y^{\mathrm{s}}_{q,k} \right) & \left(x^{\mathrm{s}}_{q,k} - x^{\mathrm{s}}_{m,k} \right) \end{bmatrix}. \tag{4.123}$$

Note, the constraint equation, as shown in Equation (4.121) has no measurement noise. Although the constraint equation with "*perfect*" measurements does not pose any theoretical problems, when augmented with Equation (4.126), the covariance of the new measurement noise may become singular. A singular noise covariance increases the possibility of numerical problems for the estimator. In addition, the *Cholesky* decomposition, which computes matrix square-roots, cannot be performed on a singular matrix. Hence, we modify the constraint equation by adding a small measurement noise to the "*perfect*" measurements

$$z^{\mathrm{c}}_{q,k} = \boldsymbol{h}^{\mathrm{c}}_{q,k} \begin{bmatrix} x_k & y_k \end{bmatrix} + n^{\mathrm{c}}_{q,k}, \tag{4.124}$$

Combining all observations, observation functions and noise terms for a fixed active unit m, we form a multidimensional observation model

$$\boldsymbol{z}_{m,k} = \begin{bmatrix} z^{\mathrm{a}}_{m,k} & z^{\mathrm{p}}_{1,k} & \dots & z^{\mathrm{p}}_{q,k} & \dots & z^{\mathrm{p}}_{\text{M-1},k} & z^{\mathrm{c}}_{q,k} \end{bmatrix} \tag{4.125}$$

$$\boldsymbol{h}_{m,k} = \begin{bmatrix} h^{\mathrm{a}}_{m,k} & h^{\mathrm{p}}_{1,k} & \dots & h^{\mathrm{p}}_{q,k} & \dots & h^{\mathrm{p}}_{\text{M-1},k} & \boldsymbol{h}^{\mathrm{c}}_{q,k} \end{bmatrix} \tag{4.126}$$

$$\boldsymbol{n}_{m,k} = \begin{bmatrix} n^{\mathrm{a}}_{m,k} & n^{\mathrm{p}}_{1,k} & \dots & n^{\mathrm{p}}_{q,k} & \dots & n^{\mathrm{p}}_{\text{M-1},k} & n^{\mathrm{c}}_{q,k} \end{bmatrix}. \tag{4.127}$$

Note that the dimension of each augmented vector will very depending on whether the constraint term is present. Finally, for all active units $m \in \mathrm{M}$, we form the complete observation model

$$\boldsymbol{z}_k = \begin{bmatrix} \boldsymbol{z}_{1,k} & \boldsymbol{z}_{2,k} & \dots & \boldsymbol{z}_{m,k} & \dots & \boldsymbol{z}_{\mathrm{M},k} \end{bmatrix} \tag{4.128}$$

$$\boldsymbol{h}_k = \begin{bmatrix} \boldsymbol{h}_{1,k} & \boldsymbol{h}_{2,k} & \dots & \boldsymbol{h}_{m,k} & \dots & \boldsymbol{h}_{\mathrm{M},k} \end{bmatrix} \tag{4.129}$$

$$\boldsymbol{n}_k = \begin{bmatrix} \boldsymbol{n}_{1,k} & \boldsymbol{n}_{2,k} & \dots & \boldsymbol{n}_{m,k} & \dots & \boldsymbol{n}_{\mathrm{M},k} \end{bmatrix}, \tag{4.130}$$

where the augmented measurement, observation model, observation noise at time k are denoted as $\boldsymbol{z}_k$, $\boldsymbol{h}_k$ and $\boldsymbol{n}_k$ respectively.

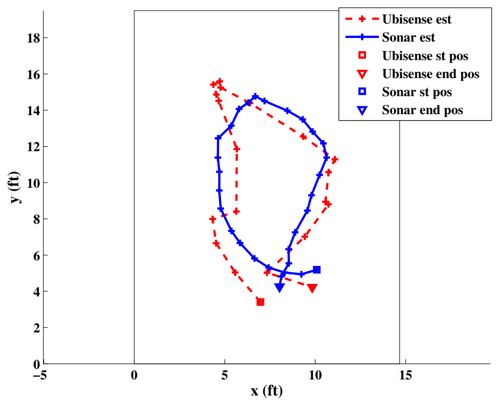
SLAM Algorithm

The dual Kalman framework shown in Section 4.6.3 is adopted. Recall that it incorporates two ISPKS filters with gating to alternately estimate the state $\boldsymbol{x}_k$ and model parameters $\boldsymbol{w}_k$ using the range observations $\boldsymbol{z}_k$. The only difference with the dual estimator in Section 4.6.3 is the dimension of the observation vector at time k. As we consider measurements coming from both the active and passive ultrasonic sensors, the dimension of each observation vector $\boldsymbol{z}_k$ is of size equal to M^2 compared to size M for the LANS-SLAM case which considers only the active sonar units.

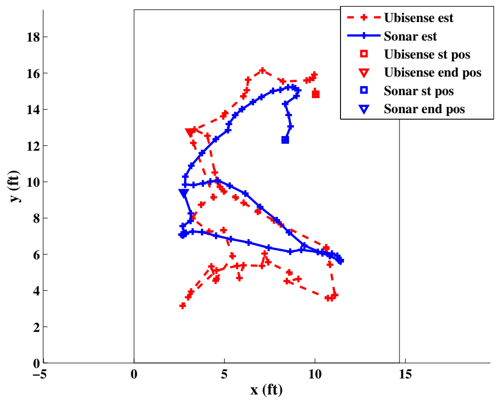
4.7.2 Experimental Results

In this section, we illustrate the performance results of the LANS-SLAM approach incorporating both the active and passive sonar measurements and compare them with the results obtained using only the active sonar data. The objective here is to evaluate how incorporating the passive sonar ranges affect the tracking accuracy. We demonstrate two random trajectories followed by two different subjects. Subject 1 took 40 s to complete the path and 96 range observations were recorded during that time period. Subject 2 completed the path in 60 s and recorded 144 measurements.

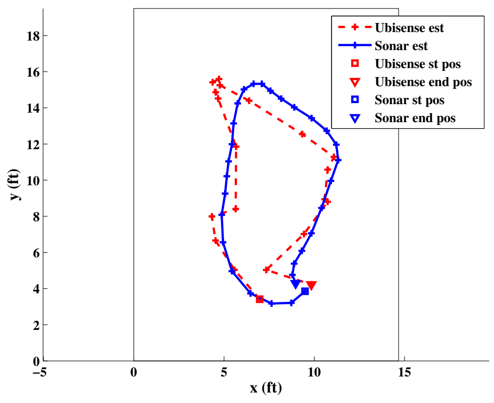
Figure 4.23(a)-4.23(d) show the LANS-SLAM estimated positions with/without passive measurements for subjects 1 and 2. Figure 4.23(a) and 4.23(b) are the LANS-SLAM position estimates using only active sonar measurements, whereas both active and passive measurements are used to generate the estimated positions shown in Figure 4.23(c) and



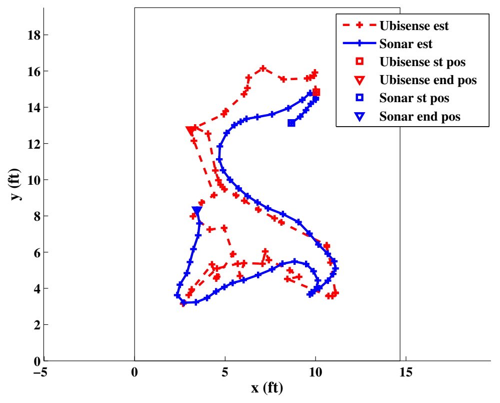
(a) Trial 1, RMSE=1.94 ft



(b) Trial 2, RMSE=2.85 ft



(c) Trial 1, RMSE=1.48 ft



(d) Trial 2, RMSE=2.01 ft

Figure 4.23: Tracking performance in Sonar-Lab using the LANS-SLAM incorporating both *active* and *passive* ultrasonic transducers, (a) and (b): LANS-SLAM estimates (*blue*) using range measurements recorded by active units, (c) and (d): LANS-SLAM estimates (*blue*) using range measurements recorded by active and passive units. The Ubisense results (*red*) provide the ground truth for comparison. Two subjects performed the walking to generate the data.

4.23(d). As before, the accuracy of our tracking results are compared in terms of RMSE between our estimates and the Ubisense engine. It is clearly evident that the position estimates shown in Figure 4.23(c) and 4.23(d) are more accurate compared to that in Figure 4.23(a) and 4.23(b).

Table 4.5 tabulates the RMSE of position estimates for 50 different trials performed by multiple subjects in the Sonar-Lab. The LANS-SLAM estimator, which obtains range

Table 4.5: Tracking accuracy of the LANS-SLAM with active and passive sonar measurements in terms of RMSE. The mean and standard deviation of RMSE is calculated by averaging over 50 different trials performed by multiple subjects in the test Lab.

Estimator	E(RMSE) (ft)	std(RMSE) (ft)
LANS-SLAM (active+passive)	1.66	0.92
LANS-SLAM (active)	2.02	1.04

measurements from the active and passive sonar units generates position estimates with superior mean and standard deviation of RMSE over the case when no passive range measurements are used. The above figures and table suitably verify that incorporating the passive ultrasonic units into the location tracking method indeed helps to improve the tracking accuracy.

4.8 Discussion and Future Work

In this chapter, we develop a novel method for indoor tracking using range measurements from ultrasonic sensors. Instead of wearing a body-borne receiver tag, we propose an unobtrusive tag-free tracking system, which requires setting up a set of ultrasonic transducers inside the indoor location. Each sonar unit records analog echoes, which are then digitized and passed through a number of signal processing steps including Bandpass filtering, Hilbert transformation, background subtraction, and clustering to calculate a set of range candidates. An observation function is generated from the calibration data by fitting nonlinear maps between the known calibration locations and active sonar ranges. The observation range maps are incorporated into an ISPKS based tracking algorithm that fuses all range measurements with a CT based dynamic model to generate 2D position, velocity and turn rate. A gating technique is also integrated in the range-map generation and state estimation process to avoid confusion from multiple range candidates.

Instead of using a fixed observation model, a simultaneous localization and mapping algorithm is designed to simultaneously estimating the state of the person and the parameters of the observation model. The TANS-SLAM method still uses observation maps in the estimation process but the map parameters are updated during tracking using the newly generated state estimates. The LANS-SLAM approach completely eliminates the offline calibration phase. A dual estimation framework is employed which operates two separate ISPKS in an alternate fashion. One SPKS localizes the user by taking input the estimated observation model parameters and the other computes the parameters given the currently estimated user position. Parameters in the LANS-SLAM correspond to the 2D sonar module locations along with a correction factor to account for various noise and modeling errors. Finally, we also incorporate the indirect range observations from the passive sonar modules into the LANS-SLAM framework and evaluate the system performance. Accuracy of all the proposed tracking algorithms are evaluated on a data set involving 50 different trials performed by multiple subjects in a test Lab. The tracking accuracy is determined through a comparison with the commercial *Ubisense* engine. It is shown that the ultrasonic sensor based tag-free tracking performs comparably with the Ubisense, which is tag based, more expensive and also encounters several calibration challenges. The LANS-SLAM algorithm with all range observations from the active and passive sonar-modules provides the highest tracking accuracy. The LANS-SLAM method is the most attractive choice for the ultrasonic sensor based tag-free indoor tracking because it requires minimal calibration, uses fewer number of parameters and demonstrates fast convergence (within 5-10 s) starting from a rough estimate of sonar locations.

Future research direction includes developing a more refined model of human walking, adding a better gating technique to determine the correct range from range candidates and extending the system to a multi-room facility. In this context, one can incorporate a biomechanical model in the tracking framework as that can mimic accurately the way human walks. Investigation on the multiple hypothesis approach can help in developing a better gating method. The current system also requires cabling to a multi-channel DAQ operating at a fast 250 kHz sampling rate to allow for subsequent digital demodulation. This could be replaced with analog demodulation, allowing for the uses of a low cost D/A and wireless transmission to the central PC. The current system is also obviously limited to single person tracking, thus limiting the use to applications such as monitoring elders in independent living facilities. Multi-person tracking may be possible through extensions in the (Sigma-Point) Kalman algorithms based on multi-target tracking methods and probabilistic data association, along with possibly other sensors to help differentiate sonar returns between people. In addition, future research also can be directed to determining optimal placement of ultrasonic transducers in an indoor location in order to maximize the tracking accuracy.

4.9 Appendix

4.9.1 Autonomous Terrain Aided Navigation in Unknown Environment

In this section, we describe the application of a dual SPKF to the problem of simultaneous estimation of the state and map parameters for localizing an unmanned vehicle. The experimental scenario considered here consists of an unmanned vehicle maneuvering through an unsurveyed environment within a 10 m×10 m bounded region. Though the terrain aided navigation algorithm presented here can be extended to a full three dimensional environment (e.g., for unmanned aerial vehicles), for ease of visualization, simulations are conducted in two dimensions. Three onboard sensors obtain terrain information and an IMU provides acceleration and angular rate in the body fixed frame. We also compare our dual SPKF based system performance to the baseline EKF in terms of vehicle state estimation accuracy. We will start with describing the specific process and observation (map) models used inside the SPKF/EKF based estimators and then moving on to the experimental results.

Vehicle process model

In the vehicle process model, the standard IMU driven kinematic process model formulation [117] that comprises an inertial navigation system (INS) mechanization component and sensor bias error components is followed. In two dimensions, the IMU includes two accelerometer and one rate gyro. Errors in the sensors include both bias and additive noise. The vehicle state vector is shown as

$$\boldsymbol{x} = \begin{bmatrix} x & y & v_x & v_y & \psi & b_x & b_y & b_\omega \end{bmatrix}^T, \tag{4.131}$$

where $\begin{bmatrix} x & y \end{bmatrix}$ is vehicle 2D position, $\begin{bmatrix} v_x & v_y \end{bmatrix}$ is 2D velocity, ψ is the Euler (heading) angle. The accelerometer bias vector is denoted as $\boldsymbol{b}_a = \begin{bmatrix} b_x & b_y \end{bmatrix}$ and b_ω is the IMU gyro rate bias. There is no separate scale-error term in the state vector, as it is found to be sufficient to model both the sensor bias and scale error as a time varying bias term. The continuous time kinematic equations of the vehicle [5] followed in this simulation can be defined as

$$\begin{aligned}
\dot{x} &= \cos\psi v_x - \sin\psi v_y \\
\dot{y} &= \sin\psi v_x + \cos\psi v_y \\
\dot{\psi} &= z_\omega - b_\omega \\
\dot{v}_x &= z_x - b_x + (z_\omega - b_\omega)\, v_y \\
\dot{v}_y &= z_y - b_y + (z_\omega - b_\omega)\, v_x \\
\dot{b}_x &= v_{b_x} \\
\dot{b}_y &= v_{b_y} \\
\dot{b}_\omega &= v_{b_\omega},
\end{aligned} \tag{4.132}$$

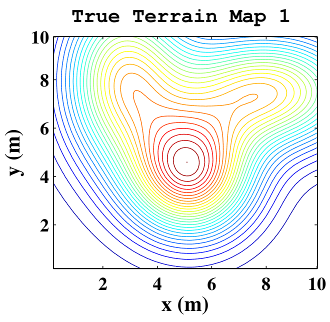
where $\boldsymbol{z}_a = \begin{bmatrix} z_x & z_y \end{bmatrix}^T$ is the noisy observation from accelerometer

$$\boldsymbol{z}_a = \boldsymbol{a} + \boldsymbol{b}_a + \boldsymbol{n}_a, \tag{4.133}$$

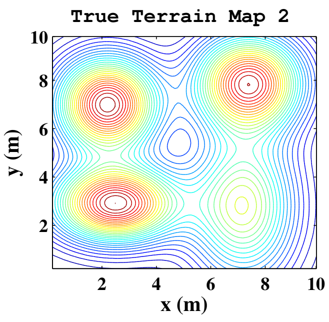
and b_ω is the noisy measurement obtained from the rate gyro

$$z_\omega = \omega + b_\omega + \boldsymbol{n}_\omega. \tag{4.134}$$

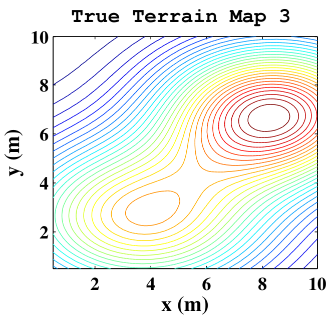
$\boldsymbol{a} = \begin{bmatrix} a_x & a_y \end{bmatrix}^T$ and ω are the vehicle's true acceleration and angular velocity respectively. Note that in reality, the accelerometer and gyro outputs slowly drift from the true values. The sensor bias terms shown in (4.133) and (4.134) are incorporated to compensate the sensor drifts. It is assumed that the accelerometer is situated exactly at the vehicle center of gravity (CG). The Gaussian white process and observation noises are denoted as $v_{(\cdot)}$ and $n_{(\cdot)}$ respectively. Sampling rate of $\delta T = 100\,\text{ms}$ is chosen in order to discretize the continuous time vehicle kinematic model in (4.132). Note that the accelerometer and gyro biases are modeled as a slowly varying random walk where v_{b_x}, v_{b_y} and v_{b_ω} are zero mean Gaussian random variable. As the biases are unknown random variable, the accelerometer and gyro sensors may not be sufficient on its own to localize the vehicle. Hence three terrain sensors are incorporated into the tracking system which provide some information about the environment at the current vehicle location.



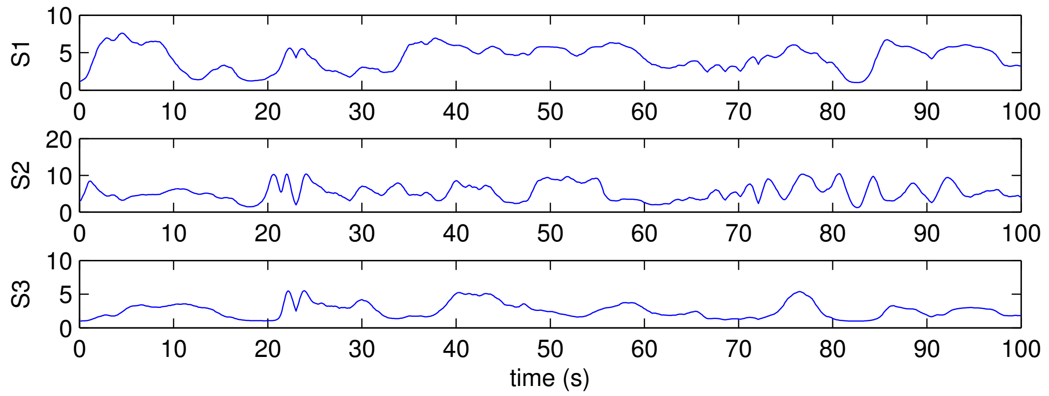
(a)



(b)



(c)



(d)

Figure 4.24: (a) to (c): True terrain maps used in the simulation, (d): Traces of three noisy terrain sensor measurements.

Vehicle observation model (Terrain map)

In this simulation, three terrain maps are used. Each map can be thought of as providing complimentary terrain information such as altitude, pressure, or visual characteristics of the environment. For this preliminary study, each map is modeled as a mixture of three, five and two Gaussian distributions respectively. Clearly, this is not a complex realistic scenario, but serves the purpose to investigate the ability of SPKF estimators to generate consistent estimates of the vehicle state and map parameters after convergence and also to provide a performance comparison between the SPKF and EKF implementation. Figure 4.24(a) - 4.24(c) illustrate the visual contour plots of these maps.

The measurements from the $M = 3$ onboard sensors correspond to each of the map values at the current true vehicle location plus additive noise. The terrain sensors obtain the measurements at the rate of $s = 10\,\mathrm{Hz}$. The role of the Kalman filter state-estimator is thus to fuse these three sensor readings with the process model to estimate the vehicle location and heading. The signal to noise ratio (SNR) at the output of each terrain sensor is kept as 20 dB. A typical sensor trace from a random trajectory is shown in Figure 4.24(d).

In the dual Kalman filter setup for SLAM, the maps are assumed unknown and some parametric representation of the maps must be simultaneously learned from M sensor traces. As described earlier, the true map in this case consists of a simple mixture of Gaussian densities. However, since we cannot assume prior knowledge of this, we attempt to learn this map using both two layer multi-layer perceptrons (MLP) and radial basis function (RBF) neural networks. The vehicle observation model can be defined as

$$z_{m,k} = h_{m,k}\left(x_k, y_k\right) + n_{m,k}, \tag{4.135}$$

where $z_{m,k}$ is the sensor measurement from the m-th terrain sensor, $m,\ 1 \leq m \leq M$, with noise $n_{m,k}$ assumed to be Gaussian with zero mean. The observation map $h_{m,k}$ can either be learned using MLP or RBF. The discrete time index is denoted as k.

The *MLP* observation map [88], $h^{\mathrm{M}}_{m,k}$, for the m-th sensor is specified by

$$h^{\mathrm{M}}_{m,k}\left(x_k, y_k\right) = \left(\boldsymbol{W}2^{\mathrm{M}}_{m,k}\right)^T \tanh\left(\left(\boldsymbol{W}1^{\mathrm{M}}_{m,k}\right)^T \begin{bmatrix} x_k & y_k \end{bmatrix} + \boldsymbol{B}1^{\mathrm{M}}_{m,k}\right) + b2^{\mathrm{M}}_{m,k}, \tag{4.136}$$

where $\boldsymbol{W}1^{\mathrm{M}}_{m,k}$ and $\boldsymbol{W}2^{\mathrm{M}}_{m,k}$ are the hidden and output layer weights and $\boldsymbol{B}1^{\mathrm{M}}_{m,k}$ and $b2^{\mathrm{M}}_{m,k}$ are the input and output biases for the m-th terrain sensor. Note that in the above equation, we have adopted a standard three layer form [88] of MLP. Assuming there are H nodes in the hidden layer,

$$\boldsymbol{W}1^{\mathrm{M}}_{m,k} = \begin{bmatrix} w1^{0,0}_{m,k} & w1^{0,1}_{m,k} & \cdots & w1^{0,H-1}_{m,k} \\ w1^{1,0}_{m,k} & w1^{1,1}_{m,k} & \cdots & w1^{1,H-1}_{m,k} \end{bmatrix} \tag{4.137}$$

$$\boldsymbol{B}1^{\mathrm{M}}_{m,k} = \begin{bmatrix} b1^{0}_{m,k} & b1^{1}_{m,k} & \cdots & b1^{H-1}_{m,k} \end{bmatrix}^T \tag{4.138}$$

$$\boldsymbol{W}2^{\mathrm{M}}_{m,k} = \begin{bmatrix} w2^{0}_{m,k} & w2^{1}_{m,k} & \cdots & w2^{H-1}_{m,k} \end{bmatrix}^T. \tag{4.139}$$

Combining all the network weights and biases a single weight vector $\boldsymbol{W}_{m,k}$ is formed for

m-th terrain sensor

$$\boldsymbol{W}_{m,k} = \begin{bmatrix} \boldsymbol{W}1^{\mathrm{M}}_{m,k} & \boldsymbol{W}2^{\mathrm{M}}_{m,k} & \boldsymbol{B}1^{\mathrm{M}}_{m,k} & b2^{\mathrm{M}}_{m,k} \end{bmatrix}^T. \tag{4.140}$$

Notice that all the individual weight terms are first vectorized before combining them to form the augmented weight vector $\boldsymbol{W}_{m,k}$. Note that the true value of $\boldsymbol{W}_{m,k}$ is unknown and it is estimated simultaneously with the vehicle state using the dual estimation framework.

Similarly for *RBF* case, the observation map $h^{\mathrm{R}}_{m,k}$ may be denoted as

$$h^{\mathrm{R}}_{m,k}(x_k, y_k) = \left(\boldsymbol{W}1^{\mathrm{R}}_{m,k}\right)^T \boldsymbol{K}^{\mathrm{R}}_{m,k}\left(\begin{bmatrix} x_k & y_k \end{bmatrix}; \boldsymbol{\mu}^{\mathrm{R}}_{m,k}, \boldsymbol{\Sigma}^{\mathrm{R}}_{m,k}\right), \tag{4.141}$$

where $\boldsymbol{K}^{\mathrm{R}}_{m,k}$ is the Gaussian kernel function [88] with mean vector $\boldsymbol{\mu}^{\mathrm{R}}_{m,k}$ and covariance matrix $\boldsymbol{\Sigma}^{\mathrm{R}}_{m,k}$,

$$\boldsymbol{\mu}^{\mathrm{R}}_{m,k} = \begin{bmatrix} \boldsymbol{\mu}^{0}_{m,k} & \boldsymbol{\mu}^{1}_{m,k} & \cdots & \boldsymbol{\mu}^{\mathrm{C}-1}_{m,k} \end{bmatrix}^T \tag{4.142}$$

$$\boldsymbol{\Sigma}^{\mathrm{R}}_{m,k} = \begin{bmatrix} \boldsymbol{\Sigma}^{0}_{m,k} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Sigma}^{1}_{m,k} & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & 0 & \boldsymbol{\Sigma}^{\mathrm{C}-1}_{m,k} \end{bmatrix}, \tag{4.143}$$

where C is the number of Gaussian kernels in the hidden layer of the RBF network and $\boldsymbol{W}1^{\mathrm{R}}_{m,k}$ is the output layer linear weights,

$$\boldsymbol{W}1^{\mathrm{R}}_{m,k} = \begin{bmatrix} w1^{0}_{m,k} & w1^{1}_{m,k} & \ldots & w1^{\mathrm{C}-1}_{m,k} \end{bmatrix}. \tag{4.144}$$

The parameter vector for the RBF model which needs to be estimated at each time k can be denoted as

$$\boldsymbol{W}_{m,k} = \begin{bmatrix} \boldsymbol{\mu}^{\mathrm{R}}_{m,k} & \boldsymbol{\Sigma}^{\mathrm{R}}_{m,k} & \boldsymbol{W}1^{\mathrm{R}}_{m,k} \end{bmatrix}^T. \tag{4.145}$$

As seen in (4.135), each MLP/RBF network implements a mapping from a two dimensional input space to one dimensional output space. Note that each terrain sensor has its own sensor map. The map learning problem is made more challenging by the fact that the inputs and outputs are limited to the random trajectory and corresponding sensor traces from the vehicle.

The observed sensor output $z_{m,k}$, terrain map $h_{m,k}$, parameter vector $\boldsymbol{W}_{m,k}$ and the observation noise $n_{m,k}$ from each terrain sensor are combined to form a multi-dimensional observation model,

$$\boldsymbol{z}_k = \begin{bmatrix} z_{1,k} & \dots & z_{m,k} & \dots & z_{\mathrm{M},k} \end{bmatrix} \tag{4.146}$$

$$\boldsymbol{h}_k = \begin{bmatrix} h_{1,k} & \dots & h_{m,k} & \dots & h_{\mathrm{M},k} \end{bmatrix} \tag{4.147}$$

$$\boldsymbol{W}_k = \begin{bmatrix} \boldsymbol{W}_{1,k} & \dots & \boldsymbol{W}_{m,k} & \dots & \boldsymbol{W}_{\mathrm{M},k} \end{bmatrix} \tag{4.148}$$

$$\boldsymbol{n}_k = \begin{bmatrix} n_{1,k} & \dots & n_{m,k} & \dots & n_{\mathrm{M},k} \end{bmatrix} \tag{4.149}$$

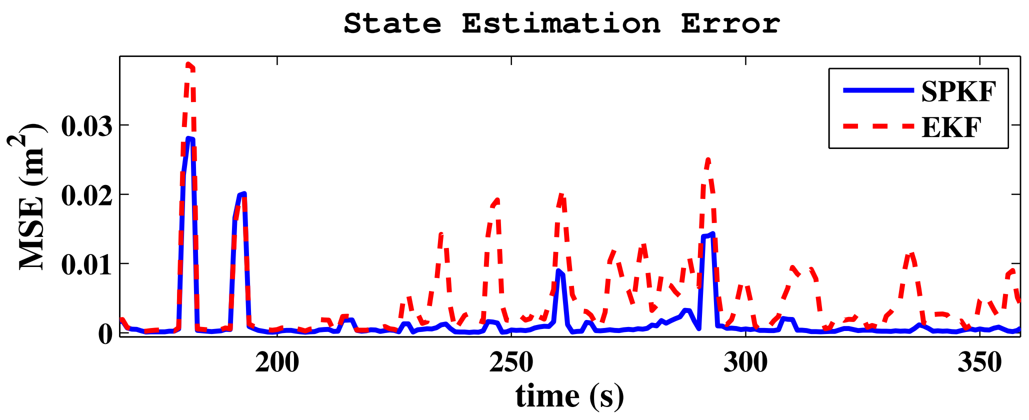
where $\boldsymbol{z}_k$ is the multi-dimensional RSSI observations emanating from each terrain sensor. Similarly $\boldsymbol{h}_k$, $\boldsymbol{W}_k$ and $\boldsymbol{n}_k$ are the augmented observation maps, augmented parameter vector and the augmented measurement noise for sensors $1 \leq m \leq \mathrm{M}$. SPKF based tracking algorithm using dual Kalman framework defined in (4.103)-(4.104) has been used to estimate both the state vector $\boldsymbol{x}_k$ and augmented parameter vector $\boldsymbol{W}_k$ at each time k.

Dual Estimation

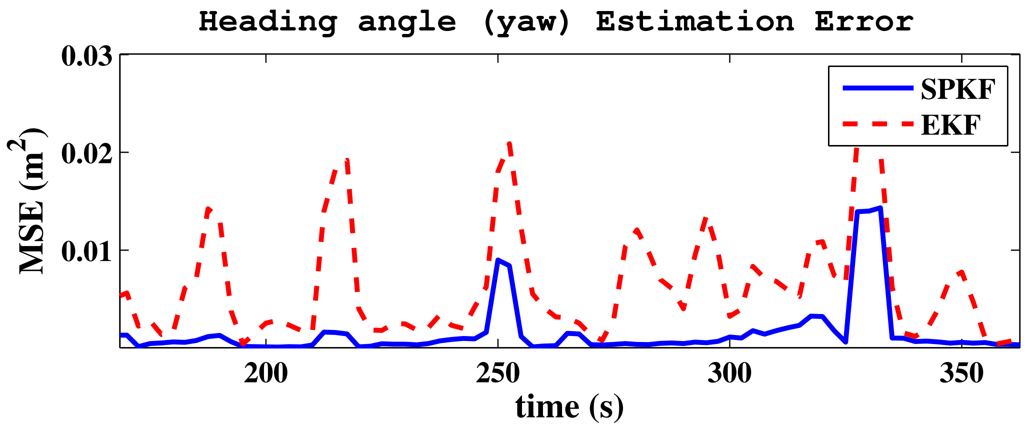
We have used a standard SPKF to track the vehicle's state and simultaneously learn the parameters of the observation maps using dual Kalman framework. Note that the SPKF time-update and measurement-update steps operate at different rates. The time-update step operates at every 10 ms, whereas we obtain measurements from the terrain sensors at every 100 ms.

Experimental Results

In this section, we demonstrate the performance of SPKF/EKF based estimators for simultaneously estimating the vehicle's position and the parameters of its surroundings given a set of terrain sensor measurements. The vehicle is driven randomly within a 10 m×10 m space. It is assumed that the vehicle is carrying onboard three terrain sensors, which observe the value of three terrain maps based on the current vehicle location. The true terrain maps from which the sensor traces were obtained are unknown. Hence we propose a MLP/RBF based parametric representation of the observation maps whose parameters



(a)



(b)

Figure 4.25: Tracking performance of dual SPKF in terms of MSE between the true and the estimated, (a): position estimation accuracy, (b) heading angle estimation accuracy.

are initially taken as random and are learned from the sensor measurements to mimic the unknown terrain maps. There is a separate observation map learned for each terrain sensor. Recall that we have employed both three layer *MLP* and *RBF* networks to represent the terrain maps, which require the knowledge of the number of hidden nodes H and Gaussian kernels C respectively. Hidden nodes H and kernel C are decided using a 10 fold cross validation technique. It is found that the optimum number of H and C were 28 and 12 respectively [118].

For the simulation purposes, we have generated a set of epochs each consisting of a random vehicle trajectory. At each epoch, $N = 1000$ terrain observations are collected

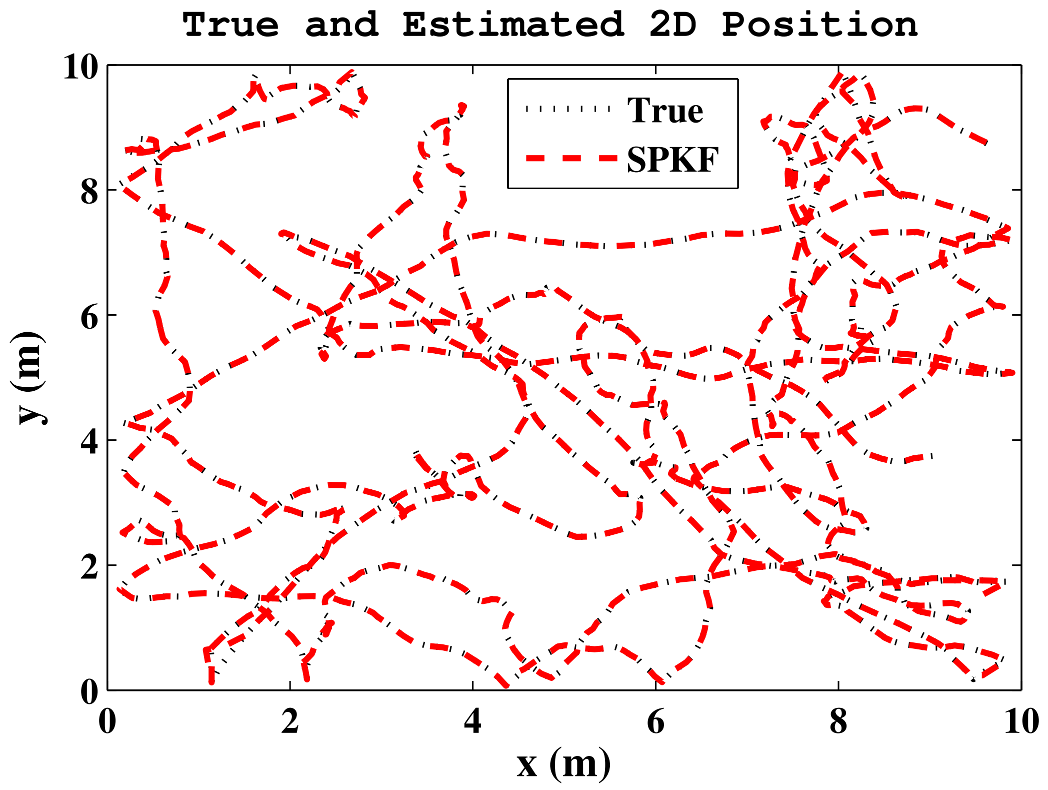


Figure 4.26: SPKF estimated vehicle position after convergence.

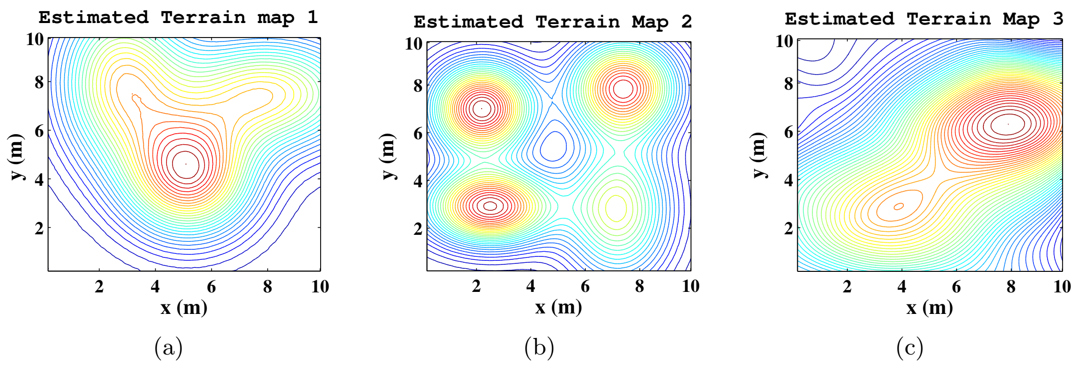


Figure 4.27: (a) to (c): Reconstructed terrain maps after convergence.

from the vehicle's trajectory. A new vehicle trajectory is generated at each epoch from the same bounded region. The estimated state and map parameters from the previous epoch are used to initialize the next epoch. It is observed that it takes about $50 - 75$ epochs for the SPKF to converge to the true state and maps. Figure 4.26 shows the estimated

Table 4.6: Performance comparison of the dual SPKF and the dual EKF after convergence (MSE shown in the table is actually the average of MSE obtained in MLP and RBF).

MSE **after convergence**	**Map 1**	**Map 2**	**Map 3**	**Vehicle Position**
EKF	0.145	0.139	0.204	0.135
SPKF	0.074	0.072	0.118	0.055

trajectory of the SPKF versus the true trajectory after convergence. As is clearly seen, the SPKF estimated trajectory is nearly indistinguishable from the true. Figure 4.25(a) and 4.25(b) demonstrates the accuracy of the SPKF based tracker after convergence in tracking the vehicle position and heading angle on the basis of MSE between the true and the estimated values. Note that neither the initial values of the state nor the maps are assumed known. The MSE between the true and the estimates is computed by a moving average window over time. Each window is 10 s long with a 4 s overlap between successive windows. As is clearly seen from the figures, the SPKF estimator can successfully track the vehicle's position and heading angle after the convergence of the vehicle's state and terrain map parameters. Dual EKF estimates are also plotted in the same figures for performance comparison with SPKF estimates. It is shown that the SPKF consistently outperformed the EKF in terms of vehicle position and yaw angle estimates. Figure 4.27(a) - 4.27(c) display the SPKF reconstructed maps after convergence. Comparing them with the true maps (Figure 4.24(a) - 4.24(c)), we can observe that the estimated maps are almost similar with the true. Reconstructed maps are shown for the RBF cases only, as the RBF result converges faster to the true values and after convergence it demonstrates lower MSE than the MLP. Table 4.6 compares the performance of the SPKF with the EKF in terms of estimation accuracy [118]. As evident from the table, the SPKF provides significant gain over the EKF in final performance.

Chapter 5

New Multiharmonic Frequency Tracking Using Sigma-Point Kalman Smoothers

5.1 Overview

In the previous chapters, we evaluate the performance of the sigma-point Kalman smoothing (SPKS) algorithms in real world indoor tracking. In this chapter, we apply the SPKS into the domain of frequency tracking where the task is to track the individual frequencies and amplitudes of a multiharmonic periodic signal. The state-space approach to tracking time-varying frequencies of multiharmonic periodic/quasi-periodic signals have recently become popular within the research community. Due to nonlinear state dynamics, several groups have proposed an extended Kalman filter/smoother (EKF/EKS) to track multiharmonic frequencies. In this chapter, our intention is to introduce a new multiharmonic frequency tracker based on the proposed SPKS and compare its performance to that of the EKS method. This work has been done in collaboration with Sunghan Kim and James McNames who are associated with the Biomedical Signal Processing (BSP) Laboratory, department of Electrical and Computer Engineering (ECE), Portland State University. They have been working on applications, including frequency tracking for quasi-periodic signals with time-varying amplitudes for many years and we have integrated our SPKS based estimator with their tracking framework.

This chapter is organized as follows. Section 5.2 introduces our approach with a brief description of current frequency tracking methodologies seen in the literature. Section 5.3 discusses the Bayesian framework and details the dynamic and observation models used in our SPKS estimator. Section 5.4 examines the different estimation algorithms, including the SPKS and the EKS, implemented in our tracking system. Experimental results are given in Section 5.5, and finally discussion and conclusions in Section 5.6.

5.2 Introduction

Many natural signals contain nearly periodic rhythms with slowly varying morphologies. Example signals with this property include tremor, speech, electrocardiogram (ECG), and arterial blood pressure (ABP). In many applications the instantaneous frequency (IF) of these signals contains useful information for further analysis.

Many signal processing methods have been applied to the problem of multiharmonic frequency tracking in quasi-periodic signals. For example, the pitch tracking in the speech signal analysis is one of the most common applications of multiharmonic frequency tracking. Pitch detection/tracking algorithms can be roughly categorized into three groups: time-domain methods such as zero-crossing, frequency-domain methods, and time-frequency-domain methods. Due to the nonstationary nature of human voice, the pitch tracking algorithms generally perform within a short window in time/frequency domain [119,120]. Recently Tabrikian *et al.* proposed a new *maximum a-posteriori* (MAP) based statistical approach using a harmonic model for pitch tracking [121]. They implemented the MAP estimator using dynamic programming based on measurements collected over several window frames. However, these *frame-by-frame* based algorithms are always not applicable especially when a local signal stationarity cannot be assumed [16, 122]. There are other methods based on adaptive schemes that have been applied to track rhythmicity (harmonic components) in nonstationary quasi-periodic signals (not necessarily speech signals) [123]. The advantage of using these adaptive schemes is that one can track frequencies *recursively* as signal samples are acquired. For details of these different frequency tracking methods, one can refer to [15].

In order to perform multiharmonic frequency tracking, we represent a time-domain signal using the Fourier series in which the amplitude, phase, and frequency of each harmonic component are allowed to vary slowly over time. We employ the dynamic state space approach in order to track the harmonics present in a given signal. The application of state space methods to continuously track the harmonics was pioneered by Parker *et al.* [122], which triggered many subsequent investigations [124–128]. Recently there have been several proposed methods based on particle filters [129, 130] but unfortunately they are highly computationally intensive and hence practically intractable.

We follow the SPKF [26, 36, 82] based Bayesian inference approach for multiharmonic tracking. Although the SPKF has been applied to a wide range of problems, we are unaware of any literature that applies the SPKF to the multiharmonic frequency tracking problem. In [16], we have successfully adopted the SPKF to track the amplitudes, frequencies and phases of all harmonics present in a quasi-periodic signal and have outlined the superior accuracy of SPKF based approach compared to the EKF. As a smoother delivers better estimates than a filter and our work was focused on an offline analysis, in this chapter we concentrate on our recently proposed fixed-interval sigma-point Kalman smoother (FI-SPKS) algorithm [35] for the tracking purpose. This dissertation provides a more detailed description and analysis of our method that was recently accepted in [131]. The FI-SPKS estimator, which uses the entire recording of signal to generate each state estimate, fuses a dynamic model with discrete signal observations in order to track the multiple harmonics. We have adopted the state dynamics and observation model from Parker's work [122] with slight modifications. The SPKS based multiharmonic frequency tracker is also compared with the traditional EKS approach based on several performance metrics and the performance advantage of the SPKS method is demonstrated.

5.3 Recursive Bayesian Inference Framework

We have used the following state space model,

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) \tag{5.1}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{n}_k\right). \tag{5.2}$$

The different components of the above model including the dynamic model f_k and observation model h_k are described below in terms of multiharmonic frequency estimation context.

5.3.1 State Space Model

We employ the following observation model called a *rectangular model* [122] in our frequency tracking system,

$$\begin{aligned} \boldsymbol{z}_k &= \boldsymbol{s}_k + \boldsymbol{n}_k \\ &= h(\boldsymbol{x}_k) + \boldsymbol{n}_k \\ &= \sum_{p=1}^{m} \boldsymbol{a}_{p,k} \cos\left(p\boldsymbol{\theta}_k\right) + \boldsymbol{b}_{p,k} \sin\left(p\boldsymbol{\theta}_k\right) + \bar{\boldsymbol{z}}_k + \boldsymbol{n}_k, \end{aligned} \tag{5.3}$$

where m is the total number of harmonics present which is assumed to be known, $\boldsymbol{\theta}_k$ is the instantaneous angle, $\boldsymbol{a}_{p,k}$ and $\boldsymbol{b}_{p,k}$ are the amplitudes of the p-th harmonic sinusoidal components, $\bar{\boldsymbol{z}}_k$ is the trend of $\boldsymbol{z}_k$, $\bar{f}$ is the mean frequency, and observation noise $\boldsymbol{n}_k$ is a white noise process with zero-mean and covariance $\boldsymbol{R}_k$. The instantaneous angle $\boldsymbol{\theta}_k$ is modeled as,

$$\begin{aligned} \boldsymbol{\theta}_k &= \sum_{i=1}^{k} 2\pi T_{\mathrm{s}} \boldsymbol{f}_i \\ &= \sum_{i=1}^{k} 2\pi T_{\mathrm{s}} \left(\boldsymbol{\xi}_i + \bar{f}\right) \\ &= \sum_{i=1}^{k} 2\pi T_{\mathrm{s}} \boldsymbol{\xi}_i + \sum_{i=1}^{k} 2\pi T_{\mathrm{s}} \bar{f} \\ &= \boldsymbol{\varphi}_k + 2\pi T_{\mathrm{s}} k \bar{f}, \end{aligned} \tag{5.4}$$

where $\bar{f}$ is the mean frequency, $\boldsymbol{\xi}_k$ is the difference between the instantaneous frequency $\boldsymbol{f}_k$ and the mean frequency $\bar{f}$, $\boldsymbol{\varphi}_k$ is the accumulative sum of $\boldsymbol{\xi}_k$ and T_{s} is the sampling interval. The definition of instantaneous angle $\boldsymbol{\theta}_k$ shown in Equation (5.4) is one of the major differences between our state-space model and the one proposed in [122]. This modification was obtained because the SPKS favors the state variables to have zero mean. Since $\boldsymbol{\varphi}_k$ is the accumulative sum of $\boldsymbol{\xi}_k = \boldsymbol{f}_k - \bar{f}$, its mean is zero. The zero mean state process increases the numerical stability for the SPKS estimator.

The dynamic model $f(.)$, which characterizes the propagation of each state-space variable from time k to time $k+1$, is formed as follows,

$$\boldsymbol{\varphi}_{k+1} = \boldsymbol{\varphi}_k + 2\pi T_{\mathrm{s}}\boldsymbol{\gamma}_k + \boldsymbol{v}_{\varphi,k} \tag{5.5}$$

$$\boldsymbol{\gamma}_{k+1} = \alpha\boldsymbol{\gamma}_k + (1-\alpha)\boldsymbol{v}_{\gamma,k} \tag{5.6}$$

$$\boldsymbol{a}_{p,k+1} = \boldsymbol{a}_{p,k} + \boldsymbol{v}_{a,k} \tag{5.7}$$

$$\boldsymbol{b}_{p,k+1} = \boldsymbol{b}_{p,k} + \boldsymbol{v}_{b,k} \tag{5.8}$$

$$\bar{\boldsymbol{z}}_{k+1} = \bar{\boldsymbol{z}}_k + \boldsymbol{v}_{\bar{z},k} \tag{5.9}$$

where $\boldsymbol{\gamma}_k$ is the fluctuating component in $\boldsymbol{\varphi}_k$, α is an autoregressive (AR) coefficient of $\boldsymbol{\gamma}_k$, and $\boldsymbol{v}_{\cdot,k}$ are mutually uncorrelated white noise processes. A value of $\alpha = 1$ results in a random walk model of $\boldsymbol{\varphi}_k$ and $\alpha = 0$ results in a white noise model. The variance $\boldsymbol{Q}_k$ of process noise $\boldsymbol{v}_{\cdot,k}$ determines how quickly the parameters are expected to change over time.

The state vector $\boldsymbol{x}_k$ is shown as,

$$\boldsymbol{x}_k = \begin{bmatrix}\boldsymbol{\varphi}_k \ \boldsymbol{\gamma}_k \ \boldsymbol{a}_{1,k} \ \ldots \ \boldsymbol{a}_{m,k} \ \boldsymbol{b}_{1,k} \ \ldots \ \boldsymbol{b}_{m,k} \ \bar{\boldsymbol{z}}_k\end{bmatrix}^T, \tag{5.10}$$

Now combining the dynamic and observation models, we can write the overall state-space framework for our proposed frequency tracker as follows,

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k) + \boldsymbol{v}_k \tag{5.11}$$

$$= \begin{bmatrix}\boldsymbol{x}_{1,k} + 2\pi T_{\mathrm{s}}\boldsymbol{x}_{2,k} \\ \alpha\boldsymbol{x}_{2,k} \\ \boldsymbol{x}_{3,k} \\ \vdots \\ \boldsymbol{x}_{2m+2,k} \\ \boldsymbol{x}_{2m+3,k}\end{bmatrix} + \begin{bmatrix}\boldsymbol{v}_{1,k} \\ \boldsymbol{v}_{2,k} \\ \boldsymbol{v}_{3,k} \\ \vdots \\ \boldsymbol{v}_{2m+2,k} \\ \boldsymbol{v}_{2m+3,k}\end{bmatrix} \tag{5.12}$$

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k) + \boldsymbol{n}_k \tag{5.13}$$

$$= \sum_{p=1}^{m} \boldsymbol{a}_{p,k}\cos(p\boldsymbol{\theta}_k) + \boldsymbol{b}_{p,k}\sin(p\boldsymbol{\theta}_k) + \bar{\boldsymbol{z}}_k + \boldsymbol{n}_k. \tag{5.14}$$

Note that the state transition function $f(\cdot)$ is linear and the observation function $h(\cdot)$ is nonlinear in the above state space model.

5.4 Multiharmonic Frequency Tracker

In this section we use the FI-SPKS and EKS based estimators for multiharmonic frequency tracking. We perform an offline analysis, in which we use the entire signal recording of whole duration in order to compute the smoothed state at each time k. We prefer the RTSSL-SPKS for this task due to its ease of implementation, low computational complexity and numerical efficiency (for details, refer to Chapter 2).

In order to demonstrate the performance advantage of the proposed SPKS based frequency tracker, we compare the performance of the SPKS with that of the EKS in terms of estimation accuracy. In the next section, we will summarize the EKS estimation algorithm. Due to the presence of several variants of the EKS, we will demonstrate below the specific version that is used for performance comparison with the SPKS. The following EKS estimator has previously been used for frequency tracking by Kim *et al.* [15].

5.4.1 EKS frequency tracker Recursions

The pseudo code for the EKS is shown below:

Forward Updates

The filtered and predicted state estimates can be computed directly from the well-known EKF recursions, which is shown as below:

- Initialization:

$$\hat{\boldsymbol{x}}_{0|0} = [0 \;\; 0 \;\; 0.1 \;\; \ldots \;\; 0.1 \;\; 0]^{\mathrm{T}}$$
$$\boldsymbol{P}_{0|0} = \mathrm{diag}(0.1 \;\; \ldots \;\; 0.1)$$

- Time-update equations:

$$F_k = \left.\frac{\partial f_k(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_k}$$

$$F_k = \begin{bmatrix} 1 & 2\pi T_s s'\left[\hat{\boldsymbol{x}}_{2,k}\right] & 0 & 0 & \dots & 0 \\ 0 & \alpha s'\left[\hat{\boldsymbol{x}}_{2,k}\right] & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

$$\boldsymbol{P}^-_{k+1} = F_k \boldsymbol{P}_k F_k^{\mathrm{T}} + \boldsymbol{Q}_k$$

$$\hat{\boldsymbol{x}}^-_{k+1} = f(\hat{\boldsymbol{x}}_k)$$

- Measurement-update equations:

$$H_k = \left.\frac{\partial h_k(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}^-_{k+1}}$$

$$H_k = \begin{bmatrix} \sum_{p=1}^{m} \hat{\boldsymbol{a}}^-_{p,k+1} k \sin(k\boldsymbol{\theta}) + \hat{\boldsymbol{b}}^-_{p,k+1} k \cos(k\boldsymbol{\theta}) \\ 0 \\ \cos(\boldsymbol{\theta}) \\ \vdots \\ \cos(m\boldsymbol{\theta}) \\ \sin(\boldsymbol{\theta}) \\ \vdots \\ \sin(m\boldsymbol{\theta}) \\ 1 \end{bmatrix}^{\mathrm{T}}$$

$$\boldsymbol{R}_{\mathrm{e},k} = \boldsymbol{R}_k + H_k \boldsymbol{P}^-_{k+1} H_k^{\mathrm{T}}$$

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}^-_{k+1} H_k^{\mathrm{T}} \boldsymbol{R}^{-1}_{\mathrm{e},k}$$

$$\hat{\boldsymbol{z}}^-_{k+1} = h(\hat{\boldsymbol{x}}^-_{k+1})$$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}^-_{k+1} + \boldsymbol{K}_{k+1}\left(\boldsymbol{z}_{k+1} - \hat{\boldsymbol{z}}^-_{k+1}\right)$$

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}^-_{k+1} - \boldsymbol{K}_{k+1} \boldsymbol{R}_{\mathrm{e},k} \boldsymbol{K}^{\mathrm{T}}_{k+1}$$

The above recursions produce both the filtered estimates $\hat{\boldsymbol{x}}_k$, the predicted estimates $\hat{\boldsymbol{x}}_{k+1}^-$ and their associated estimation error covariances at each time k, which will be required for smoothing.

Smoothing

There are many mathematically equivalent expressions for the EKS. Here we use a variant similar to that developed in [33] (see [34, p. 374]). The backward recursive update equations for the EKS start with an initialization at time N such as,

$$\boldsymbol{\psi}_N^- = 0,$$

where $\boldsymbol{\psi}$ is called the adjoint variable. The smoothed estimates can then be computed by the following recursive operations from $k = N$ to $k = 1$,

- Backward-smoothing equations:

$$\begin{aligned}
\boldsymbol{K}_{\mathrm{b},k} &= (F_k \boldsymbol{P}_k^- H_k^{\mathrm{T}})\boldsymbol{R}_{\mathrm{e},k}^{-1} \\
\boldsymbol{\psi}_k &= (F_k - \boldsymbol{K}_{\mathrm{b},k} H_k)^{\mathrm{T}} \boldsymbol{\psi}_{k+1} + H_k^{\mathrm{T}} \boldsymbol{R}_{\mathrm{e},k}^{-1} \boldsymbol{e}_k \\
\hat{\boldsymbol{x}}_k^{\mathrm{s}} &= \hat{\boldsymbol{x}}_k^- + \boldsymbol{P}_k^- \boldsymbol{\psi}_k.
\end{aligned}$$

5.5 Experimental Results

The performance of the SPKS and EKS based frequency tracker have been demonstrated on two sets of simulated signals and a photo-sensor insect activity signal. We will start with defining the input signals, parameter and performance-metric selection for the estimators before moving on to the performance results.

5.5.1 Simulated Time-Variant Harmonic Signals

We employ equations (5.3)-(5.9) to generate two sets of simulated signals with time-varying harmonics using a sampling rate of $f_\mathrm{s} = 2\,\mathrm{kHz}$. The mean frequency and the signal duration are kept as $\bar{f} = 100\,\mathrm{Hz}$ and $T_d = 3\,\mathrm{s}$, respectively. The first set of simulated signal contains the rhythmicity during the entire $3\,\mathrm{s}$ duration. The second set of simulated signal contains the rhythmicity only during the first and last one second, $0 - 1\,\mathrm{s}$ and $2-3\,\mathrm{s}$. Between $1\,\mathrm{s}$ and $2\,\mathrm{s}$, a white Gaussian noise replaces the rhythmicity. The absence of rhythmicity for a specific period of time mimics certain real world cases when the rhythmicity is intermittent.

5.5.2 Photo-Sensor Insect Activity Signal

We applied both the SPKS and EKS trackers to a photo-sensor insect activity signal. The photo-sensor insect activity signal has a clear harmonic structure, which carries important entomological information. The instantaneous frequency and the harmonic amplitudes help entomologists determine what kind of insects flew over the photo-sensor [132, 133]. The sampling frequency of the photo-sensor insect activity signal was $16\,\mathrm{kHz}$ and the signal duration was $10\,\mathrm{s}$.

5.5.3 Parameter Selection

Table 5.1 lists the user-specified parameters needed to generate the examples and the estimators. The parameter values, except for the SPKS case, are mainly adopted from the work of Kim *et al.* who have successfully designed a EKS multiharmonic frequency tracker using the same parameters [15]. Note that the chosen parameters were tuned in accordance with the best performance of the EKS tracker [15]. While designing the proposed SPKS multiharmonic tracker, we did not perform any additional tuning. Therefore, any bias incurred during the selection of the user-specified parameters would favor the EKS tracker.

5.5.4 Performance Metric Criteria

There are two major criterions that we considered when comparing the performance of frequency trackers: *accuracy* and "*lock-on time*" [15,16,131]. The accuracy quantifies how closely the tracker estimates the state. In many cases, the primary objective of frequency tracking is to estimate the fundamental frequency and all of its harmonics as accurately as possible from a noisy input signal. The lock-on time is a measure of how quickly the tracker can converge to the true state. This is particularly important especially when the rhythmicity in a given signal is intermittent. Hence a frequency tracker need to be

Table 5.1: Summary of user-specified design parameters

Name	**Symbol**	**Value**
AR Coefficient	α	0.9987
Phase process noise variance	q_θ	$10^{-5}T_s$
Frequency process noise variance	q_f	100 T_s
Amplitude process noise variance	q_s	0.0002 T_s
Average process noise variance	$q_{\bar{y}}$	0.001 T_s
Measurement noise variance	$\boldsymbol{R}_k$	1
Mean frequency	$\bar{f}$	100 Hz
Sigma-Point spread	α	0.85
Sigma-Point weighting term	β	2
Sigma-Point parameter	κ	0

where T_s is the sample interval.

robust so that it can regain the ability to track the intermittent frequencies as quickly as possible [129].

We used three metrics to compare the accuracy and speed of convergence for the SPKS and EKS multiharmonic frequency trackers. The performance metrics, which will be described next are designed by Sunghan Kim and James McNames for this application.

The first metric is the *normalized mean-square-error* (NMSE),

$$\text{NMSE} = \frac{\sum_{k=1}^{N}\left(\boldsymbol{s}_k - \hat{\boldsymbol{s}}_k\right)^2}{\sum_{k=1}^{N}\left(\boldsymbol{s}_k - \bar{\boldsymbol{s}}\right)^2}, \tag{5.15}$$

where N is the signal duration, $\hat{\boldsymbol{s}}_k$ is the estimate of the clean signal $\boldsymbol{s}_k$ and $\bar{\boldsymbol{s}}$ is the mean of the signal $\boldsymbol{s}_k$. Lower NMSE indicates that the signal estimates are closer to the true.

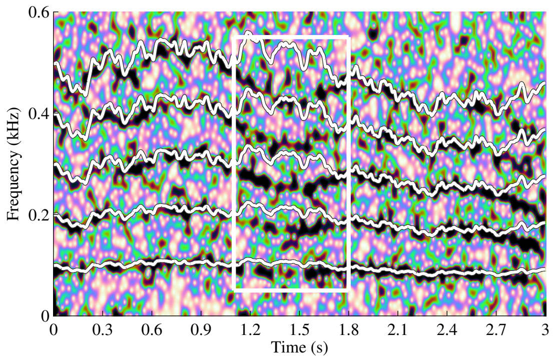
The second metric is *normalized frequency mean-square-error* (NFMSE),

$$\text{NFMSE} = \frac{\sum_{k=1}^{N}(\boldsymbol{f}_k - \hat{\boldsymbol{f}}_k)^2}{\sum_{k=1}^{N}(\boldsymbol{f}_k - \bar{f})^2}, \tag{5.16}$$

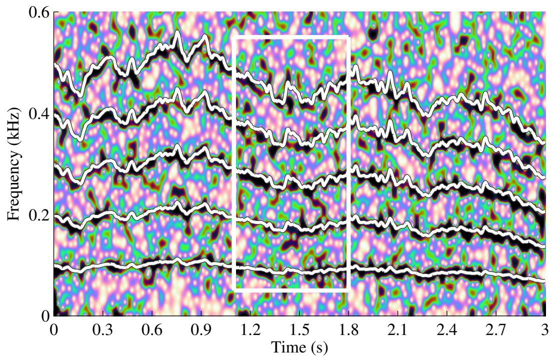
where $\boldsymbol{f}_k$ is the instantaneous frequency (IF), $\hat{\boldsymbol{f}}_k$ is the estimated IF, and $\bar{f}$ is the mean IF. NFMSE has a natural scale ranging from 0 to 1. A value NFMSE $= 1$ means that the average accuracy of the estimated IF is no better than simply using the mean IF as an estimate. NFMSE > 1 indicates poorer frequency tracking than a simple mean estimator and a value of NFMSE $\ll 1$ indicates accurate frequency tracking.

The third metric is *square-frequency-error* (SFE(k)), which can be written as,

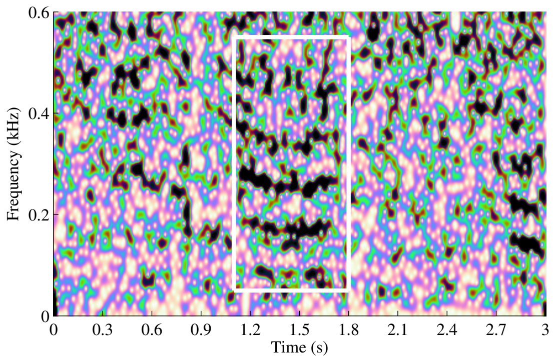
$$\text{SFE}(k) = \left(\boldsymbol{f}_k - \hat{\boldsymbol{f}}_k\right)^2. \tag{5.17}$$



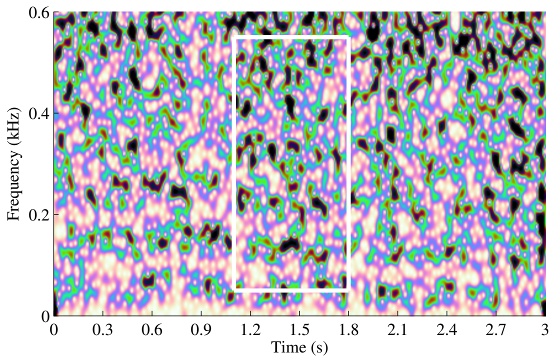
(a) EKS



(b) SPKS



(c) EKS error residual



(d) SPKS error residual

Figure 5.1: Tracking performance on a simulated signal, (a) Signal spectrogram with the estimated harmonics (white lines). The EKS frequency tracker is used to generate the frequency estimates. (b) Signal spectrogram with the estimated harmonics (white lines). The estimates are generated using the SPKS tracker. (c) Residual estimation Error for the EKS (NMSE = 8.56). (d) Residual estimation error for the SPKS (NMSE = 7.53). The black stripes along the spectrograms are the true harmonics and indicate the presence of significant amount of power at the corresponding frequencies. The white rectangular box highlights the performance improvement of SPKS over the EKS tracker. As can be seen for the SPKS case, the estimated harmonics always closely follow the true (Figure (b)). However, inside the rectangular box the EKS estimates diverge from the true (Figure (a)). The failure of the EKS to track the appropriate frequencies is responsible for the presence of harmonic structures in the EKS residual plot (Figure (c)). However, the SPKS residual plot (Figure (d)) does not display any left-over harmonic structures.

When this metric is averaged over an ensemble of simulated signals, it visualizes how rapidly the trackers lock on to the true frequency. In contrast to the NMSE and NFMSE, SFE(k) is a function of time that shows the squared difference between the true IF and its estimate at a given time. For all of our results we calculated the NFMSE, NMSE, SFE(k)

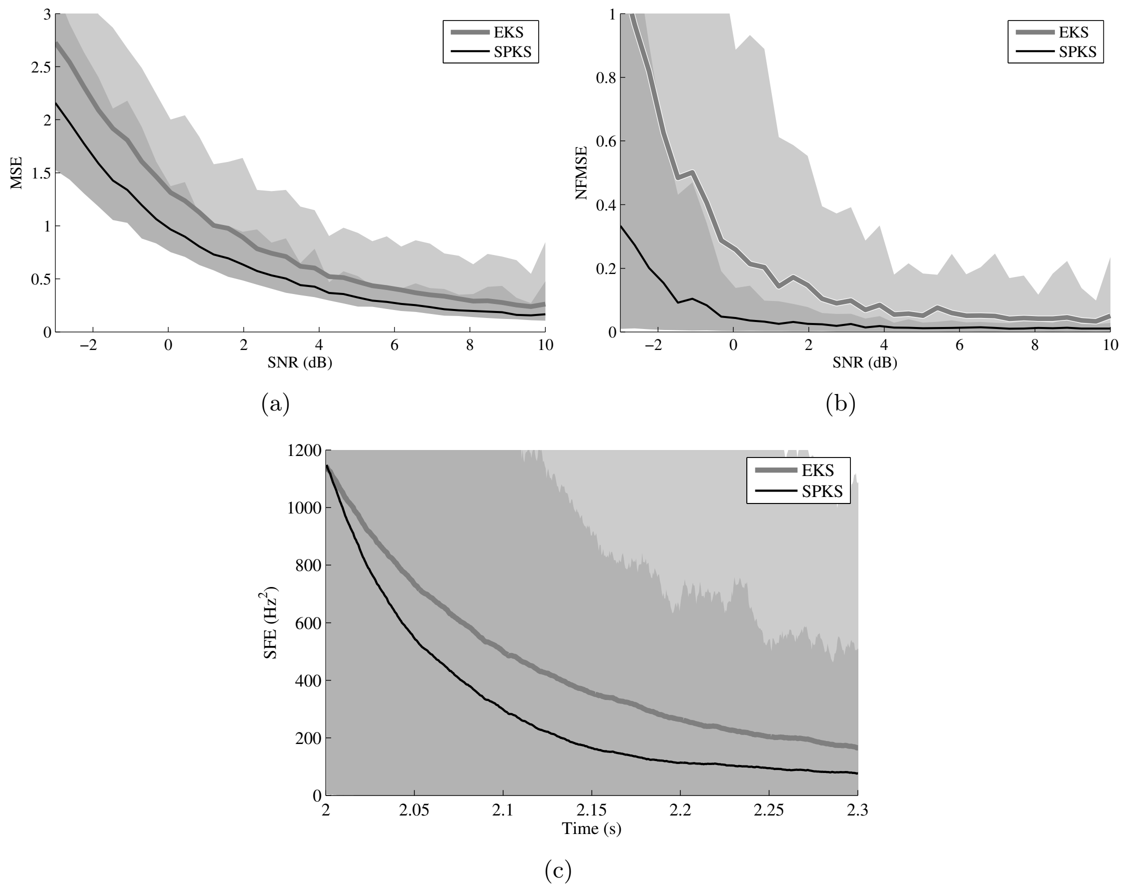


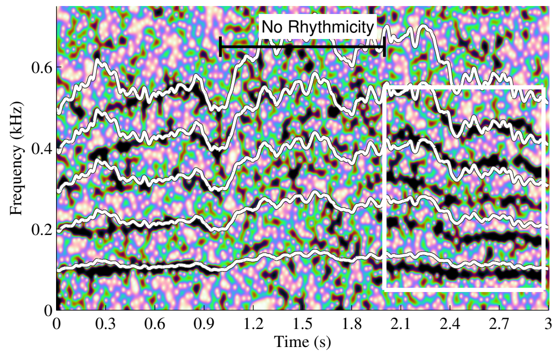
Figure 5.2: Performance comparison between the SPKS and EKS in terms of NMSE, NFMSE and SFE(k), (a) NMSE versus SNR, (b) NFMSE versus SNR, (c) SFE(k) versus time (s) at SNR $= -3\,$dB. This plot demonstrates that how quickly the output of a frequency tracker can converge to the true state after a period when no rhythmicity is present. The shaded regions displayed in the plots represent the 5th and 95th percentile ranges of the NFMSE, NMSE and SFE(k) respectively.

over an ensemble of 300 simulated signals.

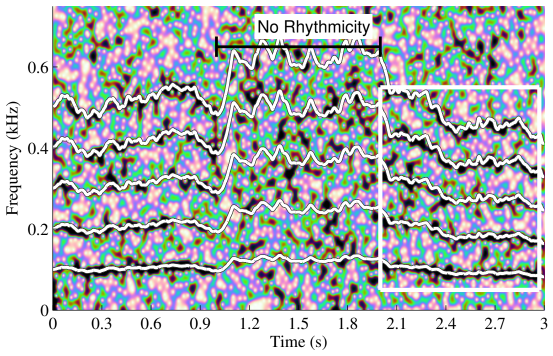
5.5.5 Results

Simulated signals

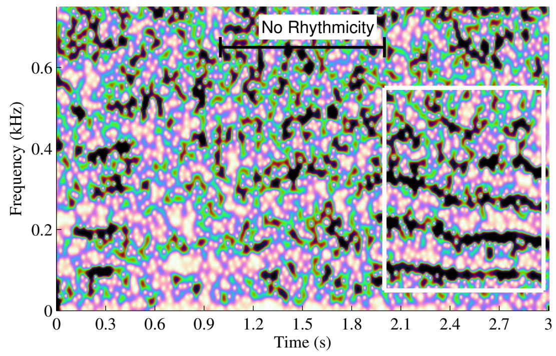
Figure 5.1 shows the estimated multiharmonic frequencies using the EKS (a) and SPKS (b) trackers on top of the spectrogram of a simulated signal. Note that the SNR of the



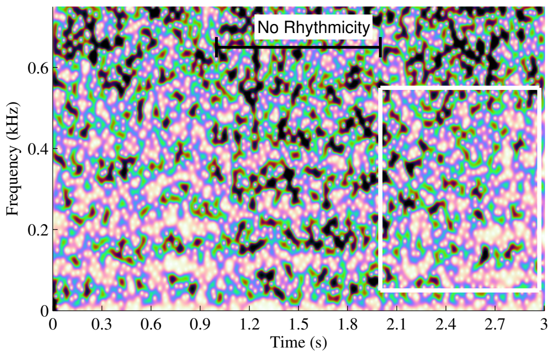
(a) EKS



(b) SPKS



(c) EKS error residual



(d) SPKS error residual

Figure 5.3: Performance comparison between the SPKS and EKS in tracking an intermittent rhythmic signal, (a) EKS estimated frequencies on top of an intermittent rhythmic signal at SNR $= -3\,\text{dB}$, (b) SPKS estimated frequencies on top of an intermittent rhythmic signal at SNR $= -3\,\text{dB}$, (c) Residual signal between the true and estimated using the EKS tracker, (d) Residual signal between the true and estimated using the SPKS tracker. Note that the rhythmicity is present only between $0-1\,\text{s}$ and $2-3\,\text{s}$. The simulated signal does not contain any harmonic structure between $1-2\,\text{s}$. With the help of these plots we evaluate how fast the SPKS/EKS frequency tracker can start tracking the harmonic components after $t = 2\,\text{s}$. The presence of harmonic structures in the EKS residual plot (see Figure (c)) indicates EKS's failure to lock on the true frequencies. As is evident from Figure (b) and (d), the SPKS estimates converge to the true harmonic frequencies at $t = 2.1\,\text{s}$ and hence it does not generate any residual harmonic patterns.

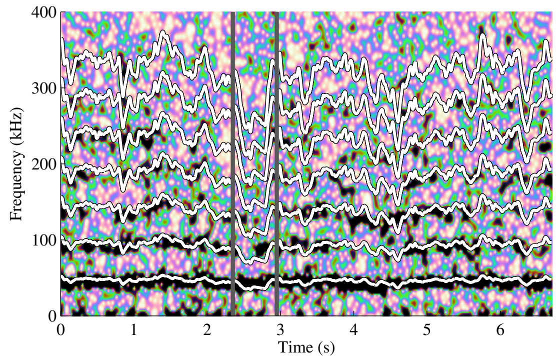
simulated signal was set to be at $-3\,\text{dB}$. As can be seen from the plots, the EKS temporarily lost its way in tracking the correct harmonics at the middle of the spectrogram between $t = 1.1\,\text{s}$ and $t = 1.9\,\text{s}$ and also toward the end, after $t = 2.7\,\text{s}$. The estimation error of EKS is responsible in producing the residual harmonic structures as shown in Figure 5.1(c). In contrast, the estimated frequencies obtained from the SPKS never diverged from the true during the entire signal duration and hence its residual error plot (Figure 5.1(d)) does not show any harmonic structures.

Plot (a) in Figure 5.2 demonstrates the NMSE versus SNR for the SPKS and EKS trackers. It shows that the SPKS tracker can track the true signal better than the EKS tracker over a wide range of SNR. Plot (b) in Figure 5.2 depicts the NFMSE versus SNR of the two multiharmonic trackers. Similar to the previous case, the SPKS clearly outperforms the EKS over the entire range of SNR. Note that the performance difference is larger at low SNR values. In fact at SNR $= -2\,\text{dB}$, the corresponding NFMSE ≈ 1 signifies that the EKS estimated IF is no better than simply using a mean IF as an estimate. However, the NFMSE for the SPKS at SNR $= -2\,\text{dB}$ is only 0.2, which proves that the SPKS is more robust to noise than the EKS. Plot (c) in Figure 5.2 depicts the SFE(k) of the SPKS and EKS multiharmonic trackers. The objective of this figure is to demonstrate how quickly the estimators can converge to the true harmonics after a period when no rhythmicity is present. It depicts that the SPKS tracker can regain its track of the true IF faster than the EKS tracker.

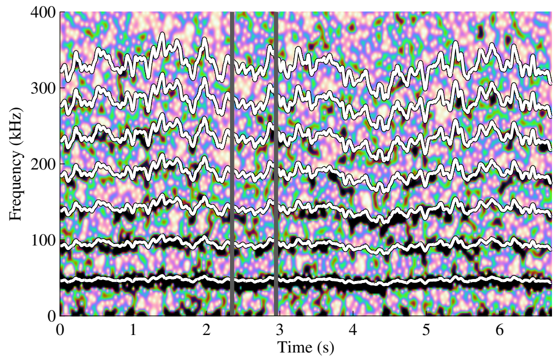
Plots in Figure 5.3 show the estimated harmonic frequencies using the EKS and SPKS trackers on top of the spectrogram of a simulated signal. Note that the rhythmicity is present only between 0-1 s and 2-3 s. After the rhythmic structure came back at $t = 2\,\text{s}$, the SPKS converged and started tracking the true harmonic frequencies accurately from $t = 2.4\,\text{s}$ (see Figure 5.3(a)). In contrast the EKS based frequency tracker completely failed to regain its track of the true IF as seen in Figure 5.3(b)). Figure 5.3(c) and 5.3(d) demonstrate the spectrograms of estimation residuals using the EKS and SPKS respectively. As visualized from the error plots, the EKS just barely started tracking the true harmonic frequencies at the very end of the signal. After the rhythmicity came back at $t = 2\,\text{s}$, it took only $0.4\,\text{s}$ for the SPKS to start tracking the true frequencies.

Photo sensor insect activity signal

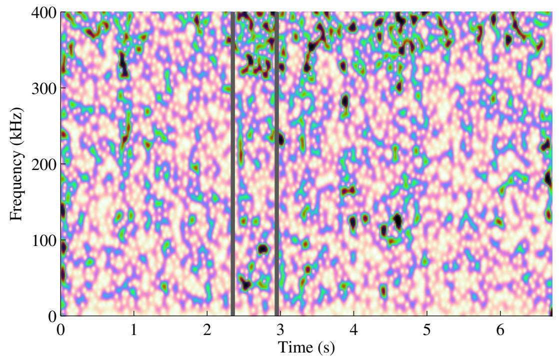
Plots (a) and (b) in Figure 5.4 show the estimated harmonics using the EKS (a) and SPKS (b) multiharmonic trackers on top of the spectrogram of a photo-sensor insect activity signal. Plots (c) and (d) in Figure 5.4 are the spectrograms of estimation residuals using the



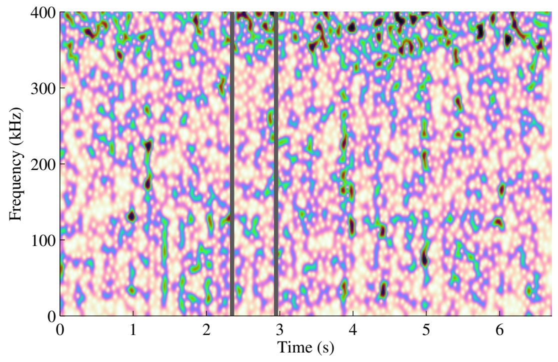
(a) EKS



(b) SPKS



(c) EKS residuals (NMSE = 0.104)



(d) SPKS residuals (NMSE = 0.038)

Figure 5.4: Performance comparison between the EKS and SPKS tracker in tracking the fundamental and harmonic frequencies of a photo sensor insect activity signal, (a) EKS estimated frequency on top of the signal spectrogram, (b) SPKS estimated frequency on top of the signal spectrogram, (c) Residual spectrogram between the true and EKS estimates, (d) Residual spectrogram between the true and SPKS estimates.

EKS (c) and SPKS (d), respectively. The NMSE between the true and reconstructed bug signal using the SPKS estimated harmonics is 0.038 while that using the EKS estimates is 0.104. As seen from the figures, the SPKS estimates closely matched with the true harmonics for the entire signal duration while the EKS tracker lost its track between 2.3 s and 2.9 s. The region where the EKS tracker diverged from the true is marked with two dark grey bars. Although the resulting error between the true and EKS estimates for the fundamental frequency is not significant, the slight error in the IF results in complete mismatch of higher harmonic frequencies. This observation is verified from the residual plot of EKS (Figure 5.4(c)), which demonstrates higher estimation errors at upper harmonic frequencies than at the fundamental frequency level.

5.6 Discussion

In this chapter we covered in detail how a SPKS based new multiharmonic tracker for tracking the amplitude, phase and frequency of each harmonic component was implemented. Recently, the EKF/EKS based state-space method has become popular to accomplish the task of frequency tracking. In this work we have demonstrated that the SPKF/SPKS is in fact a better and viable alternative than the industry standard EKF/EKS for tracking fundamental and harmonic components of a periodic signal. The dynamic and observation models used in the frequency tracking framework were taken from the work of Parker *et al.* [122] and modified according to our needs. We made a head-to-head performance comparison between the SPKS and EKS multiharmonic trackers based on a set of simulated signals and a photo sensor insect activity signal. Using three difference performance metrics, including NMSE, NFMSE and SFE, we have demonstrated that the SPKS multiharmonic tracker is significantly more accurate, converges faster to the true solution, and robust to noise than the EKS multiharmonic tracker.

Chapter 6

Error Bounds for Discrete Time Sigma-Point Kalman Filter

6.1 Overview

The SPKF based Bayesian inference algorithm has been demonstrated to perform with a superior accuracy in estimating the unknown state of the nonlinear system. Julier has demonstrated that the weighted set of sigma points that accurately capture the relevant prior mean and covariance of a random variable (RV), calculates the posterior mean and covariance correctly at least to the second order [23]. Recall that the prior sigma points are propagated over the true nonlinear dynamics to generate the posterior sigma points, which are then weighted averaged to compute the posterior mean and covariance. Julier analyzed the accuracy of the posterior sigma points in moment calculation by comparing with the Taylor series expansion around the posterior RV and verifying that the first and second order terms matched exactly. However, no specific form of the error bounds that capture the difference between the true and estimated states are derived for the SPKF. In this chapter, our focus is to derive the recursive error bounds on the mean-square error (MSE) performance of the SPKF based estimator in the context of a nonlinear dynamical system.

This chapter is organized as follows. Section 6.2 discusses about the importance of developing bounds on the performance of nonlinear estimation algorithms and demonstrates the various techniques that leads to our derivation of the performance bounds for the SPKF. Section 6.3 and 6.4 contain the derivations for computing a lower and a upper error bounds for the SPKF. Experimental results are given in Section 6.5, and finally discussion and conclusions are presented in Section 6.6.

6.2 Introduction

Recursive Bayesian estimation is a general probabilistic approach for sequentially estimating an unknown state probability density function over time using incoming noisy measurements and a known process model. Recall that if the system is linear and the state and noise densities are all Gaussian, the Kalman filter is optimal and provides for an efficient and practical solution. For the nonlinear dynamical systems, the EKF is the most popular and widely used suboptimal filtering technique for state estimation. The SPKF, which was recently introduced has been shown to outperform the EKF in a consistent manner for a number of applications [17, 26, 27, 116, 134]. Although the error behavior of the EKF and its variants have been analyzed in a rigorous mathematical way [135–140], we are not aware of any literature that provides an in-depth analysis of the estimation error bounds for the SPKF based algorithms.

Van Trees *et al.* discusses about the importance and the issues involved in attaining the bounds on the performance of nonlinear estimation algorithms [139]. As the nonlinear estimators adopt suboptimal filtering technique to generate the state estimates, the lower and upper error bounds measure the level of confidence associated with each estimate. The SPKF approximates the prior state density with a number of sigma points. The posterior state statistics, calculated at each SPKF iteration are approximated from the sigma points propagated over the nonlinear dynamics. Hence, like other nonlinear estimation algorithms, analytical error bounds in this case provide important information about the filter performance. The lower and upper bounds provide an effective analytical tool to measure the performance limitation of the filter and correspondingly help to manage expectations of the end-user before performing any data simulation.

A commonly used lower bound for estimating nonrandom parameters in the time-invariant system is equal to the famous Cramér-Rao bound (CRB), given by the inverse of the Fisher information matrix [141, 142]. Similar to the original CRB, a Bayesian version of the CRB (BCRB) is developed by Van Trees *et al.* for estimating random parameters with a prior distribution [143–145]. The BCRB (also termed as *"posterior CRB"*) is proved to be equal to the inverse of the Bayesian information matrix (BIM), which takes into account both the contribution of the data and the prior information. Developing a BCRB for the *time-variant dynamical systems* has attracted a lot of attention in the research community and as a result several variants of the BCRB have appeared in the literature [140, 146–151]. Kerr *et al.* presents an overview of the various formulations that are used to develop the BCRB for the state dynamical models [152].

Although contributions have been made in deriving the recursive BCRB for both the continuous and the discrete time systems, the continuous time dynamics seems to have received the most attention [143, 147, 148, 153]. In this respect, much of the effort was concentrated in deriving a continuous time differential equation for the BIM. Van Trees was the first to apply the Cramér-Rao theory to continuous time systems [143]. Later, the bound was improved by Snyder *et al.* under the assumption that the underlying process model is linear [153]. Bobrovsky and Zakai generalized the continuous-time BCRB so that it could be applied to the nonlinear process and observation models [146]. Their initial paper derived the BCRB for the one-dimensional state estimation case [146, 147] but later they extended their derivation to include the multidimensional states [147, 148]. Bobrovsky and Zakai were also the pioneers to derive a lower bound formulation for the discrete time dynamical systems [146]. Galdos *et al.* generalized the Bobrovsky-Zakai version of the BCRB and applied it successfully to the nonlinear multidimensional discrete time systems [149, 150]. Although their bounds are more general than the previous bounds in the literature, still they have certain limitations. The most significant one is the assumption that both the system state and the measurement vector have equal dimensions, which is not necessarily true for all cases. Tichavský *et al.* provided an efficient implementation of a recursive BCRB on the state error covariance, which is obtained from first principles and is more general than the Bobrovsky-Zakai/Galdos bound [140]. In addition, the derivation does not apply any constraints on the state dynamics or on the state/observation dimensions. The derived BCRB demonstrates a lower bound on the estimation error covariance of the nonlinear estimator. Using Tichavský's derivation, Van

Trees demonstrated a lower error bound for the EKF [139].

Similar to the filter lower bound, the convergence and stability properties of the filter also need to be addressed in order to fully quantify the estimation error. Hence efforts have been made to derive a filter upper bound for stochastic dynamical systems [135–138, 154]. Although a literature survey points to an extensive number of research papers that already exist on this topic, below we will only discuss those papers that we have found useful for our derivation. The concept of the estimation-error upper bound came from the control theory. Its derivation is based upon computing a Lyapunov function and analyzing the convergence and stability properties of the stochastic process defined by the Lyapunov system. Whereas in lower bound computation, the BCRB is solely based on the estimation error covariance of the filter, the Lyapunov function based Bayesian upper bound is a function of the state space dynamics, system nose parameters as well as the estimation error covariance matrix. Song *et al.* derived a set of stochastic stability criterions in the MSE sense under the condition that the system model is linear [135]. Independent of Song's work, the convergence and stability properties have also been specifically investigated for the linear parameter estimation framework [138, 155, 156]. Galkowski *et al.* extended the stability criteria to the nonlinear models using a special form of the nonlinear dynamics [136]. Another situation, where the upper error bound was derived includes a deterministic state space model, i.e. state dynamics with zero process and observation noises [154, 157, 158]. Recently, Reif *et al.* demonstrated that the expected estimation error of the EKF is upper bounded in the mean-square sense provided certain constraints can be applied on the system dynamics, noise terms and the estimation error covariance [137]. Reif's derivation is more general than the previous error bounds in the sense that it is based on a standard nonlinear dynamical model with the additive process and observation noises. Recently, Alessandri *et al.* and Kim *et al.* derived the upper stability bound of their variants of the EKF using Reif's formulation [138, 159]. Alessandri *et al.* further extended Reif's work by taking into account the correlation between the estimation error and system noises in his bound calculation.

In this chapter, the main contribution of our work is to derive a lower and upper error bound for the SPKF, which operates on the discrete-time nonlinear dynamical model to estimate the system's state. Both the lower and the upper bound is derived in terms of MSE between the true and estimated state. The derived BCRB for the SPKF follows the work of Tichavský and co-authors [140] as this is the most straightforward and generalized lower error bound we have found in the literature. We further demonstrate that the estimation error of the SPKF is exponentially bounded in the MSE sense by an upper bound, if the prior conditions on the estimation error covariance, state dynamics and system noise terms hold true. The derivation of the SPKF upper bound is based on the work of Reif *et al.* [137]. After a thorough literature review, we decided to follow Reif's work as we found his derivation to be the most exhaustive, theoretically sound and easy to understand. Note that the final equations of the lower and upper error bounds for the SPKF are derived analytically from first principles. However, there are some approximations needed to implement the bounds in practical examples. In the lower bound calculation, we have approximated the multidimensional expectations required to generate a closed form solution by sample weighted averaging over the extracted sigma points. This is necessary because the calculation of the multidimensional expectations is analytically intractable for the nonlinear system. This will be explained in detail in Section 6.3.3. The upper error bound is derived as a function of parameters corresponding to the lower and upper bound values of the state error covariance, system noise terms, state space parameters and the initial estimation error. However, in practice those bound values are not known in advance, hence we adopt a separate training procedure to estimate them. This will be further detailed in Section 6.4.1. Although these approximations and assumptions might affect the accuracy, the derived bounds provide a quantitative measure of confidence on the SPKF generated state estimates and help to determine whether the SPKF would achieve the expected accuracy in a particular state estimation application.

6.3 Lower Error Bound For The SPKF

Before going details into the derivation of the SPKF lower bound, we provide a short description on the CRB and the BCRB, which analyzes the performance of an estimator in terms of MSE between the true states and the estimates. To describe the lower bound, we have followed the same notations used by Van trees in his book [139].

6.3.1 Background: CRB and BCRB

Consider the process of estimating a M dimensional state vector $\boldsymbol{x}$ using a set of noisy N dimensional observation vector $\boldsymbol{z}$. In the first case, we assume that the state $\boldsymbol{x}$ is an unknown nonrandom vector. If $p(\boldsymbol{z}|\boldsymbol{x})$ denotes the observation likelihood given the state $\boldsymbol{x}$, the Cramér-Rao bound (CRB) ($\boldsymbol{C}(\boldsymbol{x})$) on the estimation error covariance matrix $\boldsymbol{P}\left(\boldsymbol{x},\hat{\boldsymbol{x}}\right)$ can be shown as

$$\boldsymbol{P}\left(\boldsymbol{x},\hat{\boldsymbol{x}}\right) \geq \boldsymbol{J}_F^{-1} \triangleq \boldsymbol{C}(\boldsymbol{x}), \tag{6.1}$$

where $\boldsymbol{P}\left(\boldsymbol{x},\hat{\boldsymbol{x}}\right)$ is the covariance between the estimated $\hat{\boldsymbol{x}}$ and the true state $\boldsymbol{x}$ and can be shown as

$$\boldsymbol{P}\left(\boldsymbol{x},\hat{\boldsymbol{x}}\right) = \mathrm{E}_{\boldsymbol{z}|\boldsymbol{x}}\left[\Big(\boldsymbol{x}(\boldsymbol{z}) - \mathrm{E}_{\boldsymbol{z}|\boldsymbol{x}}\left(\hat{\boldsymbol{x}}(\boldsymbol{z})\right)\Big)\Big(\boldsymbol{x}(\boldsymbol{z}) - \mathrm{E}_{\boldsymbol{z}|\boldsymbol{x}}\left(\hat{\boldsymbol{x}}(\boldsymbol{z})\right)\Big)^T\right]. \tag{6.2}$$

The $M \times M$ Fisher information matrix $\boldsymbol{J}_F$, which uses the likelihood distribution can be defined as

$$\left[\boldsymbol{J}_F(\boldsymbol{x})\right]_{ij} = \mathrm{E}_{\boldsymbol{z}|\boldsymbol{x}}\left[-\frac{\partial^2 \ln p(\boldsymbol{z}|\boldsymbol{x})}{\partial \boldsymbol{x}_i \partial \boldsymbol{x}_j}\right] \quad \text{for } i,j = 1,2,\ldots,M. \tag{6.3}$$

Next, the unknown state $\boldsymbol{x}$ to be estimated is defined as a random vector with a prior probability density $p(\boldsymbol{x})$, which is referred to as Bayesian estimation. Van Trees extended the classical CRB bound for the case of Bayesian estimation, which is known as the Bayesian Cramér-Rao bound (BCRB). The BCRB ($\boldsymbol{C}_B(\boldsymbol{x})$) is defined as

$$\boldsymbol{P}\left(\boldsymbol{x},\hat{\boldsymbol{x}}\right) \geq \boldsymbol{J}_B^{-1} \triangleq \boldsymbol{C}_B(\boldsymbol{x}), \tag{6.4}$$

where $\boldsymbol{P}\left(\boldsymbol{x},\hat{\boldsymbol{x}}\right)$ is the MSE matrix between the true and estimated state,

$$\boldsymbol{P}\left(\boldsymbol{x},\hat{\boldsymbol{x}}\right) = \mathrm{E}_{\boldsymbol{x}\boldsymbol{z}}\left[\left(\hat{\boldsymbol{x}} - \boldsymbol{x}\right)\left(\hat{\boldsymbol{x}} - \boldsymbol{x}\right)^T\right]. \tag{6.5}$$

The $M \times M$ Bayesian information matrix (BIM) $\boldsymbol{J}_B$ is equal to the summation of contribution from the data $\boldsymbol{J}_D$ and the contribution of the prior density $\boldsymbol{J}_P$

$$[\boldsymbol{J}_B]_{ij} = \mathrm{E}_{\boldsymbol{z},\boldsymbol{x}}\left[-\frac{\partial^2 \ln p(\boldsymbol{z},\boldsymbol{x})}{\partial \boldsymbol{x}_i \partial \boldsymbol{x}_j}\right] \quad \text{for } i,j = 1,2,\dots,M \tag{6.6}$$

$$= \mathrm{E}_{\boldsymbol{x}}\left[\boldsymbol{J}_F\right]_{ij} + \mathrm{E}_{\boldsymbol{x}}\left[-\frac{\partial^2 \ln p(\boldsymbol{x})}{\partial \boldsymbol{x}_i \partial \boldsymbol{x}_j}\right] \quad \text{for } i,j = 1,2,\dots,M \tag{6.7}$$

$$\boldsymbol{J}_B = \boldsymbol{J}_D + \boldsymbol{J}_P. \tag{6.8}$$

Note that $\boldsymbol{J}_D$ is the expectation of the Fisher information matrix $\boldsymbol{J}_F$ over the prior distribution $p(\boldsymbol{x})$. The above equation demonstrates that the BCRB depends on the information obtained from the observed data as well as a priori information of the hidden state.

6.3.2 Recursive BCRB For The Nonlinear Filtering Problem

The objective here is to derive a recursive formulation of the BCRB for the time-varying nonlinear dynamical model in order to lower bound the state estimation error covariance $\boldsymbol{P}_k$. For the nonlinear filtering problem, the BCRB can be demonstrated as

$$\boldsymbol{P}_k \geq \boldsymbol{J}_k^{-1}, \tag{6.9}$$

where $\boldsymbol{J}_k$ is the BIM. For notational convenience, we have omitted the subscript "B" from $\boldsymbol{J}_k$. Note in this case, the BIM is a function of discrete time k. Hence the key of the derivation is to find a recursive formulation of $\boldsymbol{J}_{k+1}$ in terms of $\boldsymbol{J}_k$.

In order to obtain the derivation, first recall the nonlinear discrete time dynamic system,

$$\boldsymbol{x}_{k+1} = f_k\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) \tag{6.10}$$

$$\boldsymbol{z}_k = h_k\left(\boldsymbol{x}_k, \boldsymbol{n}_k\right), \tag{6.11}$$

where $\boldsymbol{x}_k \in \mathbb{R}^M$ represents the unobserved state of the system and $\boldsymbol{z}_k \in \mathbb{R}^P$ is the sensor observation at time index k. The system dynamic model $f(.)$ and observation model $h(.)$ are assumed known. The process noise $\boldsymbol{v}_k \sim N\left(0, \boldsymbol{Q}_k\right)$ and observation noise $\boldsymbol{n}_k \sim N\left(0, \boldsymbol{R}_k\right)$ are independent white processes.

Tichavský and co-authors derived an expression for $\boldsymbol{J}_{k+1}$ from first principles, which is based on the previously computed $\boldsymbol{J}_k$, using the above nonlinear dynamical model [140],

$$\boldsymbol{J}_{k+1} = \boldsymbol{D}_k^{22} - \boldsymbol{D}_k^{21}\left(\boldsymbol{J}_k + \boldsymbol{D}_k^{11}\right)^{-1}\boldsymbol{D}_k^{12}, \tag{6.12}$$

where

$$\boldsymbol{D}_k^{11} = \mathrm{E}_{\boldsymbol{x}_k,\boldsymbol{x}_{k+1}}\left[-\Delta_{\boldsymbol{x}_k}^{\boldsymbol{x}_k}\ln p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)\right] \tag{6.13}$$

$$\boldsymbol{D}_k^{12} = \mathrm{E}_{\boldsymbol{x}_k,\boldsymbol{x}_{k+1}}\left[-\Delta_{\boldsymbol{x}_k}^{\boldsymbol{x}_{k+1}}\ln p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)\right] \tag{6.14}$$

$$\boldsymbol{D}_k^{21} = \left[\boldsymbol{D}_k^{12}\right]^T \tag{6.15}$$

$$\boldsymbol{D}_k^{22} = \mathrm{E}_{\boldsymbol{x}_k,\boldsymbol{x}_{k+1}}\left[-\Delta_{\boldsymbol{x}_{k+1}}^{\boldsymbol{x}_{k+1}}\ln p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)\right] + \mathrm{E}_{\boldsymbol{x}_{k+1},\boldsymbol{z}_{k+1}}\left[-\Delta_{\boldsymbol{x}_{k+1}}^{\boldsymbol{x}_{k+1}}\ln p(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1})\right]. \tag{6.16}$$

The gradient operator Δ_Φ^Θ is defined as

$$\Delta_\Phi^\Theta = \nabla_\Phi\left[\nabla_\Theta^T\right] \tag{6.17}$$

$$\nabla_\Theta = \begin{bmatrix} \frac{\partial}{\partial\theta_1} & \frac{\partial}{\partial\theta_2} & \cdots & \frac{\partial}{\partial\theta_M} \end{bmatrix}^T. \tag{6.18}$$

The state transition and observation probability distributions are denoted as $p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)$ and $p(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1})$ respectively. In order to maintain continuity the entire derivation of $\boldsymbol{J}_{k+1}$ is skipped here. An interested reader can refer to Tichavský's paper [140] for further details.

6.3.3 Recursive BCRB For The SPKF

We derive the lower bound of the state estimation error covariance of the SPKF using the Equation (6.12). For this purpose, we make use of statistically linearized form of the standard nonlinear state space, which is repeated here for convenience

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_{f,k}\boldsymbol{x}_k + \boldsymbol{b}_{f,k} + \boldsymbol{G}_{f,k}\left(\boldsymbol{v}_k + \boldsymbol{\epsilon}_{f,k}\right) \tag{6.19}$$

$$\boldsymbol{z}_k = \boldsymbol{A}_{h,k}\boldsymbol{x}_k + \boldsymbol{b}_{h,k} + \boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k}, \tag{6.20}$$

where the linearization coefficients, $\boldsymbol{A}_{f,k}$, $\boldsymbol{A}_{h,k}$, $\boldsymbol{b}_{f,k}$ and $\boldsymbol{b}_{h,k}$, and linearization error terms, $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ and $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ are already defined in Section 1.4.1.

By defining the state transition distribution from state k to $k+1$, $p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)$, and the observation density at time $k+1$, $p(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1})$ in terms of WSLR coefficients, we obtain

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) = \frac{1}{(2\pi)^{\frac{M}{2}} \left|\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right|^{\frac{1}{2}}} \cdot \mathrm{e}^{-\frac{1}{2}\left[\boldsymbol{x}_{k+1}-\left(\boldsymbol{A}_{f,k}\boldsymbol{x}_k+\boldsymbol{b}_{f,k}\right)\right]^T\left(\boldsymbol{Q}_k+\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1}\left[\boldsymbol{x}_{k+1}-\left(\boldsymbol{A}_{f,k}\boldsymbol{x}_k+\boldsymbol{b}_{f,k}\right)\right]} \tag{6.21}$$

$$p(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1}) = \frac{1}{(2\pi)^{\frac{P}{2}} \left|\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right|^{\frac{1}{2}}} \cdot \mathrm{e}^{-\frac{1}{2}\left[\boldsymbol{z}_{k+1}-\left(\boldsymbol{A}_{h,k+1}\boldsymbol{x}_{k+1}+\boldsymbol{b}_{h,k+1}\right)\right]^T\left(\boldsymbol{R}_{k+1}+\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right)^{-1}\left[\boldsymbol{z}_{k+1}-\left(\boldsymbol{A}_{h,k+1}\boldsymbol{x}_{k+1}+\boldsymbol{b}_{h,k+1}\right)\right]}, \tag{6.22}$$

where M is the dimension of the state vector $\boldsymbol{x}_k$ and P is the dimension of the observation vector $\boldsymbol{z}_k$. Note that we represent both of these densities as Gaussian.

Applying natural logarithm (ln) on both sides of those probability densities, we obtain

$$-\ln p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) = \frac{M}{2}\ln(2\pi) + \frac{1}{2}\ln\left|\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right| + \frac{1}{2}\left[\boldsymbol{x}_{k+1} - (\boldsymbol{A}_{f,k}\boldsymbol{x}_k + \boldsymbol{b}_{f,k})\right]^T \left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1} \left[\boldsymbol{x}_{k+1} - (\boldsymbol{A}_{f,k}\boldsymbol{x}_k + \boldsymbol{b}_{f,k})\right] \tag{6.23}$$

$$-\ln p(\boldsymbol{z}_{k+1}|\boldsymbol{x}_{k+1}) = \frac{P}{2}\ln(2\pi) + \frac{1}{2}\ln\left|\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right| + \frac{1}{2}\left[\boldsymbol{z}_{k+1} - (\boldsymbol{A}_{h,k+1}\boldsymbol{x}_{k+1} + \boldsymbol{b}_{h,k+1})\right]^T \left(\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right)^{-1} \cdot \left[\boldsymbol{z}_{k+1} - (\boldsymbol{A}_{h,k+1}\boldsymbol{x}_{k+1} + \boldsymbol{b}_{h,k+1})\right]. \tag{6.24}$$

The log-densities obtained from Equations (6.23) and (6.24) are substituted into the Equations (6.13)-(6.16), which are further simplified by applying the double gradient operator $\Delta_{\cdot}^{\cdot}$ as shown,

$$\boldsymbol{D}_k^{11} = \mathrm{E}_{\boldsymbol{x}_k}\left[\boldsymbol{A}_{f,k}^T \left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1} \boldsymbol{A}_{f,k}\right] \tag{6.25}$$

$$\boldsymbol{D}_k^{12} = -\mathrm{E}_{\boldsymbol{x}_k}\left[\boldsymbol{A}_{f,k}^T \left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1}\right] \tag{6.26}$$

$$\boldsymbol{D}_k^{21} = -\mathrm{E}_{\boldsymbol{x}_k}\left[\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1} \boldsymbol{A}_{f,k}\right] \tag{6.27}$$

$$\boldsymbol{D}_k^{22} = \mathrm{E}_{\boldsymbol{x}_k}\left[\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1}\right] + \mathrm{E}_{\boldsymbol{x}_{k+1}}\left[\boldsymbol{A}_{h,k+1}^T \left(\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right)^{-1} \boldsymbol{A}_{h,k+1}\right]. \tag{6.28}$$

Now substituting the value of $\boldsymbol{D}_k^{11}$, $\boldsymbol{D}_k^{12}$, $\boldsymbol{D}_k^{21}$ and $\boldsymbol{D}_k^{22}$ into the Equation (6.12), we obtain a recursive equation for the BIM $\boldsymbol{J}_{k+1}$ as shown below:

$$\begin{aligned}\boldsymbol{J}_{k+1} = {} & \mathrm{E}_{\boldsymbol{x}_k}\left[\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1}\right] - \mathrm{E}_{\boldsymbol{x}_k}\left[\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1}\boldsymbol{A}_{f,k}\right]\cdot \\ & \left(\boldsymbol{J}_k + \mathrm{E}_{\boldsymbol{x}_k}\left[\boldsymbol{A}_{f,k}^T\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1}\boldsymbol{A}_{f,k}\right]\right)^{-1}\mathrm{E}_{\boldsymbol{x}_k}\left[\boldsymbol{A}_{f,k}^T\left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1}\right] + \\ & \mathrm{E}_{\boldsymbol{x}_{k+1}}\left[\boldsymbol{A}_{h,k+1}^T\left(\boldsymbol{R}_{k+1} + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k+1}\right)^{-1}\boldsymbol{A}_{h,k+1}\right]. \end{aligned} \tag{6.29}$$

As proved, the recursive equation of $\boldsymbol{J}_{k+1}$ is the function of statistically linearized state and observation dynamics, statistics of the linearization error and the previous $\boldsymbol{J}_k$ computed at time k. The expression for $\boldsymbol{J}_{k+1}$ is significant due to its ability to calculate the lower bound on the SPKF performance without applying the original filter equations to the data. As recursive computation of $\boldsymbol{J}_k$ assesses the SPKF performance on a given application, it has the potential to save significant amount of time and effort in terms of computations, filter design and filter tuning compared to actual implementation of the filter.

In order to obtain a closed form solution for $\boldsymbol{J}_{k+1}$, we have to compute the multidimensional expectations over the state distribution. Note, this is necessary because the WSLR coefficients and linearization error terms at time k are defined as a function of the current state $\boldsymbol{x}_k$. Unfortunately computing the above mentioned expectations are intractable, hence statistical approximation techniques must be used in order to solve those. Monte-Carlo based simulation is one possibility to approximate the expectations. However it can be computationally intensive for calculating expectations over a state with a higher dimension. In this work, we follow the sigma-point based weighted averaging technique to solve the multidimensional expectations, which can be shown as follows.

Any expectations of the form

$$\mathrm{E}\left[g\left(\boldsymbol{x}_k\right)\right] = \int g\left(\boldsymbol{x}_k\right) p\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right) d\boldsymbol{x}_k \tag{6.30}$$

can be approximated by a sample weighted averaging of the sigma points

$$\tilde{\mathrm{E}}\left[g\left(\boldsymbol{x}_k\right)\right] = \sum_{i=0}^{2M} w_i g\left(\boldsymbol{x}_k^{(i)}\right), \tag{6.31}$$

where w_i is the weight of the i-th sigma point. Hence using the above equation, we can compute the expectations needed to obtain an expression for the lower bound. For example, to compute $\boldsymbol{D}_k^{11}$, first we extract sigma points from the expression $\boldsymbol{A}_{f,k}^T \left(\boldsymbol{Q}_k + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right)^{-1} \boldsymbol{A}_{f,k}$ and then we obtain an estimate of the true expectation by weighted averaging over all the sigma points as shown in Equation (6.31).

In this respect, there is another important thing to mention i.e. how to compute the initial value of the BIM $\boldsymbol{J}_0$. To start the recursion on $\boldsymbol{J}_k$, we need to know about $\boldsymbol{J}_0$ which can be evaluated from a priori state distribution $p\left(\boldsymbol{x}_0\right)$ as shown below

$$\boldsymbol{J}_0 = \mathrm{E}_{\boldsymbol{x}_0}\left[-\Delta_{\boldsymbol{x}_0}^{\boldsymbol{x}_0} \ln p\left(\boldsymbol{x}_0\right)\right]. \tag{6.32}$$

By taking negative logarithm on $p\left(\boldsymbol{x}_0\right)$,

$$-\ln\left(p(\boldsymbol{x}_0)\right) = \frac{M}{2}\ln\left(2\pi\right) + \frac{1}{2}\ln\left|\boldsymbol{P}_0\right| + \frac{1}{2}\left[\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right]^T \boldsymbol{P}_0^{-1}\left[\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right] \tag{6.33}$$

$$\boldsymbol{J}_0 = \mathrm{E}_{\boldsymbol{x}_0}\left[\boldsymbol{P}_0^{-1}\right] \tag{6.34}$$

$$= \boldsymbol{P}_0^{-1}. \tag{6.35}$$

In the experimental result section, we provide two different simulation examples that will demonstrate the realizations of the SPKF lower bound.

6.4 Upper Error Bound For The SPKF

In this section, we are going to analyze the error convergence behavior of the SPKF and derive an upper bound of the state estimation error in the mean square sense. However, first we describe the necessary concept and assumptions we have to make, which enable us to derive the upper error bound.

6.4.1 Conditions Of Convergence And Assumptions

We make use of the following Lemma, which states that a stochastic process can be shown to be error bounded under certain conditions, in order to analyze the error behavior of the filter. Note, Agniel *et al.* and Tarn *et al.* were the first who used this Lemma in the parameter estimation problem to prove that the error of an estimator operating on a stochastic model is upper-bounded [160, 161]. Recently Reif *et al.* also apply this to demonstrate the error behavior of the discrete time EKF [137].

Lemma 6.4.1. *If a stochastic process $\boldsymbol{V}_k(\boldsymbol{e}_k)$ as well as real numbers $c_1, c_2, c_3 > 0$ and $0 < c_4 \leq 1$ exists, such that the following criteria*

$$c_1 \left\|\boldsymbol{e}_k\right\|^2 \leq \boldsymbol{V}_k(\boldsymbol{e}_k) \leq c_2 \left\|\boldsymbol{e}_k\right\|^2 \tag{6.36}$$

$$\mathrm{E}\left[\boldsymbol{V}_{k+1}(\boldsymbol{e}_{k+1}) \,|\boldsymbol{e}_k\right] - \boldsymbol{V}_k(\boldsymbol{e}_k) \leq c_3 - c_4 \boldsymbol{V}_k(\boldsymbol{e}_k), \tag{6.37}$$

is satisfied for each $\boldsymbol{e}_k$, where $\boldsymbol{e}_k$ is the estimation error between the true and estimated state at time k

$$\boldsymbol{e}_k = \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k, \tag{6.38}$$

the stochastic process can be proved to be exponentially bounded in MSE *sense with probability one for every $k \geq 0$. In mathematical form, this can be expressed as*

$$\mathrm{E}\left[\left\|\boldsymbol{e}_k\right\|^2\right] \leq \frac{c_2}{c_1} \mathrm{E}\left[\left\|\boldsymbol{e}_0\right\|^2\right] (1 - c_4)^k + \frac{c_3}{c_1} \sum_{i=1}^{k-1} (1 - c_4)^i . \tag{6.39}$$

The proof of the Lemma is beyond the scope of this dissertation, interested readers can look at the books by Goodwin [7] and Morozan [162] and the references within for further details.

Next, we want to explicitly state the assumptions that we have to make in order to derive the upper bound of the SPKF.

Assumption 6.4.1. *Considering a statistically linearized form of the SPKF given by Equations* (6.19)-(6.20)*, the following assumptions need to be hold true for the upper error bound to exist.*

1. *There exists positive real numbers $p_{\min}$ and $p_{\max}$ which lower and upper bounds the estimation error covariance matrix $\boldsymbol{P}_k$ such that*

$$p_{\min}\boldsymbol{I} \leq \boldsymbol{P}_k \leq p_{\max}\boldsymbol{I}. \tag{6.40}$$

2. *The covariance matrices for the process and the observation noise $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are positive definite and they are lower bounded by $q_{\min} > 0$ and $r_{\min} > 0$*

$$q_{\min}\boldsymbol{I} \leq \boldsymbol{Q}_k \tag{6.41}$$

$$r_{\min}\boldsymbol{I} \leq \boldsymbol{R}_k. \tag{6.42}$$

3. *It is assumed that an upper bound exists for the norm of the WSLR coefficients* $\boldsymbol{A}_{f,k}$ *and* $\boldsymbol{A}_{h,k}$ *as follows:*

$$\|\boldsymbol{A}_{f,k}\| \leq a_{f,\max} \tag{6.43}$$

$$\|\boldsymbol{A}_{h,k}\| \leq a_{h,\max}. \tag{6.44}$$

4. *The initial estimation error and estimation error at each time* k *is finite,*

$$\|\boldsymbol{e}_0\| \leq \eta \tag{6.45}$$

$$\|\boldsymbol{e}_k\| \leq \delta_e. \tag{6.46}$$

5. *The statistical linearization errors* $\boldsymbol{\epsilon}_{f,k}$ *and* $\boldsymbol{\epsilon}_{h,k}$ *for the process and the observation models are bounded via the following equations,*

$$\|\boldsymbol{\epsilon}_{f,k}\| \leq k_{\boldsymbol{\epsilon}_f} \|\boldsymbol{e}_k\|^2 \tag{6.47}$$

$$\|\boldsymbol{\epsilon}_{h,k}\| \leq k_{\boldsymbol{\epsilon}_h} \|\boldsymbol{e}_k\|^2, \tag{6.48}$$

   where $k_{\boldsymbol{\epsilon}_f}$ *and* $k_{\boldsymbol{\epsilon}_h}$ *are the positive real numbers.*

6. *The covariances of the linearization error terms for the process and observation models*

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} = \mathrm{E}\left[\boldsymbol{\epsilon}_{f,k}\boldsymbol{\epsilon}_{f,k}^T\right] \tag{6.49}$$

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} = \mathrm{E}\left[\boldsymbol{\epsilon}_{h,k}\boldsymbol{\epsilon}_{h,k}^T\right], \tag{6.50}$$

   are assumed to be lower-bounded as follows:

$$p_{\boldsymbol{\epsilon}_f,\min}\boldsymbol{I} \leq \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} \tag{6.51}$$

$$p_{\boldsymbol{\epsilon}_h,\min}\boldsymbol{I} \leq \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}. \tag{6.52}$$

7. *Finally, we assume that the term* $\boldsymbol{G}_{f,k}$ *appears in Equation* (6.19) *are bounded via*

$$\boldsymbol{G}_{f,k}\boldsymbol{G}_{f,k}^T \leq \delta\boldsymbol{I}, \tag{6.53}$$

   for $\delta > 0$ *is a small positive number.*

The assumptions are based upon the notion that certain parameters of the system model, noise terms and the estimation error covariance are bounded and we know the exact bound values in advance. Reif *et al.* demonstrates how the existence of the above bounds can be verified during the state estimation process by taking advantage of the observability and detectability properties of the system model [137]. For example, with the help of the observability gramian they have proved that the estimation error covariance $\boldsymbol{P}_k$ is lower and upper bounded if the following conditions hold

- The initial error covariance $\boldsymbol{P}_0$ is positive definite.
- The state space model satisfies the uniform observability condition.
- The process and the observation noise covariances $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are positive definite and finite.

The details of this proof are omitted here, but can be found in [137,163].

The Equations (6.40)-(6.52) demonstrate a number of unknown parameters, such as $p_{\min}$, $p_{\max}$, $q_{\min}$ and $r_{\min}$, which determine the accuracy of the filter upper bound. In an ideal scenario, it is assumed that we already know those values and analyze the filter estimation error in terms of those parameters. But in practice, the true parameter values are unknown and hence the question is how to compute them. This is important because the upper bound of the filter estimation error is derived as a function of those parameters, hence any inaccurate computation may pose a direct effect on the accuracy of the derived error bound in practical examples. Formulating the parameters analytically should be the ideal case but as mentioned by Reif and Alessandri, the process is challenging and complex [137, 138]. One option is to resort to simulations in order to estimate them from the training data. For example, Reif *et al.* computed the maximum and minimum eigenvalue of $\boldsymbol{P}_k \ \forall k$ from the training data and used those as a bound for $\boldsymbol{P}_k$, i.e. $p_{\min}$ and $p_{\max}$. The terms $k_{\boldsymbol{\epsilon}_f}$ and $k_{\boldsymbol{\epsilon}_h}$ are estimated with respect to the highest spectral norm of the Hessian matrices of the process and the observation model. Similarly, the bounds $q_{\min}$ and $r_{\min}$ are adopted from the minimum eigenvalues over $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k \ \forall k$ respectively [137,138].

We have also adopted a training procedure similar to Reif and Alessandri in order to estimate the above bound parameters. However, instead of computing the *sample* bounds from the training data, we derive the *confidence-interval* estimate for each unknown parameter, which is an interval that contains the $N\%$ of the total probability distribution of a random variable. By definition, the confidence interval is the region within which the unknown true value of the parameter is expected to fall with approximately $N\%$ probability. Assuming a random variable $\boldsymbol{X}$ is normal distributed, the $N\%$ upper and lower confidence interval bounds can be described as follows:

$$\mu_{\boldsymbol{x}} \pm z_N \sigma_{\boldsymbol{x}}, \tag{6.54}$$

where $\mu_{\boldsymbol{x}}$ and $\sigma_{\boldsymbol{x}}$ are the mean and standard deviation of the measured value $\boldsymbol{x}$ of $\boldsymbol{X}$, which are obtained from the training data. The constant z_N specifies the size of the interval about the mean that contains $N\%$ of the probability mass under the normal distribution. The values of z_N for the two-sided $N\%$ confidence intervals can be found in any statistics textbook. Note, we have used N=95% as 95% confidence interval is most commonly used in the statistical learning theory. For N=95%, the value of $z_N = 1.96$. For details about the assumptions and the procedure, please refer to chapter 5 of the Mitchell's book [164].

We have employed the above concept to determine the unknown parameters as shown in Equations (6.40)-(6.52). For example, we run the SPKF on the training data comprising $T = 500$ independent trials to compute the estimation error covariance $\boldsymbol{P}_k$ from each trial. The minimum and the maximum values $\hat{p}^t_{\min}$ and $\hat{p}^t_{\max}$ are computed at each trial t from the corresponding trace of $\boldsymbol{P}_k$. Combining $\hat{p}^t_{\min}$ and $\hat{p}^t_{\max}$ over all T trials, we obtain

$$\hat{\boldsymbol{p}}_{\min} = \begin{bmatrix} \hat{p}^1_{\min} & \cdots & \hat{p}^t_{\min} & \hat{p}^{\mathrm{T}}_{\min} \end{bmatrix} \tag{6.55}$$

$$\hat{\boldsymbol{p}}_{\max} = \begin{bmatrix} \hat{p}^1_{\max} & \cdots & \hat{p}^t_{\max} & \hat{p}^{\mathrm{T}}_{\max} \end{bmatrix}. \tag{6.56}$$

Assuming that the distribution of the random vectors $\hat{\boldsymbol{p}}_{\min}$ and $\hat{\boldsymbol{p}}_{\max}$ as Gaussian, the $N = 95\%$ confidence interval is estimated using the Equation (6.54) for each distribution. The confidence interval upper and lower bound for the distribution of $\hat{\boldsymbol{p}}_{\min}$ can be shown as

$$\mu_{\hat{\boldsymbol{p}}_{\min}} \pm z_{95} \sigma_{\hat{\boldsymbol{p}}_{\min}}, \tag{6.57}$$

where $\mu_{\hat{\boldsymbol{p}}_{\min}}$ and $\sigma_{\boldsymbol{p}_{\min}}$ are the sample mean and the sample variance from the distribution. In other words, we can tell with 95% confidence that the true $p_{\min}$ lies between the interval defined by $\mu_{\hat{\boldsymbol{p}}_{\min}} - z_{95}\sigma_{\hat{\boldsymbol{p}}_{\min}}$ and $\mu_{\hat{\boldsymbol{p}}_{\min}} + z_{95}\sigma_{\hat{\boldsymbol{p}}_{\min}}$. In a similar way, we can also demonstrate that the true $p_{\max}$ is bounded within $\mu_{\hat{\boldsymbol{p}}_{\max}} - z_{95}\sigma_{\hat{\boldsymbol{p}}_{\max}}$ and $\mu_{\hat{\boldsymbol{p}}_{\max}} + z_{95}\sigma_{\hat{\boldsymbol{p}}_{\max}}$ with a probability equal to 95%. Hence using the above logic, we can choose the values of $p_{\min}$ and $p_{\max}$ as

$$p_{\min} = \mu_{\hat{\boldsymbol{p}}_{\min}} - z_{95}\sigma_{\hat{\boldsymbol{p}}_{\min}} \tag{6.58}$$

$$p_{\max} = \mu_{\hat{\boldsymbol{p}}_{\max}} + z_{95}\sigma_{\hat{\boldsymbol{p}}_{\max}}, \tag{6.59}$$

where $z_{95} = 1.96$.

We have implemented the exact similar procedure in order to determine the 95% confidence interval bounds for $\boldsymbol{Q}_k$, $\boldsymbol{R}_k$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$, $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$, $\boldsymbol{A}_{f,k}$ and $\boldsymbol{A}_{h,k}$. We are not repeating those derivations as we think readers can compute them by following the above procedure. In order to estimate $k_{\boldsymbol{\epsilon}_f}$ and $k_{\boldsymbol{\epsilon}_h}$, we have followed the procedure shown by Reif *et al.* [137]. The confidence interval estimates of the parameters defined here are used to compute the estimation error upper bound in practical examples shown in Section 6.5.

6.4.2 Mathematical Derivation

In this section, our objective is to form the Lyapunov function as a function of the SPKF estimation error $\boldsymbol{e}_k$ so that the Lemma 6.4.1 can be applied to prove the error bound. We start with an expression of the state estimate ($\hat{\boldsymbol{x}}_{k+1}$) at time $k+1$ in terms of previous state estimate $\hat{\boldsymbol{x}}_k$ at time k

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k + \boldsymbol{b}_{f,k} + \boldsymbol{K}_k\left(\boldsymbol{z}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k - \boldsymbol{b}_{h,k}\right), \tag{6.60}$$

where $\boldsymbol{K}_k$ is the Kalman gain and $\boldsymbol{z}_k$ is the observation at time k. The statistical linearization parameters for the process model are $\boldsymbol{A}_{f,k}$, $\boldsymbol{b}_{f,k}$, whereas that for the observation model are $\boldsymbol{A}_{h,k}$ and $\boldsymbol{b}_{h,k}$ respectively. Substituting $\boldsymbol{z}_k$ in (6.60) with the WSLR observation Equation (6.11) and also defining $\boldsymbol{e}_k = \hat{\boldsymbol{x}}_k - \boldsymbol{x}_k$, we obtain

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{A}_{f,k}\hat{\boldsymbol{x}}_k + \boldsymbol{b}_{f,k} + \boldsymbol{K}_k\left(\boldsymbol{A}_{h,k}\boldsymbol{x}_k - \boldsymbol{A}_{h,k}\hat{\boldsymbol{x}}_k + \boldsymbol{n}_k + \boldsymbol{\epsilon}_{h,k}\right). \tag{6.61}$$

Subtracting Equation (6.61) from the WSLR process Equation (6.19)

$$\begin{aligned}
\boldsymbol{x}_{k+1} - \hat{\boldsymbol{x}}_{k+1} &= \boldsymbol{A}_{f,k}\left(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\right) + \boldsymbol{\epsilon}_{f,k} + \boldsymbol{G}_{f,k}\boldsymbol{v}_k - \boldsymbol{K}_k\left(\boldsymbol{A}_{h,k}\left(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\right) + \boldsymbol{\epsilon}_{h,k} + \boldsymbol{n}_k\right)\\
\boldsymbol{e}_{k+1} &= \left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_{f,k} - \boldsymbol{K}_k\boldsymbol{\epsilon}_{h,k} + \boldsymbol{G}_{f,k}\boldsymbol{v}_k - \boldsymbol{K}_k\boldsymbol{n}_k\\
\boldsymbol{e}_{k+1} &= \left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_k^{\mathrm{t}} + \boldsymbol{N}_k^{\mathrm{t}}, \qquad (6.62)
\end{aligned}$$

where

$$\boldsymbol{\epsilon}_k^{\mathrm{t}} = \boldsymbol{\epsilon}_{f,k} - \boldsymbol{K}_k\boldsymbol{\epsilon}_{h,k}, \qquad (6.63)$$

is the combined pseudo-linearization error and

$$\boldsymbol{N}_k^{\mathrm{t}} = \boldsymbol{G}_{f,k}\boldsymbol{v}_k - \boldsymbol{K}_k\boldsymbol{n}_k, \qquad (6.64)$$

is the combined noise term. The Equation (6.62) demonstrates how the estimation error $\boldsymbol{e}_k$ propagates from time k to $k+1$.

In order to use the Lemma 6.4.1, which proves the convergence of the estimation error $\boldsymbol{e}_k$, we introduce the Lyapunov function $\boldsymbol{V}_k\left(\boldsymbol{e}_k\right)$, which is defined as

$$\boldsymbol{V}_k\left(\boldsymbol{e}_k\right) = \boldsymbol{e}_k^T\boldsymbol{P}_k^{-1}\boldsymbol{e}_k, \text{ for } k = 0, 1, \ldots, N. \qquad (6.65)$$

Using (6.40), we have

$$\frac{1}{p_{\max}}\left\|\boldsymbol{e}_k\right\|^2 \leq \boldsymbol{V}_k\left(\boldsymbol{e}_k\right) \leq \frac{1}{p_{\min}}\left\|\boldsymbol{e}_k\right\|^2 \qquad (6.66)$$

This satisfies the first condition for Lemma 6.4.1, as listed in Equation (6.36). In order to prove the second condition, we have to demonstrate that an upper bound for $\mathrm{E}\left[\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right)|\boldsymbol{e}_k\right]$ exists as in Equation (6.37). Defining $\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right)$ in terms of the error dynamics shown in (6.62)

$$\begin{aligned}
\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right) &= \left[\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_k^{\mathrm{t}} + \boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\left[\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_k^{\mathrm{t}} + \boldsymbol{N}_k^{\mathrm{t}}\right]\\
&= \boldsymbol{e}_k^T\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T\boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\left[2\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k\right] +\\
&\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^{\mathrm{t}} + 2\left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\left[\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_k^{\mathrm{t}}\right] + \left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^{\mathrm{t}}. \qquad (6.67)
\end{aligned}$$

The task is now simplified to deriving an upper bound for each term on the right hand side of Equation (6.67). The proof of this is divided into several Lemmas, which we will describe next. The Lemmas, emphasized below, are the integral parts of this derivation and is proved from first principles.

Lemma 6.4.2. *Taking into account that the Assumption 6.4.1 be fulfilled, we can demonstrate that the norm of the Kalman gain term $\boldsymbol{K}_k$ of the SPKF is satisfied by the following inequality*

$$\|\boldsymbol{K}_k\| \leq \frac{a_{f,\max} p_{\max} a_{h,\max}}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}. \tag{6.68}$$

Proof*:*

Defining Kalman gain $\boldsymbol{K}_k$ based on the coefficients of the statistically linearized dynamical system

$$\boldsymbol{K}_k = \boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T \left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{R}_k\right)^{-1} \tag{6.69}$$

$$\boldsymbol{K}_k \leq \|\boldsymbol{A}_{f,k}\| \|\boldsymbol{P}_k\| \left\|\boldsymbol{A}_{h,k}^T\right\| \left\|\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{R}_k\right)^{-1}\right\|. \tag{6.70}$$

Note, for a vector $\boldsymbol{x} \in \mathbb{R}^n$, the $\|\boldsymbol{x}\|$ is defined as the Euclidean norm of $\boldsymbol{x}$

$$\|\boldsymbol{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}. \tag{6.71}$$

Now using Assumption 6.4.1, we can upper bound each individual norm and prove the following

$$\boldsymbol{K}_k \leq \frac{a_{f,\max} p_{\max} a_{h,\max}}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}, \tag{6.72}$$

which concludes the proof.

Lemma 6.4.3. *Let the conditions of Assumption 6.4.1 hold and defining a real number $0 < \alpha < 1$, we can prove the following inequality*

$$\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T \boldsymbol{P}_{k+1}^{-1} \left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right) \leq \frac{(1-\alpha)}{(1+\epsilon)}\boldsymbol{P}_k^{-1}, \tag{6.73}$$

where $\epsilon \approx 0$ is a very small positive real number.

Proof*:*

We start with the equation of the estimation error covariance $\boldsymbol{P}_{k+1}$ at time $k+1$ in terms of the error covariance $\boldsymbol{P}_k$

$$\boldsymbol{P}_{k+1} = (1+\epsilon)\left[\boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{f,k}^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k - \boldsymbol{K}_k\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{R}_k\right)\boldsymbol{K}_k^T\right]. \tag{6.74}$$

The term $(1+\epsilon)$ *is called "covariance scaling", which is introduced by Alessandri et al. in deriving the error bound of the modified EKF algorithm [138]. Although this term inflates the covariance matrix slightly from time* k *to* $k+1$*, the advantage is that it simplifies the derivation of the error bound. Furthermore, Alessandri et al. demonstrate that there is no visible change of performance for the EKF with and without the presence of* ϵ*. Manipulating the R.H.S. of Equation* (6.74) *yields*

$$\begin{aligned}\boldsymbol{P}_{k+1} &= (1+\epsilon)\left[\boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{f,k}^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k - \boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T\boldsymbol{K}_k^T\right] \\ &= (1+\epsilon)\left[\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T\right] + \\ &\quad (1+\epsilon)\left[\boldsymbol{K}_k\boldsymbol{A}_{h,k}\boldsymbol{P}_k\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right].\end{aligned} \tag{6.75}$$

In the next step, we aim at simplifying the term $\boldsymbol{K}_k\boldsymbol{A}_{h,k}\boldsymbol{P}_k\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T$*. With Equation* (6.69)*, we demonstrate*

$$\begin{aligned}\boldsymbol{A}_{f,k}^{-1}\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k &= \boldsymbol{P}_k - \boldsymbol{A}_{f,k}^{-1}\boldsymbol{K}_k\boldsymbol{A}_{h,k}\boldsymbol{P}_k \\ &= \boldsymbol{P}_k - \boldsymbol{P}_k\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{R}_k\right)^{-1}\boldsymbol{A}_{h,k}\boldsymbol{P}_k.\end{aligned} \tag{6.76}$$

Now using the matrix inversion Lemma, the R.H.S. of the Equation (6.76) *can be simplified to*

$$\begin{aligned}\boldsymbol{A}_{f,k}^{-1}\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k &= \left(\boldsymbol{P}_k^{-1} + \boldsymbol{A}_{h,k}^T\boldsymbol{R}_k^{-1}\boldsymbol{A}_{h,k}\right)^{-1} \\ \left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right) &= \boldsymbol{A}_{f,k}\left(\boldsymbol{P}_k^{-1} + \boldsymbol{A}_{h,k}^T\boldsymbol{R}_k^{-1}\boldsymbol{A}_{h,k}\right)^{-1}\boldsymbol{P}_k^{-1} \geq 0.\end{aligned} \tag{6.77}$$

It can be seen that the R.H.S. of Equation (6.77) ≥ 0 *as all the R.H.S. matrices are either positive definite or positive semi-definite (*$\boldsymbol{A}_{f,k}$ *and* $\boldsymbol{A}_{h,k}$ *are positive semi-definite matrices and all others are positive definite). From the definition of* $\boldsymbol{K}_k$*, we can show that*

$$\boldsymbol{K}_k = \boldsymbol{A}_{f,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T\left(\boldsymbol{A}_{h,k}\boldsymbol{P}_k\boldsymbol{A}_{h,k}^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_h,k} + \boldsymbol{R}_k\right)^{-1} \geq 0. \tag{6.78}$$

Combining Equations (6.77) *and* (6.78)*,*

$$\boldsymbol{K}_k\boldsymbol{A}_{h,k}\boldsymbol{P}_k\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T \geq 0. \tag{6.79}$$

Substituting Equation (6.79) *into* (6.75)

$$\boldsymbol{P}_{k+1} \geq (1+\epsilon)\left[\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{P}_k\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T + \boldsymbol{P}_{\boldsymbol{\epsilon}_f,k} + \boldsymbol{Q}_k\right]. \tag{6.80}$$

Now multiplying $(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^{-1}$ *and* $(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^{-T}$ *by the left and right of each term on both sides*

$$\frac{1}{(1+\epsilon)}(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^{-1}\boldsymbol{P}_{k+1}(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^{-T}\geq\boldsymbol{P}_k+(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^{-1}\cdot\left(\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}+\boldsymbol{Q}_k\right)(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^{-T}. \tag{6.81}$$

Applying Lemma 6.4.2 and Assumption 6.4.1 to simplify the Equation (6.81) *as follows*

$$\begin{aligned}\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}&\leq\|\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\|\boldsymbol{I}\\&\leq(\|\boldsymbol{A}_{f,k}\|+\|\boldsymbol{K}_k\|\|\boldsymbol{A}_{h,k}\|)\boldsymbol{I}\\&\leq\left(a_{f,\max}+\frac{a_{f,\max}p_{\max}a_{h,\max}^2}{p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}}\right)\boldsymbol{I}\\&\leq a_{f,\max}\left(1+\frac{p_{\max}a_{h,\max}^2}{p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}}\right)\boldsymbol{I}.\end{aligned}\tag{6.82}$$

Interchanging the L.H.S and R.H.S. terms in Equation (6.82),

$$\left[(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^T\right]^{-1}\geq\frac{1}{a_{f,\max}\left(1+\frac{p_{\max}a_{h,\max}^2}{p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}}\right)}\boldsymbol{I}. \tag{6.83}$$

Similarly we can show that

$$\left[(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k})^T\right]^{-T}\geq\frac{1}{a_{f,\max}\left(1+\frac{p_{\max}a_{h,\max}^2}{p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}}\right)}\boldsymbol{I}. \tag{6.84}$$

From Assumption 6.4.1,

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\geq p_{\boldsymbol{\epsilon}_f,\min}\boldsymbol{I} \tag{6.85}$$

$$\boldsymbol{Q}_k\geq q_{\min}\boldsymbol{I} \tag{6.86}$$

We can express the identity matrix $\boldsymbol{I}$ *in terms of the estimation error covariance* $\boldsymbol{P}_k$ *and its upper bound* $p_{\max}$

$$\begin{aligned}\boldsymbol{P}_k&\leq p_{\max}\boldsymbol{I}\\\boldsymbol{P}_k^{-1}&\geq\frac{1}{p_{\max}}\boldsymbol{I}\\\boldsymbol{I}=\boldsymbol{P}_k\boldsymbol{P}_k^{-1}&\geq\frac{\boldsymbol{P}_k}{p_{\max}}\end{aligned}\tag{6.87}$$

The R.H.S. of Equation (6.81) *can be substituted using the results obtained from Equations* (6.83)*,* (6.84)*,* (6.86) *and* (6.87)

$$\frac{1}{(1+\epsilon)}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^{-1}\boldsymbol{P}_{k+1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^{-T}\geq \boldsymbol{P}_k\left[1+\frac{p_{\boldsymbol{\epsilon}_f,\min}+q_{\min}}{p_{\max}a_{f,\max}^2\left(1+\frac{p_{\max}a_{h,\max}^2}{p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}}\right)^2}\right]. \tag{6.88}$$

Now defining the term t_1*,*

$$t_1=\frac{p_{\boldsymbol{\epsilon}_f,\min}+q_{\min}}{p_{\max}a_{f,\max}^2\left(1+\frac{p_{\max}a_{h,\max}^2}{p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}}\right)^2}, \tag{6.89}$$

we can write Equation (6.88) *in terms of* t_1*,*

$$\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^{-1}\boldsymbol{P}_{k+1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^{-T}=(1+\epsilon)\,(1+t_1)\,\boldsymbol{P}_k. \tag{6.90}$$

Taking inverse on both sides and assuming $(1-\alpha)=\frac{1}{1+t_1}$*, we obtain*

$$\begin{aligned}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^{T}\boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right) &\leq \frac{1}{(1+\epsilon)\,(1+t_1)}\boldsymbol{P}_k^{-1}\\ &\leq \frac{(1-\alpha)}{(1+\epsilon)}\boldsymbol{P}_k^{-1},\end{aligned} \tag{6.91}$$

which concludes the proof.

Lemma 6.4.4. *Considering that the Assumption 6.4.1 holds true and there exists a real number* $t_2\in\mathbb{R}>0$*, we can prove the following inequality*

$$\left[\boldsymbol{\epsilon}_k^t\right]^T\boldsymbol{P}_{k+1}^{-1}\left[2\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\boldsymbol{\epsilon}_k^t\right]\leq t_2\left\|\boldsymbol{e}_k\right\|^3, \tag{6.92}$$

where $\boldsymbol{\epsilon}_k^t$ *is the combined pseudo-linearization error as defined in Equation* (6.63) *and* $\boldsymbol{e}_k$ *is the SPKF estimation error.*

Proof*:*

From the properties of the vector and matrix norm, the L.H.S. of the above inequality can be written as

$$\left[\boldsymbol{\epsilon}_k^t\right]^T\boldsymbol{P}_{k+1}^{-1}\left[2\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\boldsymbol{\epsilon}_k^t\right]\leq \left\|\left[\boldsymbol{\epsilon}_k^t\right]^T\right\|\left\|\boldsymbol{P}_{k+1}^{-1}\right\|\left[2\left(\left\|\boldsymbol{A}_{f,k}\right\|+\left\|\boldsymbol{K}_k\right\|\left\|\boldsymbol{A}_{h,k}\right\|\right)\left\|\boldsymbol{e}_k\right\|+\left\|\boldsymbol{\epsilon}_k^t\right\|\right] \tag{6.93}$$

We can expand $\boldsymbol{\epsilon}_k^t$ *from Equation* (6.63) *and then take the norm on both sides*

$$\left\| \left[\boldsymbol{\epsilon}_k^t\right]^T \right\| \leq \|\boldsymbol{\epsilon}_{f,k}\| + \|\boldsymbol{K}_k\| \, \|\boldsymbol{\epsilon}_{h,k}\| \tag{6.94}$$

Using Assumption 6.4.1 and Lemma 6.4.2,

$$\left\| \left[\boldsymbol{\epsilon}_k^t\right]^T \right\| \leq k_{\boldsymbol{\epsilon}_{f,k}} \|\boldsymbol{e}_k\|^2 + \frac{a_{f,\max} p_{\max} a_{h,\max}}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})} k_{\boldsymbol{\epsilon}_{h,k}} \|\boldsymbol{e}_k\|^2 . \tag{6.95}$$

The $\left\|\boldsymbol{P}_{k+1}^{-1}\right\|$ *can be shown as*

$$\left\|\boldsymbol{P}_{k+1}^{-1}\right\| \leq \frac{1}{p_{\max}}. \tag{6.96}$$

We apply Equations (6.95), (6.96) *and the Assumption 6.4.1 to substitute the R.H.S. of Equation* (6.93). *After simplification we obtain*

$$\begin{aligned}\left[\boldsymbol{\epsilon}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \left[2\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k \boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_k^t\right] \leq \left[\left(k_{\boldsymbol{\epsilon}_{f,k}} + \frac{a_{f,\max} p_{\max} a_{h,\max} k_{\boldsymbol{\epsilon}_{h,k}}}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}\right)\frac{1}{p_{\min}}\right] . \\ \left[2a_{f,\max}\left(1 + \frac{p_{\max} a_{h,\max}^2}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}\right) + \left(k_{\boldsymbol{\epsilon}_{f,k}} + \frac{a_{f,\max} p_{\max} a_{h,\max} k_{\boldsymbol{\epsilon}_{h,k}}}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}\right)\delta_e\right] \|\boldsymbol{e}_k\|^3 .\end{aligned} \tag{6.97}$$

Now introducing t_2,

$$\begin{aligned} t_2 = & \left[\left(k_{\boldsymbol{\epsilon}_{f,k}} + \frac{a_{f,\max} p_{\max} a_{h,\max} k_{\boldsymbol{\epsilon}_{h,k}}}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}\right)\frac{1}{p_{\min}}\right] . \\ & \left[2a_{f,\max}\left(1 + \frac{p_{\max} a_{h,\max}^2}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}\right) + \left(k_{\boldsymbol{\epsilon}_{f,k}} + \frac{a_{f,\max} p_{\max} a_{h,\max} k_{\boldsymbol{\epsilon}_{h,k}}}{(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min})}\right)\delta_e\right], \end{aligned} \tag{6.98}$$

and finally substituting t_2 *into Equation* (6.97)

$$\left[\boldsymbol{\epsilon}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \left[2\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k \boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_k^t\right] \leq t_2 \|\boldsymbol{e}_k\|^3 , \tag{6.99}$$

we prove the above Lemma.

Lemma 6.4.5. *Using the Assumption 6.4.1 and there exists a positive real number* $t_2 \in \mathbb{R} > 0$, *we can prove the following inequality*

$$\begin{aligned} 2\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \left[\left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k \boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k + \boldsymbol{\epsilon}_k^t\right] \leq \left(1 + \frac{1}{\epsilon}\right)\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{N}_k^t + \\ \left[\boldsymbol{\epsilon}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{\epsilon}_k^t + \epsilon \boldsymbol{e}_k^T \left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k \boldsymbol{A}_{h,k}\right)^T \boldsymbol{P}_{k+1}^{-1} \left(\boldsymbol{A}_{f,k} - \boldsymbol{K}_k \boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k \end{aligned} \tag{6.100}$$

where $\boldsymbol{N}_k^t$ *is the combined noise term as defined in Equation* (6.64) *and* $\epsilon \approx 0$ *is a very small positive real number.*

Proof*:*

The L.H.S. of the proof is divided into two parts:

$$2\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\left[\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\boldsymbol{\epsilon}_k^t\right] = 2\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\left[\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k\right] + 2\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^t. \tag{6.101}$$

The task is now to decorrelate the noise expression $\boldsymbol{N}_k$ from the expression involving the estimation error $\boldsymbol{e}_k$. For this purpose, we make use of Young's inequality theorem,

$$2\boldsymbol{a}_1^T\boldsymbol{a}_2 \le \boldsymbol{a}_1^T\boldsymbol{P}_a\boldsymbol{a}_1 + \boldsymbol{a}_2^T\boldsymbol{P}_a^{-1}\boldsymbol{a}_2, \forall \boldsymbol{a}_1,\boldsymbol{a}_2 \in \mathbb{R}^n \tag{6.102}$$

where $\boldsymbol{P}_a$ is a symmetric positive definite matrix of dimension $n \times n$. Now defining

$$\boldsymbol{a}_1 = \boldsymbol{N}_k^t \tag{6.103}$$

$$\boldsymbol{a}_2 = \boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k \tag{6.104}$$

$$\boldsymbol{P}_a = \frac{1}{\epsilon}\boldsymbol{P}_{k+1}^{-1}, \tag{6.105}$$

for the first part of the R.H.S. of Equation (6.101)*, Young's inequality provides*

$$2\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\left[\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k\right] \le \frac{1}{\epsilon}\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^t + \epsilon\boldsymbol{e}_k^T\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T\boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k. \tag{6.106}$$

Similarly Young's inequality also can be applied on the second part of the R.H.S. of Equation (6.101) *by specifying*

$$\boldsymbol{a}_1 = \boldsymbol{N}_k^t \tag{6.107}$$

$$\boldsymbol{a}_2 = \boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^t \tag{6.108}$$

$$\boldsymbol{P}_a = \boldsymbol{P}_{k+1}^{-1}, \tag{6.109}$$

$$2\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^t \le \left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^t + \left[\boldsymbol{\epsilon}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^t. \tag{6.110}$$

Combining Equations (6.106) *and* (6.110) *gives the final result:*

$$2\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\left[\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\boldsymbol{\epsilon}_k^t\right] \le \left(1+\frac{1}{\epsilon}\right)\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^t + \left[\boldsymbol{\epsilon}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^t + \epsilon\boldsymbol{e}_k^T\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T\boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k. \tag{6.111}$$

Lemma 6.4.6. *Let the Assumption 6.4.1 holds true, we can derive that for* $t_3 \in \mathbb{R} > 0$,

$$\mathrm{E}\left[\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^t\right] \leq t_3 \tag{6.112}$$

Proof*:*

Substituting the expression of $\boldsymbol{N}_k^t$ *from Equation* (6.64) *and then performing expectation on the L.H.S. of the proof,*

$$\begin{aligned}\mathrm{E}\left[\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^t\right] &= \mathrm{E}\left[\left(\boldsymbol{G}_{f,k}\boldsymbol{v}_k - \boldsymbol{K}_k\boldsymbol{n}_k\right)^T \boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{G}_{f,k}\boldsymbol{v}_k - \boldsymbol{K}_k\boldsymbol{n}_k\right)\right]\\ &= \mathrm{E}\left[\boldsymbol{v}_k^T\boldsymbol{G}_{f,k}^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{G}_{f,k}\boldsymbol{v}_k + \boldsymbol{n}_k^T\boldsymbol{K}_k^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{K}_k\boldsymbol{n}_k\right]\end{aligned} \tag{6.113}$$

Expectation of the other cross terms in Equation (6.113) *is zero as both the process and observation noise* ($\boldsymbol{v}_k$ *and* $\boldsymbol{n}_k$) *are white and have zero mean. In order to evaluate the expectation on the R.H.S. of Equation* (6.113), *we use Assumption 6.4.1 and Lemma 6.4.2 as follows:*

$$\boldsymbol{v}_k^T\boldsymbol{G}_{f,k}^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{G}_{f,k}\boldsymbol{v}_k \leq \frac{1}{p_{\min}}\boldsymbol{v}_k^T\boldsymbol{G}_{f,k}^T\boldsymbol{G}_{f,k}\boldsymbol{v}_k \tag{6.114}$$

$$\boldsymbol{n}_k^T\boldsymbol{K}_k^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{K}_k\boldsymbol{n}_k \leq \frac{a_{f,\max}^2 p_{\max}^2 a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min}\right)^2}\boldsymbol{n}_k^T\boldsymbol{n}_k, \tag{6.115}$$

where

$$\boldsymbol{P}_{k+1}^{-1} \leq \frac{1}{p_{\min}}. \tag{6.116}$$

After substituting the expressions from Equations (6.114) *and* (6.115) *into the Equation* (6.113), *it can be further simplified as*

$$\begin{aligned}&\mathrm{E}\left[\boldsymbol{v}_k^T\boldsymbol{G}_{f,k}^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{G}_{f,k}\boldsymbol{v}_k + \boldsymbol{n}_k^T\boldsymbol{K}_k^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{K}_k\boldsymbol{n}_k\right] \leq\\ &\mathrm{E}\left[\frac{1}{p_{\min}}\mathrm{trace}\left(\boldsymbol{v}_k\boldsymbol{G}_{f,k}\boldsymbol{G}_{f,k}^T\boldsymbol{v}_k^T\right)\right] + \mathrm{E}\left[\frac{a_{f,\max}^2 p_{\max}^2 a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min}\right)^2}\mathrm{trace}\left(\boldsymbol{n}_k\boldsymbol{n}_k^T\right)\right]\\ &\leq \frac{1}{p_{\min}}\mathrm{trace}\left[\mathrm{E}\left(\boldsymbol{G}_{f,k}\boldsymbol{v}_k\boldsymbol{v}_k^T\boldsymbol{G}_{f,k}^T\right)\right] + \frac{a_{f,\max}^2 p_{\max}^2 a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min}\right)^2}\mathrm{trace}\left[\mathrm{E}\left(\boldsymbol{n}_k\boldsymbol{n}_k^T\right)\right]\\ &\leq \frac{1}{p_{\min}}\mathrm{trace}\left(\boldsymbol{G}_{f,k}\boldsymbol{Q}_k\boldsymbol{G}_{f,k}^T\right) + \frac{a_{f,\max}^2 p_{\max}^2 a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min}\right)^2}\mathrm{trace}\left(\boldsymbol{R}_k\right).\end{aligned} \tag{6.117}$$

As both sides of Equation (6.113) *are scalars, applying the* trace *operation on the R.H.S. as shown above does not change its value. In the above equation, we have used*

$$\mathrm{E}\left[\boldsymbol{v}_k\boldsymbol{v}_k^T\right] = \boldsymbol{Q}_k \tag{6.118}$$

$$\mathrm{E}\left[\boldsymbol{n}_k\boldsymbol{n}_k^T\right] = \boldsymbol{R}_k. \tag{6.119}$$

The $\operatorname{trace}(\boldsymbol{Q}_k)$ *and* $\operatorname{trace}(\boldsymbol{R}_k)$ *can be shown as*

$$\operatorname{trace}(\boldsymbol{Q}_k) \leq q_{\max} q_{\text{row}}, \quad q_{\text{row}} = \textit{number of rows in } \boldsymbol{Q}_k \textit{ matrix} \tag{6.120}$$

$$\operatorname{trace}(\boldsymbol{R}_k) \leq r_{\max} r_{\text{row}}, \quad r_{\text{row}} = \textit{number of rows in } \boldsymbol{R}_k \textit{ matrix.} \tag{6.121}$$

Substituting the Equations (6.120) *and* (6.121) *into the Equation* (6.117) *and also using the bound for* $\boldsymbol{G}_{f,k}\boldsymbol{G}_{f,k}^T$, *we get the final result*

$$\begin{aligned} \mathrm{E}\left[\left[\boldsymbol{N}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{N}_k^t\right] &\leq \frac{1}{p_{\min}} q_{\max} q_{\text{row}} \delta + \frac{a_{f,\max}^2 p_{\max}^2 a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min}\right)^2} r_{\max} r_{\text{row}} \\ &\leq t_3, \end{aligned} \tag{6.122}$$

where $t_3 > 0$ *is defined as*

$$t_3 = \frac{1}{p_{\min}} q_{\max} q_{\text{row}} \delta + \frac{a_{f,\max}^2 p_{\max}^2 a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min}\right)^2} r_{\max} r_{\text{row}}. \tag{6.123}$$

Lemma 6.4.7. *Assuming that there exists Assumption 6.4.1, we can prove the following*

$$\mathrm{E}\left[\left[\boldsymbol{\epsilon}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{\epsilon}_k^t\right] \leq t_4, \tag{6.124}$$

where $t_4 \in \mathbb{R} > 0$.

Proof*:*

The proof of this Lemma follows the same path as the previous Lemma 6.4.6. First expanding $\boldsymbol{\epsilon}_k^t$ *using the Equation* (6.63) *and then taking the expectation on both sides*

$$\begin{aligned} \mathrm{E}\left[\left[\boldsymbol{\epsilon}_k^t\right]^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{\epsilon}_k^t\right] &= \mathrm{E}\left[\left(\boldsymbol{\epsilon}_{f,k} - \boldsymbol{K}_k \boldsymbol{\epsilon}_{h,k}\right)^T \boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{\epsilon}_{f,k} - \boldsymbol{K}_k \boldsymbol{\epsilon}_{h,k}\right)\right] \\ &= \mathrm{E}\left[\boldsymbol{\epsilon}_{f,k}^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{\epsilon}_{f,k}\right] + \mathrm{E}\left[\boldsymbol{\epsilon}_{h,k}^T \boldsymbol{K}_k^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{K}_k \boldsymbol{\epsilon}_{h,k}\right]. \end{aligned} \tag{6.125}$$

The cross terms obtained after multiplication in Equation (6.125) *are zero as the WSLR linearization error terms (*$\boldsymbol{\epsilon}_{f,k}$ *and* $\boldsymbol{\epsilon}_{h,k}$*) are assumed to be white and have zero mean. Applying the Assumption 6.4.1 and Lemma 6.4.2, we can show that*

$$\boldsymbol{\epsilon}_{f,k}^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{\epsilon}_{f,k} \leq \frac{1}{p_{\min}} \boldsymbol{\epsilon}_{f,k}^T \boldsymbol{\epsilon}_{f,k} \tag{6.126}$$

$$\boldsymbol{\epsilon}_{h,k}^T \boldsymbol{K}_k^T \boldsymbol{P}_{k+1}^{-1} \boldsymbol{K}_k \boldsymbol{\epsilon}_{h,k} \leq \frac{a_{f,\max}^2 p_{\max}^2 a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min} + r_{\min}\right)^2} \boldsymbol{\epsilon}_{h,k}^T \boldsymbol{\epsilon}_{h,k}, \tag{6.127}$$

where

$$\boldsymbol{P}_{k+1}^{-1} \leq \frac{1}{p_{\min}}. \tag{6.128}$$

Adding Equations (6.126) *and* (6.127)*, the R.H.S. of Equation* (6.125) *becomes*

$$\mathrm{E}\left[\boldsymbol{\epsilon}_{f,k}^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_{f,k}\right]+\mathrm{E}\left[\boldsymbol{\epsilon}_{h,k}^T\boldsymbol{K}_k^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{K}_k\boldsymbol{\epsilon}_{h,k}\right]\leq\frac{1}{p_{\min}}\boldsymbol{\epsilon}_{f,k}^T\boldsymbol{\epsilon}_{f,k}+\frac{a_{f,\max}^2p_{\max}^2a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}\right)^2}\boldsymbol{\epsilon}_{h,k}^T\boldsymbol{\epsilon}_{h,k}. \tag{6.129}$$

Again applying the trace *operation on the R.H.S*

$$\mathrm{E}\left[\left[\boldsymbol{\epsilon}_k^t\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^t\right]\leq$$
$$\frac{1}{p_{\min}}\mathrm{trace}\left[\mathrm{E}\left[\boldsymbol{\epsilon}_{f,k}\boldsymbol{\epsilon}_{f,k}^T\right]\right]+\frac{a_{f,\max}^2p_{\max}^2a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}\right)^2}\mathrm{trace}\left[\mathrm{E}\left[\boldsymbol{\epsilon}_{h,k}\boldsymbol{\epsilon}_{h,k}^T\right]\right]. \tag{6.130}$$

Defining $\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$ *and* $\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$ *as the covariance of the statistical linearization error* $\boldsymbol{\epsilon}_{f,k}$ *and* $\boldsymbol{\epsilon}_{h,k}$*,*

$$\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}=\mathrm{E}\left[\boldsymbol{\epsilon}_{f,k}\boldsymbol{\epsilon}_{f,k}^T\right] \tag{6.131}$$
$$\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}=\mathrm{E}\left[\boldsymbol{\epsilon}_{h,k}\boldsymbol{\epsilon}_{h,k}^T\right], \tag{6.132}$$

we can write the above equation as

$$\begin{aligned}\mathrm{E}\left[\left[\boldsymbol{\epsilon}_k^t\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^t\right]&\leq\frac{1}{p_{\min}}\mathrm{trace}\left[\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right]+\frac{a_{f,\max}^2p_{\max}^2a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}\right)^2}\mathrm{trace}\left[\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right]\\&\leq\frac{1}{p_{\min}}p_{\boldsymbol{\epsilon}_f,\max}p_{\boldsymbol{\epsilon}_f,\mathrm{row}}+\frac{a_{f,\max}^2p_{\max}^2a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}\right)^2}p_{\boldsymbol{\epsilon}_h,\max}p_{\boldsymbol{\epsilon}_h,\mathrm{row}}\\&\leq t_4,\end{aligned} \tag{6.133}$$

where

$$\mathrm{trace}\left[\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}\right]\leq p_{\boldsymbol{\epsilon}_f,\max}p_{\boldsymbol{\epsilon}_f,\mathrm{row}},\quad p_{\boldsymbol{\epsilon}_f,\mathrm{row}}=\textit{Number of rows in}\boldsymbol{P}_{\boldsymbol{\epsilon}_f,k}$$
$$\mathrm{trace}\left[\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}\right]\leq p_{\boldsymbol{\epsilon}_h,\max}p_{\boldsymbol{\epsilon}_h,\mathrm{row}},\quad p_{\boldsymbol{\epsilon}_h,\mathrm{row}}=\textit{Number of rows in}\boldsymbol{P}_{\boldsymbol{\epsilon}_h,k}$$

and the positive real number t_4 *is defined as*

$$t_4=\frac{1}{p_{\min}}p_{\boldsymbol{\epsilon}_f,\max}p_{\boldsymbol{\epsilon}_f,\mathrm{row}}+\frac{a_{f,\max}^2p_{\max}^2a_{h,\max}^2}{p_{\min}\left(p_{\boldsymbol{\epsilon}_h,\min}+r_{\min}\right)^2}p_{\boldsymbol{\epsilon}_h,\max}p_{\boldsymbol{\epsilon}_h,\mathrm{row}}. \tag{6.134}$$

This proves Lemma 6.4.7.

Now going back to Equation (6.67),

$$\begin{aligned}\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right)=\boldsymbol{e}_k^T\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T\boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\\\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\left[2\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k\right]+\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^{\mathrm{t}}+\\2\left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\left[\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]+\left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^{\mathrm{t}},\end{aligned} \tag{6.135}$$

we can substitute the R.H.S. terms with the results obtained from the Lemmas 6.4.2-6.4.7,

$$\begin{aligned}\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right) &= (1+\epsilon)\,\boldsymbol{e}_k^T\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)^T\boldsymbol{P}_{k+1}^{-1}\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\\ &\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\left[2\left(\boldsymbol{A}_{f,k}-\boldsymbol{K}_k\boldsymbol{A}_{h,k}\right)\boldsymbol{e}_k+\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]+\left(2+\frac{1}{\epsilon}\right)\left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^{\mathrm{t}}+\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^{\mathrm{t}}\\ &\leq (1-\alpha)\boldsymbol{e}_k^T\boldsymbol{P}_k^{-1}\boldsymbol{e}_k+t_2\left\|\boldsymbol{e}_k\right\|^3+\left(2+\frac{1}{\epsilon}\right)\left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^{\mathrm{t}}+\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^{\mathrm{t}}. \end{aligned} \tag{6.136}$$

Substituting $\boldsymbol{V}_k\left(\boldsymbol{e}_k\right)=\boldsymbol{e}_k^T\boldsymbol{P}_k^{-1}\boldsymbol{e}_k$, the expression of $\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right)$ can be denoted as

$$\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right)\leq(1-\alpha)\boldsymbol{V}_k\left(\boldsymbol{e}_k\right)+t_2\left\|\boldsymbol{e}_k\right\|^3+\left(2+\frac{1}{\epsilon}\right)\left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^{\mathrm{t}}+\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^{\mathrm{t}}. \tag{6.137}$$

Taking the expectation on both sides, we get

$$\begin{aligned}\mathrm{E}\left[\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right)|\boldsymbol{e}_k\right]\leq(1-\alpha)\,\boldsymbol{V}_k\left(\boldsymbol{e}_k\right)+t_2\left\|\boldsymbol{e}_k\right\|^3+\left(2+\frac{1}{\epsilon}\right)\mathrm{E}\left[\left[\boldsymbol{N}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{N}_k^{\mathrm{t}}\right]+\\ \mathrm{E}\left[\left[\boldsymbol{\epsilon}_k^{\mathrm{t}}\right]^T\boldsymbol{P}_{k+1}^{-1}\boldsymbol{\epsilon}_k^{\mathrm{t}}\right].\end{aligned} \tag{6.138}$$

Now we replace the two expectation terms from the R.H.S of Equation (6.138) with the results shown in Equations (6.122) and (6.133),

$$\mathrm{E}\left[\boldsymbol{V}_{k+1}\left(\boldsymbol{e}_{k+1}\right)|\boldsymbol{e}_k\right]\leq(1-\alpha)\,\boldsymbol{e}_k^T\boldsymbol{P}_k^{-1}\boldsymbol{e}_k+t_2\left\|\boldsymbol{e}_k\right\|^3+\left(2+\frac{1}{\epsilon}\right)t_3+t_4. \tag{6.139}$$

Assuming that the estimation error $\boldsymbol{e}_k$ is beta-distributed as $\boldsymbol{e}_k\in B(0,\delta)$ [137] and defining

$$\delta=\min\left(\delta_e,\frac{\alpha}{2p_{\max}t_2}\right) \tag{6.140}$$

we can derive

$$\begin{aligned}t_2\left\|\boldsymbol{e}_k\right\|^3&=t_2\left\|\boldsymbol{e}_k\right\|^2\left\|\boldsymbol{e}_k\right\|\\ &=t_2\frac{\alpha}{2p_{\max}t_2}\left\|\boldsymbol{e}_k\right\|^2.\end{aligned} \tag{6.141}$$

From the definition of $\boldsymbol{V}_k\left(\boldsymbol{e}_k\right)$ and defining

$$\boldsymbol{P}_k^{-1}\geq\frac{1}{p_{\max}}\boldsymbol{I},$$

we obtain

$$\begin{aligned}\boldsymbol{V}_k\left(\boldsymbol{e}_k\right)&=\boldsymbol{e}_k^T\boldsymbol{P}_k^{-1}\boldsymbol{e}_k\\ &\geq\frac{1}{p_{\max}}\left\|\boldsymbol{e}_k\right\|^2.\end{aligned} \tag{6.142}$$

Substituting the above expression of $\boldsymbol{V}_k(\boldsymbol{e}_k)$ into the Equation (6.141) to obtain,

$$t_2 \|\boldsymbol{e}_k\|^3 \leq \frac{\alpha}{2}\boldsymbol{V}_k(\boldsymbol{e}_k), \tag{6.143}$$

Now we substitute the above equation into (6.139) and simplify by introducing a new constant c, which is equal to

$$c = \left(2 + \frac{1}{\epsilon}\right) t_3 + t_4 > 0. \tag{6.144}$$

We can further simplify the Equation (6.139) as follows:

$$\begin{aligned}
\mathrm{E}\left[\boldsymbol{V}_{k+1}(\boldsymbol{e}_{k+1})\,|\boldsymbol{e}_k\right] &\leq \left(1 - \frac{\alpha}{2}\right)\boldsymbol{V}_k(\boldsymbol{e}_k) + c \\
\mathrm{E}\left[\boldsymbol{V}_{k+1}(\boldsymbol{e}_{k+1})\,|\boldsymbol{e}_k\right] - \boldsymbol{V}_k(\boldsymbol{e}_k) &\leq -\frac{\alpha}{2}\boldsymbol{V}_k(\boldsymbol{e}_k) + c \\
\mathrm{E}\left[\boldsymbol{V}_{k+1}(\boldsymbol{e}_{k+1})\,|\boldsymbol{e}_k\right] - \boldsymbol{V}_k(\boldsymbol{e}_k) &\leq -\beta\boldsymbol{V}_k(\boldsymbol{e}_k) + c,
\end{aligned} \tag{6.145}$$

where β is a constant defined as

$$0 < \beta = \frac{\alpha}{2} \leq 1. \tag{6.146}$$

From the definition, it can be demonstrated that the Lyapunov function $\boldsymbol{V}_k$ is bounded within

$$\begin{aligned}
\boldsymbol{V}_k(\boldsymbol{e}_k) &= \boldsymbol{e}_k^T \boldsymbol{P}_k^{-1}\boldsymbol{e}_k \\
\frac{1}{p_{\max}}\|\boldsymbol{e}_k\|^2 \leq \boldsymbol{V}_k(\boldsymbol{e}_k) &\leq \frac{1}{p_{\min}}\|\boldsymbol{e}_k\|^2.
\end{aligned} \tag{6.147}$$

Hence we have shown that for the SPKF estimation error $\boldsymbol{e}_k$ at each k, the stochastic process denoted by the Lyapunov function $\boldsymbol{V}_k(\boldsymbol{e}_k)$ follows the criteria

$$\frac{1}{p_{\max}}\|\boldsymbol{e}_k\|^2 \leq \boldsymbol{V}_k(\boldsymbol{e}_k) \leq \frac{1}{p_{\min}}\|\boldsymbol{e}_k\|^2 \tag{6.148}$$

$$\mathrm{E}\left[\boldsymbol{V}_{k+1}(\boldsymbol{e}_{k+1})\,|\boldsymbol{e}_k\right] - \boldsymbol{V}_k(\boldsymbol{e}_k) \leq -\beta\boldsymbol{V}_k(\boldsymbol{e}_k) + c, \tag{6.149}$$

where $p_{\max}, p_{\min}, c > 0$ and $0 < \beta = \frac{\alpha}{2} \leq 1$. Assuming the validity of the above conditions, we can apply the Lemma 6.4.1 to prove that the SPKF estimation error $\boldsymbol{e}_k$ is exponentially upper bounded in MSE sense at each discrete time index k as follows:

$$\mathrm{E}\left[\|\boldsymbol{e}_k\|^2\right] \leq \frac{p_{\max}}{p_{\min}}\mathrm{E}\left[\|\boldsymbol{e}_0\|^2\right](1-\beta)^k + c p_{\max}\sum_{i=1}^{k-1}(1-\beta)^i. \tag{6.150}$$

If we compare the above equation with the general error-convergence Equation (6.39), we find that they are identical if we denote

$$c_1 = \frac{1}{p_{\max}} \tag{6.151}$$

$$c_2 = \frac{1}{p_{\min}} \tag{6.152}$$

$$c_3 = c \tag{6.153}$$

$$c_4 = \beta. \tag{6.154}$$

The Lyapunov upper bound expression is a function of numerous parameters characterized by the statistically linearized dynamic model, initial estimation error and the upper and lower bounds of the estimation error covariance. For a specific application, these are computed by collecting an extensive set of training data and then running a SPKF to generate the estimates. Unfortunately, as we cannot completely get rid of the estimator in the calculation of the filter upper bound, it serves as a drawback for the proposed method. Another disadvantage is that the training data should adequately represent the unseen observations and hence care should be taken to collect it in an exhaustive manner. Hence the next research question should be how to apply the upper bound in practical examples that does not require data driven simulations. This is still an open topic and could prove to be an excellent research project.

The Equation (6.150) demonstrates that the estimation error $\boldsymbol{e}_k$ exponentially converges to a steady state in the MSE sense provided certain conditions on the estimator performance and state space model are met. These conditions, which are shown in Assumption 6.4.1 specify that the error covariance, noise terms, initialization error and certain state space parameters are upper and lower bounded. Recall, the Assumption 6.4.1 computes the lower and upper bounds of those parameters using 95% confidence interval estimates, which denote that the true value lies within those bounds with 95% probability. Note, the rate of convergence of Equation (6.150) is governed by the term β and the ratio of $p_{\max}$ and $p_{\min}$. Higher value of β and lower ratio of $\frac{p_{\max}}{p_{\min}}$ ensure faster convergence. In the experimental result section, we demonstrate two different simulated examples where the norm of the estimation error generated by the SPKF over 200 Monte-Carlo (MC) runs

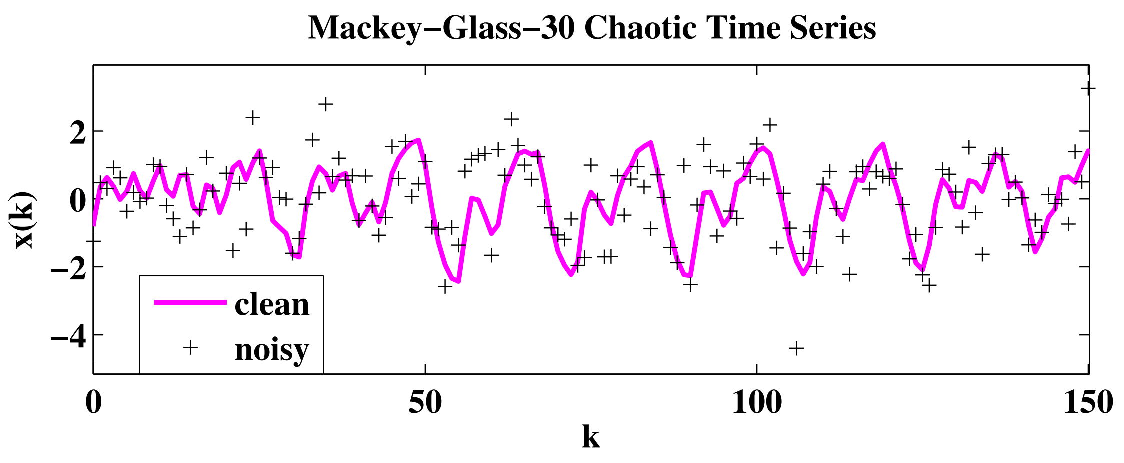


Figure 6.1: Clean and noisy Mackey-Glass time series data.

is shown to be upper bounded by the Equation (6.150).

6.5 Numerical Simulations

We evaluate our derived lower and upper bound expressions in the following scenarios:

1. Estimation of an underlying clean Mackey-Glass chaotic time series corrupted by an additive white Gaussian noise.

2. Tracking a space vehicle when it re-enters into the earth's atmosphere at a high altitude and with a high speed.

The same two examples were also used to evaluate the performance of the FBSL-SPKS and RTSSL-SPKS as demonstrated in chapter 2. Here we provide an outline of the above examples, details can be found in Section 2.5.

6.5.1 Mackey-Glass clean time series estimation

State Space Representation

In this example, the objective is to estimate the clean Mackey-Glass-30 chaotic time series which is corrupted by an additive white Gaussian noise (SNR $= 0\,$db). The clean time series is modeled as a parameterized function f by training a 6-5-1 (input-hidden-output) nodes feed-forward neural network. The M element state vector is denoted as $\boldsymbol{x}_k$, where

$$\boldsymbol{x}_k = \begin{bmatrix} x_{k-1} & x_{k-2} & \ldots & x_{k-M} \end{bmatrix}. \tag{6.155}$$

The state space configuration of the above problem is defined as:

$$\boldsymbol{x}_{k+1} = f\left(\boldsymbol{x}_k; \boldsymbol{w}\right) + \boldsymbol{G}_{f,k} v_k \tag{6.156}$$

$$z_k = \boldsymbol{H}_k \boldsymbol{x}_k + n_k, \tag{6.157}$$

which can be expanded as

- *Process Model*:

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ x_k \\ \vdots \\ x_{k-M+2} \end{bmatrix} = \begin{bmatrix} f\left(x_k, x_{k-1}, \cdots, x_{k-M+1}; \boldsymbol{w}\right) \\ \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ x_{k-1} \\ \vdots \\ x_{k-M+1} \end{bmatrix} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} v_k \tag{6.158}$$

- *Observation Model*:

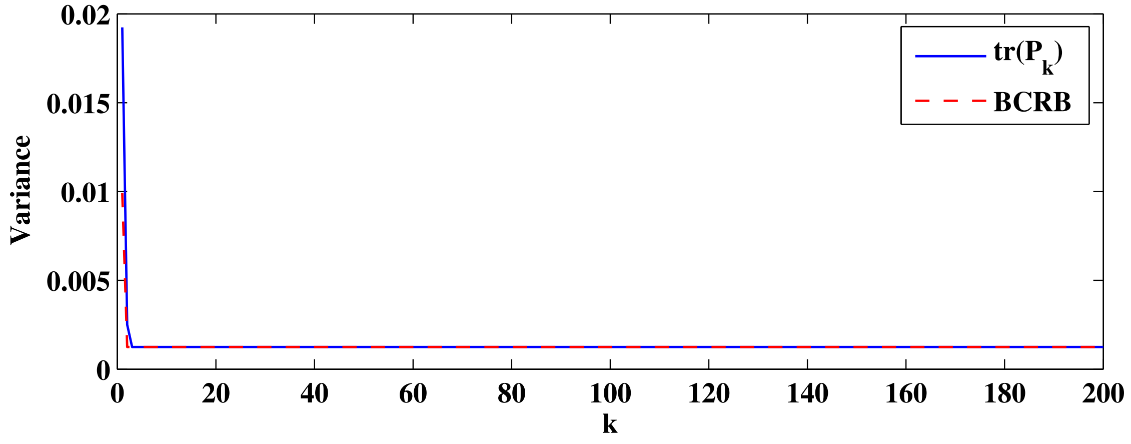
$$z_k = \begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix} \boldsymbol{x}_k + n_k, \tag{6.159}$$

where the process noise v_k is Gaussian distributed with zero mean and covariance σ_v^2, z_k is the noise corrupted time series and n_k is the measurement noise.

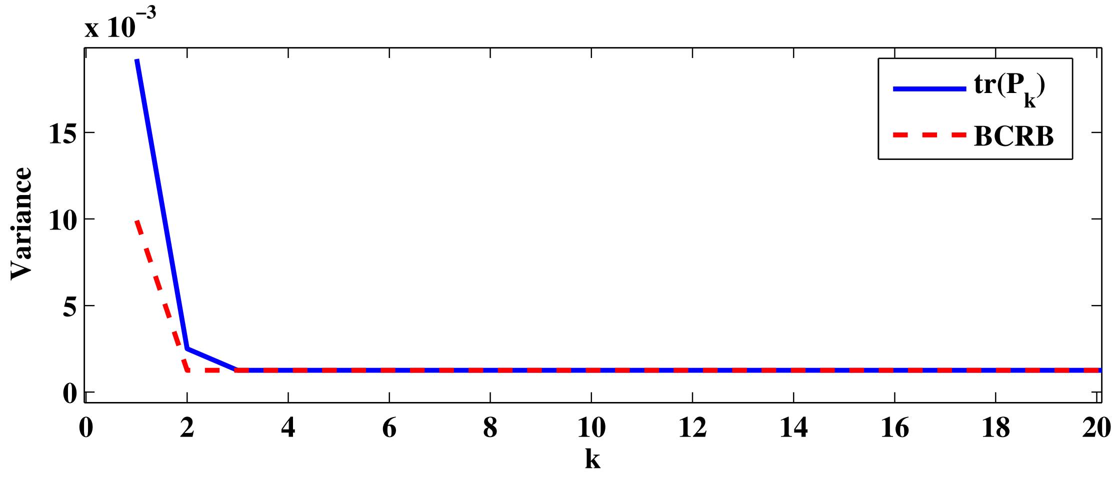
6.5.2 Vehicle Re-entry Tracking

In this case, the task involves to track a space vehicle that re-enters into the earth's atmosphere at a high altitude and with a significantly large velocity. A radar is used to measure the range and the bearing of the vehicle from the earth's surface. While entering, the vehicle is under the influence of strong nonlinear forces such as aerodynamic drag and the earth's gravity, which are functions of the vehicle's position, velocity and altitude.

The vehicle's state ($\boldsymbol{x}_k$) consists of its 2D position (x_k and y_k), 2D velocity (v_{x_k} and v_{y_k}) and a scaler parameter of aerodynamic drag (d_k). The vehicle's process model and the observation model are described below:



(a)



(b)

Figure 6.2: Demonstration of the lower error bound for the SPKF in the Mackey-Glass time series example. (a): The trace of the estimation error covariance is plotted against the derived BCRB, (b): This plot *zooms in* on a section of plot (a) in order to demonstrate that the SPKF estimation error is lower bounded by the BCRB.

- *Process Model*:

$$x_{k+1} = x_k + \delta T v_{x_k} \tag{6.160}$$

$$y_{k+1} = y_k + \delta T v_{y_k} \tag{6.161}$$

$$v_{x_{k+1}} = \left(1 + \delta T \boldsymbol{D}_k^{\mathrm{dr}}\right) v_{x_k} + \delta T \boldsymbol{G}_k^{\mathrm{g}} x_k + v_{p_{x,k}} \tag{6.162}$$

$$v_{y_{k+1}} = \left(1 + \delta T \boldsymbol{D}_k^{\mathrm{dr}}\right) v_{y_k} + \delta T \boldsymbol{G}_k^{\mathrm{g}} y_k + v_{p_{y,k}} \tag{6.163}$$

$$d_{k+1} = d_k + v_{p_{d,k}} \tag{6.164}$$

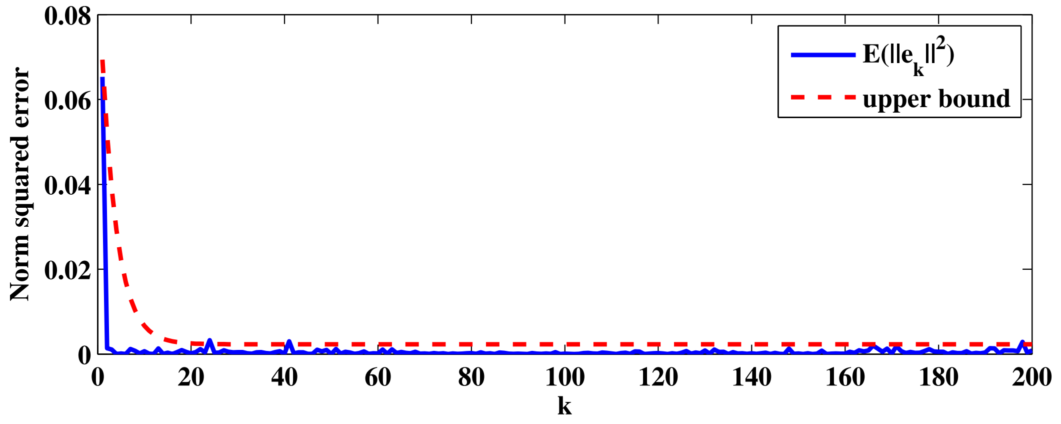


Figure 6.3: Demonstration of the upper error bound for the SPKF in the Mackey-Glass time series example. The trace of the estimation error covariance is plotted against the derived Lyapunov function based upper bound.

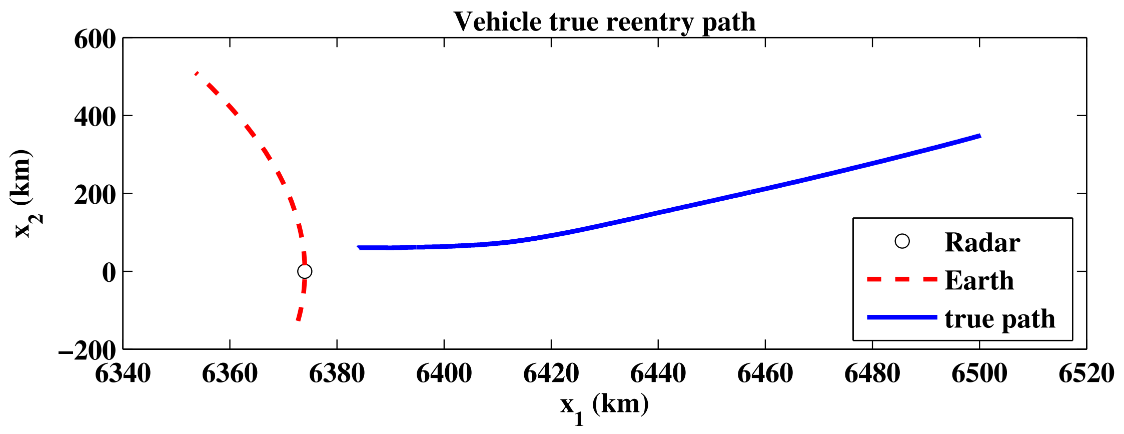


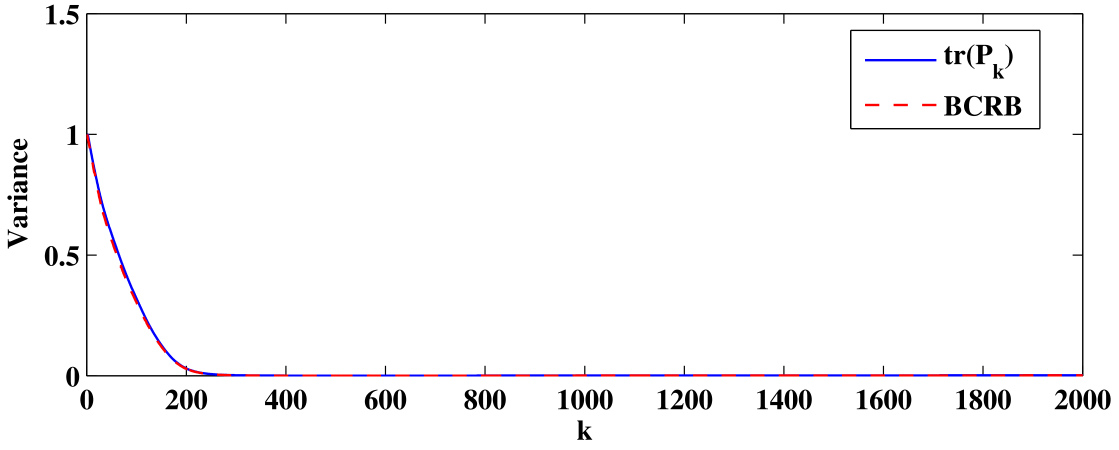
Figure 6.4: True vehicle trajectory. The solid line is the vehicle trajectory and the dashed line is the earth's surface. The radar is placed at 'o'.

where $\boldsymbol{D}_k^{\mathrm{dr}}$ and $\boldsymbol{G}_k^{\mathrm{g}}$ are the aerodynamic drag related force term and the gravity related force term respectively at each discrete time k. The integration time is denoted as δT. For further details, refer to Section 2.5.1.

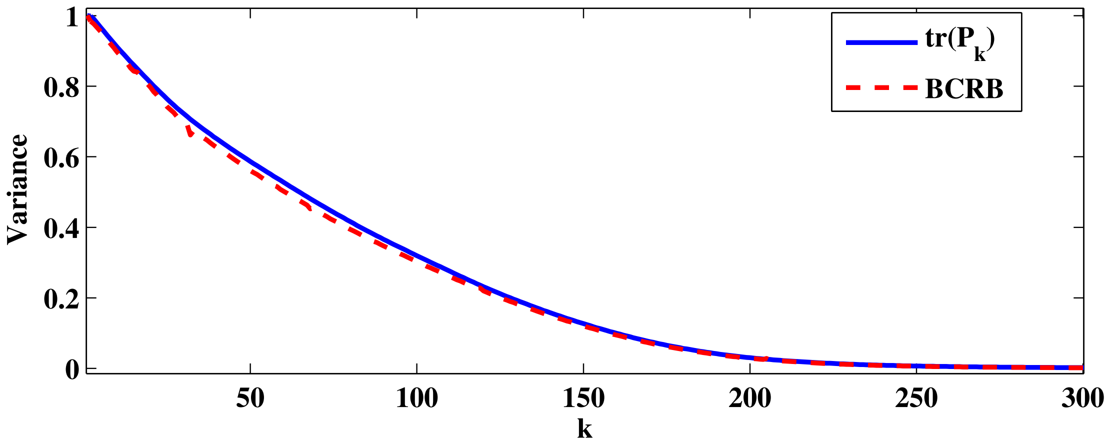
- *Observation Model*:

$$r_k = \sqrt{(x_{1,k} - x_{\mathrm{r}})^2 + (x_{2,k} - y_{\mathrm{r}})^2} + n_{1,k} \tag{6.165}$$

$$\theta_k = \arctan\left(\frac{x_{2,k} - y_{\mathrm{r}}}{x_{1,k} - x_{\mathrm{r}}}\right) + n_{2,k}, \tag{6.166}$$



(a)



(b)

Figure 6.5: Demonstration of the lower error bound for the SPKF in the vehicle re-entry tracking example. (a): The trace of the estimation error covariance is plotted against the derived BCRB, (b): This plot *zooms in* on a section of plot (a) in order to demonstrate that the SPKF estimation error is lower bounded by the BCRB.

where the measurement $\boldsymbol{z}_k = \begin{bmatrix} r_k & \theta_k \end{bmatrix}$ consists of both range and bearing. The radar is assumed to be located at $\begin{bmatrix} x_\mathrm{r} & y_\mathrm{r} \end{bmatrix}$. The measurement sampling rate is taken as 10 Hz. The observation noise $\boldsymbol{n}_k$ is uncorrelated zero mean white.

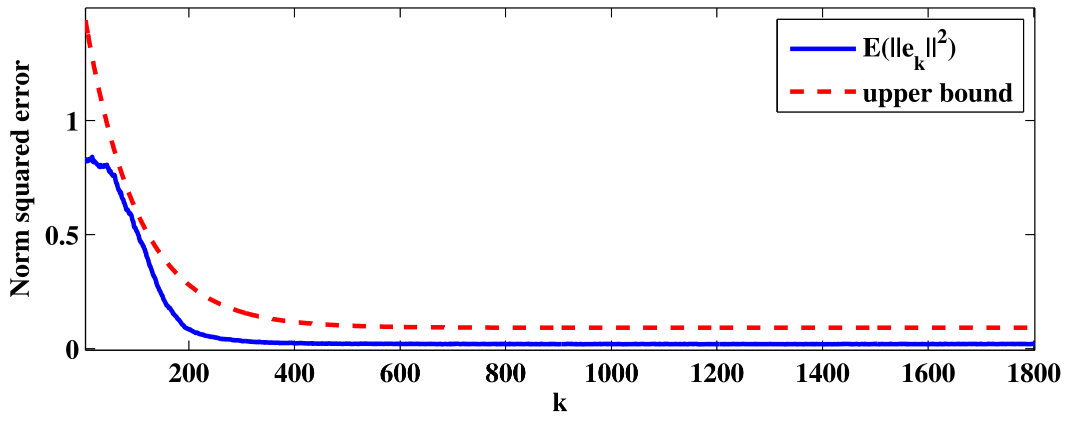


Figure 6.6: Demonstration of the upper error bound for the SPKF in the vehicle re-entry tracking example. The trace of the estimation error covariance is plotted against the derived Lyapunov function based upper bound.

6.5.3 Experimental Results

In this section, we demonstrate how our derived lower and upper bounds compare against the estimation error $\boldsymbol{e}_k$ of the SPKF for the Mackey-Glass time series example and the vehicle re-entry tracking example. In each case, the SPKF is employed to estimate the unknown states and the estimation error is computed by the difference between the true and estimated states. Note that for each example, the estimation error plotted in the figures are generated by ensemble averaging of the 200 randomly initialized Monte-Carlo (MC) runs. For each MC run, a different realization of both process and observation noises was generated.

The clean and noisy Mackey-Glass time series is displayed in Figure 6.1. Figure 6.2(a)-6.2(b) compares the trace of the estimation error covariance of the SPKF with the BCRB. As shown in Figure 6.2(a), the derived BCRB is so close at the trace of $\boldsymbol{P}_k$, we cannot distinguish each other. Hence we use Figure 6.2(b) in order to zoom in a section of Figure 6.2(a), which clearly separates the BCRB bound from $\boldsymbol{P}_k$. Both of the figures confirm that the BCRB demonstrated in Equation (6.29) indeed actually lower bounds the $\boldsymbol{e}_k$. Figure 6.3 demonstrates the estimation error upper bound for the SPKF in the Mackey-Glass time series example. In this example, the convergence parameter values i.e. β and the ratio of $p_{\max}$ and $p_{\min}$ are exclusively computed from the training data and are found as

$$\beta = 0.262 \tag{6.167}$$

$$\frac{p_{\max}}{p_{\min}} = 0.094 \tag{6.168}$$

As clearly seen in the figure, the derived Lyapunov bound acts as an upper bound for the expected norm of the estimation error $\boldsymbol{e}_k$ in the Mackey-Glass time series example.

We also illustrate the results of the experiment that we performed in the vehicle re-entry tracking case in order to verify the lower and upper bounds against the estimation error $\boldsymbol{e}_k$ of the SPKF. Figure 6.4 displays the true vehicle re-entry path and the position of the radar at the earth's surface. Figure 6.5(a) demonstrates the BCRB bound against the trace of the estimation error covariance $\boldsymbol{P}_k$. Similar to the Mackey-Glass example, a zoomed segment is shown in Figure 6.5(b) in order to distinguish the BCRB error bound from $\boldsymbol{P}_k$. As demonstrated in the figures, the SPKF estimation error covariance is lower bounded by the derived BCRB. Figure 6.6 illustrates the estimator upper bound against the norm of the filter estimation error $\boldsymbol{e}_k$. For this example, the convergence parameters are as follows:

$$\beta = 0.02 \tag{6.169}$$

$$\frac{p_{\max}}{p_{\min}} = 1.5 \tag{6.170}$$

The Lyapunov function based stochastic upper bound as shown in Equation (6.150) clearly captures the filter estimation error. Note that in the vehicle re-entry tracking case, the value of β and $\frac{p_{\max}}{p_{\min}}$ is lower and higher than that in the Mackey-Glass time series estimation case. Small β and high ratio of $p_{\max}$ and $p_{\min}$ may increase the convergence time of the upper bound. Figure 6.6 confirms the validity of the above finding. As seen in the plot, the convergence rate of the upper bound is significantly slower here than that of the Mackey-Glass case. This is primarily due to higher uncertainty of the initial vehicle state, which correlates to higher estimation error covariance $p_{\max}$ and severe nonlinearity exhibited by the system dynamics, which causes higher statistical linearization error $p_{\boldsymbol{\epsilon}_f}$ and $p_{\boldsymbol{\epsilon}_h}$. With the above two examples, we have achieved demonstrating how the derived error bounds quantify the estimation performance of the SPKF.

6.6 Discussion And Future Work

In this work, we have analyzed the estimation error behavior for the SPKF operating on a discrete time nonlinear dynamical system. In this context, step by step derivations of the lower and upper error bounds for the SPKF based estimators are presented. Although the SPKF enjoys greater success than the EKF in estimating hidden states for the nonlinear filtering problem, the analytical convergence analysis for the SPKF has mostly been ignored. We have derived both the lower and upper bound in terms of the estimation error of the filter, i.e. MSE between the true and the estimated state. Our derivation of the SPKF lower bound is based on the Van Trees version of the BCRB and its value at time k is shown to be equal to the inverse of the BIM $\boldsymbol{J}_k$. The recursive formulation of the BIM from time k to $k+1$ extends the work of Tichavský, who has derived a general expression of $\boldsymbol{J}_k$ for a nonlinear dynamical model. The derivation of the SPKF error upper bound follows the work of Reif *et al.* where we have demonstrated that the estimation error exponentially converges to a steady state. The validity of the derived lower and upper bounds are verified for the Mackey-Glass time series estimation problem and vehicle re-entry tracking problem. In each case, both the lower and upper error bounds are plotted against the estimation error of the SPKF and are shown that the estimation error lies within the above bounds.

Although the SPKF error bounds are derived analytically, several numerical approximation techniques are adopted to apply the error bounds in the practical examples. In order to formulate a closed form solution of the lower bound, one has to solve multidimensional expectations, which are analytically intractable for the nonlinear system. We have resolved the situation by using the sigma-point averaging strategy where the expectations are computed by weighted averaging over the extracted sigma points. The SPKF upper error bound is a function of the parameters involving the bound values of the estimation error covariance, system noise terms, state space model and the initial estimation error. As those parameters are not known in advance, numerical simulations are adopted and 95% confidence interval estimates are used as their values. The above approximations no doubt affect the accuracy of the error bound implementation in practical cases.

Furthermore, the training data should be collected in an exhaustive manner so that the confidence interval estimates are the true indicator of unknown parameters. Note, the approximations are needed as the process of analytically obtaining the parameter values is complex and challenging for the nonlinear system. Hence instead of devoting much time and effort on deriving the analytical expression of the parameters, we consider the numerical approximation as the first step in implementing the upper error bound in practical state estimation examples. Despite the practical limitations, the derived lower and upper error bounds are capable of analyzing the SPKF performance on a given application without applying the recursive filter equations. Most importantly this is the first attempt of deriving performance bounds for the SPKF based estimators, which can provide an excellent building block in pursuing further research. The future direction may include to apply the derived error bounds to some other state estimation problems and evaluate their performance. Future research may also extend our work by exploring the stability and observability properties of the nonlinear state space model to analytically formulate suitable bound parameters for the SPKF upper bound.

Chapter 7

Summary and Conclusions

In this chapter, we briefly summarize our work performed during the course of this dissertation.

7.1 Sigma-point Kalman Smoothers (SPKS)

In Chapter 2 the following new SPKF smoothing algorithms that include both the fixed-interval and fixed-lag methodologies are implemented and brought under a common family called *sigma-point Kalman smoothers* (SPKS).

- **Fixed-interval sigma-point Kalman smoother (FI-SPKS):**
    1. Forward-backward statistical linearized sigma-point Kalman smoother (FBSL-SPKS)
    2. Rauch-Tung-Striebel statistical linearized sigma-point Kalman smoother (RTSSL-SPKS)
- **Fixed-lag sigma-point Kalman smoother (FL-SPKS):**
    1. State-augmented sigma-point Kalman smoother (Aug-SPKS)
    2. Forward-backward a priori sigma-point Kalman smoother (FB-Priori-SPKS)
    3. Forward-backward statistical linearized sigma-point Kalman smoother (FBSL-SPKS)
    4. Rauch-Tung-Striebel statistical linearized sigma-point Kalman smoother (RTSSL-SPKS)

Both the FI-SPKS and FL-SPKS algorithms are derived from first principles by making use of the weighted statistical linear regression (WSLR) formulation of the nonlinear state dynamics. In addition, the computational complexity and memory of the FI-SPKS and FL-SPKS algorithms are analyzed. The tracking accuracy of the proposed FI-SPKS and FL-SPKS algorithms are experimentally verified using two benchmark examples: Mackey-Glass nonlinear time-series estimation and vehicle re-entry tracking. The performance of our proposed SPKS are also compared with the extended Kalman smoother (EKS) and other sigma-point smoothing approaches in terms of accuracy, computational efficiency and memory. From the results, the superiority of our estimators is clearly established.

7.2 Real-world Application of SPKS Algorithms: Unobtrusive Indoor Pedestrian Tracking

A novel SPKS based location tracking method that tracks an user in an indoor environment using unobtrusive sensors is successfully developed. We have investigated two variants of the tracking mechanism based on the requirement of carrying a receiver tag.

Chapter 3 proposes a tag-based indoor tracking system which uses RSSI as the primary sensor. The person(s) to be tracked carry a small body-borne device that periodically measures the RSSI at 3 or more standard Wi-Fi access points placed at pre-defined locations. The observation model of the tracker is generated from the RSSI calibration data by fitting nonlinear RBF maps between the known calibration locations and RSSI mean values. The RSSI observation maps are incorporated into the SPKS based tracking algorithm which combines the RSSI observations with a potential field based dynamic model. In addition to RSSI, the SPKS fuses the infra-red (IR) motion sensor and binary foot-switch measurements in the inference system. Furthermore the SPKS performs multi-rate processing, where state updates and RSSI observations occur at different rates, and handles time-varying observations. The performance of the proposed SPKS tracker is evaluated in a number of "living laboratories", where the tracking accuracy is demonstrated to be superior to the EKS and an available industry positioning engine developed by the Ekahau Inc.

Chapter 4 presents a novel tag-free solution that utilizes low cost ultrasonic transducers to track a person. Instead of wearing a body-borne receiver tag, the proposed system requires setting up multiple wall-mounted ultrasonic range sensors inside the indoor location. The active ultrasonic transducers transmit an ultrasonic wave and capture analog echoes, which are then digitized and analyzed in order to calculate the 1D range of the moving person. Signal processing techniques including Band-pass filtering, Hilbert transformation, background subtraction and clustering are used to remove interference from other static objects in the room. The range data from active and passive ultrasonic sensors are treated as observations in the SPKS based Bayesian framework to determine a person's 2D position and velocity. We adopt two different tracking procedures:

- Range-map approach, which estimates the user's state by generating RBF observation maps in the calibration phase between known calibration locations and 1D ranges.
- SLAM approach, which corresponds to simultaneously estimating the state of the person and the parameters of the observation model.

We further investigate two different SLAM approaches: TANS-SLAM and LANS-SLAM. The TANS-SLAM method still uses observation maps but the map parameters are updated during tracking using the newly generated state estimate. The LANS-SLAM method employs a dual estimation approach, which alternately uses one SPKS to localize the user given the current estimate of the observation model parameters and a second SPKS to update the parameters using the current user state. Parameters consist of the 2D locations of the ultrasonic modules and a time-varying term which corresponds to the effect of multipath, reflection/refraction on the speed of sound. The LANS-SLAM method is the most attractive choice for the ultrasonic sensor based indoor tracking because it requires minimal calibration, uses fewer number of parameters and demonstrates fast convergence starting from a rough estimate of sonar locations. The tracking accuracy of all our proposed methods are evaluated over a number of trials performed in a test-lab. It is demonstrated that the TANS-SLAM based tag-free system performs comparably with the tag-based highly accurate industry standard "Ubisense" positioning engine.

7.3 Real-world Application of SPKS Algorithms: Multiharmonic Frequency Tracking

In Chapter 5 we cover in detail how the SPKS based Bayesian inference algorithm can be implemented for tracking the phase, frequency, and amplitude of the fundamental frequency and the harmonic components present in a periodic signal. This work is done in collaboration with James McNames's research group at Portland State University. The performance of the SPKS multiharmonic tracker is compared with that of EKS using simulated signals and a photo sensor insect activity signal. It is clearly shown that the SPKS is significantly more accurate, converges faster to the true solution, and robust to noise than the EKS.

7.4 Estimation Error Bounds for SPKF

Chapter 6 analyzes the state estimation error behavior and presents the formulations of the lower and upper error bounds for the SPKF operating on a nonlinear discrete time system. The well-known Bayesian Cramér-Rao theory is used for the derivation of the lower bound. It is shown that the state estimation error covariance $\boldsymbol{P}_k$ for the nonlinear system can be lower bounded by the inverse of the Bayesian information matrix $\boldsymbol{J}_k$, which is a function of time k. We extend Tichavský's formulation to derive a recursive expression of $\boldsymbol{J}_k$. The derivation of the upper bound follows the work of Reif *et al.* to demonstrate that the expected value of the norm of state estimation error exponentially converges to a steady state. Step by step analytical derivations are shown to determine the final equations of the lower and upper bounds. To apply the error bounds in the practical examples, we adopt several numerical approximations which are clearly explained in the dissertation. The validity of the SPKF error bounds are shown using two benchmark examples: Mackey-Glass nonlinear time series estimation and vehicle re-entry tracking. In each case, it is demonstrated that the state estimation error lies within the derived lower and upper bounds.

Bibliography

[1] M. Grewal, L. R. Weil, and A. P. Andrews, *Global Positioning Systems, Intertial Navigation and Integration*, 1st ed. John Wiley and Sons, 2001.

[2] J. Kim and S. Sukkarieh, "Autonomous Airborne Navigation in Unknown Terrain Environments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, pp. 1031–1045, 2004.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics.* The MIT Press, 2005.

[4] G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 229–241, June 2001.

[5] J. Langelaan and S. Rock, "Navigation of Small UAVs Operating in Forests," in *Proceedings of AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2004.

[6] Y. Bar-Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation.* Wiley InterScience, 2001.

[7] G. Goodwin and K. Sin, *Adaptive Filtering, Prediction and Control.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[8] A. Gelb, Ed., *Applied Optimal Estimation.* Cambridge, MA: MIT Press, 1984.

[9] J. Mendel, *Lessions in Digital Estimation Theory.* Englewood Cliffs, NJ: Prentice-Hall, 1986.

[10] M. Safonov, *Stability and Robustness of Multivariable Feedback Systems.* Cambridge, MA: MIT Press, 1980.

[11] D. Simon, *Optimal State Estimation.* Wiley InterScience, 2006.

[12] F. Lewis, *Optimal Estimation with an Introduction to Stochastic Control Theory.* John Wiley and Sons, 1986.

[13] G. Puskorius and L. Feldkamp, “Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks,” *IEEE Transactions on Neural Networks*, vol. 5, pp. 279–297, 1994.

[14] J. McNames and M. Aboy, “Statistical modeling of cardiovascular signals and parameter estimation based on the extended Kalman filter,” *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 1, pp. 119–129, January 2008.

[15] S. Kim and J. McNames, “Tracking intermittent tremor frequency with a particle filter,” in *Proceedings of IEEE/SP workshop on Statistical Signal Processing*, August 2007, pp. 171–175.

[16] S. Kim, A. Paul, E. Wan, and J. McNames, “Multiharmonic tracking using sigma-point Kalman filter,” in *Proceedings of the International Conference of IEEE Engineering in Medicine and Biology (EMBS)*, 2008.

[17] R. van der Merwe, “Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models,” Ph.D. dissertation, OGI School of Science and Engineering at Oregon Health & Science University (OHSU), 2004.

[18] R. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” *Transactions of the ASME, Journal of Basic Engineering*, vol. 82, pp. 35–45, March 1960.

[19] A. Nelson, “Nonlinear Estimation and Modeling of Noisy Time Series By Dual Kalman Filtering Methods,” Ph.D. dissertation, OGI School of Science and Engineering, 2000.

[20] S. Haykin, Ed., *Kalman Filtering and Neural Networks*, ser. Adaptive and Learning Systems for Signal Processing, Communications and Control. Wiley InterScience, 2001.

[21] A. Jazwinsky, *Stochastic Processes and Filtering Theory.* Academic Press, 1970.

[22] S. Julier, J. Uhlmann, and H. Durrant-Whyte, "A new approach for filtering nonlinear systems," in *Proceedings of American Control Conference*, 1995, pp. 1628–1632.

[23] S. J. Julier and J. K. Uhlmann, "Unscented Filtering and Nonlinear Estimation," *Proceedings of the IEEE*, vol. 92, pp. 401–422, March 2004.

[24] K. Ito and K. Xiong, "Gaussian Filters for Nonlinear Filtering Problems," *IEEE Transactions on Automatic Control*, vol. 45, no. 5, pp. 910–927, May 2000.

[25] M. Norgaard, N. Poulsen, and O. Ravn, "New Developments in State Estimation for Nonlinear Systems," *Automatica*, vol. 36, pp. 1627–1638, November 2000.

[26] E. Wan and R. van der Merwe, *Kalman Filtering and Neural Networks*, 1st ed. John Wiley & Sons, 2001, ch. 7, pp. 221–280.

[27] R. van der Merwe and E. A. Wan, "The Square-Root Unscented Kalman Filter for State and Parameter Estimation," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 6, Salt Lake City, Utah, May 2001, pp. 3461–3464.

[28] T. Lefebvre, H. Bruyninckx, and J. D. Schutter, "Comment on A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators," *IEEE Transactions on Automatic Control*, vol. 47, no. 8, pp. 1406 – 1409, August 2002.

[29] D. Fraser and J. Potter, "The optimum linear smoother as a combination of two optimum linear filters," *IEEE Transactions on Automatic Control*, vol. 14, no. 4, pp. 387–390, 1969.

[30] H. Rauch, F. Tung, and C. Striebel, "Maximum Likelihood Estimates of Linear Dynamic Systems," *AIAA*, vol. 3, no. 8, pp. 1445–1450, 1965.

[31] C. Leondes, J. Peller, and E. Stear, "Nonlinear Smoothing Theory," *IEEE Transactions on System, Science and Cybernetics*, vol. 6, no. 1, pp. 63–71, January 1970.

[32] S. Sarkka, "Unscented Rauch-Tung-Striebel Smoother," *IEEE Transactions on Automatic Control*, vol. 53, no. 3, pp. 845–849, 2007.

[33] A. E. Bryson and M. Frazier, "Smoothing for linear and nonlinear dynamic systems," Aerospace Systems Division, Wright-Patterson Air Force Base, Tech. Rep. TDR 63-119, 1963.

[34] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation.* Prentice Hall, 2000.

[35] A. S. Paul and E. A. Wan, "A New Formulation For Nonlinear Forward-Backward Smoothing," in *Proceedings of International Conference on Acoustics, Specch and Signal Processing(ICASSP)*, 2008.

[36] R. van der Merwe and E. Wan, "Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models," in *Proceedings of Workshop on Advances in Machine Learning*, June 2003.

[37] P. Costa, "Adaptive model architecture and extended kalman-bucy filters," *IEEE Transaction on Aerospace and Electronic systems*, vol. 30, pp. 525–533, April 1994.

[38] R. Mehra, "A comparison of several nonlinear filters for reentry vehicle tracking," *IEEE Transaction on Automatic Control*, vol. AC-16, pp. 307–319, August 1971.

[39] J. Austin and C. Leondes, "Statistically linearized estimation of reentry trajectories," *IEEE Transaction on Aerospace and Electronic System*, vol. AES-17, pp. 54–61, January 1981.

[40] R. Want, A. Hopper, V. Falcao, and J. Gibbons, "The Active Badge Location System," *ACM Transactions on Information Systems*, vol. 40, no. 1, pp. 91–102, January 1992.

[41] P. Bahl and V. Padmanabhan, "RADAR: An In-Bulding RF-Based User Location and Tracking System," in *Proceedings of IEEE Infocomm 00*, April 2000.

[42] N. Priyantha, A. Charraborty, and H. Balakrishnan, "The Cricket Location-support System," in *Proceedings of ACM Mobicomm 00*, July 2000.

[43] J. Hightower and G. Borriello, "Particle filters for location estimation in ubiquitious computing: A case study," in *Proceedings of International Conference on Ubiquitious Computing (UBICOMP)*, 2004.

[44] [Online]. Available: http://www.sonitor.com/

[45] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert5, P. Powledge, G. Borriello, and B. Schilit, "Place Lab: Device Positioning Using Radio Beacons in the Wild," in *Proceedings of International Conference on Pervasive Computing (Pervasive)*, 2005.

[46] Y. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm, "Accuracy Characterization for Metropolitan-scale Wi-Fi Localization," in *Proceedings of 3rd international conference on Mobile systems, applications, and services*, 2005, pp. 233–245.

[47] M. Youssef and A. Agrawala, "The Horus WLAN Location Determination System," in *Proceedings of Third International Conference Mobile Systems, Applications and Services*, 2005, pp. 205–218.

[48] [Online]. Available: http://www.ekahau.com/

[49] [Online]. Available: http://www.ubisense.net/

[50] X. Li and K. Pahlavan, "Super-resolution TOA Estimation with Diversity for Indoor Geolocation," *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, pp. 224–234, 2004.

[51] A. Kushki, K. Plataniotis, and A. Venetsanopoulos, "Kernel-Based Positioning in Wireless Local Area Networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 689–705, June 2007.

[52] E. Elnahrawy, X. Li, and R. Martin, "The limits of localization using signal strength: A comparative study," in *Proceedings of the 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, October 2004, pp. 406–414.

[53] N. Patwari, J. Ash, S. Kyperountas, A. H. III, R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, 2005.

[54] C. Wang and X. Li, "Sensor localization under limited measurement capabilities," *IEEE Networks*, vol. 21, no. 3, pp. 16–23, 2007.

[55] V. Kaseva, M. Kohvakka, M. Kuorilehto, M. Hannikainen, and T. Hamalainen, "A wireless sensor network for RF-based indoor localization," *EURASIP Journel on Advances in Signal Processing*, vol. 731835, pp. 1–27, 2008.

[56] A. Smailagic and D. Kogan, "Location sensing and privacy in a context-aware computing environment," *IEEE Wireless Communications*, vol. 9, no. 5, pp. 10–17, 2002.

[57] T. Hodes, R. Katz, E. Servan-Schreiber, and L. Rowe, "Composable ad-hoc mobile services for universal interaction," in *Proceedings of the 3rd Annual International Conference on Mobile Computing and Networking (MOBICOM 97)*, September 1997.

[58] J. Pan, J. Kwok, Q. Yang, and Y. Chen, "Accurate and Low-Cost Location Estimation Using Kernels," in *Proceedings of International Joint Conference on Artificial Intelligence*, 2005, pp. 1366–1370.

[59] K. Kaemarungsi and P. Krishnamurthy, "Modeling of Indoor Positioing Systems Based On Location Fingerprinting," in *Proceedings of INFOCOM*, vol. 2, 2004, pp. 1012–1022.

[60] T. Roos, P. Myllymki, H. Tirri, P. Misikangas, and J. Sievnen, "A Probabilistic Approach to WLAN User Location Estimation," *International Journal on Wireless Information Networks*, vol. 9, no. 3, pp. 155–164, 2002.

[61] A. Ladd, K. Bekris, A. Rudys, G. Marceau, L. Kavraki, and D. Wallach, "Robotics Based Location Sensing Using Wireless Ethernet," in *Proceedings Mobicom*, 2002, pp. 227–238.

[62] M. Youssef, A. Agrawala, and A. Shankar, "WLAN location determination via clustering and probability distributions," in *Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications (Percom 2003)*, September 2003, pp. 143–150.

[63] K. Lorincz and M. Welsh, "Motetrack:a robust decentralized approach to RF-based location tracking," *IEEE Transaction on Pervasive and Ubiquitous Computing*, vol. 11, pp. 489–503, 2007.

[64] Z. Xiang, S. Song, J. Chen, H. Wang, J. Huang, and X. Gao, "A wireless LAN-based indoor positioning technology," *IBM Journal Research and Development*, vol. 48, no. 5/6, pp. 617–626, 2004.

[65] R. Singh, L. Macchi, C. Regazzoni, and K. Plataniotis, "A Statistical Modelling Based Location Determination Method Using Fusion in WLAN," in *Proceedings of International Workshop on Wireless Ad-Hoc Networks*, 2005.

[66] P. Barsocchi, S. Lenzi, S. Chessa, and G. Giunta, "A novel approach to indoor RSSI localization by automatic calibration of the wireless propagation model," in *Proceedings of the IEEE Vehicular Technology Conference (VTC)*, April 2009.

[67] A. Goldsmith, *Wireless Communications.* Cambridge University Press, 2005.

[68] D. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello, "Bayesian Filtering for Location Estimation," *IEEE pervasive computing*, vol. 2, no. 3, pp. 24–33, July-September 2003.

[69] A. Elfes, E. Prassler, and J. Scholz, "Tracking people in a railway station during rush hour," in *Proceedings of computer vision system, ICVS 99*, 1999.

[70] A. Fod, A. Howard, and M. Mataric, "Laser-based people tracking," in *Proceedings of the IEEE International Conference on Robotics & Automation*, 2002.

[71] C.-T. Hsieh, C.-Y. Chen, and Y.-K. Wu, "People Tracking System using Kalman Filter," in *Proceedings of Signal and image processing*, 2002.

[72] J. Krumm and E. Horvitz, “Locadio:Inferring motion and location from Wi-Fi signal strengths,” in *Proceedings of International Conference onMobile and Ubiquitious Systems:Networking and Services(MobiQuitous 04)*, 2004.

[73] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. Nordlund, “Particle Filters for Positioning, Navigation and Tracking,” *IEEE Transactions on Signal Processing*, vol. 50, pp. 425–435, 2002.

[74] B. Ferris, D. Haehnel, and D. Fox, “Gaussian Processes for Signal Strength-Based Location Estimation,” in *In Proc. of Robotics Science and Systems*, 2006.

[75] J. Letchner, D. Fox, and A. LaMarce, “Large-Scale Localization from Wireless Signal Strength,” in *Proceeding of the National Conference on Artificial Intelligence (AAAI-05)*, 2005, pp. 15–20.

[76] D. Hhnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose, “Mapping and Localization with RFID Technology,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

[77] Q. Meng, Y. Sun, and Z. Cao, “Adaptive extended Kalman filter (AEKF)-based mobile robot localization using sonar,” *Robotica*, vol. 18, pp. 459–473, 2000.

[78] J. Leonard and H. Durrant-Whyte, “Mobile Robot Localization by Tracking Geometric Beacons,” *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 376–382, June 1991.

[79] S. Seidal and T. Rappaport, “914 MHz path loss prediction models for indoor wireless communications in multifloored buildings,” *IEEE Transactions on Antennas and Propagation*, vol. 40, no. 2, pp. 207–217, 1992.

[80] A. LaMarca, J. Hightower, I. Smith, and S. Consolvo, “Self-mapping in 802.11 location systems,” in *Proceedings of International Conference on Ubiquitous Computing (UbiComp)*, 2005.

[81] A. Schwaighofer, M. Grigoras, V. Tresp, and C. Hoffmann, "GPPS: A Gaussian process positioning system for cellular networks," in *Proceedings of Advances In Neural Information Processing Systems (NIPS)*, 2003.

[82] E. A. Wan and R. van der Merwe, "The Unscented Kalman Filter for Nonlinear Estimation," in *Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC)*, Octobor 2000.

[83] A. Paul and E. Wan, "RSSI based indoor localization and tracking using sigma-point Kalman smoothers," *IEEE Journal on Special Topics in Signal Processing: Special Issue on Advanced Signal Processing for GNSS and Robust Navigation*, vol. 3, no. 5, pp. 860–873, October 2009.

[84] A. S. Paul and E. A. Wan, "Wi-Fi Based Indoor Localization and Tracking Using Sigma-Point Kaman Filtering Methods," in *Proceedings of IEEE/ION Position Location and Navigation Symposium 2008 (PLANS 2008)*, May 2008.

[85] X. Li and V. Jilkov, "A Survey of Maneuvering Target Tracking-Part II," in *Proc. 2001 SPIE conference on Signal and Data Processing of Small Targets*, 2001.

[86] ——, "Survey of Maneuvering Target Tracking-Part I: Dynamic Models," *IEEE Transactions Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, Oct. 2003.

[87] J. Borenstein and Y. Koren, "Real-time Obstacle Avoidance for Fast Mobile Robots," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, Sept./Oct. 1989.

[88] S. Haykin, *Neural networks : a comprehensive foundation.* Maxwell Macmillan International, 1994.

[89] K. Plataniotis and C. Regazzoni, "Visual-Centric Surveillance Networks and Services," *IEEE Signal Processing magazine, special issue on video and signal processing for survelliance networks and services*, vol. 22, no. 2, pp. 12 – 15, March 2005.

[90] Y. Ivanov, A. Sorokin, C. Wren, and I. Kaur, "Tracking People in Mixed Modality Systems," in *Proc. of SPIE Conference on Visual Communications and Image Processing (VCIP)*, vol. 6508, January 2007.

[91] M. A. Ali, S. Indupalli, and B. Boufama, "Tracking multiple people for video surveilliance," in *Proc. of CRV06 Workshop on Video Processing for Security (VP4S-06)*, June 2006.

[92] M. Brubaker, L. Sigal, and D. Fleet, "Video-based people tracking," in *Handbook on Ambient Intelligence and Smart Environments*, H. Nakashima, H. Aghajan, and J. Augusto, Eds. Springer Verlag, November 2009.

[93] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 249–265, 1987.

[94] S. Williams, P. Newman, J. Rosenblatt, G. Dissanayake, and H. Durrant-Whyte, "Autonomous underwater navigation and control," *Robotica*, vol. 19, no. 5, pp. 481–496, August 2001.

[95] B. Xerri, J. Cavassilas, and B. Borloz, "Passive tracking in underwater acoustic," *Signal Processing*, vol. 82, pp. 1067–1085, 2002.

[96] W. Knight and R. Pridham, "Digital signal processing for sonar," *Proceedings of IEEE*, vol. 69, no. 11, pp. 1451–1507, 1981.

[97] E. Ivanjko, I. Petrovic, and M. Vasak, "Sonar based pose tracking of indoor mobile robots," *Automatica*, vol. 45, no. 3-4, pp. 145–154, 2004.

[98] O. Wijk and H. Christensen, "Triangulation-Based Fusion of Sonar Data with Application in Robot Pose Tracking," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 740–752, December 2000.

[99] M. Betke and L. Gurvits, "Mobile Robot Localization Using Landmarks," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 2, pp. 251–263, April 1997.

[100] M. Drumheller, “Mobile Robot Localization Using Sonar,” Massachusetts Institute of Technology, Tech. Rep., January 1985.

[101] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, “Monte Carlo Localization: Efficient Position Estimation for Mobile Robots,” in *Proc. of the National Conference on Artificial Intelligence*, 1999.

[102] J. Laaksonen, “Mobile Robot Localization Using Sonar Ranging,” Master’s thesis, Lappeenranta University of Technology, 2007.

[103] E. A. Wan and A. S. Paul, “A tag-free solution to unobtrusive indoor tracking using wall-mounted ultrasonic transducers,” in *Proceedings of the 2010 IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010.

[104] A. Oppenheim and R. Schafer, *Discrete-Time Signal Processing.* Prentice-Hall, 1998.

[105] R. Duda, P. Hart, and D. Stork, *Pattern Classification.* John Wiley & Sons, 2001.

[106] G. Sibley, G. Sukhatme, and L. Matthies, “The iterated sigma point Kalman filter with applications to long range stereo,” in *Proceedings of Robotics: Science and Systems II*, August 2006.

[107] J. D. Jr and R. Schnabel, *Numerical Methods for unconstrained optimization and nonlinear equations.* Soc for Industrial & Applied Math, 1996.

[108] S. Thrun, *Spatial Mapping Approaches in Robotic and Natural Mapping Systems.* Berlin: Springer Tracts in Advanced Robotics, 2006, ch. Simultaneous localization and mapping.

[109] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit, “FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges,” in *Proceedings of the 18th international joint conference on Artificial intelligence*, Acapulco, Mexico, 2003, pp. 1151–1156.

[110] C.-C. Wang, C. Thorpe, and S. Thrun, "Online Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects: Theory and Results from a Ground Vehicle in Crowded Urban Areas," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[111] S. Localization and M. with Unknown Data Association Using FastSLAM, "M. Montemerlo and S. Thrun," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[112] M. Csorba and H. F. Durrant-Whyte, "New approach to map building using relative position estimates," in *Proceedings of the SPIE*, vol. 3087, April 1997, pp. 115–125.

[113] H. J. S. Feder, J. J. Leonard, and C. M. Smith, "Adaptive mobile robot navigation and mapping," *International Journal of Robotics Research, Special Issue on Field and Service Robotics,*, vol. 18, pp. 650–688, 1999.

[114] J. Kosecka and F. Li, "Vision based topological Markov localization," in *Proceedings of the 23rd IEEE International Conference on Robotics and Automation*, 1999.

[115] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2d range scans," *Journal of Intelligent and Robotic Systems*, vol. 18, pp. 249–275, 1997.

[116] E. Wan, R. van der Merwe, and A. Nelson, "Dual estimation and the unscented transformation," in *Proceedings of the Neural Information Processing system (NIPS)*, vol. 12, 2000, pp. 666–672.

[117] P. Savage, "Strapdown inertial navigation integration algorithm design part 1: Attitude algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 21, pp. 19–28, January 1998.

[118] A. S. Paul and E. A. Wan, "Dual Kalman filters for autonomous terrain aided navigation in unknown environments," in *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, 2005.

[119] R. J. McAulay and T. F. Quatieri, "Speech Analysis/Synthesis Based on a Sinusoidal Representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-34, no. 4, pp. 744–754, August 1986.

[120] M. Wu, D. Wan, and G. J. Brown, "A Multipitch Tracking Algorithm for Noisy Speech," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 3, pp. 229–241, May 2003.

[121] J. Tabrikian, S. Dubnov, and Y. Dickalov, "Maximum A-posteriori probability pitch tracking in noisy environments using harmonic model," *IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING*, vol. 12, no. 1, pp. 76–87, Jan 2004.

[122] P. J. Parker and B. Anderson, "Frequency Tracking of Nonsinusoidal Periodic Signals in Noise," *Signal Processing*, vol. 20, no. 2, pp. 127–152, June 1990.

[123] D. Li and R. Jung, "Tracking rhythmicity in nonstationary quasi-periodic biomedical signals using adaptive time-varying convariance," *Computers in Biology and Medicine*, vol. 32, pp. 261–282, 2002.

[124] B. James, B. D. Anderson, and R. C. Williamson, "Conditional mean and maximum likelihood approaches to multiharmonic frequency estimation," *IEEE Transactions on Signal Processing*, vol. 42, no. 6, pp. 1366–1375, June 1994.

[125] B. F. La Scala and R. R. Bitmead, "Design of an extended Kalman filter frequency tracker," *IEEE Transactions on Signal Processing*, vol. 44, no. 3, pp. 739–742, March 1996.

[126] B. F. La Scala, R. R. Bitmead, and B. G. Quinn, "An exteded Kalman filter frequency tracker for high-noise environments," *IEEE Transactions on Signal Processing*, vol. 44, no. 2, pp. 431–434, Feb. 1996.

[127] S. Bittanti and S. M. Savaresi, "On the Parameterization and Design of an Extended Kalman Filter Frequency Tracker," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1718–1724, September 2000.

[128] L. A. Johnston and V. Krishnamurthy, “Derivation of a Sawtooth Iterated Extended Kalman Smoother via the AECM Algorithm,” *IEEE Transactions on Signal Processing*, vol. 49, no. 9, pp. 1899–1909, September 2001.

[129] E. Fischler and B. Z. Bobrovsky, “Mean time to loose lock of phase tracking by particle filtering,” *Signal Processing*, vol. 86, p. 34813485, 2006.

[130] C. Dubois and M. Davy, “Joint detection and tracking of time-varying harmonic components: A flexible Bayesian approach,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 4, p. 12831295, May 2007.

[131] S. Kim, A. Paul, E. Wan, and J. McNames, “New multiharmonic frequency tracking using the sigma-point Kalman smoother,” *EURASIP Journal on Advances in Signal Processing*, vol. 2010, pp. 1–13, February 2010.

[132] A. Moore, J. Miller, B. Tabashnik, and S. Gage, “Automated identification of flying insects by analysis of wingbeat frequencies,” *Journal of Economic Entomology*, vol. 79, no. 6, p. 17031706, 1986.

[133] A. Moore and R. Miller, “Automated identification of optically sensed aphid (homoptera:aphidae) wingbeat waveforms,” *Annals of the Entomological Society of America*, vol. 95, no. 1, pp. 1–8, 2002.

[134] R. van der Merwe and E. Wan, “Sigma-Point Kalman Filters for Integrated Navigation,” in *Proceedings of the 60th Annual Meeting of The Institute of Navigation (ION)*, June 2004.

[135] T. Song and J. Speyer, “A stochastic analysis of a modified gain extended Kalman filter with application to estimation with bearing only measurement,” *IEEE Transactions on Automatic Control*, vol. AC-30, pp. 940–949, 1985.

[136] P. Galkowski and M. Islam, “An alternative derivation of the modified gain function of Song and Speyer,” *IEEE Transaction on Automatic Control*, vol. 36, pp. 1323–1326, 1991.

[137] K. Reif, S. Gunther, E. Yaz, and R. Unbehauen, "Stocastic stability of the discrete time extended Kalman filter," *IEEE Transactions on Automatic Control*, vol. 44, no. 4, pp. 714–728, April 1999.

[138] A. Alessandri, M. Cuneo, S. Pagnan, and M. Sanguineti, "A recursive algorithm for nonlinear least-squares problems," *Computational Optimization and Applications*, vol. 38, no. 2, pp. 195–216, 2007.

[139] H. V. Trees and K. Bell, Eds., *Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking.* IEEE Press, 2007.

[140] P. Tichavsky, C. Muravchik, and A. Nehorai, "Posterior Cramer-Rao bounds for Discrete Time Nonlinear Filtering," *IEEE Transactions on Signal Processing*, vol. 46, no. 5, pp. 1386–1396, May 1998.

[141] H. Cramer, *Mathematical Methods of Statistics.* Princeton, NJ: Princeton University Press, 1946.

[142] C. Rao, "Information and accuracy attainable in the estimation of statistical parameters," *Bull. Calcutta Math Soc.*, vol. 37, pp. 81–91, 1945.

[143] H. V. Trees, "Bounds on the accuracy attainable in the estimation of continuous random processes," *IEEE Transactions on Information Theory*, vol. 12, no. 3, pp. 298–305, July 1966.

[144] ——, *Detection, Estimation and Modulation Theory, Part I.* New York: Wiley, 1968.

[145] ——, *Detection, Estimation and Modulation Theory, Part II.* New York: Wiley, 2003.

[146] B. Bobrovsky and M. Zakai, "A lower bound on the estimation error for Markov processes," *IEEE Transactions on Automatic Control*, vol. AC-20, pp. 785–788, December 1975.

[147] ——, "A lower bound on the estimation error for certain diffusion processes," *IEEE Transaction on Information Theory*, vol. IT-22, no. 1, pp. 45–52, January 1976.

[148] B. Bobrovsky, M. Zakai, and O. Zetouni, "Error bounds for the nonlinear filtering of signals with small diffusion coefficients," *IEEE Transaction on Information Theory*, vol. 34, no. 4, pp. 710–721, July 1988.

[149] J. Galdos, "A lower bound on filtering error with application to phase demodulation," *IEEE Transaction on Information Theory*, vol. 25, no. 4, pp. 452–462, July 1979.

[150] ——, "A Cramer-Rao bound for multidimensional discrete time dynamical systems," *IEEE Transactions on Automatic Control*, vol. AC-25, pp. 117–119, February 1980.

[151] P. Doerschuk, "Cramer-Rao bounds for discrete time nonlinear filtering problems," *IEEE Transaction on Automatic Control*, vol. 40, no. 8, pp. 1465–1469, August 1995.

[152] T. Kerr, "Status of CR-like lower bounds for nonlinear filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 25, pp. 590–600, September 1989.

[153] D. Snyder and I. Rhodes, "Filtering and control performance bounds with implications on asymptotic separation," *Automatica*, vol. 8, pp. 747–753, November 1972.

[154] V. Fromion and G. Ferreres, "A characterization of some global properties of the Extended Kalman filter," in *Proceedings of American Control Conference*, 1997.

[155] L. Ljung, "Asymptotic behaviour of the extended Kalman filter as a parameter estimator for linear systems," *IEEE Transactions on Automatic Control*, vol. AC-24, pp. 36–50, 1979.

[156] B. Ursin, "Asymptotic convergence properties of the extended Kalman filter using filtered state estimates," *IEEE Transactions on Automatic Control*, vol. AC-25, pp. 1207–1211, 1980.

[157] M. Bontayeb, H. Rafarlaky, and M. Daorach, “Converegence analysis of the extended kalman filter used as an observer for nonlinear discrete time systems,” *IEEE Transaction on Automatic Control*, vol. 42, pp. 581–586, 1997.

[158] Y. Song and J. Grizzle, “The extended Kalman filter as a local asymptotic observer for discrete-time nonlinear systems,” *Journal on Math System Estimation and Control*, vol. 5, pp. 59–78, 1995.

[159] K. Kim, G. Jee, C. Park, and J. Lee, “The Stability Analysis of the Adaptive Fading Extended Kalman Filter Using The Innovation Covariance,” *International Journal of Control, Automation and Systems*, vol. 7, no. 1, pp. 49–56, 2009.

[160] R. Agniel and E. Jury, “Almost sure boundedness of randomely sampled systems,” *SIAM Journel on Control*, vol. 9, pp. 372–384, 1971.

[161] T. Tarn and Y. Rasis, “Observers for nonlinear stochastic systems,” *IEEE Transaction on Automatic Control*, vol. AC-21, pp. 441–448, 1976.

[162] T. Morozan, *Boundedness Properties for Stochastic Systems.* Berlin, Germany: Springer-Verlag, 1972, pp. 21–34.

[163] B. Anderson and J. Moore, “Detectability and stabilisability of time-varying discrete time linear systems,” *SIAM journal Control and Optomization*, vol. 19, pp. 20–32, 1981.

[164] T. Mitchell, *Machine Learning.* The McGraw-Hill Companies Inc., 1997.

Biographical Note

Anindya S. Paul received his B.Eng. in Electronics and Communication from Sikkim Manipal Institute of Technology, India in 2001. In 2003 he obtained his M.S. in Electrical Engineering from Wright State University, Dayton, OH, USA, where he collaborated with the Wright Patterson Air Force Base for his research work. In 2003 he accepted the offer to work as a graduate research assistant for the Adaptive Systems Lab, Dept. of Biomedical Engineering (formerly OGI School of Science and Engineering) at Oregon Health & Science University (OHSU), Beaverton, OR, USA. There he received "Special Award for Collaborative Achievements" for research contributions in multiple collaborative projects and research groups in 2007. His broad research interests lie in the fields of probabilistic inference and Bayesian learning, statistical pattern recognition, artificial neural networks and machine learning. His research work in OHSU involves unobtrusive indoor pedestrian tracking, formulating a self-calibration method for indoor tracking using the simultaneous localization and mapping (SLAM), developing new sigma-point Kalman smoother formulations for nonlinear systems and analyzing error behavior and convergence criterions for the sigma-point Kalman filter. He graduated with a PhD degree in 2010.

"I have become my own version of an optimist.
If I can't make it through one door,
I'll go through another door - or I'll make a door.
Something terrific will come no matter how dark the present."

- Rabindranath Tagore