Gaussian Process Methods for Group, Extended and Point Target Tracking and Smoothing

The
University
Of
Sheffield.

Waqas Aftab

Department of Automatic Control and Systems Engineering
University of Sheffield

This dissertation is submitted for the degree of
Doctor of Philosophy

September 2020

To my loving parents, wife and kids . . .

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this or any other university. This dissertation is my work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Waqas Aftab
September 2020

Acknowledgements

Writing of this dissertation during the last four years was a daunting task. However, after completing the thesis I feel obliged to thank everyone who helped me through this journey.

Thank you to my supervisor, Prof. L. Mihaylova, for providing continuous guidance, support, and encouragement. I would like to pay special regards to my second supervisor, Dr. M. Arvaneh, and co-authors of some of my publications, Dr. R. Hostettler and Dr. A. DeFreitas, for sharing their valuable knowledge, feedback and experience.

I wish to express my deepest gratitude to my family. My parents and sisters, for their love and belief in me. My wife, for being extremely supportive throughout this process and for making those sacrifices every day. My children, for keeping me relaxed and happy in tough times.

Lastly, I wish to express my deepest gratitude to Pakistan Air Force (Government of Pakistan) and Department of ACSE (University of Sheffield) for granting me the maintenance and tuition fee scholarships, respectively.

Abstract

Target tracking is an important component of many modern automation systems. It deals with state estimation of targets of interest from noisy sensor data. Traditional methods rely on model-based tracking approaches. This thesis proposes model-free tracking methods, using Gaussian process regression and demonstrates the performance gain. The proposed methods are in the domain of point, extended, and group target tracking.

Extended and group target tracking deals with kinematics and shape estimation of extended and group targets. A Gaussian process convolution particle filter is proposed for tracking of single and multiple irregularly shaped extended and group targets in the presence of clutter. The proposed approach is demonstrated using challenging real and simulated scenarios to track irregularly shaped targets using surface measurements. The model-free Gaussian process approach and the kernel estimation method is able to track the target without requiring any prior information regarding the target generated measurement statistics. A generic spatio-temporal Gaussian process method is proposed for non-rigid extended and group target tracking and smoothing. The temporal dependence of the model improves the shape estimates while tracking of non-rigid objects. The proposed filter and smoother is shown, using an example approach, to provide a performance improvement over the state-of-the-art Gaussian process approach on both real and simulated data.

Point target tracking deals with kinematics estimation of point targets. A model-free Gaussian process motion tracker is proposed for the target state prediction and filtering. The proposed approach is demonstrated to be overall better than the compared model-based approaches on multiple challenging manoeuvring scenarios. A study on the performance of the Gaussian process motion tracker using various covariance kernels is given. Based on this study, different covariance kernels are proposed for real target trajectories. A recursive Gaussian process motion tracker and smoother are proposed for online tracking and smoothing. The target trajectory is tracked using a non-stationary Gaussian process, where the hyperparameters are learned online. This recursive tracker is shown to be comparable in performance to

the Gaussian process motion tracker, which is non-recursive and computationally expensive as compared to the recursive approach. It has also been shown to be overall better than the compared model-based approaches for tracking of challenging target trajectories.

List of Publications

This thesis constitutes of several works that are either already published or in press. The list of the publications is given below:

Paper I : ©2017 IEEE. Reprinted, with permission, from

Waqas Aftab, Allan De Freitas, Mahnaz Arvaneh, and Lyudmila Mihaylova. A Gaussian process approach for extended object tracking with random shapes and for dealing with intractable likelihoods. In *Proceedings of the 22nd International Conference on Digital Signal Processing (DSP)*, pages 1–5, London, UK, 2017. IEEE [1]

Paper II : ©2018 IEEE. Reprinted, with permission, from

Waqas Aftab, Allan De Freitas, Mahnaz Arvaneh, and Lyudmila Mihaylova. A Gaussian process convolution particle filter for multiple extended objects tracking with non-regular shapes. In *Proceedings of the 21st International Conference on Information Fusion (FUSION)*, pages 1–8, Cambridge, UK, 2018. IEEE [2]

Paper III : ©2019 IEEE. Reprinted, with permission, from

Waqas Aftab, Roland Hostettler, Allan De Freitas, Mahnaz Arvaneh, and Lyudmila Mihaylova. Spatio-temporal Gaussian process models for extended and group object tracking with irregular shapes. *IEEE Transactions on Vehicular Technology*, 68(3):2137–2151, 2019 [3]

Paper IV : ©2019 IEEE. Reprinted, with permission, from

Waqas Aftab and Lyudmila Mihaylova. A Gaussian process regression approach for point target tracking. In *Proceedings of the 22nd International Conference on Information Fusion (FUSION)*, pages 1–8, Ottawa, Canada, 2019. IEEE [4]

Paper V : ©2020 IEEE. Reprinted, with permission, from

Waqas Aftab and Lyudmila Mihaylova. On the impact of different kernels and training data on a Gaussian process approach for tracking. In *Proceedings of* *the 23rd International Conference on Information Fusion (FUSION)*, South Africa, 2020. IEEE. In press [5]

Paper VI : ©2020 IEEE. Reprinted, with permission, from

Waqas Aftab and Lyudmila Mihaylova. A learning Gaussian process approach for manoeuvring target tracking and smoothing. *IEEE Transactions on Aerospace and Electronic Systems*, 2020. In press [6]

Paper VII : ©2019 NATO STO. Reprinted, with permission, from

L Mihaylova and W Aftab. Recent advances in extended and group objects tracking. In *Proceedings of the Artificial Intelligence for Military Multisensor Fusion Engines(STO-MP-SET-262)*, pages 14.1–14.12. IEEE, 2019 [7]

Table of Contents

List of Figures

List of Tables

List of Acronyms/ Abbreviations and Symbols

Roman Symbols

$\boldsymbol{F}$	State Update Matrix
$\boldsymbol{f}$	State Update Function
$\boldsymbol{H}$	Measurement Matrix
$\boldsymbol{h}$	Measurement Function
$\boldsymbol{I}_N$	N-dimensional Identity Matrix
$\boldsymbol{K}$	Gaussian Process Covariance Matrix
$\boldsymbol{P}$	State Error Covariance
$\boldsymbol{Q}$	Process Noise Covariance Matrix
$\boldsymbol{R}$	Measurement Noise Covariance Matrix
$\boldsymbol{v}$	Measurement Noise Vector
$\boldsymbol{w}$	Process Noise Vector
$\boldsymbol{x}$	State Vector
$\boldsymbol{z}$	Measurement Vector
$\mathbb{E}[\cdot]$	Expectation Operator
$\mathcal{K}$	Parzen-Rozenblatt Kernels
$\mathcal{N}(\cdot,\cdot)$	Normal Distribution
$\mathcal{P}$	Polygon
$\mathcal{U}[\cdot,\cdot]$	Uniform Distribution
B	Number of Keypoints
GP	Gaussian Process
$k(\cdot,\cdot)$	Gaussian Process Kernel
$m(\cdot)$	Gaussian Process Mean Function
n_p	Number of particles
$p(\cdot)$	Probability Density Function
$STGP$	Spatio-temporal Gaussian process
T	Sampling Time
u	Gaussian Process Input
$\mathcal{F}[\cdot]$	Fourier Transform

Greek Symbols

$\boldsymbol{\eta}$	Hyperparameters Vector
$\delta(\cdot)$	Dirac measure
σ^2	Measurement Noise Variance
τ_m	Manoeuvre Time Constant
$\Gamma(\cdot)$	Gamma Function

Other Symbols

$\cdot$	Dot Product

$\otimes$	Kronecker Product

Acronyms / Abbreviations

CMKF	Converted Measurement Kalman Filter
CoO	Center of Object/target
CPF	Convolution Particle Filter
CT	Coordinated Turn
CV	Constant Velocity
DBSCAN	Density Based Spatial Clustering of Applications with Noise
EKF	Extended Kalman Filter
ETT	Extended Target Tracking
FGIMM	Fixed Grid Interacting Multiple Model
GP	Gaussian Process
GPCPF	Gaussian Process Convolution Particle Filter
GPMT	Gaussian Process Motion Tracker
HMM	Hidden Markov Model
i.i.d	independent and identically distributed
IMM	Interacting Multiple Model
IRP	Internal Reference Point
ML	Maximum Likelihood
MM	Multiple Model
MMSE	Minimum Mean Square Error
MTT	Multiple Target Tracking
NCT	Nearly Coordinated Turn
NCV	Nearly Constant Velocity
RDGP*MS	Recursive Derivative of Gaussian Process Motion Smoother
RDGP*MT	Recursive Derivative of Gaussian Process* Motion Tracker
RGP	Recursive Gaussian Process
RGP*MS	Recursive Gaussian Process* Motion Smoother
RGP*MT	Recursive Gaussian Process* Motion Tracker
RHM	Random Hypersurface Model
RMSE	Root Mean Square Error
SE	Squared Exponential
SIS	Sequential Importance Sampling
SMC	Sequential Monte Carlo
UKF	Unscented Kalman Filter

Part I

Theoretical Perspective

Chapter 1

Introduction

This chapter gives a brief overview of the research conducted for the writing of this thesis. The broad area of research is multiple target tracking. This thesis proposes solutions to the sub-problems related to the field of target tracking. The research motivation is given in Section 1.1 and the contributions of the thesis are covered in Section 1.2. The outline of the thesis is given in Section 1.3. This chapter avoids the use of technical terms, as much as possible, for brevity.

1.1 Motivation and Applications

Multiple target tracking (MTT) is a core component of various commercial and non-commercial automation systems. The earliest MTT algorithms were proposed for radar data processing, after the second world war [8], and applied in military air surveillance systems. Since then, it has been an important component of various military systems as well as modern command and control systems. During the last three decades, the application of these methods has seen an exponential rise in non-military systems as well. The spectrum of the commercial applications is wide and diverse e.g. human cells tracking in biomedicine [9], objects tracking for environment perception in autonomous vehicles [10, 11], road-map estimation [12], robot navigation [13], crowd analysis [14] and tracking of heavenly objects for space exploration [15], etc. Other interesting areas include, but are not limited to, crowd tracking for urban management [16], tracking of chemical, biological, radiological and nuclear pollutant clouds for public safety [17], a flock of birds tracking for aircraft hit avoidance [18], eddy currents tracking for oceanography [19], air traffic control [20], sea surveillance [21], and human and objects tracking based on Microsoft Kinect sensors [22].

MTT requires solving a complex target state estimation problem using noisy sensor measurements. The various challenges include the unknown target trajectory and number, the unknown measurement noise characteristics, the measurement origin uncertainty, and the presence of the clutter measurements (false alarms). The states of interest are the target kinematics e.g. position, velocity, heading, etc., and the target attributes such as the aircraft identification mode. These are estimated using a subset of the kinematics and the attributes information contained in the measurement data. Sometimes, the measurement kinematics data can be processed further to estimate high level target features such as shape, orientation[1], and size/volume. This thesis focuses on the target kinematics and feature estimation.

MTT requires the processing of time-series measurement data for the above-mentioned state estimation problem. Typically, the problem is modelled in a probabilistic framework and solved using various model-based approaches. Recently, machine learning methods have gained popularity in the domain of statistical state estimation methods. This thesis develops the machine learning methods for MTT using the Gaussian process (GP), a model-free approach, and demonstrates it's flexibility. GP, one of the promising machine learning methods, is a kernel based approach. It is a popular method for solving problems related to the regression and classification. Unlike other machine learning methods, such as neural networks, the GP provides an uncertainty measure in addition to the mean of the estimate. Moreover, the Bayesian neural network output converges to a GP for infinite hidden units [23]. In this thesis, the GP has been chosen, due to the above-mentioned strengths and flexibility, to solve various sub-problems of the MTT.

MTT is classified into two categories in this thesis, multiple point and extended/group targets tracking [24, 25]. A point target/object [2] tracking refers to the situation when the measurement data is used to infer the target/object kinematics and the attributes. An extended target/object tracking refers to the inference of the target/object features, in addition to the kinematics and the attributes, using the measurement data. A group target tracking requires estimating the mean kinematics, attributes, and the shape of a group of point targets moving under similar dynamics. Although extended and group target tracking applications are different, the tracking methods proposed for the extended targets are equally applicable to

[1] The direction of motion of a target, also called heading, can be different from it's orientation.

[2] Target and object are interchangeable terminologies in this thesis.

the group targets tracking and vice-versa. The thesis focuses on both the point and extended/group targets tracking problems.

1.2 Contributions

This thesis encompasses solutions to the MTT sub-problems using the GP. The GP is proposed for feature estimation of a single extended/group target in [26] and multiple extended/group targets in [27]. The attributes are not estimated in both the papers, as well as in this thesis. Other unknowns such as the kinematics, number, the measurement origin uncertainty, and clutter measurements, are solved using non-GP methods. Although the proposed feature model [26] can adopt any shape, the evaluation is done on standard shapes and rigid targets only. The feature estimation of irregularly shaped targets is proposed using a GP convolution particle filter (GPCPF) in Paper I [1]. The measurements can be assumed to originate from the target boundary, called *contour* measurements, or from the inside of the boundary, called *surface* measurements. The shape estimation problem using the surface measurements is more complex as compared to the case of the contour measurements. The methods proposed for the processing of the surface measurements require prior knowledge of the statistical properties of the target [26–28] e.g. the mean size, the variance of the measurements around the centre of the target, etc. The GPCPF, of Paper I [1], is proposed for both cases, the contour and the surface measurements. It does not require any prior knowledge of the target statistical characteristics for the handling of the surface measurements.

Paper I [1]: A Gaussian Process Approach for Extended Target Tracking with Irregular Shapes and for Dealing with Intractable Likelihoods. A GP convolution particle filter (GPCPF), proposed in this work, is a novel shape estimation approach for an irregular target moving in clutter. The proposed filtering approach, convolution particle filter (CPF), is a kernel based estimation approach. A new kernel is proposed for the arbitrarily shaped target, based on a GP, which also deals with clutter. The same kernel handles the contour and the surface measurements without additional prior knowledge. A model-based approach is proposed for the kinematics estimation. The performance is evaluated on simulated data and the mean precision and recall of the proposed approach are around 0.9. The performance is not compared with other methods as most of these methods assume the target shape to be a standard geometrical shape such as line, circle, rectangle, ellipse, etc. The methods proposed for non-standard shapes such as random hypersurface [29], GP [26], and a few others, do not have any published code for tracking a single target in clutter. The performance evaluation on real data is not possible as the publicly available real data consists of multiple targets moving in clutter. *I proposed the GPCPF for an irregularly shaped target for the case of contour and surface measurements. I also carried out the performance evaluation of the approach under the guidance of my co-authors.*

The GPCPF, proposed in Paper I [1], is extended for tracking of multiple irregularly shaped extended targets in Paper II [2]. The number of targets and the measurement origin is considered unknown. A multiple GPCPF is proposed and tested on real data for the considered problem.

Paper II [2]: A Gaussian Process Convolution Particle Filter for Multiple Extended Objects Tracking with Irregular Shapes. In this work, a new approach for tracking multiple irregularly shaped targets using the surface measurements is proposed. A convolution kernel is proposed to determine the multi-target likelihood. A target birth and death model is proposed for automatic initiation and deletion of the targets. The proposed approach is validated on real-world LiDAR data which shows that it is efficient in tracking multiple irregularly shaped targets in challenging scenarios involving occlusion, dense clutter, and low target detection. The comparison with other methods is not done due to the unavailability of the code. *I proposed the multiple GPCPF for tracking multiple irregularly shaped targets in clutter using surface measurements. I also carried out the performance evaluation of the approach under the guidance of my co-authors.*

The GP based shape estimation methods proposed in [26], Paper I [1] and Paper II [2] are applicable to non-rigid (changes shape with time) targets. The model for temporal changes in the shape is adopted using a random process noise model. The tracking of non-rigid extended targets, such as clouds, can be improved using complex spatio-temporal models. A spatio-temporal GP has been derived in Paper III [3] for tracking of a non-rigid irregularly shaped target. Additionally, some assumptions in the GP approach of [26] are relaxed and a generic framework is proposed.

Paper III [3]: Spatio-temporal Gaussian Process for Extended and Group Target Tracking with Irregular Shapes. This work presents a generic spatio-temporal Gaussian process (STGP) method for tracking an irregularly shaped extended target. The complex shape is represented by key points and their parameters are estimated both in space and time. This is achieved by factorization of the power spectral density function of the STGP covariance function. A new form of the temporal covariance kernel is derived with the theoretical expression of the filter likelihood function. Solutions to both the filtering and the smoothing problems are presented. Various approaches based on the proposed method can be derived for tracking of rigid and non-rigid targets. A thorough evaluation of the performance in a simulated environment shows that an example approach based on the proposed STGP method outperforms the state-of-the-art GP extended Kalman filter approach [26], with up to 90% improvement in the accuracy in position, 95% in velocity, and 7% in shape, while tracking a simulated asymmetric non-rigid target. The tracking performance improvement for a non-rigid irregular real target is up to 43% in position, 68% in velocity, 10% in the recall, and 115% in the precision measures. *The idea was proposed by my co-author. I derived the model, the filter, and the smoother. I also performed the performance evaluation using the simulated and real data. The code for the compared method GP-EKF [26] is written by one of my co-authors.*

The above three works, Paper I [1], Paper II [2], and Paper III [3], and the MTT in general, rely on single and multiple model based approaches for the target kinematics estimation. In many applications, the targets can adopt an unknown and infinite number of linear or nonlinear trajectories. Multiple models (MM) based approaches, typically used in complex scenarios, provide improved performance as compared to the single model based filters. These MM methods provide the state estimate using a finite set of motion models. Each model corresponds to a possible target trajectory. The performance of these model-based approaches degrades when there is a mismatch between the true target trajectory and the selected model set. A GP is exploited for the kinematics estimation, as it can adopt infinite trajectories, based on the results of Neal [23]. A GP based kinematics estimation approach has been proposed in Paper IV [4] for a point target. A single measurement per target per time sample is assumed. The multiple measurements per target per time sample is a more complex scenario and is considered as future work.

Paper IV [4]: A Gaussian Process Regression Approach for Point Target Tracking. A point target tracker assumes scarce measurements or in most cases a single measurement per target per time sample. In this work, a Gaussian process based approach, namely, Gaussian process motion tracker (GPMT) is proposed for position estimation. The estimation of the velocity and the higher order positional derivatives has been proposed using the derivative of the GP. The evaluation of the proposed approach is performed on challenging scenarios and is compared with popular single and MM based approaches. The results show high accuracy of the predicted and filtered target position and velocity over challenging manoeuvring scenarios. *I* *proposed the model and carried out the performance evaluation on the simulated data under the guidance of my co-author.*

The GPMT, proposed in Paper IV [4], uses a squared exponential covariance kernel for the target tracking. The GPMT is a flexible approach, where different covariance kernels are chosen depending upon characteristics of the target dynamics. A simulation based study is done in Paper V [5] to compare the effects of the kernels and other parameters.

Paper V [5]: On the Impact of Different Kernels and Training Data on a Gaussian Process Approach for Tracking. This paper presents a simulation-based study on the practical aspects of a very promising and recently proposed Gaussian process approach, namely the Gaussian process motion tracker [4]. The paper also provides design guidelines on the choice of covariance kernel, size of training data, and impact of the unknown measurement noise variance on the above-mentioned tracking approach by comparing the rational quadratic and Matérn based GPMT with the originally proposed squared exponential GPMT of Paper IV [4]. The study shows that the squared exponential GPMT performs best on constant velocity and Singer model based target trajectories. The rational quadratic GPMT performs best on the coordinated turn trajectories. The accuracy of the Matérn GPMT improves with an increasing number of training data. The accuracies of rational quadratic GPMT and squared exponential GPMT decreases with increasing data on the Singer model based trajectory. Lastly, it has been shown that all three GPMT approaches are robust to the changing measurement noise variances. *I carried out the study on the simulated data under the guidance of my co-author.*

The GPMT relies on training and learning processes for the tracking approach proposed in Paper IV [4]. The learning proposed in Paper IV [4], based on maximum likelihood optimization, is slow and makes the approach non-real time. The earliest and most of the modern MTT applications require real-time tracking of the targets. The work in the Paper VI [6] aims to reduce the processing time of the learning to make it real-time.

Paper VI [6]: A Learning Gaussian Process Approach for Manoeuvring Target Tracking and Smoothing. This paper proposes a data-driven approach that represents the possible target trajectories using a distribution over an infinite number of functions. Recursive Gaussian process and derivative based Gaussian process approaches for target tracking and smoothing are developed, with online training and parameter learning. The performance evaluation over two highly manoeuvring scenarios shows that the proposed approach provides 80% and 62% performance improvement in the position and 49% and 22% in the velocity estimation, respectively, as compared to the best model-based filter. *I proposed the approach and evaluated the performance under the guidance of my co-author.*

This thesis also includes the contents of a survey published as Paper VII [7]. The contents of this publication are not given as a separate chapter.

Paper VII [7]: Recent Advances in Extended and Group Objects Tracking. We live in an era of increasing data and information from multiple sensors. The multiple complementary types of sensors introduce a variety of challenges, especially in systems with different levels of autonomy, such as Unmanned Aerial Vehicles and surveillance systems. Autonomous systems require quick situation awareness, including tracking of the location and size of the targets of interest, for instance, extended and groups that give rise to multiple measurements. Examples of extended targets are pedestrians, convoys of vehicles and clouds of bio-chemical contaminants. Most of the current approaches rely on well-defined mathematical models. However, the changes both in multiple group and extended target dynamics and in the environment require flexible approaches able to learn and adapt to the changes. Hence, this work overviews the state-of-the-art approaches and focuses on data-driven approaches such as Gaussian processes for spatio-temporal representations of extended targets and groups. We share our vision for future trends in this area. *I contributed to this survey paper as a second author.*

1.3 Thesis Outline

The thesis has been divided into two parts, namely a *theoretical perspective* and *publications*. The first part introduces the background knowledge. An introduction to the Gaussian process methodology is given in Chapter 2. The theoretical background and popular state-of-the-art methods of multiple point, extended and group target tracking problems are given in Chapters 3 and 4, respectively. The second part presents each of the proposed methods and approaches as separate chapters. The content of each chapter has either been published or is in press. The thesis is concluded and future work is recommended in Chapter 11.

Chapter 2

Introduction to Gaussian Process Methodology

Artificial intelligence deals with the development of systems that think and act like humans [30]. Machine learning deals with the study of model-free methods for solving the problems related to the field of artificial intelligence. The tasks successfully demonstrated using machine learning methods include classification [31], regression [32], pattern recognition [33], and many more. These methods rely on data, rather than explicit models, to execute the task at hand. For example, motion models (Section 3.4) are used for a target state estimation using the model-based approaches. For a machine learning based target tracker, the motion model does not need to be explicitly defined or programmed. The algorithm processes the historical data to learn a precise model. The state estimation is done using the learned model.

Machine learning methods are classified based on the type of learning as follows [34]:

1. **Supervised**. These methods require a given input and output dataset. The algorithm learns the functional mapping, from the input to the output, based on the given data. The learned model can predict the output at the unknown inputs.

2. **Semi-supervised**. These methods require both labelled and unlabelled data for training.

3. **Unsupervised**. These methods do not require explicit labelling of the input data. The algorithm is able to determine the hidden patterns within the data.

4. **Reinforcement**. These methods require a feedback on the algorithm output. The algorithm is able to learn from the external feedback and improve over time.

Machine learning methods have been applied to solve different multiple target tracking sub-problems. For example, manoeuvring (point) targets state estimation [35], data association of multiple (point) targets [36, 37], shape estimation of an extended or group target [26] etc. Neural networks, a machine learning method, based methods have also been proposed for the multiple (point) targets tracking [38]. Recently, GP, another machine learning method, has been proposed for target tracking [26, 37]. Neural networks provide point estimates whereas the GP provides the uncertainty measures, in addition to the point estimates. Moreover, the Bayesian neural networks output converges to a GP for infinite hidden units [23]. The above two advantages of the GP over neural networks is the motivation behind the methods proposed in this thesis. The GP based methods have been proposed for solving various multiple target tracking sub-problems. This chapter gives a brief overview of the GP theory.

2.1 Gaussian Process Regression

GP can be used as an unsupervised, semi-supervised, and supervised learning method. It is an abstract concept and an informal introduction is given before presenting the formal definition and concepts. GP concept is analogous to a multivariate Gaussian distribution. The random vector of the distribution can be seen analogously to a random function of the GP. The mean vector and the covariance matrix of the distribution has a parallel mean and covariance function in the GP. The multiple variables are modelled jointly Gaussian for the case of the distribution. Similarly, the functional realizations are assumed jointly Gaussian for the GP.

A GP models an unknown deterministic function f. The input and output of the function can be multidimensional but a one-dimensional input and output spaces are assumed in this section for clarity. Given an input variable u, the GP model is given below:

$$f(u) \sim GP(m(u), k(u, u')), \tag{2.1}$$

where $GP(m(u), k(u, u'))$ denotes a GP model with mean function denoted as $m(u)$ and a covariance kernel denoted as $k(u, u')$ and u and u' denote all the possible pairs of the input variables. The covariance kernel is a symmetric positive semi-definite function. The mean function and the covariance kernel encapsulate the prior on the unknown function f. Hence, a GP is also defined as a distribution over functions. The functional realization of the GP model at finite input locations is mutually Gaussian distributed as given below:

$$f(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{m}(\boldsymbol{u}), \boldsymbol{K}(\boldsymbol{u}, \boldsymbol{u})), \tag{2.2}$$

$$\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}, \quad f(\boldsymbol{u}) = \begin{bmatrix} f(u_1) \\ f(u_2) \\ \vdots \\ f(u_N) \end{bmatrix}, \quad \boldsymbol{m}(\boldsymbol{u}) = \begin{bmatrix} m(u_1) \\ m(u_2) \\ \vdots \\ m(u_N) \end{bmatrix}, \tag{2.3}$$

$$\boldsymbol{K}(\boldsymbol{u}, \boldsymbol{u}') = \boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}'} = \begin{bmatrix} k(u_1, u_1') & k(u_1, u_2') & \cdots & k(u_1, u_{N'}') \\ k(u_2, u_1') & k(u_2, u_2') & \cdots & k(u_2, u_{N'}') \\ \vdots & \vdots & \ddots & \vdots \\ k(u_N, u_1') & k(u_N, u_2') & \cdots & k(u_N, u_{N'}') \end{bmatrix}, \quad \boldsymbol{u}' = \begin{bmatrix} u_1' \\ u_2' \\ \vdots \\ u_{N'}' \end{bmatrix}, \tag{2.4}$$

where $f(\boldsymbol{u})$ denotes the output vector at N-dimensional input vector $\boldsymbol{u}$, $\mathcal{N}(\cdot, \cdot)$ represents a multivariate normal distribution, $\boldsymbol{m}(\boldsymbol{u})$ and $\boldsymbol{K}(\boldsymbol{u}, \boldsymbol{u})$ are, respectively, the GP mean vector and the covariance matrix and $\boldsymbol{u}'$ is an N'-dimensional input vector. A general expression of the covariance matrix is given in (2.4) using two different input vectors $\boldsymbol{u}$ and $\boldsymbol{u}'$ as it will be later used in the regression equations. An alternate notation of the GP covariance matrix $\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}'}$ is introduced here and used throughout this dissertation.

2.1.1 Mean Function

The GP mean function encapsulates the prior information of the mean behaviour of the modelled function f. It is a parametric function and the parameters are called *hyperparameters*. The mean behaviour of most stochastic processes is either difficult to model or not known apriori. For example, suppose the aircraft trajectory is tracked using a GP. The tracker has no prior information on the mean trajectory of the aircraft. Typically, the mean function of a GP is set to zero [39]. The non-zero mean function can be useful in some scenarios. For example, the mean function can be modelled as a fixed basis function and the parameters of the basis function are learned from the given data. The long term prediction of the GP converges to the mean function. In applications requiring long term predictions, a non-zero model of the mean function may provide an efficient solution. Some examples of non-zero mean functions are given below:

Constant

The mathematical expression is given below:

$$m_c(u) = c, \tag{2.5}$$

where c is a constant. The behaviour of the zero mean GP and a constant mean GP is similar. However, the long term prediction of the constant mean GP converges to the constant (non-zero) value.

Linear

The mathematical expression is given below:

$$m_{Lin}(u) = qu + c, \tag{2.6}$$

where q is the unknown slope. This is used for systems with a linear mean behaviour.

Basis function

The mathematical expression is given below:

$$m_{Bas}(u) = g(u)^T e, \tag{2.7}$$

where $g(u)$ denotes a set of fixed basis function, $\cdot^T$ represents the transpose function and e is a parameter. For example, a quadratic behaviour can be modelled using $g(u) = (1, u, u^2)$.

2.1.2 Covariance Kernel

A function with two input arguments, $u \in \mathbb{U}$ and $u' \in \mathbb{U}'$, mapping the inputs to a real number space $\Re$ is called a *kernel*. Hence, the covariance function is commonly called covariance kernel. The covariance kernel elicits a prior on the unknown function f modelled using the GP. It captures the covariance structure of the input variables. Similar to the mean function, the covariance kernel is also parametric and its parameters are also called hyperparameters. It should be noted that the input space of this kernel function can be multi-dimensional but one-dimensional input space is assumed in this section. As a result, the input of the kernel function is represented by a normal font u which represents a scalar input instead of the bold font which represents a vector input. When the input to kernel are two multi-dimensional vectors then the output is a one-dimensional or scalar. Hence, it is important to understand the difference between the kernel function and the GP covariance matrix given in (2.4), where the latter gives a matrix output for the vector inputs. The choice of the kernel is often critical, as the mean function is commonly set to zero [39]. Some important covariance kernels are given below:

Linear

The mathematical expression is given below:

$$k_{Lin}(u, u') = \sigma_m^2 u \cdot u' + \sigma_b^2, \tag{2.8}$$

where $\cdot$ denotes a product and σ_m^2 and σ_b^2 are, respectively, the magnitude and the bias variance hyperparameters. For vector inputs, $\cdot$ denotes a vector dot product. Using a GP for the linear regression problems is not recommended as it is computationally expensive as compared to popular linear regression methods.

Exponential

The mathematical expression is given below:

$$k_{Exp}(u, u') = \sigma_m^2 \exp\left(- \frac{|u - u'|}{l} \right), \tag{2.9}$$

where $|u - u'|$ denotes the distance between u and u', l is the length-scale hyperparameter. When the input arguments are vectors, $|\cdot|$ represents the distance between the two vectors. The lengthscale controls the width of the correlation in the input space. A large value means the input variable is correlated over a longer range and vice-versa. This kernel is used to model an Ornstein-Uhlenbeck process [39].

Squared Exponential

The mathematical expression is given below:

$$k_{SE}(u,u') = \sigma_m^2 \exp\left(-\frac{(|u-u'|)^2}{2l^2} \right). \tag{2.10}$$

This is the most commonly used kernel [39]. It is a smooth kernel and is infinitely differentiable.

Rational Quadratic

The mathematical expression is given below:

$$k_{RQ}(u,u') = \sigma_m^2 \left(1 + \frac{(|u-u'|)^2}{2\gamma l^2} \right)^{-\gamma}, \tag{2.11}$$

where γ is a scaling factor. The rational quadratic kernel behaves as a sum of squared exponential kernels with different lengthscales. The lengthscales are varied using the γ hyperparameter. The rational quadratic kernel meets the squared exponential kernel as $\gamma \to \infty$.

Matérn

The mathematical expression is given below:

$$k_{Mat,\nu}(u,u') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}(|u-u'|)}{l} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}(|u-u'|)}{l} \right), \tag{2.12}$$

where $\nu > 0, \Gamma(\cdot)$ is the gamma function and K_ν is a modified Bessel function. Unlike other kernels, this function gives a class of kernels by changing the value of ν. As $\nu \to \infty$, the kernel approaches a squared exponential kernel. As $\nu \to 0$, the kernel approaches an exponential kernel. The two popular kernels from this class are given below:

$$k_{Mat,\nu=\frac{3}{2}}(u,u') = \left(1 + \frac{\sqrt{3}(|u-u'|)}{l} \right) \exp\left(-\frac{\sqrt{3}(|u-u'|)}{l} \right), \tag{2.13}$$

$$k_{Mat,\nu=\frac{5}{2}}(u,u') = \left(1 + \frac{\sqrt{5}(|u-u'|)}{l} + \frac{5(|u-u'|)^2}{3l^2} \right) \exp\left(-\frac{\sqrt{5}(|u-u'|)}{l} \right). \tag{2.14}$$

Periodic

The mathematical expression is given below:

$$k_{per}(u,u') = \sigma_m^2 \exp\left(-\frac{2sin^2\left(\frac{|u-u'|}{p}\right)}{l^2}\right), \tag{2.15}$$

where p controls the periodicity. For example, $p = 2$ implies a periodicity of 2π. This kernel models the periodic behaviour of the unknown function.

All of the above kernels, except the linear, are a function of $(u - u')$ and are *stationary* kernels. These are invariant to the translations in the input domain. The kernels which are a function of $|u - u'|$ are called *isotropic* e.g. the k_{SE} kernel. Other useful covariance kernels can be derived e.g. by adding or multiplying two covariance kernels results in a new covariance kernel [39].

2.2 Regression Equations

GP can be built upon to solve the regression problems. This is achieved by modelling a GP prior, through the mean function and the covariance kernel. The posterior is calculated using a given data, also called the *training data*. Given the prior GP mean and covariance kernel and the training data, the predictive distribution is determined using the properties of the GP models. Consider the following measurement model:

$$\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{u}) + \boldsymbol{v}, \quad \boldsymbol{v} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_N), \tag{2.16}$$

where $\boldsymbol{z} = [z_1, z_2, \cdots, z_N]^T$ represents the one-dimensional measured output vector, $\boldsymbol{v}$ denotes the additive zero-mean Gaussian measurement noise vector whose elements are independent and identically distributed (i.i.d) with variance σ^2 and $\boldsymbol{I}_N$ is an N-dimensional identity matrix. A regression problem aims to estimate the function vector at a test input vector $\boldsymbol{u}^\star$ using the training data of the GP i.e. $\boldsymbol{z}$ and $\boldsymbol{u}$. The elements of $\boldsymbol{u}^\star$ can be from the training data vector i.e. $\boldsymbol{u}$ or different from it. The various realizations of a GP are mutually Gaussian distributed. Based on this property, the training and test outputs can be written as given below:

$$\begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{f}(\boldsymbol{u}^\star) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{m}(\boldsymbol{u}) \\ \boldsymbol{m}(\boldsymbol{u}^\star) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{uu}} + \sigma^2 \boldsymbol{I}_N & \boldsymbol{K}_{\boldsymbol{uu}^\star} \\ \boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}} & \boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}^\star} \end{bmatrix} \right), \tag{2.17}$$

where $\boldsymbol{f}(\boldsymbol{u}^\star)$ is the GP regression output at $\boldsymbol{u}^\star$. The GP regression equations given below:

$$\mathbb{E}[\boldsymbol{f}(\boldsymbol{u}^\star)] = \boldsymbol{m}(\boldsymbol{u}^\star) + \boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}}(\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}} + \sigma^2\boldsymbol{I}_N)^{-1}(\boldsymbol{z} - \boldsymbol{m}(\boldsymbol{u})), \tag{2.18}$$

$$\mathbb{C}[\boldsymbol{f}(\boldsymbol{u}^\star)] = \boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}^\star} - \boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}}(\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}} + \sigma^2\boldsymbol{I}_N)^{-1}\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}^\star}, \tag{2.19}$$

where $\mathbb{E}[\boldsymbol{f}(\boldsymbol{u}^\star)]$ and $\mathbb{C}[\boldsymbol{f}(\boldsymbol{u}^\star)]$ denote, respectively, the mean and covariance of the test output vector and $\cdot^{-1}$ is the matrix inverse. The computational complexity of the GP regression is $\mathcal{O}(N^3)$ due to the matrix inverse.

2.3 Learning of Hyperparameters

GP regression is a powerful model-free method. The GP regression (2.18) and (2.19) requires the calculation of different covariance matrices. These covariance matrices, determined using (2.4), depend on the modelled covariance kernel and the chosen hyperparameters. Although this approach is flexible, the efficiency can be improved by selecting the hyperparameters based on the training data. This process of hyperparameter determination from the training data is called *learning*. Further improvements can be made by automating the selection of the covariance kernel, from a family of kernels, using the training data. This is termed as *model selection* [39]. In this thesis, the model selection has not been used in any of the proposed methods. The learning is achieved through the maximization of the likelihood. The log of the marginal likelihood is given below:

$$\begin{aligned}\log p(\boldsymbol{z}|\boldsymbol{u}, \boldsymbol{\eta}, \sigma^2) = &-\frac{1}{2}\boldsymbol{z}^T(\boldsymbol{K}(\boldsymbol{u}, \boldsymbol{u}, \boldsymbol{\eta}) + \sigma^2\boldsymbol{I}_N)^{-1}\boldsymbol{z} - \frac{1}{2}\log|\boldsymbol{K}(\boldsymbol{u}, \boldsymbol{u}, \boldsymbol{\eta}) + \sigma^2\boldsymbol{I}_N| \\ &- \frac{N}{2}\log 2N,\end{aligned} \tag{2.20}$$

where $\boldsymbol{\eta}$ denotes the GP mean function and covariance kernel hyperparameters vector and $|\cdot|$ is the matrix determinant. The notation of the GP covariance matrix is modified to show its dependence on the hyperparameters. The above marginalization refers to the marginalization of the unknown function $\boldsymbol{f}(\boldsymbol{u})$. Maximizing (2.20) with respect to the hyperparameters, including the measurement noise variance hyperparameter, is a non-convex optimization problem. The point estimates of the hyperparameters are obtained using this method. This method can be slow depending upon the computational complexity of the optimization technique. An online learning method has been proposed in [40] using a state space model and a recursive filter. This method provides the uncertainty of the hyperparameters, in addition to the mean (point) estimates.

Chapter 3

Multiple Point Target Tracking

Multiple target tracking (MTT) is classified as multiple point, extended, and group target tracking in this thesis. This chapter introduces and formulates the problem of point target tracking. It also gives a brief overview of various methods proposed for solving this problem.

3.1 Point Target Tracking

Point target tracking relates to the kinematics and attributes estimation of multiple point targets. The size of the targets is small as compared to the sensor resolution cell. As a result, scarce or, in most cases, a single sensor measurement from a single target is received in a particular measurement sample. A point target is explained in Fig. 3.1. The dashed lines represent the width of a single beam of electromagnetic energy directed by the sensor and the dotted lines denote the range bins. The polygon, made by the beam and the range bins, is called a sensor resolution cell. Typically, a target occupying a single resolution cell is represented by a single measurement or a point (filled circle) and is called a point target. Typically in the point target tracking methods, including those proposed in this thesis, the following assumptions are made:

PT1 A single measurement is available per target per sample.

PT2 The states of each target evolve independent of the states of other targets.

In this thesis, the focus is on the point targets kinematics estimation.

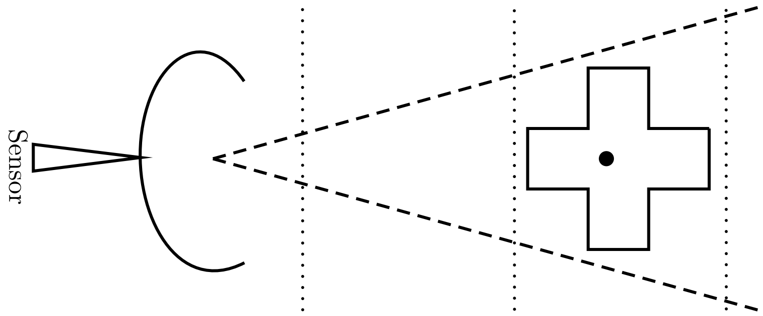


Fig. 3.1 **Point target**. This figure shows a point target (plus shape) illuminated by a sensor.

3.1.1 Kinematics Estimation

The kinematics estimation requires a sensor scanning a region of interest for the detection of the targets. The target detections, also called sensor measurements, are processed by a tracking algorithm for the state estimation. The point target tracking algorithm relies on a decision and a filtering step for the required kinematics estimation. The decision step, also called data association, involves the determination of the number of targets and their associations to the available measurements. Additionally, it also determines the false or clutter measurements. An additional challenge common to this step is the determination of missed detections. All practical sensors have a probability of detection value less than 1. Consequently, the measurements of some of the targets are missed by the sensor randomly in each sample and these are called missed detections. The filtering step involves state estimation using the predicted state and the assigned measurements. Typically, the kinematics estimation relies on state prediction and filtering steps. The state prediction requires prediction of the states at a future time sample, for which the measurements have not been received. The state filtering uses the predicted state and the noisy measurement at the current time to filter the state at the current time. The term 'filter 'means filtering out of the measurement noise.

The kinematics states include position, velocity, and higher order positional derivatives (with respect to time). Typically, the sensor provides a subset of the positional measurements. The active radars provide a complete set of positional measurements such as range, bearing, and altitude. The passive radars generally report a subset of position measurement, that is they typically do not give range measurements. The measurements are directly or indirectly related to some or all of the states. The estimates of the indirectly related states are inferred from the measurements using this (indirect) functional relationship. For example, the relation of the target velocity to the position is used to estimate the velocity. Similarly, the bearing measurements are used to determine the target position (range and bearing) in the bearing only tracking applications [41]. Linear and non-linear filters are employed based on the type of relation between the measurements and the corresponding states.

Data association is one of the hardest problems of the point target tracking. Various soft and hard decision based approaches have been proposed [42]. A Gaussian process (GP) based approach has also been proposed using an overlapping mixture of Gaussian process [37]. Sometimes the attribute information aid in the data association process [43]. For example, the airliners transmit their identification and other attributes that are received by the radars on the ground. This information aids in determining the correct measurement to target assignment. In this thesis, a GP based approach is proposed for the filtering process, and the data association is assumed known. The data association methods are hence not covered in this chapter.

3.2 Bayesian Inference

Consider a sensor reports measurement vector $\boldsymbol{z}_k$ at sample number k. It is assumed that the measurement vector belongs to a single point target and a single measurement per sample is received. The objective of the tracker or estimator is to determine the target state vector $\boldsymbol{x}_k$. Typically, the problem is formulated in a Bayesian framework and the state is estimated using Bayesian inference methods. Bayesian inference is a statistical inference method that updates the probability of a hypothesis as more and more information is received. It relies on the hypothesis prior probability and a measurement likelihood function to determine the hypothesis posterior probability. The posterior probability encapsulates all the required information of the estimate e.g. its mean, mode, median, etc. The corresponding uncertainty is also derived from the posterior. These methods are popular in decision based systems which require both the mean and the uncertainty measures, as a pre-requisite.

Let the hypothesis be denoted as h and the measurement as z. The posterior probability is updated using the Bayes formula as given below:

$$p(h|z) = \frac{p(z|h)p(h)}{p(z)} = \frac{p(z|h)p(h)}{\int p(z|h)p(h)dh} \quad , \tag{3.1}$$

where $p(\cdot)$ is the probability density function, $p(h|z)$, $p(z|h)$, and $p(h)$ denote, respectively, the posterior, likelihood, and the prior and $p(z)$ is the marginal likelihood or evidence. The evidence can be computationally expensive to determine. Moreover, a closed form solution to the inference may not be always possible.

3.2.1 Bayesian Estimation for Point Target Kinematics Tracking

Point target tracking requires state estimation of sequential measurement data. Bayesian inference can be done on incomplete data, that is, the estimates can be calculated on the sequential data. In such cases, Bayesian inference provides a solution through the recursive update of the posterior probability. This is also called Bayesian recursion. The objective of the point targets kinematics estimation, using the Bayesian inference, is to calculate the posterior pdf of all the states, given all the measurements up to the sample k. The mathematical expression of this recursion is given below:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k, \boldsymbol{z}_{1:k-1})p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})}, \tag{3.2}$$

where $\boldsymbol{x}$ and $\boldsymbol{z}$ denote, respectively, the state and the measurement vectors, $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$, $p(\boldsymbol{z}_k|\boldsymbol{x}_k, \boldsymbol{z}_{1:k-1})$, $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})$ and $p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})$ represent, respectively, the posterior, likelihood, prior and evidence and $1 : a$ denotes the samples $1, 2, \cdots, a$. The likelihood at k is assumed conditionally independent of the previous measurements given the state at k. This assumption is mathematically written below:

$$p(\boldsymbol{z}_k|\boldsymbol{x}_k, \boldsymbol{z}_{1:k-1}) = p(\boldsymbol{z}_k|\boldsymbol{x}_k). \tag{3.3}$$

It is further assumed that the states evolve in time according to a Markovian process such that,

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{1:k-1}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}), \tag{3.4}$$

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_{1:k-1}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}). \tag{3.5}$$

Using (3.5), the prior is determined as given below:

$$\begin{aligned} p(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) &= p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_{1:k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) \\ &= p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}). \end{aligned} \tag{3.6}$$

Integrating both sides with respect to $\boldsymbol{x}_{k-1}$ gives the Chapman-Kolmogrov [44] equation as given below:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1}, \tag{3.7}$$

where $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$ represents the posterior at sample $k-1$ and $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ is the state evolution. Plugging (3.7) and (3.3) in (3.2) gives:

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) &= \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)\int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1}}{p(\boldsymbol{z}_k|\boldsymbol{x}_k)}, \\ &= \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)\int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1}}{\int p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1}}. \end{aligned} \tag{3.8}$$

The various components of (3.8) are popularly referred to as given below:

$$\begin{aligned} &\text{posterior pdf at k} = \text{likelihood at k}\times \\ &\qquad \frac{\int \text{state evolution from } k-1 \text{ to } k \times \text{posterior pdf at } k-1}{\text{Normalization constant or Evidence}}. \end{aligned} \tag{3.9}$$

The likelihood (also called measurement model) and the state evolution models are design parameters of the Bayesian recursion. The product of the prior pdf and the state evolution is also called *predictive distribution*. The posterior distribution can be processed to determine the required estimates. For example, a minimum mean square error estimate of the state is denoted as $\boldsymbol{x}_k^{MMSE}$ and given below:

$$\boldsymbol{x}_k^{MMSE} = \mathbb{E}[\boldsymbol{x}_k|\boldsymbol{z}_{1:k}] = \int \boldsymbol{x}_k p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})d\boldsymbol{x}_k, \tag{3.10}$$

where $\mathbb{E}[\cdot]$ denotes the mathematical expectation operator. The Bayesian recursion is summarized in Fig. 3.2. The initial state distribution is required to process the first measurement sample only. The two models, the state evolution and measurement models, specify the prior, the likelihood and the normalization constant of the Bayesian inference. The mismatch between the models and the real world are captured by the respective model input disturbances. The Bayesian inference gives the required posterior distribution which is processed further to give the required estimate.

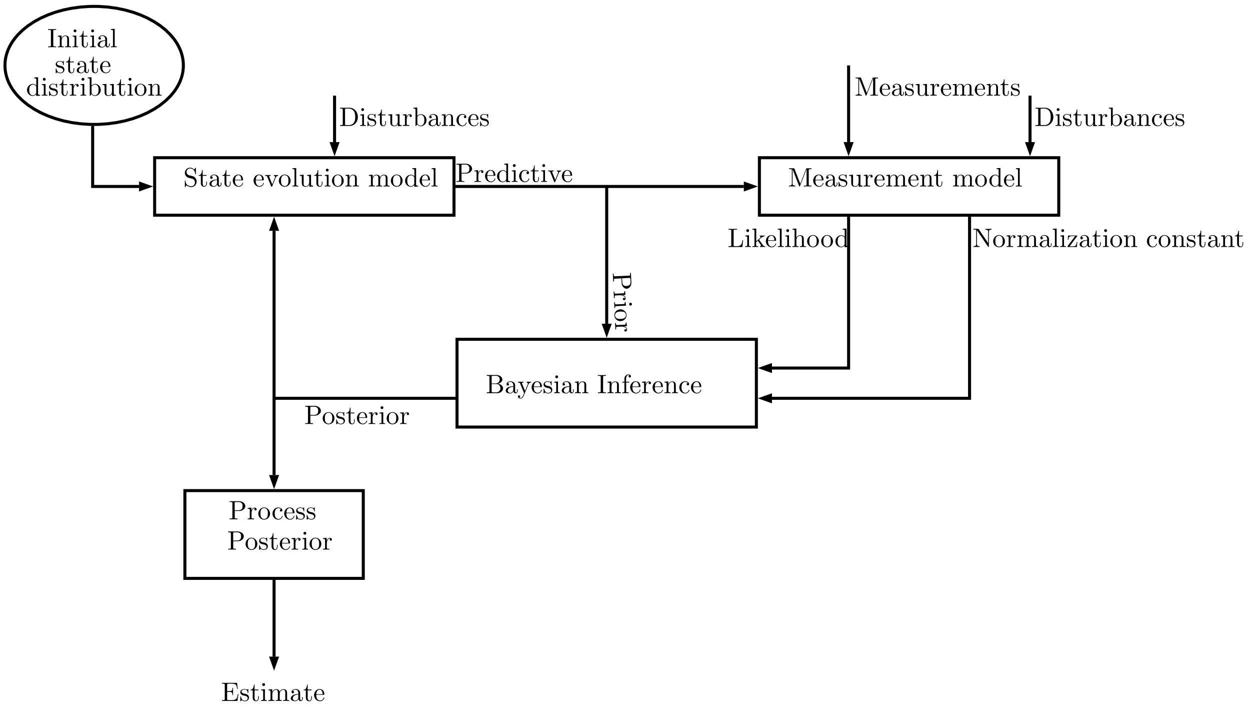


Fig. 3.2 **Bayesian Recursion**. This figure shows flow diagram of Bayesian recursion for the kinematics estimation of the point target.

3.3 Inference Methods

In this section, various Bayesian inference methods used for the point target tracking are given. The Bayesian recursion is a generic state estimation method based upon Markovian assumptions. Various linear and non-linear inference methods have been proposed using the Bayesian recursion. Some of the popular discrete time inference methods are given below:

3.3.1 Kalman Filter

Kalman filter [45] is a linear filter which determines the first two moments of the posterior distribution using the Bayesian recursion. It is based upon the following assumptions:

KF1 The state evolution is known and linear.

KF2 The state evolution is driven by a known input and an additive process noise.

KF3 The measurement model is known and is a linear function of the states.

KF4 The measurement and the process noise are additive, zero mean white (uncorrelated) and their respective covariances are known.

KF5 The process and the measurement noise are mutually uncorrelated.

KF6 The initial state is random with a known mean and error covariance.

KF7 The initial state errors are uncorrelated with the process and the measurement noises.

The Kalman filter is based upon the following linear state-space model[1]:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{F}_k \boldsymbol{x}_k + \boldsymbol{w}_k, \quad \boldsymbol{w}_k \sim \mathcal{N}(0, \boldsymbol{Q}_k), && (3.11) \\ \boldsymbol{z}_k &= \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{R}_k), && (3.12) \end{aligned}$$

where $\boldsymbol{F}_k$ and $\boldsymbol{H}_k$ represent, respectively, the state update and the measurement matrices, $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ denote, respectively, the additive process and the measurement noise vectors and $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are the process and the measurement covariance matrices, respectively. Based on the above assumptions, the Kalman filter is a minimum mean square error (MMSE) estimator for the Gaussian case, that is, the initial state error covariance and all the noises are Gaussian. For the case of non-Gaussian, it is the best linear MMSE estimator, that is, it is the best among all the linear estimators [42]. The Kalman filter recursion is given below:

$$\begin{aligned} \boldsymbol{x}_{k+1|k} &= \boldsymbol{F}_k \boldsymbol{x}_{k|k}, && (3.13) \\ \boldsymbol{P}_{k+1|k} &= \boldsymbol{F}_k \boldsymbol{P}_{k|k} \boldsymbol{F}_k^T + \boldsymbol{Q}_k, && (3.14) \\ \boldsymbol{S}_{k+1} &= \boldsymbol{R}_{k+1} + \boldsymbol{H}_{k+1} \boldsymbol{P}_{k+1|k} \boldsymbol{H}_{k+1}^T, && (3.15) \\ \boldsymbol{U}_{k+1} &= \boldsymbol{P}_{k+1|k} \boldsymbol{H}_{k+1}^T \boldsymbol{S}_{k+1}^{-1}, && (3.16) \\ \boldsymbol{x}_{k+1|k+1} &= \boldsymbol{x}_{k+1|k} + \boldsymbol{U}_{k+1}(\boldsymbol{z}_{k+1} - \boldsymbol{H}_{k+1} \boldsymbol{x}_{k+1|k}), && (3.17) \\ \boldsymbol{P}_{k+1|k+1} &= (\boldsymbol{I} - \boldsymbol{U}_{k+1} \boldsymbol{H}_{k+1}) \boldsymbol{P}_{k+1|k}, && (3.18) \end{aligned}$$

where $\boldsymbol{P}$ denotes the state error covariance matrix, $(\cdot)_{k|k}$ and $(\cdot)_{k+1|k}$ represent, respectively, the filtered and one-step predicted state, $\boldsymbol{S}_{k+1}$ is the innovation covariance and $\boldsymbol{U}_{k+1}$ represents the Kalman gain.

[1] The control input is assumed zero in the model and this thesis.

3.3.2 Extended Kalman Filter

The extended Kalman filter (EKF) [42] is a suboptimal filter for the case of non-linear system evolution or measurement model or both. The rest of the Kalman filter assumptions, **KF2**, **KF4**, **KF5**, **KF6** and **KF7**, are valid for the EKF as well. It is based upon the linearization of the non-linear functions at the estimated state using the Taylor series expansion. The first order expansion gives the first order EKF and the second order expansion gives the second order EKF. The filter may diverge as the linearization is not done on the true state, which is unknown. Another reason for the degraded performance is that the higher order terms of the Taylor series are neglected. The EKF is also sensitive to the initial conditions. The non-linear state space model is given below:

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k) + \boldsymbol{w}_k, \quad \boldsymbol{w}_k \sim \mathcal{N}(0, \boldsymbol{Q}_k), \tag{3.19}$$

$$\boldsymbol{z}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{R}_k), \tag{3.20}$$

where $\boldsymbol{f}_k$ and $\boldsymbol{h}_k$ denote, respectively, the non-linear state evolution and the measurement functions. The EKF recursion is given below:

$$\boldsymbol{x}_{k+1|k} = \boldsymbol{f}_k(\boldsymbol{x}_{k|k}), \quad \boldsymbol{\mathcal{J}}^f_{k+1} = \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}=\boldsymbol{x}_{k|k}}, \quad \boldsymbol{\mathcal{J}}^h_{k+1} = \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}=\boldsymbol{x}_{k+1|k}}, \tag{3.21}$$

$$\boldsymbol{P}_{k+1|k} = \boldsymbol{\mathcal{J}}^f_{k+1}\boldsymbol{P}_{k|k}(\boldsymbol{\mathcal{J}}^f_{k+1})^T + \boldsymbol{Q}_{k+1}, \tag{3.22}$$

$$\boldsymbol{S}_{k+1} = \boldsymbol{\mathcal{J}}^h_{k+1}\boldsymbol{P}_{k+1|k}(\boldsymbol{\mathcal{J}}^h_{k+1})^T + \boldsymbol{R}_{k+1}, \tag{3.23}$$

$$\boldsymbol{U}_{k+1} = \boldsymbol{P}_{k+1|k}(\boldsymbol{\mathcal{J}}^h_{k+1})^T\boldsymbol{S}^{-1}_{k+1}, \tag{3.24}$$

$$\boldsymbol{x}_{k+1|k+1} = \boldsymbol{x}_{k+1|k} + \boldsymbol{U}_{k+1}[\boldsymbol{z}_{k+1} - \boldsymbol{h}_k(\boldsymbol{x}_{k+1|k})], \tag{3.25}$$

$$\boldsymbol{P}_{k+1|k+1} = \boldsymbol{P}_{k+1|k} - \boldsymbol{U}_{k+1}\boldsymbol{\mathcal{J}}^h_{k+1}\boldsymbol{P}_{k+1|k}, \tag{3.26}$$

where $\boldsymbol{\mathcal{J}}^f_{k+1}$ and $\boldsymbol{\mathcal{J}}^h_{k+1}$ denote, respectively, the Jacobian with respect to the state vector of the state update function evaluated at the current state and the measurement function evaluated at the predicted state. In point target tracking, the state evolution models are typically linear. The EKF is commonly used for the case of the non-linear measurement models such as the bearing-only tracking [41].

3.3.3 Unscented Kalman Filter

The unscented Kalman filter (UKF) is also a non-linear estimation method. The EKF approximates a non-linear function through the linearization. It is not able to achieve good approximations of the error covariances. A better estimator, as compared to the EKF, is the UKF [46, 47]. The filter approximates the mean and the covariance of the state using sigma points. These are deterministically determined[2] and are a minimal representation of the distribution. The mean and the covariance of the sigma points match that of the state. The sigma points are propagated through the modelled non-linear transformations. The measurement update of the UKF on the sigma points is done using the usual Kalman filter equations. The updated sigma points are used to derive the statistics of the estimate for example the mean, covariance, etc. The system model is the same as given in (3.19) and (3.20). The UKF recursion for an n^x-dimensional state vector using $2n^x + 1$ sigma points is given below [48]:

$$\boldsymbol{x}_k^0 = \boldsymbol{x}_k, \quad w_{\boldsymbol{x}}^0 = \frac{\lambda}{n^x + \lambda}, \quad w_{\boldsymbol{P}}^0 = \frac{\lambda}{n^x + \lambda} + (1 - \alpha^2 + \beta), \tag{3.27}$$

$$\boldsymbol{x}_k^i = \boldsymbol{x}_k + (\sqrt{(n^x + \lambda)\boldsymbol{P}_k})_i, \quad \text{for } i = \{1, \cdots, n^x\}, \tag{3.28}$$

$$\boldsymbol{x}_k^i = \boldsymbol{x}_k - (\sqrt{(n^x + \lambda)\boldsymbol{P}_k})_{i-n^x}, \quad \text{for } i = \{n^x + 1, \cdots, 2n^x\}, \tag{3.29}$$

$$w_{\boldsymbol{x}}^i = w_{\boldsymbol{P}}^i = \frac{1}{2(n^x + \lambda)}, \quad \text{for } n^x = \{1, \cdots, 2n^x\}, \tag{3.30}$$

$$\boldsymbol{x}_{k+1|k}^i = \boldsymbol{f}_k(\boldsymbol{x}_k^i), \quad \text{for } i = \{1, \cdots, 2n^x\}, \tag{3.31}$$

$$\boldsymbol{x}_{k+1|k} = \sum_{j=0}^{2n^x} w_{\boldsymbol{x}}^j \boldsymbol{x}_{k+1|k}^j, \tag{3.32}$$

$$\boldsymbol{P}_{k+1|k} = \sum_{j=0}^{2n^x} w_{\boldsymbol{P}}^j (\boldsymbol{x}_k^j - \boldsymbol{x}_{k+1|k})(\boldsymbol{x}_k^j - \boldsymbol{x}_{k+1|k})^T + \boldsymbol{Q}_k, \tag{3.33}$$

$$\boldsymbol{z}_{k+1|k}^i = \boldsymbol{h}_k(\boldsymbol{x}_k^i), \quad \text{for } i = \{1, \cdots, 2n^x\}, \tag{3.34}$$

$$\boldsymbol{z}_{k+1|k} = \sum_{j=0}^{2n^x} w_{\boldsymbol{x}}^j \boldsymbol{z}_{k+1|k}^j, \tag{3.35}$$

$$\boldsymbol{S}_{k+1} = \sum_{j=0}^{2n^x} w_{\boldsymbol{P}}^j (\boldsymbol{z}_k^j - \boldsymbol{z}_{k+1|k})(\boldsymbol{z}_k^j - \boldsymbol{z}_{k+1|k})^T + \boldsymbol{R}_k, \tag{3.36}$$

$$\boldsymbol{C}_{k+1}^{\boldsymbol{xz}} = \sum_{j=0}^{2n^x} w_{\boldsymbol{P}}^j (\boldsymbol{x}_k^j - \boldsymbol{x}_{k+1|k})(\boldsymbol{z}_k^j - \boldsymbol{z}_{k+1|k})^T, \tag{3.37}$$

$$\boldsymbol{U}_{k+1} = \boldsymbol{C}_{k+1}^{\boldsymbol{xz}} \boldsymbol{S}_{k+1}^{-1}, \tag{3.38}$$

$$\boldsymbol{x}_{k+1|k+1} = \boldsymbol{x}_{k+1|k} + \boldsymbol{U}_{k+1}[\boldsymbol{z}_{k+1} - \boldsymbol{z}_{k+1|k}], \tag{3.39}$$

[2] Some sigma point filters such as the particle filters determine them randomly.

$$\boldsymbol{P}_{k+1|k+1} = \boldsymbol{P}_{k+1|k} - \boldsymbol{U}_{k+1}\boldsymbol{S}_{k+1}\boldsymbol{U}_{k+1}^T, \tag{3.40}$$

where $\boldsymbol{x}^i$ and z^i represent, respectively, the i^{th} sigma point state and the measurement vectors, $w_{\boldsymbol{x}}$ and $w_{\boldsymbol{P}}$ are the weight vectors corresponding to the state and the covariance, respectively, $\lambda = \alpha^2(n^x + \kappa) - n^x$ represents the scaling parameter, κ also controls the scaling and is usually set very close to zero or zero, α determines the size of the points spread around the mean, β is used to include the prior information of the state pdf[3], $\left(\sqrt{(n^x + \lambda)\boldsymbol{P}_k}\right)_i$ denotes the i^{th} row of the square root matrix and $\cdot^T$ is the matrix transpose. In point target tracking, the UKF is sometimes applied for the case of the non-linear measurement models such as the bearing-only tracking applications [49]. The second order Taylor expansion of the EKF and the UKF approximate the first and second order moments of the distribution. Although, the UKF has been claimed to be better than the EKF but this is not always true. For example, the second order EKF performs better than the UKF for the nonlinear function $\boldsymbol{x}^T\boldsymbol{x}$, as shown in [50].

3.3.4 Interacting Multiple Model Filter

The Kalman filter, EKF and UKF are single model based estimation methods. The estimation performance is improved using multiple models (MM) based filters. These methods are highly adaptive and assume that at any particular time the true target dynamics can be represented by a mode. The true target mode is captured by a set of parallel filters, each corresponding to a possible mode. These methods are classified as follows [51]:

- **Decision based methods**. The decision based methods are implemented in two steps, a decision step followed by a filtering step. The decision step picks a single filter, out of the set of filters, based on the match between the filter and the target mode. The state estimate is given by the output of the chosen filter. These types of filters are applicable if the true mode can be chosen. This is restrictive for most point target tracking applications where choosing a true target mode is hardly possible.

- **Interaction based methods**. Unlike decision based methods whose output at each sample is given by a single filter, the interaction based methods give output by combining the output of multiple filters. These are classified into

[3] β is set to 2 for Gaussian

> three generations in [51]. The first generation methods [52, 53], provide a processed output from an independently working set of filters. The second generation, popularly called interacting MM (IMM), was proposed in [54] and improved in [55]. The IMM, in addition to the processed output, provides a framework in which the filters cooperate to improve the estimate. This has been successfully applied in various tracking systems [56]. The third generation, termed as the variable structure IMM (VSIMM) [57], in addition to the output processing and filters cooperation, provides flexibility in the structure or the set of filters active at any given time instant. The first two generations are based on a fixed structure and it was assumed that the target mode lies within this structure all the time. If the mode is changing, then the first two generations provide degraded estimates in situations where the target mode lies outside this structure. The VSIMM handles this situation by providing an additional capability of selecting and de-selecting different structures.

An optimal MM estimator considers all the mode sequence hypothesis, which grow with time. Due to the hypothesis explosion, the implementation in a practical system is not possible. The computational complexity is of the order $\mathcal{O}(m^k)$, where m represents the number of modes. A general pseudo-Bayesian estimator (GPB$_n$) of order n keeps the depth of the hypothesis tree to n recent samples and has a computational complexity of order $\mathcal{O}(m^n)$. The interacting multiple model, also called Fixed Grid interacting multiple model (FGIMM) to differentiate from the VSIMM, has the computational complexity of a GPB$_1$ and the estimation accuracies of the GPB$_2$. This is the most cost effective method [42]. The FGIMM recursion for m parallel filters starts with the interaction of the state estimates and the error covariances at k using the mixing probabilities as given below [51]:

for $i = 1, \cdots, m$,

$$w^i_{k+1|k} = \sum_{j=1}^{m} \mathbf{\Pi}_{ij} w^j_k, \quad w^{j|i}_k = \frac{\mathbf{\Pi}_{ij} w^j_k}{w^i_{k+1|k}}, \tag{3.41}$$

$$\grave{\boldsymbol{x}}^i_{k|k} = \sum_{j=1}^{m} \boldsymbol{x}^j_{k|k} w^{j|i}_k, \tag{3.42}$$

$$\grave{\boldsymbol{P}}^i_{k|k} = \sum_{j=1}^{m} [\boldsymbol{P}^j_{k|k} + (\grave{\boldsymbol{x}}^i_{k|k} - \boldsymbol{x}^j_{k|k})(\grave{\boldsymbol{x}}^i_{k|k} - \boldsymbol{x}^j_{k|k})^T] w^{j|i}_k, \tag{3.43}$$

where w denotes the model probability, $\mathbf{\Pi}$ is the mode transition probability matrix and $\grave{\boldsymbol{x}}$ and $\grave{\boldsymbol{P}}$ are, respectively, the mixed state and the state error covariance of each filter. These mixed states and the error covariance are prior for the corresponding filters. The filter set can be a mix of linear and non-linear single model based filters such as the Kalman filter [58], the EKF [59], or the UKF [60]. The output of each filter is used to update the mode probabilities and subsequently combined to give the filtered state as given below:

$$w^i_{k+1|k+1} = \frac{w^i_{k+1|k}\Lambda^i}{\sum_{j=1}^r w^j_{k+1|k}\Lambda^j}, \quad \text{for } i = 1, \cdots, r, \quad \text{where } \Lambda^i \sim \mathcal{N}(0, \boldsymbol{S}^i_{k+1}), \tag{3.44}$$

$$\hat{\boldsymbol{x}}_{k+1|k+1} = \sum_{i=1}^{r} \boldsymbol{x}^i_{k+1|k+1} w^i_{k+1|k+1}, \tag{3.45}$$

$$\hat{\boldsymbol{P}}_{k+1|k+1} = \sum_{i=1}^{r} [\boldsymbol{P}^i_{k+1|k+1} + \Delta\boldsymbol{x}_{k+1|k+1}\Delta\boldsymbol{x}^T_{k+1|k+1}] w^i_{k+1|k+1}, \tag{3.46}$$

$$\Delta\boldsymbol{x}_{k+1|k+1} = (\hat{\boldsymbol{x}}_{k+1|k+1} - \boldsymbol{x}^i_{k+1|k+1}), \tag{3.47}$$

where $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{P}}$ denote, respectively, the combined filtered state and the corresponding state error covariance and Λ represents the measurement likelihood.

3.3.5 Particle Filters

The Bayesian recursion (3.8), in most practical scenarios including the point target tracking, is intractable. The analytical solution to the integrals is not possible. The numerical integration methods are applied when accuracy matters more than the processing time, as these methods are computationally expensive. The particle filter [61] is one such method, which approximates the (prior) pdf using point masses and the posterior pdf is approximated using a Monte Carlo integration. These are also called sequential Monte Carlo (SMC) methods. These methods are not restricted by the linearity of the system models and the Gaussianity of the noise pdfs. The general non-linear state space model is given below:

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{w}_k), \tag{3.48}$$

$$\boldsymbol{z}_k = \boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{v}_k). \tag{3.49}$$

The Bayesian recursion, given in Section 3.2.1, is based upon a hidden Markov model (HMM) shown in Fig. 3.3. The state, x, evolves from the sample number 0 to the sample k and is observed through the measurement z.

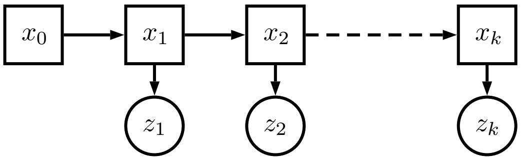


Fig. 3.3 **Bayesian Network Diagram of HMM**. This figure shows a network diagram of the hidden Markov model.

HMM is a representation of the probability distribution of the observation sequence and is characterized by two properties. Firstly, the states and their evolution process are hidden or latent and are observable through a measurement process. Secondly, the state evolves under a Markov process. Based on (3.4) and (3.5) and the whiteness of the noise, the joint pdf of the states up to sample k is given below:

$$p(\boldsymbol{x}_0, \cdots, \boldsymbol{x}_k) = p(\boldsymbol{x}_0) \prod_{i=1}^{k} p(\boldsymbol{x}_i | \boldsymbol{x}_{i-1}). \tag{3.50}$$

Similarly, the whiteness of the measurement noise and the model assumption (3.3) gives the joint conditional likelihood as given below:

$$p(\boldsymbol{z}_1, \cdots, \boldsymbol{z}_k | \boldsymbol{x}_1, \cdots, \boldsymbol{x}_k) = \prod_{i=1}^{k} p(\boldsymbol{z}_i | \boldsymbol{x}_i). \tag{3.51}$$

Various particle filters have been proposed for target tracking [62]. The bootstrap particle filter [61], the first feasible version, is explained next. At a given sample k, the pdf (3.50) is approximated using n_p samples. Each sample is associated with a weight. These samples are propagated through the state evolution process and the measurement process. The measurement likelihood (3.51) is used to update the weights of the samples. The updated samples are used to approximate the posterior pdf. For a large sample size, this method approaches the optimal Bayesian estimate.

The recursive bootstrap filter at sample $k+1$ begins with the sampling of the posterior distribution at k using samples or particles. Each particle has an associated point state and an associated weight. If the posterior at k is not available then a suitable prior is elicited. The weight is initialized equally for all the particles. The point samples or particles are passed through the state and the measurement process given in (3.48) and (3.49), respectively. The likelihood function updates the weight of i^{th} particle as given below:

$$w_{k+1}^{i} = w_{k}^{i} \frac{p(z_k|\tilde{x}_{k+1|k}^{i})}{\sum_{j=1}^{n_p} p(z_k|\tilde{x}_{k+1|k}^{j})}, \tag{3.52}$$

where w^i and $\tilde{x}^i$ denote, respectively, the weight and the state vector of the i^{th} particle. The posterior can then be approximated using the following point mass function:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_k) = \sum_{j=1}^{n_p} w_k^i \delta(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_{k+1|k}^{i}), \tag{3.53}$$

where $\delta(\cdot)$ is the Dirac delta function. The posterior mean is calculated as given below:

$$\boldsymbol{x}_{k+1|k+1} = \sum_{i=1}^{n_p} w_k^i \tilde{\boldsymbol{x}}_{k+1|k}^{i}. \tag{3.54}$$

The following two problems and their corresponding solutions are common to these methods:

- **Importance Sampling**. The sampling of the prior pdf may be difficult due to its high-dimensionality and complexity. An importance sampling technique is proposed to resolve this problem. An appropriate distribution, which is easy to sample, is chosen to be a representative of the prior pdf, also called the importance density. The samples are chosen from the importance density. A well known importance density is the multivariate Gaussian. If the structure of the importance density is similar to (3.50), then it is called sequential importance sampling (SIS). The weight update of the SIS is given below:

$$w_{k+1}^{i} = w_{k}^{i} \frac{p(z_k|\tilde{x}_{k+1|k}^{i})p(\tilde{x}_{k+1|k}^{i}|\tilde{x}_{k|k}^{i})}{q(\tilde{x}_{k+1|k}^{i}|\tilde{x}_{k|k}^{i})}, \tag{3.55}$$

  where $q(\cdot)$ represents the importance density.

- **Resampling**. The particle variance increases with the increasing samples. As a result of this scattering, the weights of some particles become insignificant. In the worst case, all the particles, except one, have a weight close to zero. This is also called particle degeneracy. The solution to this problem is called resampling. This involves replacing the lower weighted particles with the copies of higher weighted particles. This could be done at every update step or when required.

3.4 State Evolution Models

This section presents the popular state evolution models used for the kinematics estimation of the point targets. The state evolution model is also referred to as state update, state transition, and process models. The point targets kinematics estimation has been proposed by assuming a linear (3.11),(3.12) or non-linear (3.19),(3.20) system evolution model. For relatively complicated scenarios, complex non-linear model of the form (3.48),(3.49) is also applied. In most target tracking applications, the measurement process provides limited information. The inference accuracy significantly depends upon the choice of the state evolution model. Hence, the choice of the system evolution model is one of the critical steps in the point target tracking application. A comprehensive survey of these models is given in [63]. The models are derived by making assumptions on the process noise. The process noise is modelled either as white noise, a Markov process, or a semi-Markov jump process. The commonly used linear models (linear with respect to the state vector), for the linear and the non-linear target motion (with respect to time), are given below:

3.4.1 Nearly Constant Velocity Model

The nearly constant velocity model (NCV), also called constant velocity (CV) in literature, is obtained by the discretization of a constant white noise acceleration model. It assumes that the target acceleration can be modeled using a white noise process. The state vector consists of the position and the velocity and the system matrices are given below:

$$\boldsymbol{F}^{NCV} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{Q}^{NCV} = q^{NCV} \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix}, \tag{3.56}$$

where T is the sampling time and q^{NCV} is the power spectral density of the white noise model. This model is preferred for targets moving under uniform motion e.g. airliners.

3.4.2 Nearly Constant Acceleration Model

This is also called a jerk model and is obtained by the discretization of a system activated by a zero mean white noise jerk process noise. It assumes that the target jerk (time-derivative of the acceleration) can be modelled using a white noise process. The state vector consists of the position, velocity, and acceleration. The mathematical model is given below:

$$\boldsymbol{F}^{NCA} = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{Q}^{NCA} = q^{NCA} \begin{bmatrix} \frac{T^5}{20} & \frac{T^4}{8} & \frac{T^3}{6} \\ \frac{T^4}{8} & \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^3}{6} & \frac{T^2}{2} & T \end{bmatrix}, \tag{3.57}$$

where q^{NCA} is the power spectral density of the white noise model. This model is useful when acceleration is constant over a longer period of time.

3.4.3 Singer Model

The Singer model [64] assumes the target acceleration is a Markov process. The white noise process models assume that the noise is uncorrelated in time. Conversely, the Markov process based models, like Singer, assume that the noise is correlated in a specific neighbourhood. The Singer model assumes that the target acceleration is a first-order Markov process and its continuous time state space form is given below:

$$\dot{a}(t) = -\alpha a(t) + w(t), \tag{3.58}$$

where $\dot{a}(t)$ and $a(t)$ are, respectively, the jerk (first order time derivative of the acceleration) and the acceleration, α is a co-efficient and $w(t)$ is a zero mean white noise process with constant power spectral density. The state vector consists of the position, velocity, and acceleration. The discrete time state space representation of the model is given below:

$$\boldsymbol{F}^{S} = \begin{bmatrix} 1 & T & \frac{\alpha T - 1 + \exp(-\alpha T)}{\alpha^2} \\ 0 & 1 & \frac{1 - \exp(-\alpha T)}{\alpha} \\ 0 & 0 & \exp(-\alpha T) \end{bmatrix}, \quad \boldsymbol{Q}^{S} = 2\alpha\sigma_a^2 \begin{bmatrix} \frac{T^5}{20} & \frac{T^4}{8} & \frac{T^3}{6} \\ \frac{T^4}{8} & \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^3}{6} & \frac{T^2}{2} & T \end{bmatrix}, \tag{3.59}$$

where $\alpha = \frac{1}{\tau_m}$ is the reciprocal of the manoeuvre time constant τ_m and σ_a^2 is the instantaneous variance of the acceleration. As τ_m increases, the Singer model becomes equivalent to the NCA model given in Section 3.4.2. For the reciprocal case, as τ_m decreases, the Singer model becomes equivalent to the NCV model given in Section 3.4.1. The instantaneous variance of the acceleration is designed using $\sigma_a^2 = \frac{a_{max}^2}{3}(1 + 4P_{max} - P_0)$, where a_{max} is maximum possible acceleration, P_{max} and P_0 denote, respectively, the probability of maximum and no acceleration.

3.4.4 Nearly Coordinated Turn Model

The nearly coordinated turn model (NCT), also called a coordinated turn (CT), models the turn trajectories of the point targets. The target velocity and the turn rate are close to a constant value. The state vector consists of the position and the velocity in two dimensions, x and y. For a known turn rate the NCT model is given below:

$$\boldsymbol{F}^{NCT} = \begin{bmatrix} 1 & \frac{\sin\omega T}{\omega} & 0 & -\frac{1-\cos\omega T}{\omega} \\ 0 & \cos\omega T & 0 & -\sin\omega T \\ 0 & \frac{1-\cos\omega T}{\omega} & 1 & \frac{\sin\omega T}{\omega} \\ 0 & \sin\omega T & 0 & \cos\omega T \end{bmatrix}, \quad \boldsymbol{Q}^{NCT} = q^{NCT} \begin{bmatrix} \frac{T^4}{4} & \frac{T^3}{2} & 0 & 0 \\ \frac{T^3}{2} & T^2 & 0 & 0 \\ 0 & 0 & \frac{T^4}{4} & \frac{T^3}{2} \\ 0 & 0 & \frac{T^3}{2} & T^2 \end{bmatrix}, \quad (3.60)$$

where ω denotes the turn-rate and q^{NCT} is the process noise variance.

3.5 Measurement Models

This section discusses the measurement models of the point targets. The sensor measurements and the kinematics states are commonly modelled in a Cartesian or a polar frame. Typically, in the point target tracking, the positional measurements are available. The measurement model in such cases is linear when both the measurements and the states are in the same coordinate frame. The measurement model is non-linear when these are in different frames. For example, the sensor reports polar positional measurements and the target position is estimated in the Cartesian frame. In such cases, the measurement model is non-linear and non-linear inference methods are applied for the kinematics estimation. The above-mentioned example problem can also be solved using the linear inference method. This technique is called a converted measurement Kalman filter (CMKF) [42]. In CMKF, the measurement pdf in the polar coordinates is converted to the Cartesian. The measurement pdf in the Cartesian coordinates is then used in the Kalman filter update. The CMKF approach is sensitive to the polar to Cartesian conversion. The common issues are that it can induce a bias and may also give inaccurate covariances. A well known approach for the point target kinematics estimation is the modified unbiased converted measurement [65]. Further details are given in Appendix A. In some cases, the sensor may provide limited positional information. For example, passive sensors[4] report the bearing and sometimes the bearing rate of the target. The range of the target is estimated by the filter using these bearing related measurements. Such cases are not considered in this thesis.

[4] A passive sensor receives the electromagnetic radiation from the target and reports the angle.

Chapter 4

Multiple Extended and Group Target Tracking

Multiple target tracking (MTT) is classified as multiple point, extended, and group target tracking in this thesis. Chapter 3 gives an overview of the point target tracking methods. This chapter describes the multiple extended and group target tracking methods.

4.1 Extended and Group Target Tracking

Multiple extended target tracking relates to the kinematics, attributes, and feature estimation of multiple extended targets using noisy measurements. An extended target problem arises when the assumption (**PT1**) of the point target tracking is no more valid. Contrary to this assumption, the extended target is large as compared to the sensor resolution cell. It occupies multiple cells, due to which, multiple measurements per target per sample are possible. This is explained in Fig.4.1. The dashed lines represent ends of the multiple beams of the electromagnetic energy directed by the sensor. The dotted lines denote the range bins. The polygon made by the beam endings and the range bins is called a sensor resolution cell. Typically, all targets within a single cell are represented by a single measurement (denoted by a filled circle). The target occupies multiple resolution cells and multiple measurements are made available by the sensor per sample. Tracking of a target in this scenario is called extended target tracking. As a result of multiple measurements, the spatial features of the target can also be estimated, in addition to the kinematics and the attributes.

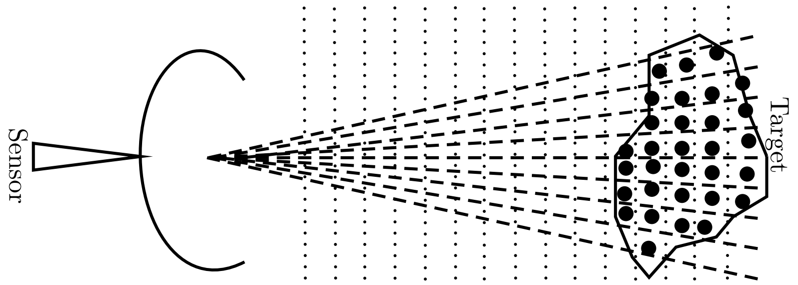


Fig. 4.1 **Extended target**. The figure shows an extended target illuminated by a sensor.

The multiple group target tracking relates to the kinematics, attributes and feature estimation of multiple group targets using noisy measurements. This scenario occurs when the assumption (**PT2**) of the point target tracking is no more valid. Contrary to this assumption, the kinematics of the multiple point targets are correlated. The aim of the group target tracking is to estimate the average group kinematics and the group extent as shown in Fig.4.2. The dashed lines represent the ends of the multiple beams of the electromagnetic energy directed by the sensor. The dotted lines denote the range bins. The polygon made by the beam endings and the range bins is called a sensor resolution cell. Typically, all targets within a single cell are represented by a single measurement (denoted by a filled circle). Each target is smaller than the resolution cell and is called a point target. The multiple point targets can be represented by a single "triangle like shaped" group target (thick solid line). The group target kinematics is the mean of the point targets kinematics. The group extent relates to the determination of the smallest polygon enclosing all the point targets belonging to a single group. In some applications, group orientation is also estimated.

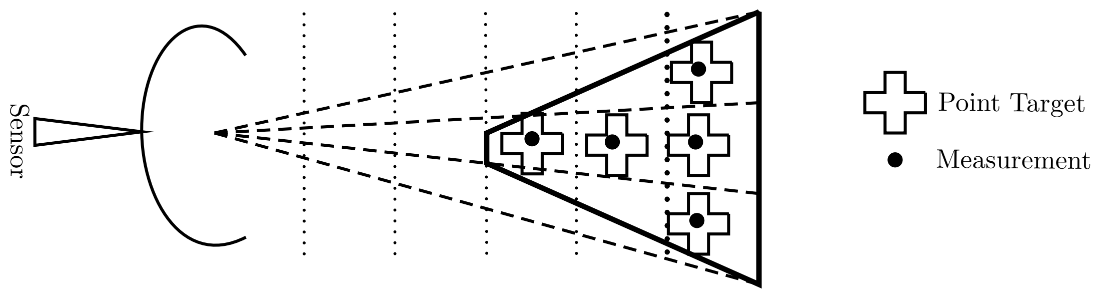


Fig. 4.2 **Group target**. The figure shows multiple point targets illuminated by a sensor.

The pioneering survey on the subject of group tracking is given in [66]. A bibliographical survey of the related methods up-till the year 2004 is given in [67]. The methods for multiple extended and group target tracking are covered in two comprehensive survey papers [24] and [25]. A recent survey focused on the application of the machine learning methods to extended and group target tracking is given in Paper VII [7]. The methods for small group targets model the interactions among the point targets. The large group target tracking methods do not model these interactions. As a result, the large group target tracking problem and the extended target tracking problem become similar. In this thesis, the large group and the extended target tracking problems are considered. The pre-fix "large" is not used from here-on with the group target for brevity.

The multiple extended and group target tracking, similar to the multiple point target tracking, requires a solution to two sub-problems. These are the data association and state estimation. The attributes estimation is not considered for the extended and group targets in this thesis. Typically, for the group target tracking, each point target is assumed to produce a single measurement per sample. A one-to-one correspondence can be made between the number of the point targets and the measurements, provided there are no missed detections. For the case of the extended targets, the measurements are assumed to originate from an infinite [26, 3] or finite [68] number of points on the target. These points are also called the *measurement sources* or *reflection points*. In this thesis, infinite measurement sources are assumed. The challenge in the infinite measurement sources case is that it is practically impossible to receive measurements of all the infinite measurement sources. Moreover, the measurement sources in the consecutive samples can be mutually exclusive. The infinite measurement source problem, like the large group tracking problem, is modelled by considering no interaction among the measurements between samples. Both the problems, the data association and state estimation, are discussed in this chapter.

4.2 Important Terminologies

A couple of important terminologies, common to the extended and group target tracking literature, are introduced in this section below:

- **Surface and Contour Measurements**. The measurement sources are assumed to lie either on the whole body or only at the boundary of the target. The measurements arising in the former case are called the *surface* whereas in the latter case are termed as the *contour*. The surface measurement case has additional complexities, as compared to the contour case. These include classifying measurements into contour and non-contour measurements and the target extent estimation using non-contour measurements. The models explained in this chapter assume contour measurements unless specified for the case of the surface measurements.

- **Rigid and Non-rigid Targets**. The target shape does not change for the *rigid* targets, whereas it can change for the *non-rigid* targets. Tracking of a car is an example of a rigid target tracking whereas tracking of a cloud is an example of a non-rigid target tracking.

4.3 Single Extended and Group Target Tracking

In this section, the state estimation of a single extended or group target is presented. The data association is assumed known and no clutter is assumed. Once the data association is resolved, the methods given in this section can be applied to estimate the states of the multiple targets. The states of interest in an extended and group target are the kinematics, attributes, and features. In this thesis, the attributes estimation is not considered. The kinematics and features are estimated using a sequence of noisy measurements. For the state estimation of extended or group target, multiple measurements are available per sample.

4.3.1 Problem Formulation

Consider a sensor that reports a measurement vector $\boldsymbol{z}_k$ at sample number k. It is assumed that all the measurements belong to a single target. The objective of the tracker is to estimate the state vector $\boldsymbol{x}_k$. Typically, the problem is formulated in a Bayesian framework and solved using a Bayesian recursion, as explained in Sections 3.2 and 3.2.1, respectively. After the initialization using a suitable prior density, the important steps of the Bayesian recursion are the predictive and the posterior density calculation. These are obtained using the Chapman-Kolmogorov (3.7) and the Bayesian inference equations (3.8), respectively. Given the measurement vector $\boldsymbol{z}_k$, the design elements are the state vector $\boldsymbol{x}_k$, the state evolution model $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ and the measurement likelihood $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$. Based on the above models, the linear or non-linear recursive Bayesian inference is used for the sequential state estimation, as described in Section 3.3. The focus of the extended and group target tracking research is upon the states, the state evolution and the measurement models.

4.3.2 Centre and Extent States

Typically, the extended or group target tracking is modelled as two sub-estimation problems, the estimation of the kinematics of the centre and the extent estimation. As a result, the states of interest relate to the centre and extent of the target, as given below:

$$\boldsymbol{x}_k = \begin{bmatrix} (\boldsymbol{x}_k^C)^T & (\boldsymbol{x}_k^E)^T \end{bmatrix}^T, \tag{4.1}$$

where $\boldsymbol{x}^C$ and $\boldsymbol{x}^E$ denote, respectively, the states related to the centre and the extent. These are explained below:

1. **Centre/Average**. The states relating to the centre of the target consist of its location, kinematics, and sometimes orientation. The kinematics states include the respective time derivatives e.g. velocity, acceleration, orientation rate, etc. The orientation states are assumed uncorrelated with those of the centre. In the case of a group target, the centre states can also be interpreted as the average location and the mean kinematics of the group. For example, it can be used to determine how fast a flock of birds is flying.

2. **Extent**. The extent states consist of the shape, volume, etc. Depending on the extent model, different state vectors are obtained. These are discussed in more detail below.

4.3.3 State Evolution Model

Typically, the evolution of the centre and the extent states are modelled independent of each other [26]. A typical example model is given below:

$$\begin{bmatrix} \boldsymbol{x}_{k+1}^C \\ \boldsymbol{x}_{k+1}^O \\ \boldsymbol{x}_{k+1}^E \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_k^C & 0 & 0 \\ 0 & \boldsymbol{F}_k^O & 0 \\ 0 & 0 & \boldsymbol{F}_k^E \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^C \\ \boldsymbol{x}_k^O \\ \boldsymbol{x}_k^E \end{bmatrix} + \begin{bmatrix} \boldsymbol{w}_k^C \\ \boldsymbol{w}_k^O \\ \boldsymbol{w}_k^E \end{bmatrix}, \tag{4.2}$$

$$\begin{bmatrix} \boldsymbol{w}_k^C \\ \boldsymbol{w}_k^O \\ \boldsymbol{w}_k^E \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \boldsymbol{Q}_k^C & 0 & 0 \\ 0 & \boldsymbol{Q}_k^O & 0 \\ 0 & 0 & \boldsymbol{Q}_k^E \end{bmatrix} \right), \tag{4.3}$$

where $\boldsymbol{F}^C$, $\boldsymbol{F}^O$ and $\boldsymbol{F}^E$ represent, respectively, the state evolution of the centre, orientation and extent states, $\boldsymbol{w}^C$, $\boldsymbol{w}^O$ and $\boldsymbol{w}^C$ are, respectively, the centre, orientation and extent process noise vectors with corresponding process noise covariance matrices denoted by $\boldsymbol{Q}^C$, $\boldsymbol{Q}^O$ and $\boldsymbol{Q}^E$, respectively.

The state evolution of the centre and orientation of the target is typically modelled in a similar way to the point target state evolution, as given in Section 3.4. Simple models have been proposed for the state evolution of the extent states. The most common are a random-walk or a damped-random-walk around the current states. The different extent states are assumed mutually uncorrelated. The random-walk and the damped-random-walk models are given below:

$$\boldsymbol{F}^E_{rw} = \boldsymbol{I}_{n^e}, \tag{4.4}$$

$$\boldsymbol{F}^E_{drw} = \exp(-\alpha_{drw}T)\boldsymbol{I}_{n^e}, \tag{4.5}$$

where $\cdot_{rw}$ and $\cdot_{drw}$ denote, respectively, the variables belonging to the random-walk and the damped-random-walk models, α_{drw} represents the damping coefficient and n^e is the number of extent states. These two efficiently model the case of rigid targets. The performance may degrade for tracking of the non-rigid targets.

4.3.4 Extent Models

Recently, the main focus of the extended and group target tracking has been on the extent models. Typically, all the models assume either a static target or estimate the location of the centre. The extent is modelled with respect to the location of the target's centre. Typically the centre is modelled in two or three dimensional Cartesian frame. Assuming the centre is known, the extent models require estimation of the target extent from the target centre in all directions. A wide variety of models have been proposed which are classified based upon the complexity of the target shape. These are given below:

1. **Basic or Regular Shape**. These models assume or fit a basic geometrical shape on the target. The example models are stick [69], circle [69], rectangle [70], and ellipse [71, 72].

2. **Arbitrary or Irregular Shape**. These models assume or fit an irregular shape on the target extent. The example models are Gaussian process [26], random hypersurface model [73], a mixture of ellipsoids [74], and a mixture of subtargets [75]. The Gaussian process and the random hypersurface models

assume a star-convex[1] target extent. The radial extent of the target is modelled as a non-linear function of the angle from its centre as given below:

$$r = f(\theta), \tag{4.6}$$

where r denotes the radial extent, f is the unknown non-linear function and θ represents the corresponding angle. This function f is explained in Fig.4.3. The left axis shows an extended target (thick solid line) with its centre at the origin. A point p (filled circle) on the contour is at polar coordinates (r, θ). As we move this point on the contour of the target, the radial value changes as a function of the angle. The right axis plots this non-linear function as $r = f(\theta)$. The function is periodic with time period 2π. The star-convex extent models are based on the estimation of the periodic non-linear function f. These models are explained below:

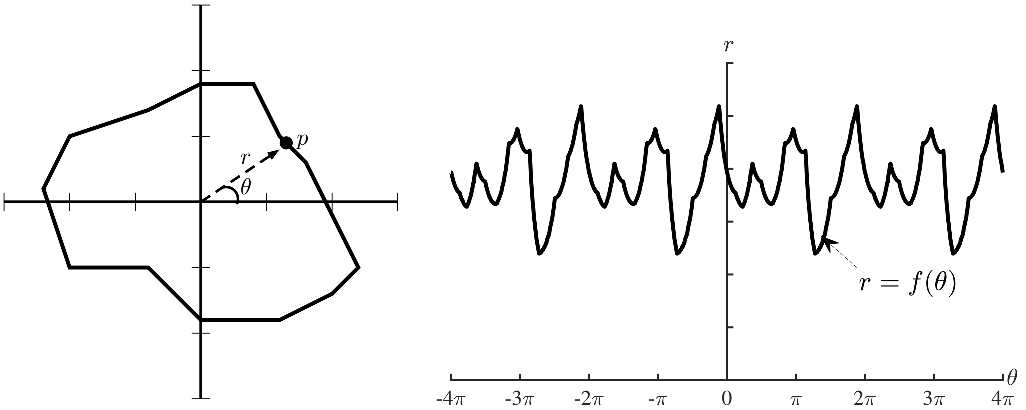


Fig. 4.3 **Arbitrary (star-convex) extent model**. The figure explains the arbitrary extent model of the extended target.

The tracking performance using basic geometric shape based models has been demonstrated on real data, e.g. tracking of a cyclist using a stick model [25], a car using a rectangular model [76], a ship using an ellipsoidal model [77], etc. Although these methods have been proved to be simple and efficient on real-world data, the shapes of these and other extended targets are different from the basic geometric shapes. Better target shape estimates give better tracking performance [25] especially in challenging environments such as low SNR [78].

[1] A star-convex set is a set S in the Euclidean space, if there exists a point $s_0 \in S$ such that the line segments from s_0 to all the points in S are all contained in S. In case of the extended or group targets the centre of the target is assumed as the arbitrary point s_0.

4.3.5 Gaussian Process Extent Model

The Gaussian process extent model was proposed in [26]. A Gaussian process is a statistical regression method that relies on a given training data to learn the unknown non-linear function. The proposed approach assumes that the model parameters (also called hyperparameters) are known. The Gaussian process model is explained in Chapter 2. The non-linear function f is modelled as given below:

$$f(\theta) \sim GP(m(\theta), k(\theta, \theta')). \tag{4.7}$$

The function is sampled at fixed points, also called the inducing or keypoints, distributed evenly on the input space θ, between 0 and 2π as shown in Fig. 4.4. The unknown non-linear extent function (thick solid line) is estimated using $B = 8$ equidistant inducing points (circle). The measurements (cross) at each sample are used to update the inducing points. These inducing points and the Gaussian process predict the non-linear extent function on the complete input space.

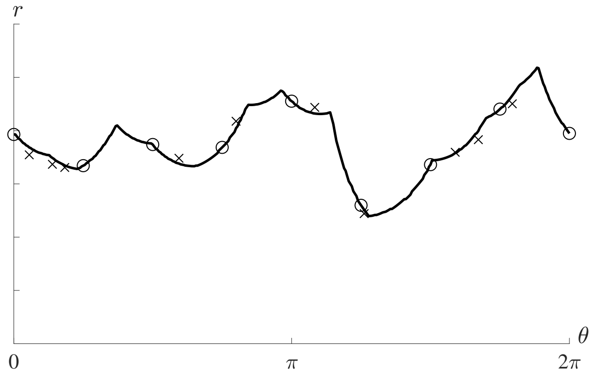


Fig. 4.4 **Gaussian process extent model**. The figure explains the Gaussian process extent model.

The functional values at the inducing points are the extent states of the state vector. For B inducing points, the extent state vector is given below:

$$\boldsymbol{x}_k^E = \begin{bmatrix} f_k(\theta_1) & f_k(\theta_2) & \cdots & f_k(\theta_B) \end{bmatrix}^T, \tag{4.8}$$

where $f_k(\theta_i)$ denotes the functional value or the target radial extent at the i^{th} inducing point. The measurements at each sample are used as the training data for updating the (inducing point) extent states using the Gaussian process. The trained Gaussian process predicts the radial values on the complete input space, subsequently. The extent state evolution is modelled using a damped-random-walk model (4.2) and (4.5). The state vector proposed in [26] consists of the centre states and the orientation, in addition to the inducing point states. The state evolution of these states is modelled using an NCV model, as given in Section 3.4.1. An extension of the model is also proposed for the case of the surface measurements.

4.3.6 Random Hypersurface Extent Model

The random hypersurface extent model was proposed in [73]. This model parametrizes the unknown extent function f using a Fourier series expansion of the radial extent function, as given below:

$$r(\psi, \boldsymbol{x}_k^E) = \frac{1}{2}a_k^0 + \sum_{j=1}^{B} \left(a_k^j \cos(j\psi) + b_k^j \sin(j\psi)\right) = \boldsymbol{r}(\psi) \cdot \boldsymbol{x}_k^E, \tag{4.9}$$

$$\boldsymbol{r}(\psi) = \begin{bmatrix} \frac{1}{2} & \cos(\psi) & \sin(\psi) & \cdots & \cos(B\psi) & \sin(B\psi) \end{bmatrix}, \tag{4.10}$$

$$\boldsymbol{x}_k^E = \begin{bmatrix} a_k^0 & a_k^1 & b_k^1 & \cdots & a_k^B & b_k^B \end{bmatrix}^T, \tag{4.11}$$

where $r(\psi, \boldsymbol{x}_k^E)$ denotes the extent radial value function, ψ is the Fourier frequency, a_k^i and b_k^i denote the Fourier coefficients, $B + 1$ is the number of coefficients, and $\boldsymbol{r}(\psi)$ is the frequency vector. The smaller frequencies (indices) estimate the course features of the extent function whereas the higher frequencies encode the finer details. The Fourier frequencies, $\boldsymbol{r}(\psi)$ are assumed known and fixed. The extent evolution is proposed using a random walk model as given in (4.2) and (4.4). An extension of the model is also proposed to handle the surface measurements.

4.3.7 Measurement Model for Star-convex Extent

The measurements are typically assumed to be either in the polar or the Cartesian coordinate system. For the case of the polar measurements, the polar to the Cartesian conversion of the measurement pdf is a recommended pre-processing step. This conversion is discussed in Appendix A. From here-on, the Cartesian measurements are assumed for clarity. The case of the large group and the infinite measurement sources based extended target is considered in this thesis. The measurement model typical to the star-convex shape, without the orientation state, is explained below.

Consider a two dimensional Cartesian coordinate measurement. The star-convex extent function is modelled in the polar coordinates (4.6) with the origin at the centre of the target. Hence, the extent state vector is defined in the polar coordinate system, whereas the centre state vector is in the Cartesian coordinate system. The measurement equation is given below:

$$\boldsymbol{z}_{i,k}^{sen,c} = h^i\left(\boldsymbol{x}_{k|k-1}^{p} + p_{\perp}^{i}\left(\measuredangle(\boldsymbol{z}_{i,k}^{sen,c} - \boldsymbol{x}_{k|k-1}^{p}), \boldsymbol{x}_{k|k-1}^{E}, \grave{\boldsymbol{v}}_{i,k}^{sen,c}\right), \acute{\boldsymbol{v}}_{i,k}^{sen,c}\right), \tag{4.12}$$

where $\boldsymbol{z}_{i,k}^{sen,c}$ represents the i^{th} measurement at the k^{th} sample in Cartesian coordinates and sensor centred frame, $h^i(\cdot)$ denotes the nonlinear measurement function with measurement noise $\acute{\boldsymbol{v}}_{i,k}^{sen,c}$, $\boldsymbol{x}_{k|k-1}^{p}$ is the predicted centre positional states, $p_{\perp}^{i}(\cdot)$ represents the nonlinear function of the measurement extent projection, $\measuredangle(\boldsymbol{v}_1 - \boldsymbol{v}_2)$ denotes the angle of the difference vector $(\boldsymbol{v}_1 - \boldsymbol{v}_2)$, $\boldsymbol{x}_{k|k-1}^{E}$ is the predicted extent state vector and $\grave{\boldsymbol{v}}_{i,k}^{sen,c}$ represents the measurement projection noise. The measurement model is explained in Fig. 4.5. The figure shows an extended target (thick solid line) observed in a sensor (filled diamond) centred Cartesian frame and a measurement (cross symbol) $\boldsymbol{z}_{i,k}^{sen,c}$. The predicted target centre position is at $\boldsymbol{x}_{k|k-1}^{p}$ with uncertainty depicted as a dotted ellipse. The angle of the measurement in a polar coordinate system with origin at the target centre is shown as $\measuredangle(\boldsymbol{z}_{i,k}^{sen,c} - \boldsymbol{x}_{k|k-1}^{p})$. The uncertainty in the angle calculation (dotted lines) is due to the measurement noise and the uncertainty of the predicted target centre. This angle is input to the extent estimation function f which predicts the target radial extent $r_{i,k}^{obj,p}$ at given angle with some uncertainty. The angle and the predicted radial extent are combined to form the predicted polar measurement with respect to the target centred polar frame. The uncertainty of the predicted measurement (dotted ellipse) is due to the angle and the predicted extent uncertainties. The polar predicted measurement is projected on a target centred Cartesian frame. The projected predicted coordinates are (p_x^i, p_y^i) with projected uncertainties (dotted lines). The projected coordinates are translated to the sensor centred Cartesian frame to give the predicted measurement coordinates $(z_{k|k-1}^{i,x}, z_{k|k-1}^{i,y})$. The translated coordinates uncertainty (dotted lines) is shown greater due to the additional uncertainty in the operation arising from the predicted target centre uncertainty. In practical cases, the predicted measurement uncertainty is correlated (rotated ellipse). The angle uncertainties are ignored in [26, 73].

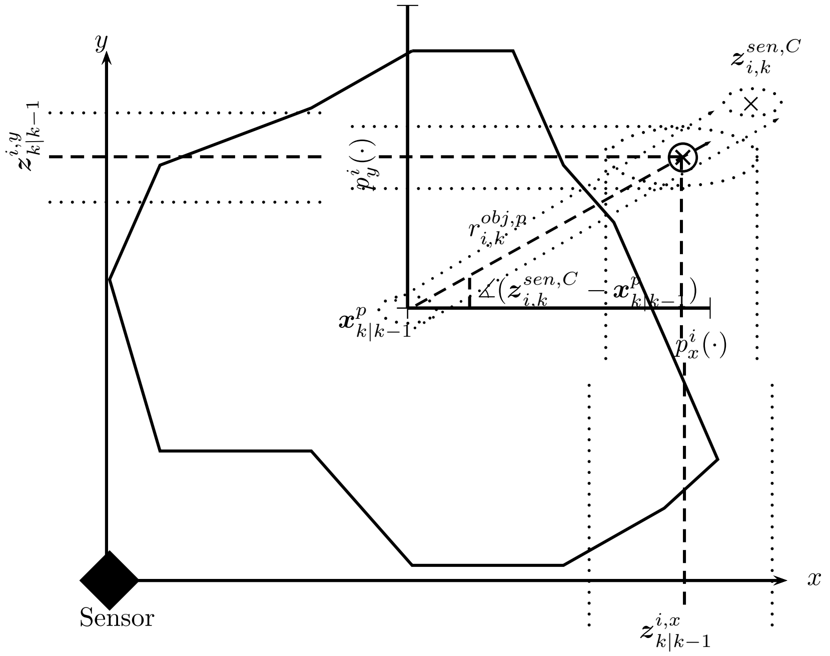

y
$\boldsymbol{z}^{i,y}_{k|k-1}$
$p^i_y(\cdot)$
$\boldsymbol{z}^{sen,C}_{i,k}$
$r^{obj,p}_{i,k}$
$\measuredangle(\boldsymbol{z}^{sen,C}_{i,k} - \boldsymbol{x}^p_{k|k-1})$
$\boldsymbol{x}^p_{k|k-1}$
$p^i_x(\cdot)$
x
Sensor
$\boldsymbol{z}^{i,x}_{k|k-1}$

Fig. 4.5 **Measurement model for star-convex extent**.

Typically, the following assumptions are made to simplify the model:

SC1 The uncertainty of the predicted location $\boldsymbol{x}^p_{k|k-1}$ is ignored in the projection function $\boldsymbol{p}_\perp$.

SC2 The noises $\grave{\boldsymbol{v}}^{sen,c}_{i,k}$ and $\acute{\boldsymbol{v}}^{sen,c}_{i,k}$ are additive Gaussian and uncorrelated.

The simplified model is given below:

$$\begin{aligned}\boldsymbol{z}^{sen,c}_{i,k} &= \boldsymbol{x}^p_{k|k-1} + p^i_\perp\left(\measuredangle(\boldsymbol{z}^i_k - \boldsymbol{x}^p_{k|k-1}), \boldsymbol{x}^E_{k|k-1}\right) + \grave{\boldsymbol{v}}^{sen,c}_{i,k} + \acute{\boldsymbol{v}}^{sen,c}_{i,k}\\ &= \boldsymbol{x}^p_{k|k-1} + p^i_\perp\left(\measuredangle(\boldsymbol{z}^i_k - \boldsymbol{x}^p_{k|k-1}), \boldsymbol{x}^E_{k|k-1}\right) + \boldsymbol{v}^{sen,c}_{i,k}\\ &= z_{k|k-1} + \boldsymbol{v}^{sen,c}_{i,k},\end{aligned} \tag{4.13}$$

where $z_{k|k-1}$ denotes the predicted measurement and $\boldsymbol{v}^{sen,c}_{i,k}$ is the the measurement noise. When the target orientation is considered, the measurement equation is given below:

$$\boldsymbol{z}^{sen,c}_{i,k} = h^i\left(\boldsymbol{x}^p_{k|k-1} + p^i_\perp\left(\measuredangle(\boldsymbol{z}^i_k - \boldsymbol{x}^p_{k|k-1}) - x^O_{k|k-1}, \boldsymbol{x}^E_{k|k-1}, \grave{\boldsymbol{v}}^{sen,c}_{i,k}\right), \acute{\boldsymbol{v}}^{sen,c}_{i,k}\right), \tag{4.14}$$

where $x^{O}_{k|k-1}$ denotes the predicted target orientation. For the multiple measurements, it is further assumed that the measurement noise is mutually uncorrelated. The simplified measurement equation for N_k measurements at k is given below:

$$\boldsymbol{z}^{sen,c}_{k} = [\boldsymbol{x}^{p}_{k|k-1}]_{\times N_k} + \boldsymbol{p}_{\perp}\left(\measuredangle\left(\boldsymbol{z}_k - [\boldsymbol{x}^{p}_{k|k-1}]_{\times N_k}\right) - x^{O}_{k|k-1}, \boldsymbol{x}^{E}_{k|k-1}\right) + \boldsymbol{v}^{sen,c}_{k}, \quad (4.15)$$

where $[\boldsymbol{x}^{p}_{k|k-1}]_{\times N_k}$ is a vector stacked with N_k copies of $\boldsymbol{x}^{p}_{k|k-1}$, $\boldsymbol{p}_{\perp}$ denotes the multiple measurement extent projection function and $\boldsymbol{v}^{sen,c}_{k}$ is the multiple measurement noise vector. For the case of the surface measurements, a one-dimensional scaling factor $s \in [0,1]$ is defined [73, 26]. It is modelled as a random variable and the scaling parameter of different measurements is assumed mutually uncorrelated. The measurement sources are assumed to be uniformly distributed over the surface. Based on the above assumptions, the squared scaling factor is shown to be uniformly distributed in the modelled range i.e. $s^2 \sim \mathcal{U}[0,1]$ [73], where $\mathcal{U}[\cdot,\cdot]$ represents the uniform distribution. For inference, the scaling parameter is modelled using a Gaussian with mean and variance derived analytically from the uniform distribution. The measurement equation of the i^{th} surface measurement is given below:

$$\boldsymbol{z}^{sen,c}_{i,k} = \boldsymbol{x}^{p}_{k|k-1} + s^{i}_{k}p^{i}_{\perp}\left(\measuredangle\left(\boldsymbol{z}^{sen,c}_{i,k} - \boldsymbol{x}^{p}_{k|k-1}\right), \boldsymbol{x}^{E}_{k|k-1}\right) + \boldsymbol{v}^{sen,c}_{i,k}, \quad (4.16)$$

where s^{i}_{k} is the scaling parameter. The inference accuracy depends on the similarity of the Gaussian to the statistical properties of the measurement sources. In practical systems these properties are difficult to model apriori.

The simplified state space model of a typical single (with a star-convex extent model) extended or group target tracking problem is given below:

$$\begin{bmatrix}\boldsymbol{x}^{C}_{k+1}\\ \boldsymbol{x}^{O}_{k+1}\\ \boldsymbol{x}^{E}_{k+1}\end{bmatrix} = \begin{bmatrix}\boldsymbol{F}^{C}_{k} & 0 & 0\\ 0 & \boldsymbol{F}^{O}_{k} & 0\\ 0 & 0 & \boldsymbol{F}^{E}_{k}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}^{C}_{k}\\ \boldsymbol{x}^{O}_{k}\\ \boldsymbol{x}^{E}_{k}\end{bmatrix} + \begin{bmatrix}\boldsymbol{w}^{C}_{k}\\ \boldsymbol{w}^{O}_{k}\\ \boldsymbol{w}^{E}_{k}\end{bmatrix}, \quad (4.17)$$

$$\boldsymbol{z}^{sen,c}_{k} = [\boldsymbol{x}^{p}_{k|k-1}]_{\times N_k} + P^{i}\left(\measuredangle\left(\boldsymbol{z}^{sen,c}_{k} - [\boldsymbol{x}^{p}_{k|k-1}]_{\times N_k}\right) - x^{O}_{k|k-1}, \boldsymbol{x}^{E}_{k|k-1}\right) + \boldsymbol{v}^{sen,c}_{k}. \quad (4.18)$$

The measurement equation is non-linear and non-linear inference methods are applied for the state estimation. Bayesian non-linear inference methods, reviewed in Section 3.3, are applicable. The Gaussian process extent model is demonstrated using an extended Kalman filter in [26]. The performance degrades for tracking of targets exhibiting sharp manoeuvres. The performance is improved in [79] using a Rao-Blackwellised particle filter (RBPF). The particle filter samples the kinematics states of target and a GP regression based Kalman filter is used to track the extent. The state estimation of the random hypersurface model is done with an unscented Kalman filter in [73].

4.4 Multiple Extended and Group Objects Tracking

The multiple extended or group target tracking requires the state estimation of multiple targets in presence of the following uncertainties:

ME1 The total number of targets is unknown and time varying.

ME2 The measurement sources on a single extended target are infinite or a single group is large.

ME3 The measurements are noisy and may contain clutter.

ME4 The number of measurements from a single target is unknown and varies from sample to sample.

ME5 The measurement origin is unknown.

ME6 There are missed detections, that is, no measurement is received from some of the targets.

ME7 The missed detections are random between samples.

Consider a sensor reports measurement vector z_k at sample number k corresponding to N extended or group targets. The objective of the tracker is to determine the unknown multi-target posterior pdf. The problem is typically modelled in a Bayesian framework. An overview of the Bayesian inference is given in Section 3.2 and the Bayesian recursion is described in Section 3.2.1. The posterior multiple target state is determined as given below:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)\int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1}}{\int p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1}}, \tag{4.19}$$

$$\text{posterior pdf at k} = \frac{\text{likelihood at k} \times \text{predictive pdf at k}}{\text{Normalization constant}}, \tag{4.20}$$

where the predictive pdf is given by the Chapman-Kolmogorov equation (3.7) using the posterior pdf at $k-1$ and the state evolution model. The inference relies on the multi-target likelihood and the state evolution models. Typically, this problem requires a solution to two sub-problems, the data association and the state estimation. The data association deals with the measurements to targets assignments. The state estimation requires an update of the multi-target state vector and higher order moments of the state using the assigned measurements.

Various methods for the multiple point target tracking have been extended for the case of the multiple extended or group target tracking. The joint probabilistic data association tracker proposed in [80] considers all the measurements to targets associations for the multi-target state estimation. The performance degrades for closely spaced targets moving together over a longer period. A probabilistic multiple hypothesis tracker for the extended target tracking has been proposed in [81]. Methods, which do not rely on the explicit measurement to target association, have also been proposed in [82, 83]. These model the measurements as a Poisson process and solve it using sequential Monte Carlo based inference methods. Many random finite sets based methods have also been proposed [84]. The full Bayesian implementation of these methods is not tractable. Various approximate methods have been proposed, such as the probability hypothesis density filter [85], and the multi-Bernoulli filter [86].

The measurements to target associations are modelled in the above mentioned methods. In some applications, such as with LiDAR sensor, the measurement vector at a particular sample is large. The inference based on all the measurements to targets assignments is computationally infeasible. Typically, a pre-processing step is added to all of the above methods. This method is called the measurement clustering in general and the measurement set partitioning for the random sets based methods. It divides all the measurements into clusters before the application of a chosen inference method. All clusters are mutually exclusive and all the measurements in a cluster are assumed to be from the same target or clutter, for example, the density based spatial clustering of applications with noise (DBSCAN) [83]. The measurement set partitioning approach divides the random measurement set into all possible subsets. For example, consider a random measurement set $Z_k = \{z_1, z_2\}$ consists of two measurements. The power set of Z_k gives all the possible measurement set partitions. The most likely subsets are chosen for the multi-target posterior pdf update. The typical criteria for the acceptance or rejection of the different subsets is the distance. A stochastic optimization based approach [87] has been demonstrated to track closely moving targets by proposing an efficient way to determine the most likely sets.

Part II

Publications

Chapter 5

A Gaussian Process Approach for Extended Target Tracking with Irregular Shapes and for Dealing with Intractable Likelihoods

Abstract

Tracking of arbitrarily shaped extended targets is a complex task due to the intractable analytical expression of measurement to target associations. The presence of sensor noise and clutter worsens the situation. Although significant work has been done on the extended target tracking problems, most of the developed methods are restricted by assumptions on the shape of the target such as stick, circle, or other axis-symmetric properties, etc. This chapter proposes a novel Gaussian process approach for tracking an irregularly shaped extended target using a convolution particle filter. The new approach is shown to track irregularly shaped targets efficiently in the presence of measurement noise and clutter. The mean recall and precision values for the shape, calculated by the proposed approach on simulated data are around 0.9, by using 1000 particles.

5.1 Introduction

This chapter proposes a Gaussian process approach for the group and extended target tracking (ETT) problems [24]. In most ETT approaches [82, 88], the target shape is modelled using basic geometrical shapes e.g. stick [69] , circle [89], rectangle [70], ellipse [88, 71]. The tracking performance can be improved by considering a more detailed shape model, as in [26, 90] (see Section 4.3.4) and the proposed approach. The analytical form of likelihood cannot be derived for the ETT problem, as the measurements relate to the states non-linearly. The approach proposed in this chapter does not require an explicit likelihood function for estimation. The complex shape tracking approaches [26, 79, 90] require prior knowledge of the distribution of surface measurements (see Section 4.3.7). The performance degrades when the statistical properties of actual measurements, which are not known in real world problems, are different from the modelled distribution. In the proposed approach, the statistical properties of the surface measurements are not required by the filter. In [70, 71], the convolution particle filter (CPF) is used to track an extended target but the target shape is assumed to be basic i.e. rectangular and ellipse. In the proposed approach the CPF is used to track using a more complex (star-convex) shaped model. Moreover, a highly non-linear target kinematics model is considered along with sensor clutter.

The approach proposed in this chapter is called Gaussian process convolution particle filter (GPCPF). It relies on a GP for modelling of the target extent and a CPF for the state estimation. A CPF samples both the kinematics and extent states. As many GPs, as the number of particles, are trained on the extent samples. The GP is used to define the CPF kernel by estimating a hypersphere in measurement space. The CPF resolves the data association in the presence of sensor noise and clutter using the hypersphere and updates the particle weights to determine the filtered state.

The rest of the chapter is organized as follows. Section 5.2 presents the background knowledge of the CPF and Section 5.3 formulates the problem using the GP model in the CPF framework. The performance validation of the approach is done in Section 5.4 followed by conclusions in Section 5.5.

5.2 Convolution Particle Filter

The CPF was first proposed in [91]. Consider the general non-linear state space model given in (3.48) and (3.49). The aim is to estimate the posterior density of the state given all the measurements up to the current time step k as given in (3.8). A discrete set of points $\tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{z}}_{1:k}$ are simulated. This process starts by sampling from an initial distribution $p(\boldsymbol{x}_0)$. The particle samples of the initial distribution have equal weights. These samples are predicted using (3.48). The predicted samples are simulated through the measurement function (3.49). The simulated states and measurements are used to determine the posterior distribution. The empirical estimates from these simulated states and measurements are given below:

$$p(\boldsymbol{x}_k, \boldsymbol{z}_{1:k}) \approx \frac{1}{n_p} \sum_{i=1}^{n_p} \delta(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k^i, \boldsymbol{z}_{1:k} - \tilde{\boldsymbol{z}}_{1:k}^i), \tag{5.1}$$

where n_p denotes the number of particles and $\delta(\cdot)$ is Dirac measure. The above density is not smooth due to the point estimates. The smoothed densities can be obtained by the convolution of the simulated states and the measurements with the respective kernels as given below:

$$p^{n_p}(\boldsymbol{x}_k, \boldsymbol{z}_{1:k}) = \frac{1}{n_p} \sum_{i=1}^{n_p} \mathcal{K}_b^{\tilde{x}}(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k^i) \mathcal{K}_b^{\tilde{z}}(\boldsymbol{z}_{1:k} - \tilde{\boldsymbol{z}}_{1:k}^i), \tag{5.2}$$

$$p^{n_p}(\boldsymbol{z}_{1:k}) = \frac{1}{n_p} \sum_{i=1}^{n_p} \mathcal{K}_b^{\tilde{z}}(\boldsymbol{z}_{1:k} - \tilde{\boldsymbol{z}}_{1:k}^i), \tag{5.3}$$

$$\mathcal{K}_b^{\tilde{z}}(\boldsymbol{z}_{1:k} - \tilde{\boldsymbol{z}}_{1:k}^i) = \prod_{j=1}^{k} \mathcal{K}_b^{\tilde{z}}(\boldsymbol{z}_{1:k} - \tilde{\boldsymbol{z}}_j^i), \tag{5.4}$$

where $p^{n_p}(\cdot)$ denotes the kernel estimate of the true density, $\mathcal{K}_b^{\tilde{x}}$ and $\mathcal{K}_b^{\tilde{z}}$ are Parzen-Rozenblatt kernels of appropriate dimensions and b is the kernel bandwidth. The posterior $p^{n_p}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$, also called the CPF density, is estimated using (3.8) and (5.2) - (5.4) as given below:

$$p^{n_p}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{\sum_{i=1}^{n_p} \mathcal{K}_b^{\tilde{x}}(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k^i) \mathcal{K}_b^{\tilde{z}}(\boldsymbol{z}_{1:k} - \tilde{\boldsymbol{z}}_{1:k}^i)}{\sum_{i=1}^{n_p} \mathcal{K}_b^{\tilde{z}}(\boldsymbol{z}_{1:k} - \tilde{\boldsymbol{z}}_{1:k}^i)}. \tag{5.5}$$

5.3 Gaussian Process Convolution Particle Filter

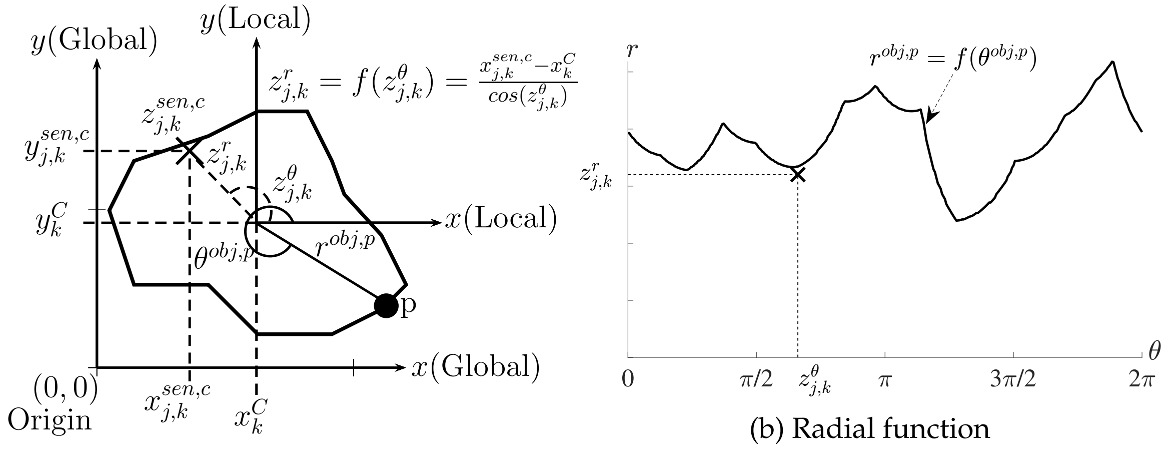
The GPCPF tracks the centre of the object or target (CoO) simultaneously with its extent. The state update model of the CoO and the extent is linear. The extent is modelled as a function of the angles with respect to the CoO. Modelling in this way converts the complex problem of associating the measurements to the respective point targets to a relatively simple problem. The resultant measurement model is non-linear. The non-linear relation between the angle and the radial extent is modelled using a zero mean Gaussian process with a novel (periodic) kernel. The state estimation is done using a novel kernel based CPF.

5.3.1 Extent Model

The shape of the target is assumed to be star convex and is modelled using a GP as proposed in [26]. The target extent is modelled as a function of angle from the CoO as given below:

$$r^{obj,p} = f(\theta^{obj,p}), \tag{5.6}$$

where $r^{obj,p}$ and $\theta^{obj,p}$ denote, respectively, the polar radial extent and angle in object frame. The input (angle) domain is periodic with period 2π. The extended target extent model is explained in Fig. 5.1a. Two coordinate frames, namely global or sensor (origin at the sensor) and local or object (origin at the CoO), along with their relationship are also shown in Fig. 5.1a. Sensor measurements and filter output are given in the global frame. The GP is modelled in the local frame. The function f is visualized in Fig. 5.1b and modelled using a GP. This function is realized as the point p rotates around the CoO. The j^{th} measurement is also shown for comparison.



(a) Group / Extended Target

(b) Radial function

Fig. 5.1 **Group/extended target model**. The figure (5.1a) shows an irregularly shaped extended target, a point p and the j^{th} measurement $z^{sen,c}_{j,k}$. The figure (5.1b) visualizes the radial function $r^{obj,p} = f(\theta^{obj,p})$ shown in the subfigure 5.1a.

The $L-$dimensional measurement vector $z_k^{sen,c}$ is as follows:

$$z_k^{sen,c} = \begin{bmatrix} x_{1,k}^{sen,c} & y_{1,k}^{sen,c} & x_{2,k}^{sen,c} & y_{2,k}^{sen,c} & \cdots & x_{L,k}^{sen,c} & y_{L,k}^{sen,c} \end{bmatrix}^T, \tag{5.7}$$

where $x_{j,k}^{sen,c}, y_{j,k}^{sen,c}$ are the Cartesian coordinates of the j^{th} measurement. Let $(r_{j,k}^{obj,p}, \theta_{j,k}^{obj,p})$ be the respective polar coordinates in the object frame and (x_k^C, y_k^C) be the origin of the local (object) frame then,

$$x_{j,k}^{sen,c} = x_k^C + r_{j,k}^{obj,p} \cos(\theta_{j,k}^{obj,p}) + \breve{v}_{j,k}^{sen,c}, \quad \breve{v}_{j,k}^{sen,c} \sim \mathcal{N}(0, \sigma_x^2), \tag{5.8}$$

$$y_{j,k}^{sen,c} = y_k^C + r_{j,k}^{obj,p} \sin(\theta_{j,k}^{obj,p}) + \bar{v}_{j,k}^{sen,c}, \quad \bar{v}_{j,k}^{sen,c} \sim \mathcal{N}(0, \sigma_y^2), \tag{5.9}$$

where $\breve{v}$ and $\bar{v}$ denote, respectively, the measurement noises in x and y coordinates. Using (5.6), the equations (5.8) and (5.9) are written as follows:

$$x_{j,k}^{sen,c} = x_k^C + f(\theta_{j,k}^{obj,p}) \cos(\theta_{j,k}^{obj,p}) + \breve{v}_{j,k}^{sen,c}, \tag{5.10}$$

$$y_{j,k}^{sen,c} = y_k^C + f(\theta_{j,k}^{obj,p}) \sin(\theta_{j,k}^{obj,p}) + \bar{v}_{j,k}^{sen,c}. \tag{5.11}$$

The function f is modelled using a GP i.e. $f(\theta^{obj,p}) \sim GP(m(\theta^{obj,p}), k(\theta^{obj,p}, \theta'^{obj,p}))$.

5.3.2 GP Mean, Covariance Kernel and Hyperparameters

The mean is modelled as a constant, $m(\theta^{obj,p}) = c \geq 0$. This model ensures that the radial extent does not go below 0 and the star convex shape is maintained. The covariance kernel is the core of the GP regression. The kernel is periodic in the input (angle) domain. A novel periodic kernel, inspired from the Von-Mises distribution is proposed as given below:

$$k_{vm}(\theta^{obj,p}, \theta'^{obj,p})) = \sigma_m^2 \exp\left(\frac{\cos(\theta^{obj,p} - \theta'^{obj,p})}{l^2} \right), \tag{5.12}$$

where $k_{vm}(\cdot,\cdot)$ denotes the Von-Mises covariance kernel, σ_m^2 and l^2 are, respectively, the magnitude and the lenghtscale hyperparameters. As given in [26], the unknown constant mean is included in the model by setting prior on the unknown mean $c \sim \mathcal{N}(0, \sigma_c^2)$. The resulting GP model is zero mean i.e. $f(\theta^{obj,p}) \sim GP(0, k(\theta^{obj,p}, \theta'^{obj,p}))$. The covariance kernel (5.12) is modified as given below:

$$k(\theta^{obj,p}, \theta'^{obj,p}) = k_{vm}(\theta^{obj,p}, \theta'^{obj,p}) + \sigma_c^2. \tag{5.13}$$

The extent at a given angle is not correlated with the distant angles. This allows the shape to be irregular. This is achieved by choosing pre-hand a set of hyperparameters that allow this behaviour. As a result, the hyperparameters need not be learned and the computational complexity is reduced. The super-script obj, p is omitted from here-on for brevity.

5.3.3 State Sampling

The states inferred at each time step k are assumed to be in $2D$ Cartesian frame and are represented by the state vector $\boldsymbol{x}_k$:

$$\boldsymbol{x}_k = \begin{bmatrix} (\boldsymbol{x}_k^C)^T & (\boldsymbol{x}_k^E)^T \end{bmatrix}^T. \tag{5.14}$$

The extent states are sampled at B input locations between 0 and 2π as given below:

$$\boldsymbol{x}_k^C = \begin{bmatrix} x_k^C & \dot{x}_k^C & y_k^C & \dot{y}_k^C \end{bmatrix}^T, \quad \boldsymbol{x}_k^E = \begin{bmatrix} r_k^1 & r_k^2 & \cdots & r_k^B \end{bmatrix}^T. \tag{5.15}$$

where x_k^C, y_k^C and $\dot{x}_k^C, \dot{y}_k^C$ are, respectively, the position and the velocity of the CoO, r_k^i denotes the radial values of the extent corresponding to the input vector $\boldsymbol{\theta}^E$ given below:

$$\boldsymbol{\theta}^E = \begin{bmatrix} \theta^1 & \theta^2 & \cdots & \theta^B \end{bmatrix}^T, \quad \theta^l = (l-1)\frac{2\pi}{B}. \tag{5.16}$$

The B points are also called the inducing or keypoints. At each sample, the function values at these keypoints are estimated. The filtered states and the GP model are used to predict the extent on the complete input domain. The kinematics of the target centre is modelled using a correlated velocity model [70]. The CoO states of the i^{th} particle are sampled as given below:

$$\tilde{\boldsymbol{x}}_k^{i,C} = \boldsymbol{F}^C \tilde{\boldsymbol{x}}_{k-1}^{i,C} + \boldsymbol{w}_k^{i,C}, \quad \boldsymbol{w}_k^{i,C} \sim \mathcal{N}(0, \boldsymbol{Q}^C), \tag{5.17}$$

$$\boldsymbol{F}^C = \mathrm{diag}(\boldsymbol{F}^{Co}, \boldsymbol{F}^{Co}), \tag{5.18}$$

$$\boldsymbol{F}^{Co} = \begin{bmatrix} 1 & \frac{1-\exp(-\alpha T)}{\alpha} \\ 0 & \exp(-\alpha T) \end{bmatrix}, \tag{5.19}$$

$$\boldsymbol{Q}^C = 2\alpha q^{Co} \mathrm{diag}(\boldsymbol{Q}^{Co}, \boldsymbol{Q}^{Co}), \tag{5.20}$$

$$\boldsymbol{Q}^{Co} = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix}, \tag{5.21}$$

$$q_{11} = \frac{4\exp(-\alpha T) - 3 - \exp(-2\alpha T) + 2\alpha T}{2\alpha^3}, \tag{5.22}$$

$$q_{12} = q_{21} = \frac{\exp(-2\alpha T) + 1 - 2\exp(-\alpha T)}{2\alpha^2}, \tag{5.23}$$

$$q_{22} = \frac{1 - \exp(-2\alpha T)}{2\alpha}, \tag{5.24}$$

where $\cdot^{i,\cdot}$ denotes the variable corresponds to the i^{th} particle, $\tilde{\boldsymbol{x}}_k^{i,C}$ represents the sampled CoO state, $\boldsymbol{w}_k^{i,C}$ is the zero mean normally distributed process noise vector with covariance matrix $\boldsymbol{Q}^C$, $\boldsymbol{F}^{Co}$ and $\boldsymbol{Q}^{Co}$ denote, respectively, the state update and process noise covariance matrices of the correlated velocity model, $\alpha = \frac{1}{\tau_m}$ is the velocity correlation constant, $\text{diag}(\cdot, \cdot)$ denotes a diagonal matrix, q^{Co} represents the variance of the process noise. The extent state update is modelled using a random walk model. The sampled extent state of the i^{th} particle is given below:

$$\tilde{\boldsymbol{x}}_k^{i,E} = \tilde{\boldsymbol{x}}_{k-1}^{i,E} + \boldsymbol{w}_k^{i,E}, \quad \boldsymbol{w}_k^{i,E} \sim \mathcal{N}(0, \boldsymbol{Q}^E = \sigma_E^2 \boldsymbol{I}_B), \tag{5.25}$$

where $\tilde{\boldsymbol{x}}_k^{i,E}$ represents the sampled extent state of the i^{th} particle, $\boldsymbol{w}_k^{i,E}$ is the zero mean extent process noise vector whose elements are i.i.d. with variance σ_E^2. The target can have any shape, which in other words means that all the radial values can be uncorrelated. As a result, the covariance matrix is set as diagonal. Smaller variance means the filter takes more time to lock the extent parameters and vice versa. The number of particles required for the filter to converge will increase if the variance is large.

5.3.4 Measurement Simulation and Weight Update

For each sampled state, simulate the measurements of the i^{th} particle as given below:

$$\tilde{\boldsymbol{z}}_k^i = \begin{bmatrix} \tilde{x}_k^{i,C} & \tilde{y}_k^{i,C} \end{bmatrix}_{\times B}^T + \boldsymbol{J} \odot \left(\tilde{\boldsymbol{x}}_k^{i,E} \otimes \begin{bmatrix} 1 & 1 \end{bmatrix}^T \right) + \tilde{\boldsymbol{v}}_k^i, \tag{5.26}$$

$$\tilde{\boldsymbol{v}}_k^i \sim \mathcal{N}\left(0, \boldsymbol{I}_B \otimes \text{diag}(\sigma_x^2, \sigma_y^2)\right),$$

$$\boldsymbol{J} = \begin{bmatrix} \cos(\theta^1) & \sin(\theta^1) & \cdots & \cos(\theta^B) & \sin(\theta^B) \end{bmatrix}^T, \tag{5.27}$$

where $\tilde{\boldsymbol{z}}_k^i$ denotes the sampled measurement, $\tilde{x}_k^{i,C}$ and $\tilde{y}_k^{i,C}$ represent, respectively, the Cartesian coordinates of the centre of the i^{th} particle, $\otimes$ is the Kronecker product and $\tilde{\boldsymbol{v}}_k^i$ denotes the measurement noise vector. The sampled measurement $\tilde{\boldsymbol{z}}_k^i$ also represents a hypersphere in the measurement space. This hypersphere lies in a

2D-plane and it is called a polygon denoted as $\mathcal{P}_k^i$. All measurements within this polygon are considered belonging to the i^{th} particle.

The measurement vector $z_{j,k}^{sen,c}$ (5.7) is in the global Cartesian frame. It is converted to the polar frame with origin located at the sampled centre location of the respective particle. It is important to note that the centre is unique for each particle. As a result, the Cartesian to polar conversion is also unique for all the particles. Let $\boldsymbol{z}_k^{r,i} = \begin{bmatrix} z_{1,k}^{r,i} & \cdots & z_{L,k}^{r,i} \end{bmatrix}^T$ and $\boldsymbol{z}_k^{\theta,i} = \begin{bmatrix} z_{1,k}^{\theta,i} & \cdots & z_{L,k}^{\theta,i} \end{bmatrix}^T$ denote the polar range and the polar angle measurement vectors, respectively, with respect to the sampled state of the i^{th} particle. The polar coordinates of the j^{th} measurement are given below:

$$\begin{bmatrix} z_{j,k}^{r,i} & z_{j,k}^{\theta,i} \end{bmatrix}^T = cart2pol\left(\begin{bmatrix} x_{j,k}^{sen,c} & y_{j,k}^{sen,c} \end{bmatrix}^T - \begin{bmatrix} \tilde{x}_k^{i,C} & \tilde{y}_k^{i,C} \end{bmatrix}^T \right). \tag{5.28}$$

where $cart2pol(\cdot)$ denotes a Cartesian to polar conversion function. The above polar converted measurements are used for the gating with the polygon. The radial extent of the i^{th} particle is determined at the polar angle measurement vector $\boldsymbol{z}_k^{\theta,i}$. The GP regression (2.18),(2.19) is used for the calculation of the radial extent. The training data of the GP is the sampled extent state $\tilde{\boldsymbol{x}}_k^{i,E}$ and the test input vector is $\boldsymbol{z}_k^{\theta,i}$. The radial extent is calculated using the GP regression (2.18) as given below:

$$\tilde{\boldsymbol{z}}_k^{r,i} = \boldsymbol{K}_{\boldsymbol{z}_k^{\theta,i}\boldsymbol{\theta}^E}(\boldsymbol{K}_{\boldsymbol{\theta}^E\boldsymbol{\theta}^E})^{-1}\tilde{\boldsymbol{x}}_k^{i,E}, \tag{5.29}$$

where $\tilde{\boldsymbol{z}}_k^{r,i}$ is the sampled measurement range vector of the i^{th} particle in the polar frame. Each sampled measurement is compared with the respective value in the converted measurement vector $\boldsymbol{z}_k^{r,i}$. The measurements with a converted value smaller than the sampled value are considered gated with the particle.

A novel kernel is proposed for the CPF likelihood. It is a uniform kernel with interval support at $\mathcal{P}_k^i$ for a gated observation and at λ otherwise (clutter). The kernel is given below:

$$\mathcal{K}_h^{\tilde{z}_k^i}(\boldsymbol{z}) = \begin{cases} \mathcal{U}_{\mathcal{P}_k^i}(\boldsymbol{z}), & \text{if } \boldsymbol{z} \in \mathcal{P}_k^i, \\ \mathcal{U}_\lambda(\boldsymbol{z}), & \text{otherwise,} \end{cases} \tag{5.30}$$

where $\mathcal{U}_{\mathcal{P}_k^i}(\boldsymbol{z})$ is a uniform kernel with support defined over polygon $\mathcal{P}_k^i$ and $\mathcal{U}_\lambda(\boldsymbol{z})$ is a uniform kernel with support defined over the complete surveillance area and it represents clutter measurements. The uniform distributions are given below:

$$\mathcal{U}_{\mathcal{P}_k^i}(z) = \frac{1}{\text{Polygon } \mathcal{P}_k^i \text{ Area}}, \tag{5.31}$$

$$\mathcal{U}_{\lambda}(z) = \frac{1}{\text{Surveillance Area}}. \tag{5.32}$$

The kernel function returns different values for different particles based on the area of polygon. Hence, it is an adaptive CPF [70]. The particle weight update at scan k is given below:

$$w_k^i = w_{k-1}^i \prod_{j=1}^{L} K_h^{\tilde{z}_k^i} \left(\begin{bmatrix} x_{j,k}^{sen,c} & y_{j,k}^{sen,c} \end{bmatrix}^T \right). \tag{5.33}$$

5.3.5 State Update

The conditional state density for CPF is as follows:

$$p(\boldsymbol{x}_k | \boldsymbol{z}_{1:k}^{sen,c}) = \frac{p(\boldsymbol{x}_k, \boldsymbol{z}_{1:k}^{sen,c})}{\int p(\boldsymbol{x}_k, \boldsymbol{z}_{1:k}^{sen,c}) d\boldsymbol{x}_k}, \tag{5.34}$$

Along the lines of adaptive CPF modelled in [71], the kinematic and the extent states are sampled separately. The posterior distribution is calculated given below;

$$p_k^{n_p}(\boldsymbol{x}_k | \boldsymbol{z}_{1:k}^{sen,c}) = \frac{\sum_{i=1}^{n_p} w_k^i \tilde{\boldsymbol{x}}_k^i \mathcal{K}_h^{\tilde{z}_k^i}(\boldsymbol{z}_{1:k}^{sen,c})}{\sum_{i=1}^{n_p} \mathcal{K}_h^{\tilde{z}_k^i}(\boldsymbol{z}_{1:k}^{sen,c})}, \tag{5.35}$$

$$\mathcal{K}_h^{\tilde{z}_k^i}(\boldsymbol{z}_{1:k}^{sen,c}) = \prod_{j=1}^{k} \mathcal{K}_h^{\tilde{z}_k^i}(\boldsymbol{z}_{j,k}^{sen,c}). \tag{5.36}$$

The state update is given below:

$$\hat{\boldsymbol{x}}_k = \frac{\sum_{i=1}^{n_p} w_k^i \tilde{\boldsymbol{x}}_k^i}{\sum_{i=1}^{n_p} w_k^i}. \tag{5.37}$$

If the number of keypoints is less, the shape estimate can be coarse and the area calculation in (5.31) can be inaccurate. To determine a fine target shape and an accurate area, from the coarse shape estimate, sample the input domain finer than the basis vector B. Let B' be the new number of sample points which are greater than B. The corresponding input vector is given below:

$$\boldsymbol{\theta'}^{E} = \left[\theta^1 \quad \theta^2 \quad \cdots \quad \theta^{B'}\right]^T, \quad \theta^l = (l-1)\frac{2\pi}{B'}. \tag{5.38}$$

The extent vector $\hat{\boldsymbol{x}'}^E_k$ at the new input vector $\boldsymbol{\theta'}^E$ is given below:

$$\hat{\boldsymbol{x}'}^E_k = \boldsymbol{K}_{\boldsymbol{\theta'}^E\boldsymbol{\theta}^E}(\boldsymbol{K}_{\boldsymbol{\theta}^E\boldsymbol{\theta}^E})^{-1}\hat{\boldsymbol{x}}^E_k, \tag{5.39}$$

The state vector estimate with a finer shape is $\hat{\boldsymbol{x}'}_k = [\hat{\boldsymbol{x}}^C_k, \hat{\boldsymbol{x}'}^E_k]^T$. The GP-CPF recursion is summarized in Table 5.1.

Table 5.1 GP-CPF recursion

1	for $k \leq 2$, $\boldsymbol{\theta}_B = 0 : 360/B : 360$ find $\hat{\boldsymbol{x}}_2$ as given in Section 5.4
2	at $k = 3$ initialize $\tilde{\boldsymbol{x}}^i_3 \sim \mathcal{N}(\hat{\boldsymbol{x}}_2, \hat{\boldsymbol{P}}_2)$, and $w^i_3 = \frac{1}{N}$
3	for $k \geq 3$
3a	for $k > 3$ Re-sample : Residual Re-sampling as in [92].
3b	State Sample: for $i = 1, 2, \cdots, n_p$ use (5.17) and (5.25) to determine $\tilde{\boldsymbol{x}}^i_k$
3c	Measurement Simulation : Simulate measurements using (5.26)
3d	Weight Update : for $i = 1, 2, \cdots, n_p$ use (5.33) to update weight w^i_k
3e	Normalize Weight : for $i = 1, 2, \cdots, n_p$ determine $w^i_k = \frac{w^i_k}{\sum_{i=1}^{np} w^i_k}$
3f	State Update : Use (5.37) to update the state $\hat{\boldsymbol{x}}_k$
3g	Output : For $\boldsymbol{\theta'} = 0 : 360/B' : 360$, use (5.39) to determine $\hat{\boldsymbol{x}}^{E'}_k$. Output is $\hat{\boldsymbol{x}'}_k = \left[(\hat{\boldsymbol{x}}^C_k)^T \quad (\hat{\boldsymbol{x}'}^E_k)^T\right]^T$

5.4 Performance Validation

5.4.1 Evaluation Parameters

The performance is evaluated on 50 Monte Carlo runs of a simulated group object. The performance of the CoO estimates is validated using root mean square error (RMSE). The RMSE in N_{MC} Monte Carlo runs is calculated as given below:

$$RMSE_{a_k} = \frac{1}{N_{MC}} \sqrt{\sum_{i=1}^{N_{MC}} (a_k^i - \hat{a}_k^i)^2}, \tag{5.40}$$

where $RMSE$ denotes the root mean square error, a represents the ground truth and $\hat{a}$ is the respective estimate. The mean Precision (P_k^μ) and Recall (R_k^μ) graphs are produced for evaluating the shape estimates. This scheme has been used in computer vision for evaluating rectangular targets detection performance [93]. The recall describes the ground truth area that has been correctly recalled by the algorithm. The precision illustrates an incorrectly detected area. If calligraphic E i.e. $\mathcal{E}$ represents the estimated polygon and calligraphic T i.e. $\mathcal{T}$ denotes the ground truth polygon then the formulae for N_{MC} Monte Carlo runs are given below:

$$R_k^\mu = \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} \frac{Area(\mathcal{T}_k^i \cap \mathcal{E}_k^i)}{Area(\mathcal{T}_k^i)}, \tag{5.41}$$

$$P_k^\mu = \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} \frac{Area(\mathcal{T}_k^i \cap \mathcal{E}_k^i)}{Area(\mathcal{E}_k^i)}, \tag{5.42}$$

where $Area(\cdot)$ calculates the polygonal area and $\cap$ is the intersection operator.

5.4.2 Test Scenarios

The irregularly shaped group object is simulated in two different scenarios. In the first scenario, the surface measurements are assumed uniformly distributed on the object surface. In the second scenario, the surface measurements are assumed Gaussian distributed on the object surface. The mean of the distribution lies on the object center and the standard deviation is set to 50% of the maximum radial length of the object from the center. The above two scenarios are considered to study the performance of the GPCPF in tracking irregularly shaped objects using measurements having different statistical properties.

5.4.3 Simulation Parameters

The simulations are performed on an irregular pentagon shaped extended target. The mean number of measurements is Poisson distributed with mean $\lambda_z = 500$, the total number of scans $K = 100$ and sampling time is $T = 0.125\ s$. The initial state for centre of motion is $[100\ m, 0\ m/s, 50\ m, 0\ m/s]^T$. The velocity correlation time constant is $\tau_m = 15\ s$, the process noise variance is set as $q^{Co} = 100\ m^2/s^2$, and extent process noise variance is $\sigma_E^2 = 0.01\ m^2/s^2$. The standard deviation of the sensor measurement error is $\sigma_x = \sigma_y = 0.1\ m$. The clutter density is $\rho = 1 \times 10^{-3}$ within a circular surveillance region of radius 100 m.

5.4.4 GP-CPF Parameters

The kinematic parameters are matched to those set in the simulation. The initial states are randomly sampled from the respective priors as given below:

$$p(x_2^C) = \mathcal{N}\left(\begin{bmatrix} x_2^C \\ \dot{x}_2^C \\ y_2^C \\ \dot{y}_2^C \end{bmatrix}, \begin{bmatrix} 0.25^2\ m^2 & 0 & 0 & 0 \\ 0 & 1\ m^2/s^2 & 0 & 0 \\ 0 & 0 & 0.25^2\ m^2 & 0 \\ 0 & 0 & 0 & 1\ m^2/s^2 \end{bmatrix}\right), \tag{5.43}$$

$$p(x_2^E) = \mathcal{N}\left(\begin{bmatrix} R_2 & \cdots & R_2 \end{bmatrix}^T, \mathrm{diag}(\begin{bmatrix} 1e^{-8}\ m^2 & \cdots & 1e^{-8}\ m^2 \end{bmatrix}\right), \tag{5.44}$$

where (x_i^C, y_i^C) are coordinates of the CoO at $k = i$, $\dot{x}_i^C = \frac{x_i^C - x_{i-1}^C}{T}$, $\dot{y}_i^C = \frac{y_i^C - y_{i-1}^C}{T}$ and R_i is the maximum radial value at sample number i. All the particles are initialized to equal weights. The CoO is determined using k-means clustering during initialization i.e. when $k = 1, 2$. The extent process noise standard deviation is $\sigma_E^2 = \frac{10}{3}\ m^2/s^2$, hyperparameters values are $l = 0.18\ radian$, $\sigma_c^2 = 1\ m^2$ and $\sigma_m^2 = \frac{1}{40}\ m^2$. The number of particles is $n_p = 1000$, number of keypoints is $B = 16$ and the number of output basis is $B' = 1440$.

5.4.5 Results

The results of the first scenario i.e. uniformly distributed measurements are shown in Figs. 5.2 and 5.3. The average positional RMSE is around $1m$ and velocity RMSE is around $2m/sec$. The mean precision is around 0.87 most of the time which means almost 13% of the estimated shape is different from the ground truth. The mean recall is around 0.96 which means that almost 96% of ground truth shape has been recalled. The initial recall is higher i.e. close to 1 and decreases with time. This is because the filter is initialized using a circular shape of radius equal to the maximum radial extent of the target from the corresponding initial CoO. As a result, the whole target is enclosed in the initial shape, which in other words means that the whole target is recalled. The CPF kernel is uniform and it weighs the particles based on the number of gated measurements. Hence, sometimes the CoO estimates are slightly away from the true CoO whereas the shape estimate remains accurate at all times. Figure 5.3 shows snapshots of simulated and estimated targets at time steps $k = 2, 30, 63$ and 96. The estimate of the path and the shape of the extended target is close to the ground truth in all steps given that the shape was initialized as a circle (at $k = 2$). The shapes have been displayed at chosen time steps for clarity. The filter runs in quick time and the mean simulation time is $49s$ for 100 time steps. The program was run on MATLAB R2016b on a Windows 10 (64 bit) Desktop computer installed with Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz(4 CPUs) and 8GB RAM.

The results of the second scenario i.e. Gaussian distributed measurements are shown in Figs. 5.4 and 5.5. The average positional RMSE is around 1 m and velocity RMSE is around 2.5 m/sec. The mean of P_{μ}^{k} is 0.81 which means almost 19% of the estimated shape is different from the ground truth. The mean of R_{μ}^{k} is 0.92 which means that almost 92% of ground truth shape has been recalled. The performance is comparable to that in the first scenario i.e. with uniformly distributed measurements. It should be noted that the same parameters for the GPCPF are used in both scenarios. It shows that the proposed algorithm does not require any prior knowledge of the statistical properties of the measurements in the considered scenarios.

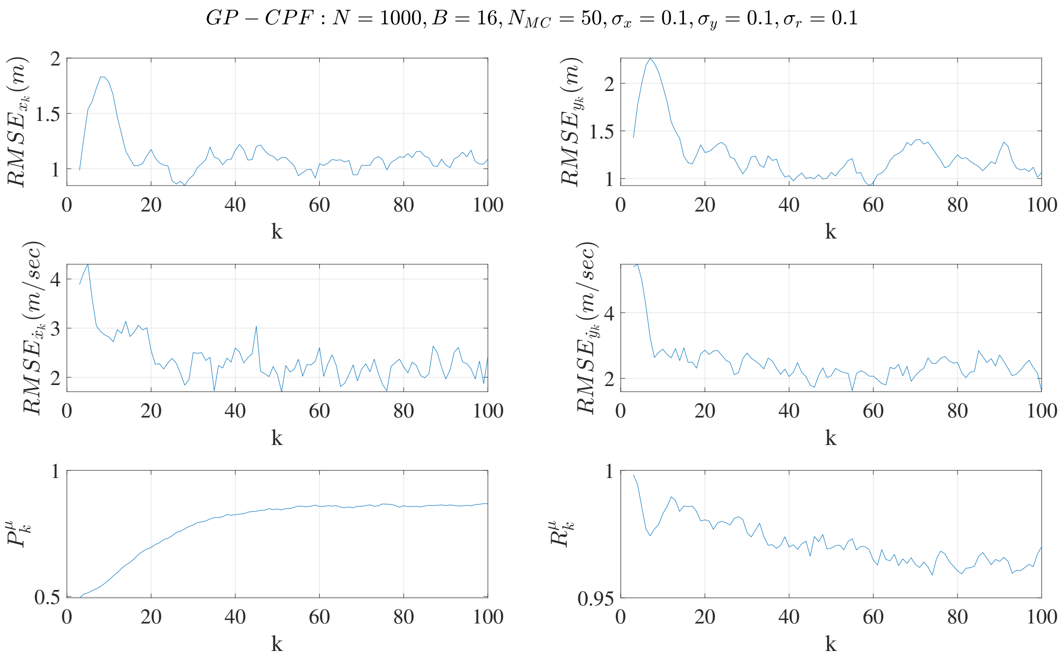


Fig. 5.2 **Results of 50 Monte Carlo runs on uniformly distributed surface measurements**. This figure shows performance results of GPCPF tracking an irreularly shaped target using uniformly distributed surface measurements.

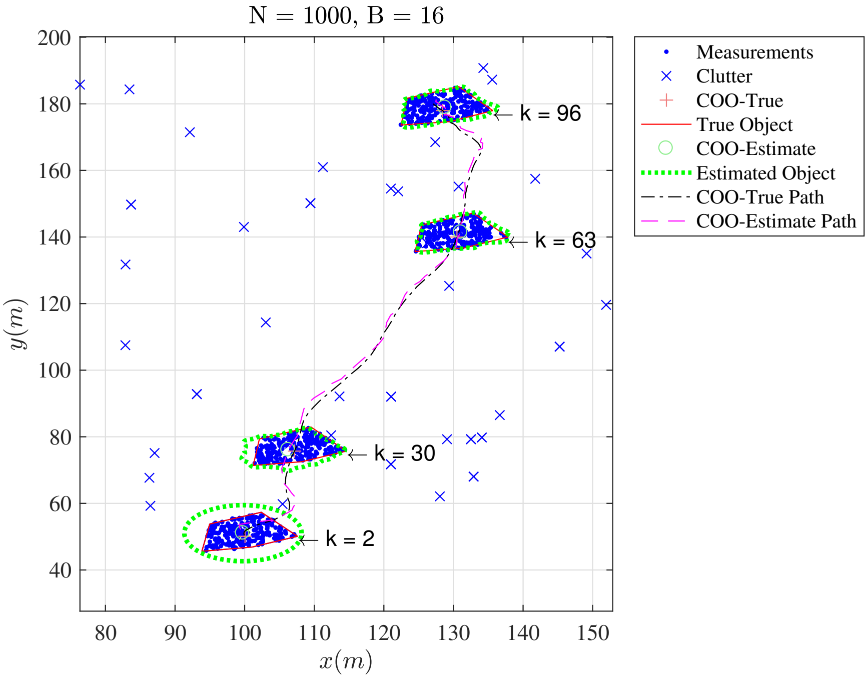
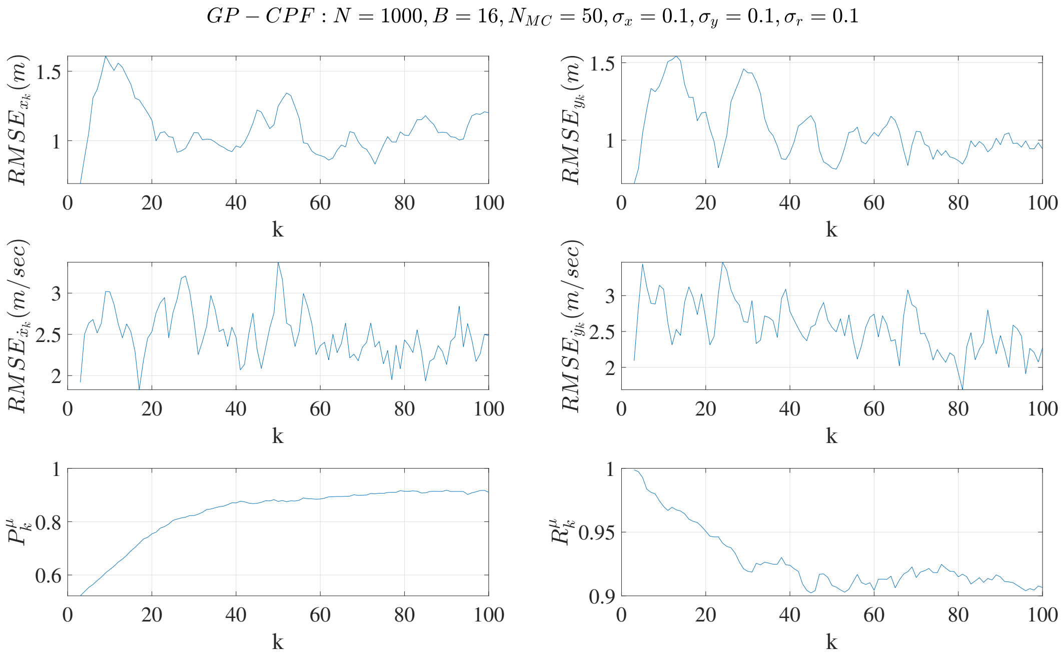
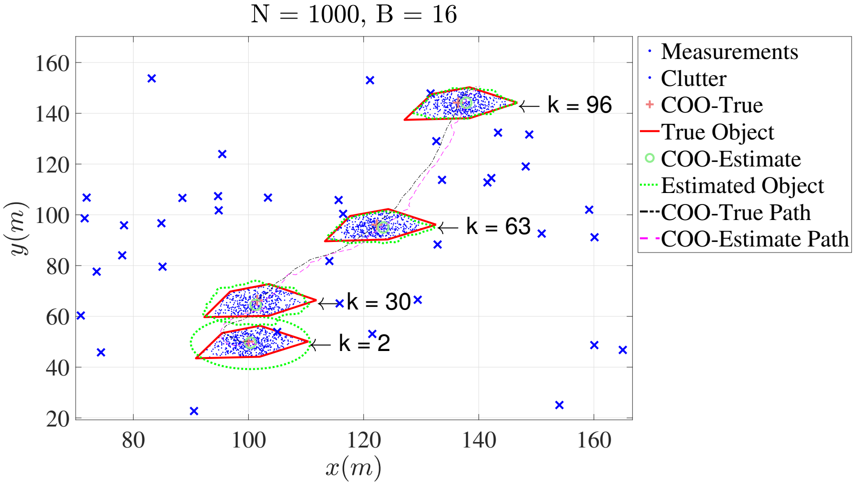


Fig. 5.3 **Tracking snapshots on uniformly distributed surface measurements**. The figure shows tracking snapshots at samples 2, 30, 63 and 96.



Results of 50 Monte Carlo runs on Gaussian distributed surface measurements. This figure shows performance results of GPCPF tracking an irreularly shaped target using Gaussian distributed surface measurements.

Fig. 5.4



Tracking snapshots on Gaussian distributed surface measurements. The figure shows tracking snapshots at samples 2, 30, 63 and 96.

Fig. 5.5

5.5 Conclusions

This chapter proposes a Gaussian process approach in a convolution particle filter framework to track an arbitrarily shaped extended or group target using noisy sensor data with clutter. The extended target is a physical entity which occupies several sensor resolution cells. Each cell generates a sensor measurement from a portion of the target, also called reflection point. The reflection points change from sample to sample and are assumed to follow a motion pattern that is sufficiently described by the target kinematics. The group target consists of many physical entities or point targets moving with similar kinematics. The point targets are assumed to follow a motion pattern that is sufficiently described by the kinematics of the group. Both the problems, extended and group target tracking, aim at recursive estimation of the target kinematics and shape or extent estimation using sensor measurements.

The target shape is modelled using a novel Von-Mises covariance kernel based Gaussian process. The Gaussian process model keeps track of the extent using the state samples and measurement simulations of the convolution particle filter. A novel convolution kernel is proposed for the measurement likelihood and clutter handling. The resulting Gaussian process convolution particle filter can work in a quick time when the extended target is described by a highly non-linear kinematic model.

The performance is evaluated by simulating an irregular pentagon shaped target moving under a correlated velocity model using noisy sensor measurements with clutter. The sensitivity of the approach towards the statistical properties of the measurements is validated by assuming two different types of distributions namely uniform and Gaussian. The performance in both scenarios is comparable. The positional accuracy is approximately 1 m in 50 Monte Carlo runs. The shape precision and recall are around 0.9. Future work will be focused on tracking of multiple extended or group targets using real data.

Chapter 6

A Gaussian Process Convolution Particle Filter for Multiple Extended Targets Tracking with Irregular Shapes

Abstract

In this chapter, a new approach for tracking multiple irregularly shaped extended targets using surface measurements is proposed. The Gaussian Process Convolution Particle Filter [1] (Chapter 5), designed to track a single extended or group target, is extended for tracking of multiple extended targets. A convolution kernel is proposed to determine the multi-target likelihood. A target birth and death model, based on the proposed approach, is also introduced for automatic initiation and deletion of the targets. The proposed approach is validated on real-world LiDAR data which shows that the approach is efficient in tracking multiple irregularly shaped extended targets in challenging scenarios involving occlusion, dense clutter and low target detection.

6.1 Introduction

This chapter extends the GPCPF approach for tracking of single extended or group target, proposed in Chapter 5 [1], to tracking of multiple extended targets. The data association and the state estimation for the multiple targets are achieved simultaneously using the proposed approach. The contributions of this chapter are enumerated in Section 6.1.1. The CoO, extent, and measurement models are presented in Section 6.2. The proposed multiple ETT GPCPF is given in Section 6.3. This section includes the details of the multiple target state estimation. The target birth, existence, and death models for tracking the number of targets is also given in this section. The extension of the proposed approach for target merging, splitting, and spawning is also presented. The performance validation and results are given in Section 6.4 followed by conclusions in Section 6.5.

6.1.1 Contributions

The contributions of this work are as follows; (i) A new Gaussian process convolution particle filter (GPCPF) based approach for tracking multiple extended targets having irregular shapes is proposed. (ii) A new convolutional kernel is proposed to track different complex shaped targets using surface measurements without any prior knowledge of the measurement statistics. The typical complex-shaped multiple extended targets tracking methods require prior information of the target size or the statistical properties of the measurements (see Section 4.3.7) [28, 27]. (iii) A new target birth and death model based on the likelihood is proposed. This framework treats the target detection, false-alarm rejection, target existence, and death in a probabilistic framework without the requirement of an explicit likelihood function. (iv) The performance validation of the proposed approach on real data is presented in the results section.

6.2 System Model

6.2.1 System Dynamics Model

The dynamics of the centre of the object (CoO) are assumed independent of those of the target shape. The multiple targets states are assumed to evolve independent of each other. The discrete time CoO state update equation is given below:

$$\boldsymbol{x}_k^C = (\boldsymbol{I}_{N_k} \otimes \boldsymbol{F}^C)\boldsymbol{x}_{k-1}^C + \boldsymbol{w}_k^C, \quad \boldsymbol{w}_k^C \sim \mathcal{N}(0, \boldsymbol{I}_{N_k} \otimes \boldsymbol{Q}^c), \tag{6.1}$$

where $\boldsymbol{x}_k^C = \left[(\boldsymbol{x}_{1,k}^C)^T, (\boldsymbol{x}_{2,k}^C)^T, \cdots, (\boldsymbol{x}_{N_k,k}^C)^T \right]^T$ represents the multiple targets CoO state vector, $\boldsymbol{x}_{t,k}^C$ is the CoO state of the t^{th} target and N_k represents the number of extended targets at time k. The extent states dynamics is modelled as a random walk (see (5.25)) [1] and is described by the following equation:

$$\boldsymbol{x}_k^E = (\boldsymbol{I}_{N_k} \otimes \boldsymbol{I}_B)\boldsymbol{x}_{k-1}^E + \boldsymbol{w}_k^E, \quad \boldsymbol{w}_k^E \sim \mathcal{N}\big(0, \boldsymbol{I}_{N_k} \otimes \boldsymbol{Q}^E\big), \tag{6.2}$$

where $\boldsymbol{x}_k^E = \Big[\big(\boldsymbol{x}_{1,k}^E\big)^T, \big(\boldsymbol{x}_{2,k}^E\big)^T, \cdots, \big(\boldsymbol{x}_{N_k,k}^E\big)^T\Big]^T$ represents the multiple targets extent state vector, $\boldsymbol{x}_{t,k}^E$ is the extent state of the t^{th} target and B is according to (5.16). The process noise covariance matrix $\boldsymbol{Q}^E$ can be modelled based on the prior knowledge of the targets being tracked, e.g. if the targets are axis-symmetric then an axisymmetric covariance kernel can be used to determine this matrix. If there is no prior knowledge of the target shape, then it can be modelled according to (5.25) [1].

6.2.2 Multiple Target State Vector

The multiple targets state vector $\boldsymbol{x}_k$ is given below:

$$\boldsymbol{x}_k = \Big[(\boldsymbol{x}_{1,k})^T \quad (\boldsymbol{x}_{2,k})^T \quad \cdots \quad (\boldsymbol{x}_{N_k,k})^T\Big]^T, \tag{6.3}$$

$$\boldsymbol{x}_{t,k} = \Big[(\boldsymbol{x}_{t,k}^C)^T \quad (\boldsymbol{x}_{t,k}^E)^T\Big]^T, \tag{6.4}$$

where $\boldsymbol{x}_{t,k}$ represents the t^{th} target state. The extent states consist of the radial extent of the target at B different angles from the CoO. The CoO and the extent states are given below:

$$\boldsymbol{x}_{t,k}^C = \Big[x_{t,k} \quad \dot{x}_{t,k} \quad y_{t,k} \quad \dot{y}_{t,k}\Big]^T, \tag{6.5}$$

$$\boldsymbol{x}_{t,k}^E = \Big[r_{t,k}^1 \quad r_{t,k}^2 \quad \cdots \quad r_{t,k}^B\Big]^T. \tag{6.6}$$

where $(x_{t,k}, y_{t,k})$ and $(\dot{x}_{t,k}, \dot{y}_{t,k})$ represent, respectively, position and velocity of the t^{th} target, $r_{t,k}^n$ represents the radial value of the t^{th} target at the n^{th} angle of the input vector $\boldsymbol{\theta}^E$ (see (5.16)).

6.2.3 Shape Model and the GP

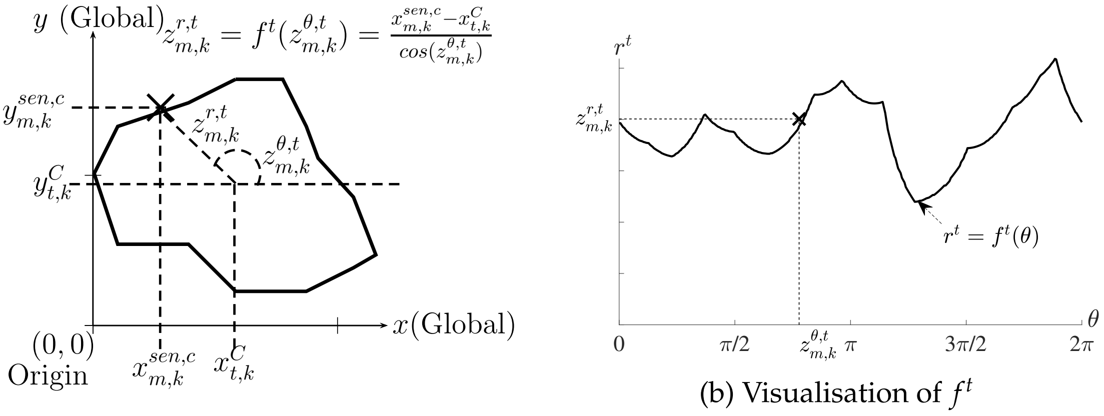
The shape of the target is assumed to be star-convex[1] and is modelled using a Gaussian Process (GP) as proposed in [26]. The extent is modelled as a function of

[1] A polygon is called star-convex, if all line segments from its centre to the boundary lie inside the same polygon.

the angle from the CoO. The angle to radial extent mapping function of the t^{th} target is given by the following equation:

$$r^t = f^t(\theta), \tag{6.7}$$

where r^t represents the radial and f^t represents the mapping function of the t^{th} target extent. As the target can have any arbitrary shape hence f^t is a non-linear function and a GP is used to model this mapping function. This is explained in Figs 6.1a and 6.1b. Fig. 6.1a shows a t^{th} extended target (thick solid line) in a global (sensor) Cartesian frame. The sensor measurements and the CoO kinematics are modelled in the global frame. The extent states are modelled in the polar frame local to each target. The t^{th} target's local frame has origin located at $(x^C_{t,k}, y^C_{t,k})$. The radial extent r^t of the target from the CoO is modelled as a function f^t of the angle θ in the local frame given by $r^t = f^t(\theta)$. The coordinates of the m^{th} measurement (cross) are shown in both the global and the t^{th} targets local frame. The non-linear relation between the two frames is also given for this measurement. Fig. 6.1b visualizes the non-linear radial function f^t of Fig. 6.1a. The origin corresponds to the centre of the t^{th} target. The m^{th} measurement is shown for comparison.



(b) Visualisation of f^t

(a) Extended target model

Fig. 6.1 **Group and extended target model**. Fig. 6.1a shows a t^{th} extended target (thick solid line) in a global (sensor) Cartesian frame. Fig. 6.1b visualizes the non-linear radial function f^t of Fig. 6.1a.

A modified Von Mises covariance kernel (see (5.12) and (5.13)) [1] is chosen as the covariance kernel.

6.2.4 Multiple Target Measurement Model

The sensor reports Cartesian coordinate measurements. The measurement noise vector is Gaussian distributed whose elements are assumed to be i.i.d. The measurement equation is given below:

$$\boldsymbol{z}_k^{sen,c} = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k^{sen,c}, \quad \boldsymbol{v}_k^{sen,c} \sim \mathcal{N}\Big(0, \boldsymbol{I}_L \otimes \mathrm{diag}(\sigma_x^2, \sigma_y^2)\Big), \tag{6.8}$$

where $\boldsymbol{z}_{j,k}^{sen,c} = \begin{bmatrix} x_{1,k}^{sen,c} & y_{1,k}^{sen,c} & x_{2,k}^{sen,c} & y_{2,k}^{sen,c} & \cdots & x_{L,k}^{sen,c} & y_{L,k}^{sen,c} \end{bmatrix}^T$ denotes the measurement vector, $x_{m,k}^{sen,c}$ and $y_{m,k}^{sen,c}$ are, respectively, the x and y coordinates of the m^{th} measurement, L represents the total number of measurements and $\boldsymbol{v}_k^{sen,c}$ denotes the measurement noise vector.

6.3 Multiple Extended and Group Targets Tracking using Gaussian Process Convolution Particle Filter

The multivariate density of the multiple extended targets states is multi-modal. Each mode of the density corresponds to an extended target in the real-world. The number of targets and hence the number of modes is assumed unknown. Furthermore, the measurements origin and the target kinematics (both of the CoO and the extent) are also assumed unknown. The sensor data is assumed to be noisy, gives multiple surface measurements per extended target and reports measurements from the extended targets as well as clutter. The existence, birth, and death probabilities of the targets are also unknown.

The multitarget state sampling is determined using the system dynamics defined in (6.1) and (6.2). The measurement sampling step maps the (Gaussian) state samples as irregularly shaped regions in the measurement space. The sensor measurements update the posterior density using the CPF kernel estimation.

6.3.1 State Sampling

The multitarget state sample of the i^{th} particle at time k is given below:

$$\tilde{\boldsymbol{x}}_k^i = \begin{bmatrix} (\tilde{\boldsymbol{x}}_{1,k}^i)^T & (\tilde{\boldsymbol{x}}_{2,k}^i)^T & \cdots (\tilde{\boldsymbol{x}}_{N_k,k}^i)^T \end{bmatrix}^T, \quad \tilde{\boldsymbol{x}}_{t,k}^i = \begin{bmatrix} (\tilde{\boldsymbol{x}}_{t,k}^{i,C})^T & (\tilde{\boldsymbol{x}}_{t,k}^{i,E})^T \end{bmatrix}^T, \tag{6.9}$$

where $\tilde{\boldsymbol{x}}^i_k$ represents the multiple targets state sample and $\tilde{\boldsymbol{x}}^i_{t,k}$ is the t^{th} target state. The CoO and the extent states samples are determined using equations (6.1) and (6.2), respectively.

6.3.2 Measurement Simulation

The measurement simulation of the i^{th} particle is given below:

$$\tilde{\boldsymbol{z}}^i_k = \Big[(\tilde{\boldsymbol{z}}^i_{1,k})^T \quad (\tilde{\boldsymbol{z}}^i_{2,k})^T \quad \cdots \quad (\tilde{\boldsymbol{z}}^i_{N_k,k})^T\Big]^T, \tag{6.10}$$

where $\tilde{\boldsymbol{z}}^i_k$ represents the simulated measurement vector of the i^{th} particle and $\tilde{\boldsymbol{z}}^i_{t,k}$ is the measurement simulation of the corresponding t^{th} target. The target measurement simulation is given below:

$$\begin{aligned}
\tilde{\boldsymbol{z}}^i_{t,k} &= \begin{bmatrix}\tilde{x}^{i,C}_{t,k} & \tilde{y}^{i,C}_{t,k}\end{bmatrix}^T_{\times B} + \boldsymbol{J} \odot \left(\tilde{\boldsymbol{x}}^{i,E}_{t,k} \otimes \begin{bmatrix}1 & 1\end{bmatrix}^T\right) + \tilde{\boldsymbol{v}}^i_{t,k}, && (6.11)\\
\tilde{\boldsymbol{v}}^i_{t,k} &\sim \mathcal{N}\Big(0, \boldsymbol{I}_B \otimes \mathrm{diag}(\sigma_x^2, \sigma_y^2)\Big), &&\\
\boldsymbol{J} &= \Big[\cos(\theta^1) \quad \sin(\theta^1) \quad \cdots \quad \cos(\theta^B) \quad \sin(\theta^B)\Big]^T, && (6.12)
\end{aligned}$$

where $\tilde{x}^{i,C}_{t,k}$ and $\tilde{y}^{i,C}_{t,k}$ represent, respectively, the Cartesian coordinates of the centre of the i^{th} particle and the t^{th} target, $\odot$ is an element-wise vector product and $\tilde{\boldsymbol{v}}^i_{t,k}$ denotes the measurement noise vector.

6.3.3 The Convolution Particle Filter Kernel

The CPF kernel is used for the multi-modal density estimation. The CPF in a state space setting for the multiple targets tracking relies on two kernels. The first kernel $\mathcal{K}^{\tilde{x}}_b$ is defined for the sampling distribution in the state space. The second kernel $\mathcal{K}^{\tilde{z}}_b$ is defined for calculating the likelihood in the measurement space. As proposed in Chapter 5 [1], the sampled state vector $\tilde{\boldsymbol{x}}_k$ maps to multiple regions in the measurement space and is equivalent to the functionality of the kernel $\mathcal{K}^{\tilde{x}}_b$. Hence, it is not required to be explicitly defined. The likelihood kernel $\mathcal{K}^{\tilde{z}}_b$ is defined for the measurements originating from the targets as well as the clutter. The proposed multitarget likelihood kernel is given below:

$$\mathcal{K}_b^{\tilde{z}_k^i}(z) = \begin{cases} \mathcal{U}_{\cup_{t=1}^{N_k}\mathcal{P}_k^{t,i}}(z), & z \in \cup_{t=1}^{N_k}\mathcal{P}_k^{t,i}, \\ \mathcal{U}_\lambda(z), & \text{otherwise}, \end{cases}, \tag{6.13}$$

$$\mathcal{U}_{\cup_{t=1}^{N_k}\mathcal{P}_k^{t,i}}(z) = \frac{1}{\text{Polygon } \cup_{t=1}^{N_k}\mathcal{P}_k^{t,i} \text{ Area}}, \quad \mathcal{U}_\lambda(z) = \frac{1}{\text{Surveillance Area}}, \tag{6.14}$$

where $\mathcal{U}_{\cup_{t=1}^{N_k}\mathcal{P}_k^{t,i}}(z)$ is a uniform kernel with support defined over an area defined by the union of the N_k polygons and $\mathcal{U}_\lambda(z)$ is a uniform kernel with support defined over the complete surveillance area and it represents the clutter measurements. Each polygon corresponds to a probable target. The measurement simulation vector $\tilde{z}_k^i$ is a collection of N_k polygons in the measurement space. These polygons are coarse and a GP model is employed to smooth them. The polygons train the corresponding GP models. The GP regression (2.18) and (2.19) is employed to determine the fine regions in the measurement space, corresponding to each polygon. The union of these regions for an i^{th} particle is denoted as $\cup_{t=1}^{N_k}\mathcal{P}_k^{t,i}$.

6.3.4 Likelihood Calculation and Weight Update

Consider L measurements are received at time k from the sensor in Cartesian coordinates. To perform the likelihood calculation and the weight update, the measurements are first gated with the particle measurement samples. This gating is done in three steps. First, the measurements are gated based on their locations and subsequently, the measurement clustering information is included to improve the gating process.

The measurement vector $z_k^{sen,c}$ is clustered using DBSCAN [94]. This clustering algorithm classifies the measurements in groups. It is an unsupervised approach and is suitable for the target tracking algorithms, where the number of targets is unknown. The Cartesian measurement vector $z_k^{sen,c} = \begin{bmatrix} x_{1,k}^{sen,c} & y_{1,k}^{sen,c} & \cdots & x_{L,k}^{sen,c} & y_{L,k}^{sen,c} \end{bmatrix}^T$ is converted to the polar measurement vector with respect to t^{th} target of the i^{th} particle as given below:

$$\begin{bmatrix} z_{t,k}^{r,i} & z_{t,k}^{\theta,i} \end{bmatrix}^T = cart2pol\left(z_k^{sen,c} - \begin{bmatrix} \tilde{x}_{t,k}^{i,C} & \tilde{y}_{t,k}^{i,C} \end{bmatrix}_{\times L}^T \right), \tag{6.15}$$

where $\boldsymbol{z}^{r,i}_{t,k} = \left[z^{r,i}_{1,k}, \cdots, z^{r,i}_{L,k}\right]^T$ and $\boldsymbol{z}^{\theta,i}_{t,k} = \left[z^{\theta,i}_{1,k}, \cdots, z^{\theta,i}_{L,k}\right]^T$ represent, respectively, the polar range and angle measurement vectors. The GP model, trained on the simulated measurement vector (6.10), is used to predict the radial extent of the target at the angle vector $z^{\theta,i}_{t,k}$. The GP predicted (also called simulated) radial extent is given below:

$$\tilde{\boldsymbol{z}}^{r,i}_{t,k} = \boldsymbol{K}_{z^{\theta,i}_{t,k}\boldsymbol{\theta}^E}(\boldsymbol{K}_{\boldsymbol{\theta}^E\boldsymbol{\theta}^E})^{-1}\tilde{\boldsymbol{x}}^{i,E}_{t,k}, \tag{6.16}$$

where $\tilde{z}^{r,i}_{t,k}$ is the predicted radial extent vector.

As a first step of the gating process, all measurements are considered belonging to the t^{th} target of the i^{th} particle whose simulated range is less than or equal to the measured range, that is, $\tilde{z}^{r,i}_{t,k} \leq z^{r,i}_{t,k}$. The clustering information is used in the second and third steps of the gating process. The cluster labels of all the measurements, successfully gated in the first step, are combined in a label vector and its mode is determined. The measurements with a label different from the mode of the label vector are considered not gated. In the third step, the measurements are also considered not-gated if a specific percentage (15% value used in Section 6.4) of the remaining cluster is not gated. The measurements gated with one target are not considered for updating the other targets. The particle weights update equation is as follows:

$$w^i_k = w^i_{k-1}\prod_{j=1}^{L}\mathcal{K}^{\tilde{z}^i_k}_b\left(\left[x^{sen,c}_{j,k} \quad y^{sen,c}_{j,k}\right]^T\right). \tag{6.17}$$

6.3.5 Multiple Targets State Update

The conditional multi-target state density can be written as:

$$p(\boldsymbol{x}_k|\boldsymbol{z}^{sen,c}_{1:k}) = \frac{p(\boldsymbol{x}_k, \boldsymbol{z}^{sen,c}_{1:k})}{\int p(\boldsymbol{x}_k, \boldsymbol{z}^{sen,c}_{1:k})d\boldsymbol{x}_k}, \tag{6.18}$$

where $\boldsymbol{z}^{sen,c}_{1:k}$ represents all the measurements from time-step 1 to k. Along the lines of adaptive CPF modelled in [71], the kinematic and extent states are sampled separately. The state update equations are given below:

$$p^{n_p}_k(\boldsymbol{x}_k|\boldsymbol{z}^{sen,c}_{1:k}) = \sum_{m=1}^{N_k} p^{n_p}_k(\boldsymbol{x}_{t,k}|\boldsymbol{z}^{sen,c}_{1:k}), \tag{6.19}$$

$$p_k^{n_p}(\boldsymbol{x}_{t,k}|\boldsymbol{z}_{1:k}^{sen,c}) = \frac{\sum_{i=1}^{n_p} w_{t,k}^i \tilde{\boldsymbol{x}}_{t,k}^i \mathcal{K}_b^{\tilde{\boldsymbol{z}}_k^i}(\boldsymbol{z}_{1:k}^{sen,c})}{\sum_{i=1}^{n_p} \mathcal{K}_b^{\tilde{\boldsymbol{z}}_k^i}(\boldsymbol{z}_{1:k}^{sen,c})}, \tag{6.20}$$

where n_p is the number of particles and $w_{t,k}^i$ is the weight of the t^{th} target of the i^{th} particle. The t^{th} target state update is given below:

$$\hat{\boldsymbol{x}}_{t,k} = \frac{\sum_{i=1}^{n_p} w_{t,k}^i \tilde{\boldsymbol{x}}_{t,k}^i}{\sum_{i=1}^{n_p} w_{t,k}^i}. \tag{6.21}$$

6.3.6 Target Existence, Birth and Death Model

The targets enter, move across and leave the area of interest. The sensors can also report clutter. These scenarios are modelled using different processes, which are a modification of the method proposed in [95]. The entry is modelled by a pre-birth and birth process, the pass-through is modelled by an existence process while the exit is modelled by a disappearance or death process. The sensor clutter is modelled as a false alarm process. Each extended target state is augmented by an existence variable $e_{t,k} \in \{0, 1, 2\}$ which specifies the existence state of the t^{th} extended target at time k. The relation between the different processes and the existence variable is shown in Table 6.1.

Table 6.1 Existence processes

Process	$e_{t,k}$	$e_{t,k-1}$
Pre-birth	0	0
Birth	1	0
Existing	1	1
Death	0	1
False alarm	2	0

The existence variable is assigned a value based on the target likelihood $\lambda_{t,k}$ and is given below:

$$\lambda_{t,k} = \frac{\sum_{i=1}^{n_p} w_{t,k}^i \prod_{j=1}^{L} K_b^{\tilde{\boldsymbol{z}}_k^i}\left(\begin{bmatrix} x_{j,k}^{sen,c} & y_{j,k}^{sen,c} \end{bmatrix}^T \right)}{\sum_{i=1}^{n_p} w_{t,k}^i}. \tag{6.22}$$

Two thresholds are defined to detect the target existence process. These are the birth threshold T_b and the death threshold T_d. The thresholds are related to the existence variable as given below:

$$e_{t,k} = \begin{cases} 1 & \lambda_{t,k} \geq T_b, e_{t,k-1} = 0, \\ 0 & \lambda_{t,k} < T_b, e_{t,k-1} = 0, \\ 0 & \lambda_{t,k} \leq T_d, e_{t,k-1} = 1, \\ 1 & \lambda_{t,k} > T_d, e_{t,k-1} = 1, \\ 2 & \lambda_{t,k} \leq T_d, e_{t,k-1} = 0, \end{cases} \tag{6.23}$$

At any given time, the pre-birth, birth and the existing targets are part of the extended target state vector $\boldsymbol{x}_k$. The death and false-alarm targets are removed from this state vector at the end of the processing step. As a result, the size of this state vector changes, which is also depicted by the time-dependence of number of targets variable N_k.

The targets can appear from a region called birth region e.g it can be a door to the building entrance in a crowd tracking in a building problem. There are N_b number of birth regions in the area of interest. Each birth region is defined by a centre (x_b, y_b), which specifies the location of the centre of the birth targets, an initial velocity $(\dot{x}_b, \dot{y}_b)$ and a circular region of radius r_b, which specifies the initial shape of the target. The values of these parameters can be tuned according to the application.

6.3.7 Target Merging, Splitting and Spawning

The target merging occurs by design through the gating process. The splitting or spawning can be included through a modification of the birth process. All the unassociated measurements in a particular scan are classified using DBSCAN clustering method. All these unassociated clusters are considered as birth regions for the next scan. The mean and the variance of the positional measurements are used to define the centre and the size of the birth region, respectively. As a result, the target splitting or spawning is achieved.

6.4 Performance Validation

A sample of publicly available real-world data is considered for the performance validation [96]. This is a recorded data of the sensors installed on a car for real-world computer vision benchmarking. The benchmarking problems are related to an autonomous vehicle project. Multiple sensors data, installed on an observer vehicle, is available for various scenarios. The sensors include two grayscale cameras, two colour cameras and one laser scanner. The data of the laser scanner (HDL-64E LiDAR) sensor is considered for the performance evaluation of the proposed approach. The ground truth data is not available and is constructed manually using the image data from one of the colour cameras. The ground truth of the states is calculated only for those time samples when the complete target is visible in the image data.

The 3D LiDAR data is reported in the body-fixed frame of the observer vehicle. The data is synchronized with the images obtained from the cameras. A 3D to 2D transformation matrix is used to project the data on the 2D image frame. The ETT is done in the image frame and compared with the ground truth data, which is also available in the image frame.

The given data sample consists of static and moving extended targets. The moving targets are considered as targets of interest for the performance validation. Hence, the static extended targets are treated as clutter. Four moving targets (cars) cross in front of the observer vehicle during a total of 66 time samples. These are available in the scene at different time instants which are explained next. The first target in samples $k = 1 - 8$, the second in $k = 1 - 22$, the third in $k = 18 - 43$ and the fourth in $k = 35 - 60$. The time samples when the targets are completely visible are as follows. The first target in samples $k = 1 - 3$, the second target in $k = 3 - 20$, the third in $k = 23 - 39$ and the fourth in $k = 39 - 55$.

The different challenges in the data are a large number of the LiDAR data i.e. on average 0.1 million measurements are received per time sample, dense (static) clutter, occlusion and one of the targets is not perfectly detected by the sensor i.e. it is a stealthy target. This stealthy target poses an additional challenge of tracking similar extended targets having different measurement statistics.

The evaluation of the multiple targets state is done using the mean cardinality $card_k^{\mu}$ comparison, the positional and velocity root mean square errors (RMSE) of the CoO and the mean shape precision and recall in 200 Monte Carlo runs. The shape recall and precision have been used in computer vision for evaluating rectangular targets detection performance [93]. The RMSE errors and the shape recall and precision in N_{MC} Monte Carlo runs are calculated as given below:

$$\hat{E}_k^a = \frac{1}{N_{MC}} \sqrt{\sum_{i=1}^{N_{MC}} (a_k^i - \hat{a}_k^i)^2}, \tag{6.24}$$

$$R_k^\mu = \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} \frac{Area(\mathcal{T}_k^i \cap \mathcal{E}_k^i)}{Area(\mathcal{T}_j^i)}, \tag{6.25}$$

$$P_k^\mu = \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} \frac{Area(\mathcal{T}_k^i \cap \mathcal{E}_k^i)}{Area(\mathcal{E}_k^i)}, \tag{6.26}$$

where $\hat{E}_k^a$ represents the RMSE of the evaluation parameter a at time k, a represents the true and $\hat{a}$ represents the estimated value, R_k^μ and P_k^μ represent, respectively, the mean shape recall and precision at time k.

6.4.1 System Dynamics

The CoO dynamics are modelled using a nearly constant velocity (NCV) motion model given in Section 3.4.1. The extent process noise covariance is modelled using a periodic covariance kernel [39] and is given below:

$$\boldsymbol{Q}^E = \boldsymbol{K}^{per}_{\boldsymbol{\theta}^E \boldsymbol{\theta}^E}, \tag{6.27}$$

where $\boldsymbol{K}^{per}_{\boldsymbol{\theta}^E \boldsymbol{\theta}^E}$ represents a GP covariance matrix calculated using a periodic covariance kernel (2.15) and (2.4).

6.4.2 Birth and Death Model for the Test Scenario

The objective is to track the moving targets of interest i.e. pedestrians, cyclists, vehicles etc. The birth or death model is enhanced based on the problem at hand. To detect and track only the moving targets, two speed thresholds are introduced in the birth or death model. These are the low speed threshold V_l and high speed threshold V_h. The targets of interest are assumed faster than V_l and slower than V_h.

6.4.3 Measurement Clustering

The LiDAR data is in 3D. The measurements coming from the ground and from very high targets (which cannot be considered moving targets on the roads) are filtered based on the height information. The filtered data is clustered based on the depth value using 1D DBSCAN clustering. The measurements are then projected to the 2D image frame. The projected measurements are clustered using 2D DBSCAN clustering.

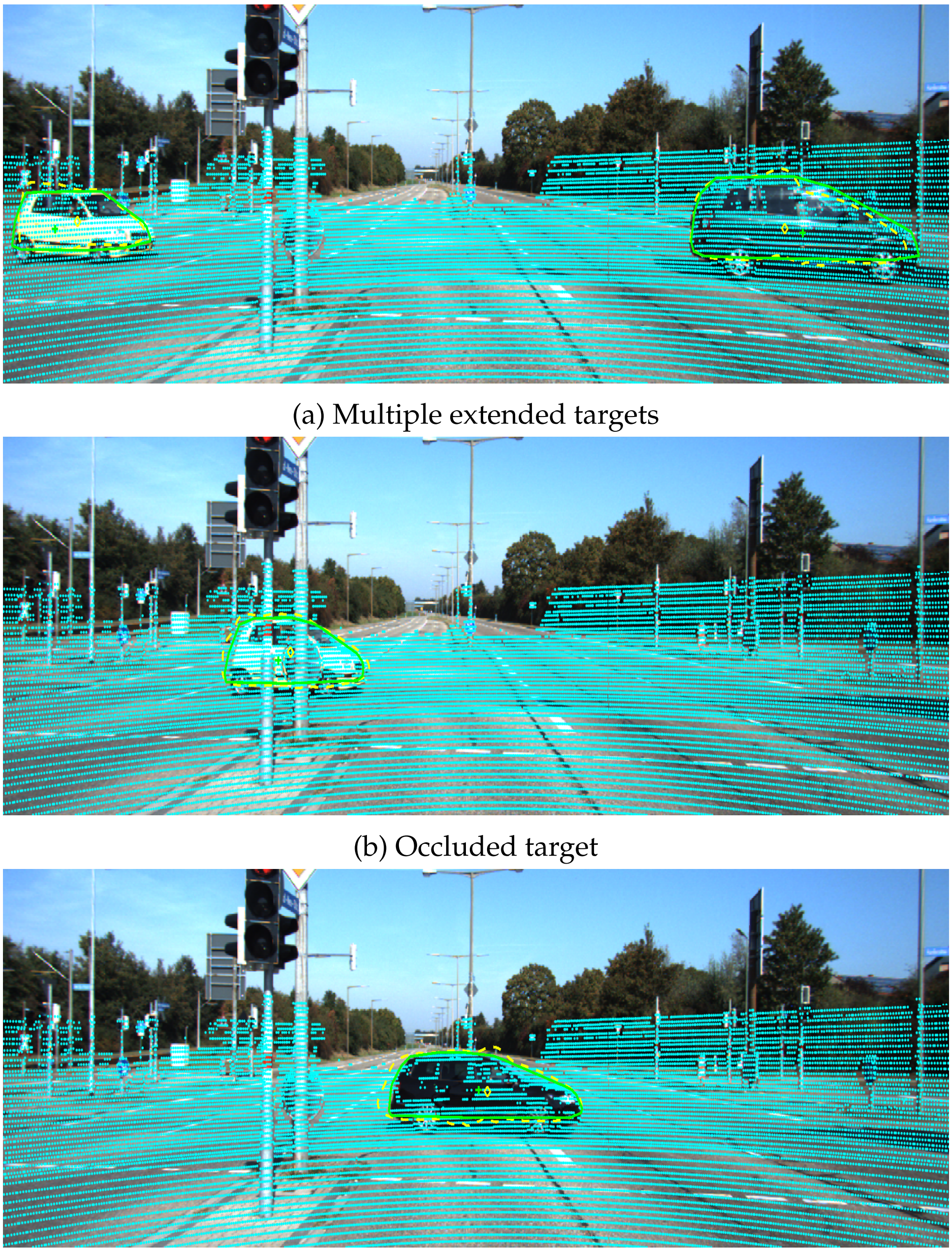
6.4.4 Filter Parameters

The filter parameters are given as follows. The total number of time samples are $K = 66$ samples, the sampling time is $T = 0.1\ s$, the velocity standard deviations are $q_x^{NCV} = 250^2\ pixel^2/s^2$ and $q_y^{NCV} = 25^2\ pixel^2/s^2$, the hyperparameters of the extent process noise covariance kernel are $\sigma_m^2 = \frac{1}{10}\ pixel^2$, $\sigma_c^2 = 0$ and $l = 0.2\ radian$, the surveillance volume is $Area(\lambda) = 1242\ pixel \times 345\ pixel$. The hyperparameters of the GPCPF kernel are $l = 0.22\ radian$, $\sigma_c^2 = 1\ pixel^2$ and $\sigma_m^2 = \frac{1}{40}\ pixel^2$. The number of particles is $n_p = 500$, number of basis is $B = 36$, the birth threshold is $T_b = 0.01$, death threshold is 0.001, the low speed threshold is $V_l = 200\ pixel/s$ and the high speed threshold is $V_h = 1000\ pixel/s$. The 1D DBSCAN clustering parameters are $epsilon = 1.25\ pixel$ and the minimum number of points are 24. The 2D DBSCAN clustering parameters are $epsilon = 50\ pixel$ and the minimum number of points are 80. The sensor noise variances are $\sigma_x^2 = \sigma_y^2 = 0.0025\ pixel^2$.

6.4.5 Results

The challenging scenarios and the tracking results at three chosen time samples are shown in Fig 6.2. The multiple moving extended target tracking, occlusion and differently distributed surface measurements are the complexities of the considered data depicted in subfigures 6.2a, 6.2b and 6.2c, respectively. The projected LiDAR data (cyan dots) is overlayed on the camera image. The ground truth target is represented by the (green) solid line and the estimated target is represented by (yellow) dotted lines. The ground truth CoO is represented by (green) plus and the estimated CoO is represented by a (yellow) diamond. Fig. 6.2a shows two moving extended targets (cars) are tracked whereas the static extended targets (signal post, trees, etc.) are treated as clutter. Fig. 6.2b shows the tracking of the moving target (white car) is occluded by the two static extended targets. Fig. 6.2c shows the front half of the car is picked up by the sensor while few measurements are reported from the back half of the car. Moreover, the statistical properties of the sensor measurements from the black car are different from the other moving and static extended targets. The measurement density is different from other similar targets.

The proposed algorithm detects and tracks this target, which shows that it is not sensitive to the statistical properties of the sensor measurements.



(a) Multiple extended targets

(b) Occluded target

(c) Differently distributed surface measurements

Fig. 6.2 **Challenging scenarios**. This figure shows tracking snapshots to depict the three challenges i.e. tracking multiple moving extended targets, occlusion and differently distributed surface measurements in subfigures 6.2a, 6.2b and 6.2c, respecively.

The mean cardinality is shown in Fig. 6.3. A delay in the target detection can be observed for all four targets. This is since the shape is not detected in the initial time steps as the complete target is not visible. Moreover, a moving target can be determined from the data of minimum two time samples. An error in the cardinality is observed between samples $25 - 30$. It is because the measurement statistics of the black car change considerably during these time samples. A large number of particles can be used to improve the cardinality at these time samples at the expense of the processing time. It can be observed that although dense clutter is present, no false alarms are observed.

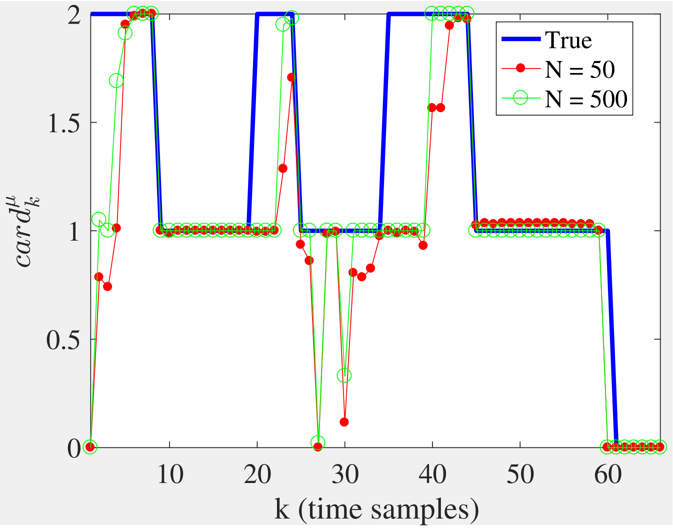
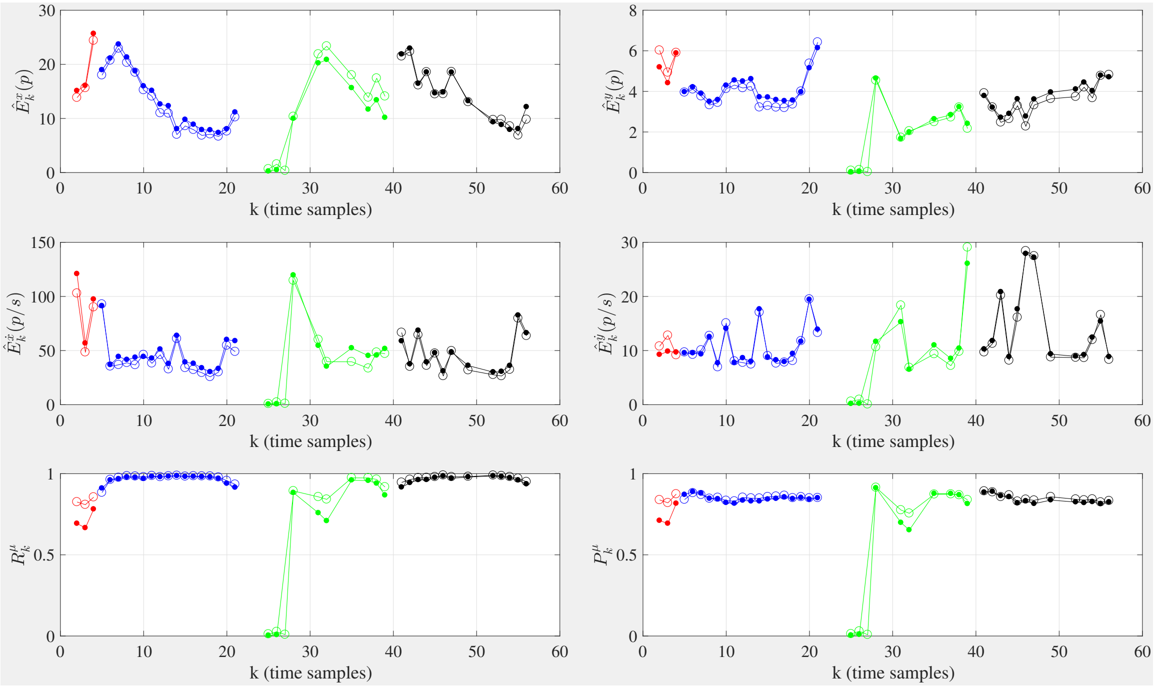


Fig. 6.3 **Mean cardinality**. The figure shows the true (blue thick line) and mean estimated cardinality using $n_p = 500$ (green circle line) and $n_p = 50$ (red dot line).

The average state estimate errors are shown in Fig. 6.4. The distance unit is in the frame of reference of the image that is pixels denoted as p. The positional RMSE in x is less than 25 p, y is less than 7 p, $\dot{x}$ is less than 110 p/s and $\dot{y}$ is less than 30 p/s. The mean shape recall is greater than 0.9 for most of the time steps, which shows that more than 90% of the true shape has been recalled all the time. The mean shape precision is more than 0.8 most of the time which shows that less than 20% of the estimated shape is false.

The program was run on MATLAB R2016b and a Windows 10 (64 bit) Desktop computer installed with an Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz(4 CPUs) and 8GB RAM. The computational time is 52 s per time sample for $n_p = 500$ particles. This is due to a large number of sensor measurements per time sample. The processing time is improved to 4.5 s using $n_p = 50$ particles. The mean cardinality and the state estimate errors for $n_p = 50$ are also given in the Figs. 6.3 and 6.4, respectively. It can be observed that all four targets are tracked and there is no false alarm. The performance of the state estimates is almost similar and the cardinality estimates are slightly degraded. The processing time can be further improved by optimizing the code and running in C++.



	T-1	T-2	T-3	T-4
$n_p = 50$	—●—	—●—	—●—	—●—
$n_p = 500$	—○—	—○—	—○—	—○—

Fig. 6.4 **State errors**. The state estimation errors are shown for the four moving targets (T-1 in red, T-2 in blue, T-3 in green and T-4 in black) for two different number of particles that is $n_p = 500$ (solid line with dots) and $n_p = 50$ (solid line with circles).

6.5 Conclusions

This chapter proposes a Gaussian process convolution particle filter approach for tracking of irregularly shaped multiple extended or group targets using noisy measurements with clutter. The target extent is modelled using a Von-Mises based Gaussian process. The multi-target state is sampled from a mixture of Gaussian state samples. The multi-target state is simulated using the measurement model to give simulated measurements. A novel convolution kernel is proposed for the multi-target likelihood calculation. The resulting Gaussian process convolution particle filter recursively estimates the kinematics and the extent of multiple targets. The time varying number of targets is estimated using a novel target birth/existence/death model. The proposed target model is a decision based approach. The decisions are carried out in a statistical framework by relating various thresholds to the multi-target likelihood. A pre-processing step, measurement clustering using DBSCAN, is recommended to improve the data association and computation complexity.

The performance of the proposed multiple extended or group target tracking approach is evaluated using publicly available real world data. It consists of recorded data of cameras and a LiDAR installed on a car for developing autonomous driving algorithms. The LiDAR data is considered for tracking and compared to the ground truth, created using camera data. The considered data poses four challenges namely dense clutter, multiple static and moving extended targets, occlusion and stealthy (low observable) targets. The precision and recall of the proposed filter are above 0.8 and 0.9, respectively. The processing time is shown to improve by reducing the number of particles from 500 to 50 with a slight compromise on the performance. The evaluation also shows that the proposed approach is insensitive to the statistical properties of the measurements.

In the future, the GPCPF will be extended to the tracking scenarios involving closely moving irregularly shaped extended or group targets. The birth/existence/death model will be enhanced to include the probability of detection measures.

Chapter 7

Spatio-temporal Gaussian Process for Extended and Group Target Tracking with Irregular Shapes

Abstract

This chapter presents a generic spatio-temporal Gaussian process (STGP) method for tracking an irregularly shaped extended target. The complex shape is represented by key points and their parameters are estimated both in space and time. This is achieved by factorization of the power spectral density function of the STGP covariance function. A new form of the temporal covariance kernel is derived with the theoretical expression of the filter likelihood function. Solutions to both the filtering and the smoothing problems are presented. A novel concept for determining of the center of the target and its kinematics is given. Various approaches based on the proposed method can be derived for tracking of rigid and non-rigid irregularly shaped extended targets. A thorough evaluation of the performance in a simulated environment shows that an example approach based on the proposed STGP method outperforms the state-of-the-art GP extended Kalman filter approach [26], with up to 90% improvement in the accuracy in position, 95% in velocity and 7% in shape, while tracking a simulated asymmetric non-rigid target. The tracking performance improvement for a non-rigid irregular real target is up to 43% in position, 68% in velocity, 10% in the recall and 115% in the precision measures.

7.1 Introduction

This chapter proposes an efficient Gaussian process method for tracking of non-rigid extended and group targets. This is achieved by modelling the extent of the target using a two dimensional, spatial and temporal, model also called the spatio-temporal GP (STGP). In [97], it has been shown that if one of the input dimensions of the STGP is stationary (and some other conditions), then it has an equivalent state space representation which can be solved using Rauch-Tung-Streibel smoothing. Various real-time approaches based on the proposed method can be realized according to the application area. The main contributions of this work are given in Section 7.1.1. The theoretical background of the STGP is covered in Section 7.2.1 and the Rauch-Tung-Streibel Smoother is described in Section 7.2.2. The proposed method of the ETT is explained in Section 7.3 and demonstrated using an example is given in Section 7.4. The performance evaluation is presented in Section 7.5 followed by conclusions.

7.1.1 Contributions

The contribution area of the proposed model in the ETT literature is depicted in Fig. 7.1. A complex extent model namely STGP (bold) is proposed in this work.

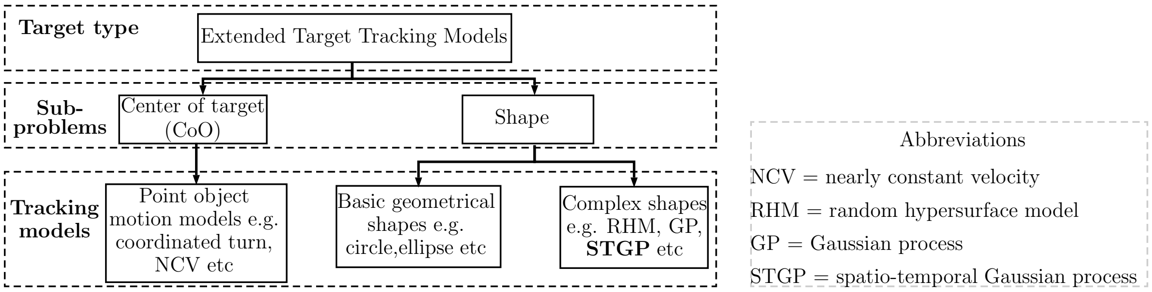

Target type
Extended Target Tracking Models
Sub-problems
Center of target (CoO)
Shape
Tracking models
Point object motion models e.g. coordinated turn, NCV etc
Basic geometrical shapes e.g. circle,ellipse etc
Complex shapes e.g. RHM, GP, **STGP** etc
Abbreviations
NCV = nearly constant velocity
RHM = random hypersurface model
GP = Gaussian process
STGP = spatio-temporal Gaussian process

Fig. 7.1 **The proposed innovation**. The figure gives a hierarchical representation of ETT research and highlights the contribution of this work within this paradigm.

The key contributions of this chapter are as follows; (i) A novel interpretation of the centre of an asymmetric extended target is presented (given in Section 7.3.1). (ii) The existing methods for ETT, including the GP approach (see Section 4.3.5) [26, 27], use simple models for the temporal changes in shape. As a result, the performance is compromised while tracking non-rigid extended targets. In this work, a novel non-rigid extended target tracker is proposed based on an STGP model, which includes both the spatial and the temporal correlations of the extent (See Section 7.3.3). (iii) Traditionally, the GP is a batch processing method and cannot be used for real-time applications. A GP based recursive filter for real-time ETT has been proposed in [26, 27] (see Section 4.3.5). Based on the theoretical results of [97], in this chapter, the full GP regression is proposed to be approximated using a fixed-lag Rauch-Tung-Streibel smoother to obtain a quasi-realtime method. This is the first time in the literature of ETT that the theoretical fundamentals of the equivalence between a batch and the recursive GP regression are described for deep understanding (See Section 7.3.8). (iv) The computational complexity of the Rauch–Tung-Streibel smoother, derived in [97], increases with time. After a specific number of samples, the smoother might not be implementable in real-time. This limitation is avoided by proposing a real-time fixed lag smoother based on the STGP model and real-time smoothing is achieved (See Sections 7.3.8 and 7.6.2). (v) The measurement likelihood is derived considering all noises (given in Section 7.3.5). Due to the complex relationship between the states and the measurements, the previous GP based implementations of the ETT ignored part of the noise (see Section 4.3.7). (vi) The performance validation of the proposed method is provided on real and simulated data. The computational complexity and the effect of smoother lag are also evaluated (given in Section 7.5).

7.2 Background Knowledge

7.2.1 Spatio-Temporal Gaussian Processes

An STGP is a stochastic process model for systems evolving in both space and time [97]. Let the spatial input be represented by u and the temporal input is represented by t, then an STGP can be used to model a functional mapping as given below:

$$f(u,t) \sim STGP(m(u,t), k(u,u';t,t')), \tag{7.1}$$

where $STGP(\cdot,\cdot)$ denotes the STGP model, $m(u,t)$ and $k(\theta,\theta';t,t')$ represent, respectively, the mean function and the covariance kernel of the STGP model. The STGP regression can be determined in the same way as the GP regression explained in Section 2.2. The time complexity of determining an STGP regression on a model trained at T time steps for N input locations is $\mathcal{O}(N^3T^3)$. As time progresses the computational expense increases beyond desired for most applications that require real-time processing. In [97] it has been shown that under some conditions, the STGP regression is equivalent to an infinite dimensional state space model. An infinite dimensional recursive filter and a smoother can then be used to perform the inference instead of using the batch processing method. An additional separability assumption, given below, simplifies the resulting model:

$$k(u,u';t,t') = k_u(u,u')k_t(t,t'), \tag{7.2}$$

where $k_u(\cdot,\cdot)$ and $k_t(\cdot,\cdot)$ represent the spatial and temporal covariance kernels, respectively. The conditions are given below:

ST1 The temporal (process) covariance is stationary

$$k_t(t,t') = k_t(t-t'). \tag{7.3}$$

ST2 The power spectral density (PSD) of the process is rational

$$S(\omega_u,\omega_t) = \mathcal{F}[k(u,u';t,t')] = \frac{\text{constant w.r.t } \omega_t}{\text{polynomial in } \omega_t^2}, \tag{7.4}$$

where $S(\cdot)$ represents the PSD of the process, ω_u and ω_t represent the Fourier frequency in the u and t domains, respectively and $\mathcal{F}[\cdot]$ denotes the Fourier transform .

ST3 The order of the temporal PSD is a multiple of 2

$$S(\omega_u,\omega_t) = \frac{q_t S(\omega_u)}{S(\omega_t^2)}, \tag{7.5}$$

where q_t denotes the spectral density of a white noise process driving the temporal dynamics.

ST4 The spectral factorization of PSD gives a stable transfer function i.e.

$$S(\omega_u,\omega_t) = G(\imath\omega_t)S(\omega_u)G(-\imath\omega_t)^T, \tag{7.6}$$

where $G(\imath\omega_t)$ and $G(-\imath\omega_t)$ represent the unstable and the stable transfer function components, respectively, and $\imath\omega_t$ represents the complex Fourier frequency.

As a result, the corresponding GP covariance matrices are also separable. Under the above conditions, the spatio-temporal stochastic process can be equivalently represented by an infinite dimensional dynamic system given below:

$$\frac{\partial \boldsymbol{f}(\boldsymbol{u},t)}{\partial t} = \boldsymbol{A}\boldsymbol{f}(\boldsymbol{u},t) + \boldsymbol{L}\boldsymbol{w}(\boldsymbol{u},t), \tag{7.7}$$

where $\boldsymbol{f}(\boldsymbol{u},t)$ is a function of the spatial input $\boldsymbol{u}$ at time t, $\boldsymbol{A}$ is the state transition matrix, $\boldsymbol{L}$ represents the noise effect and $\boldsymbol{w}(\cdot,\cdot)$ represents a zero mean continuous time white noise process.

The measurements are assumed to be arriving at discrete time. The equivalent discrete time model is given below:

$$\boldsymbol{f}(\boldsymbol{u},t_k) = \boldsymbol{F}_k\boldsymbol{f}(\boldsymbol{u},t_{k-1}) + \boldsymbol{w}_k(\boldsymbol{u}), \quad \boldsymbol{w}_k(\boldsymbol{u}) \sim \mathcal{N}(0,\boldsymbol{Q}(\boldsymbol{u},\boldsymbol{u}';T)), \tag{7.8}$$

$$\boldsymbol{z}_k = \boldsymbol{H}_k\boldsymbol{f}(\boldsymbol{u},t_k) + \boldsymbol{v}_k, \tag{7.9}$$

where $\boldsymbol{w}_k(\boldsymbol{u})$ represents the zero mean white process noise with corresponding covariance matrix $\boldsymbol{Q}(\cdot,\cdot;\cdot)$.

Given a system model of the form (7.8)–(7.9), recursive Bayesian filtering and smoothing solutions can be developed to estimate the function $\boldsymbol{f}(\boldsymbol{u},t_k)$. As a result, the computational complexity of the STGP regression is reduced to $\mathcal{O}(N^3T)$ and becomes linear in time.

7.2.2 Fixed Lag Rauch-Tung-Streibel Smoother

The state estimation for the model defined by (7.8) and (7.9) can be done using Bayesian inference methods given in Sections 3.2 and 3.2.1. The filtered states are further improved using a smoother. A fixed lag Rauch-Tung-Streibel Smoother (RTSS) is used to smooth the filtered output using a fixed lag or smoothing length. Given a smoothing length k_s, the smoothed state $\bar{\boldsymbol{x}}_k$ and the corresponding state error covariance $\bar{\boldsymbol{P}}_k$ are recursively updated using the following recursion [98], which is performed for the time-steps $\{k-1, k-2, \cdots, k-k_s\}$;

$$\boldsymbol{G}_k = \boldsymbol{P}_{k|k}\boldsymbol{F}^T(\boldsymbol{P}_{k+1|k})^{-1}, \tag{7.10}$$

$$\bar{\boldsymbol{x}}_k = \boldsymbol{x}_{k|k} + \boldsymbol{G}_k[\bar{\boldsymbol{x}}_{k+1} - \boldsymbol{x}_{k+1|k}], \tag{7.11}$$

$$\bar{\boldsymbol{P}}_k = \boldsymbol{P}_{k|k} + \boldsymbol{G}_k[\bar{\boldsymbol{P}}_{k+1} - \boldsymbol{P}_{k+1|k}]\boldsymbol{G}_k^T, \tag{7.12}$$

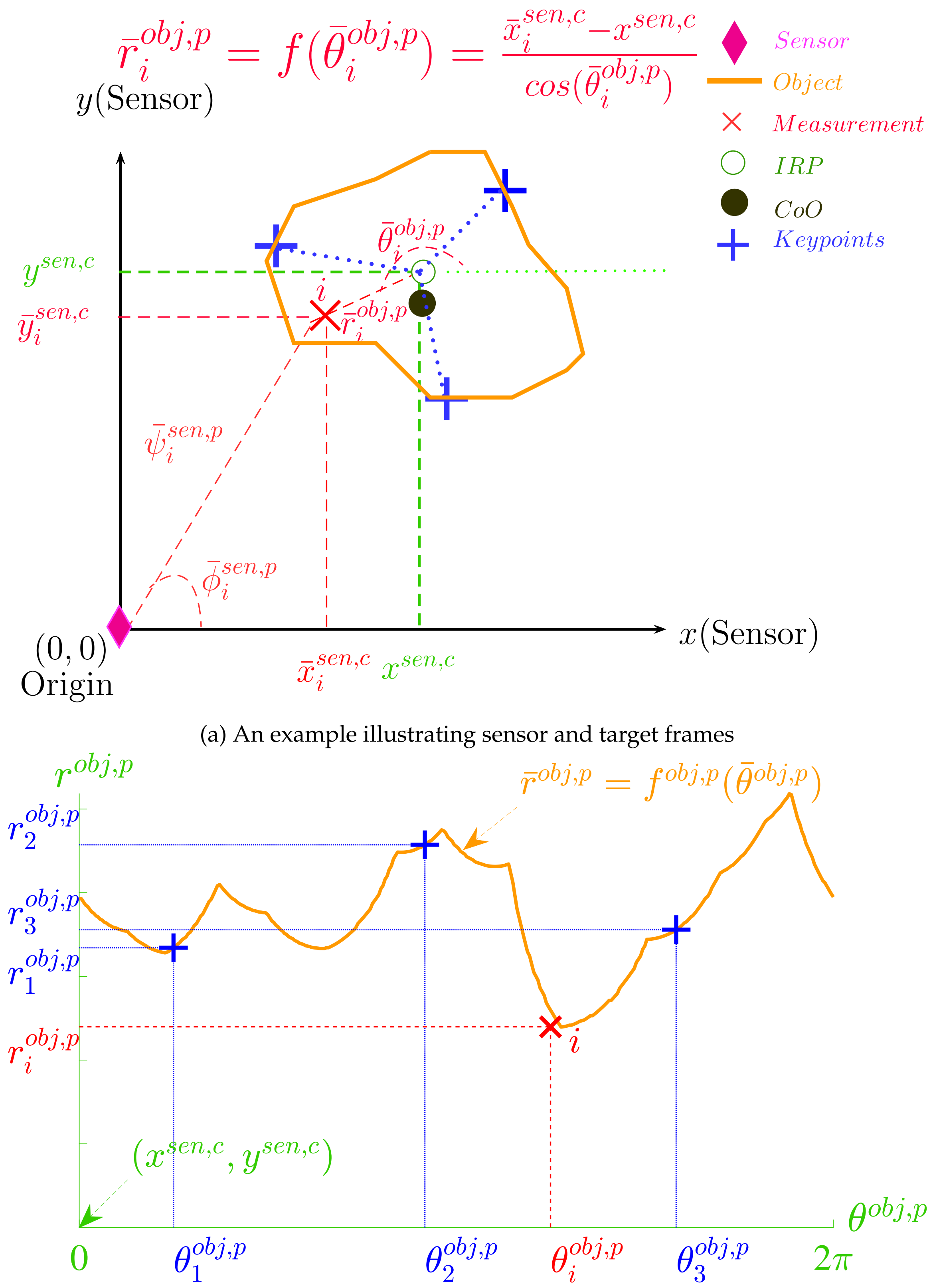
where $\boldsymbol{G}_k$ represents the gain matrix, and $\boldsymbol{x}$ and $\boldsymbol{P}$ represent, respectively, the state and error covariance of the Kalman filter. The smoother is initialized at the current time step k as $\bar{\boldsymbol{x}}_k = \boldsymbol{x}_{k|k}$ and $\bar{\boldsymbol{P}}_k = \boldsymbol{P}_{k|k}$.

7.3 The Proposed Extended Target Method

In this section, the proposed method for ETT and the associated multiple measurements likelihood is derived. A novel extent dynamical model based on a spatio-temporal GP (STGP) is used. A GP based model is preferred as it is a non-parametric method and can model complex shapes. Additionally, the STGP includes both the spatial and the temporal dynamics to give better shape estimation. The inference can be done using STGP batch regression, however in this work real-time ETT using an STGP model is presented. The real-time processing requires modelling the STGP in a state space form and deriving a recursive filtering and smoothing solution to the STGP state space model. The target is modelled as a star convex [26] shape as shown in Figs. 7.2a and 7.2b. Fig. 7.2a shows a sensor, an extended target, the CoO, two (Cartesian and polar) sensor frames with the origin at the sensor and a target (polar) frame (origin at internal reference point). The superscripts with the coordinates represent the frame it belongs to e.g. the coordinates of the internal reference point in the Cartesian sensor frame are $(x^{sen,c}, y^{sen,c})$. The state vector consists of radial extent values at equidistant points in the angle domain. A sensor measurement i is reported at $\bar{z}_i^{sen,p} = [\bar{\psi}_i^{sen,p}, \bar{\phi}_i^{sen,p}]^T$ in polar sensor frame. The coordinates of i in Cartesian sensor frame are $\bar{z}_i^{sen,c} = [\bar{x}_i^{sen,c}, \bar{y}_i^{sen,c}]^T$ and in target polar frame are $\bar{z}_i^{obj,p} = [\bar{r}_i^{obj,p}, \bar{\theta}_i^{obj,p}]^T$. Fig. 7.2b shows the radial function $\bar{r}^{obj,p} = f(\bar{\theta}^{obj,p})$ (orange) on the polar axis (green). The origin corresponds to the internal reference point i.e. $(x^{sen,c}, y^{sen,c})$ given in Fig. 7.2a. The i^{th} measurement (red cross) is shown for comparison with Fig. 7.2a. The extent state vector (blue plus) consists of radial values (shown on y-axis), which are equidistant points on the $\theta^{obj,p}$ axis. In this figure, the number of extent states is $B = 3$. The function is periodic with a period equal to 2π. An STGP, trained on coordinates of i and other measurements reported by the sensor, models the extent at complete angle domain.

7.3.1 Center of a Non-rigid Asymmetric Extended Target

The definition of the CoO depends on the application, for example, for uniformly dense targets the geometric centre (centroid) of the target shape is considered as the CoO. For non-uniformly dense targets, it can be defined as the centre of gravity or the centre of mass. In this work, targets with uniform density are considered. The CoO of rigid targets is assumed to lie on the same position relative to the target extent at all times. In such cases, a filter with nearly accurate initialization and appropriately modelled dynamics can provide efficient CoO estimates. In contrast, the CoO of a



(a) An example illustrating sensor and target frames

(b) Visualization of the nonlinear function estimated using the STGP

Fig. 7.2 **Extended Target Model**

non-rigid asymmetric extended target can shift relative to the target extent. This displacement of the CoO needs to be considered in the CoO kinematics model. In this work, it is proposed that the estimator does not model the CoO kinematics. Instead, the kinematics of a reference point and the extent states relative to this reference point is modelled. This point lies anywhere inside the target boundary and is called the internal reference point (IRP). The CoO kinematic parameters are determined from the IRP and the extent estimates.

7.3.2 Sensor and Target Reference Frames

The extended target tracking problem is modelled in two frames, the sensor (global), and the target (local) frames. The sensor measurements are reported in the sensor (polar or Cartesian) frame. The kinematics of the CoO parameters and the IRP states are modelled in the Cartesian sensor frame whereas the extent states and their kinematics are modelled in the target (polar) frame as shown in Fig. 7.2a. The extent states are radial values of the target extent at an angle from the IRP that is $r^{obj,p} = f(\theta^{obj,p})$, where $(\cdot)^{obj,p}$ denotes the variable is in the polar target frame, $r^{obj,p}$ represents the radial extent and $\theta^{obj,p}$ represents the angle from the IRP. This is shown in Fig. 7.2b. The frames and coordinates superscripts are omitted from hereon for brevity.

7.3.3 Dynamic Model

The IRP dynamics are modelled using point target motion models given in Section 3.4 [63, 51]. The extent dynamics are designed as separable kernels, which satisfy the conditions **ST1** to **ST4**, as given below:

$$k^E(\theta, \theta'; t, t') = k^E_\theta(\theta, \theta')k^E_t(t, t'), \tag{7.13}$$

where θ and θ' represent either the training or test points of the input spatial domain, t and t' denote either the training or test points of the input temporal domain, $k^E(\cdot)$ represents the spatio-temporal covariance kernel, $k^E_\theta(\cdot)$ represents the spatial and $k^E_t(\cdot)$ represents the temporal covariance kernel. A periodic (2.15) [39] or Von-Mises (5.12) [1] covariance kernel can be used to model $k^E_\theta(\cdot)$. The temporal kernel $k^E_t(\cdot)$ can be modelled in several ways, e.g. squared exponential (2.10) or Whittle-Matèrn (2.13), which shows the generality of the proposed method. The proposed model is converted to a transfer function form and subsequently to an equivalent state space representation using steps given in Section 7.2.1. The dynamics of the IRP and the extent states are assumed independent of each other. The dynamical models are given below:

$$\boldsymbol{x}_k^I = \boldsymbol{f}_k^I(\boldsymbol{x}_{k-1}^I) + \boldsymbol{w}_k^I, \quad \boldsymbol{w}_k^I \sim \mathcal{N}(0, \boldsymbol{Q}_k^I), \tag{7.14}$$

$$\boldsymbol{x}_k^E = \boldsymbol{F}_k^E \boldsymbol{x}_{k-1}^E + \boldsymbol{w}_k^E, \quad \boldsymbol{w}_k^E \sim \mathcal{N}(0, \boldsymbol{Q}_k^E), \tag{7.15}$$

where $(\cdot)^I$ and $(\cdot)^E$ denote the vector or matrix corresponds, respectively, to the IRP and the extent, $\boldsymbol{f}^I$ represents the nonlinear IRP dynamics function, $\boldsymbol{F}^E$ represents the linear extent dynamics function as derived in (7.8), $\boldsymbol{w}$ and $\boldsymbol{Q}$ denote the corresponding process noise vector and process noise covariance matrix, respectively. The IRP state transition models the target motion and determines the maximum velocity or acceleration limits. For slowly moving targets, simple motion models can be used. However, for fast manoeuvring targets, complex motion models [63],[51] can be adopted. The resulting state vector at time k is given below:

$$\boldsymbol{x}_k = \begin{bmatrix} (\boldsymbol{x}_k^I)^T & (\boldsymbol{x}_k^E)^T \end{bmatrix}^T, \tag{7.16}$$

where $\boldsymbol{x}_k \in \mathbb{R}^{n^x}$ represents the overall state vector, $\boldsymbol{x}_k^I = \begin{bmatrix} (\boldsymbol{p}_k)^T, (\boldsymbol{p}_k')^T \end{bmatrix}^T \in \mathbb{R}^{n^I}$ denotes the IRP kinematic states, $\boldsymbol{x}_k^E = \begin{bmatrix} (\boldsymbol{r}_k)^T, (\boldsymbol{r}_k')^T \end{bmatrix}^T \in \mathbb{R}^{n^E}$ represents the extent dynamics states, $\boldsymbol{p}_k$ and $\boldsymbol{p}_k'$ denote, respectively, the position and its higher order time derivatives, $\boldsymbol{r}_k$ and $\boldsymbol{r}_k'$ represent, respectively, the radial extent and its higher order time derivatives. The spatial input of the STGP model is denoted as $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_B]^T$ which consists of B keypoints in the angle domain between 0 and 2π, as shown in Fig. 7.2b.

7.3.4 Measurement Model

The total number of measurements received from the target contour at time k are L_k. The coordinates of the sensor measurements can be either polar $\tilde{\boldsymbol{z}}_k^{sen,p}$ or Cartesian $\tilde{\boldsymbol{z}}_k^{sen,c}$. The polar measurement vector is represented as $\tilde{\boldsymbol{z}}_k^{sen,p} = \begin{bmatrix} \tilde{\boldsymbol{z}}_{1,k}^{sen,p}, \cdots, \tilde{\boldsymbol{z}}_{L_k,k}^{sen,p} \end{bmatrix}^T$. Each measurement is modelled as an i.i.d. Gaussian $\tilde{\boldsymbol{z}}_{i,k}^{sen,p} \sim \mathcal{N}(\boldsymbol{\mu}_{i,k}^{sen,p}, \boldsymbol{R}_{i,k}^{sen,p})$. The coordinate converted measurement vector in sensor (Cartesian) frame is represented by $\tilde{\boldsymbol{z}}_k^{sen,c} = \begin{bmatrix} \tilde{\boldsymbol{z}}_{1,k}^{sen,c}, \cdots, \tilde{\boldsymbol{z}}_{L_k,k}^{sen,c} \end{bmatrix}^T$ and the corresponding pdf of the i^{th} measurement at time k is approximated to a correlated Gaussian $\tilde{\boldsymbol{z}}_{i,k}^{sen,c} \sim \mathcal{N}(\boldsymbol{\mu}_{i,k}^{sen,c}, \boldsymbol{R}_{i,k}^{sen,c})$. For the Cartesian sensor measurements case, this approximation is not required. After translating $\tilde{z}_k^{sen,c}$ to the IRP and converting the coordinates to polar, the measurement vector $\tilde{z}_k^{obj,p} = \left[\tilde{z}_{1,k}^{obj,p}, \cdots, \tilde{z}_{L_k,k}^{obj,p}\right]^T$ is obtained. The corresponding pdf of the i^{th} measurement at time k is approximated to a Gaussian $\tilde{z}_{i,k}^{obj,p} \sim \mathcal{N}(\boldsymbol{\mu}_{i,k}^{obj,p}, \boldsymbol{R}_{i,k}^{obj,p})$. The relationship among $\tilde{z}_k^{sen,p}$, $\tilde{z}_k^{sen,c}$ and $\tilde{z}_k^{obj,p}$ is explained in Fig. 7.2a and given in Appendix A. The resulting measurement model is given below:

$$z_k^{sen,c} = \boldsymbol{h}(\boldsymbol{x}_k^C, \boldsymbol{x}_k^E, \tilde{z}_k^{obj,p}, \boldsymbol{v}_k), \tag{7.17}$$

where $\boldsymbol{h}(\cdot)$ is a generic measurement function (linear or nonlinear).

7.3.5 Derivation of the Measurement Likelihood Function

The measurement likelihood is derived in this subsection assuming contour measurements. For the surface measurements case, the model derived in this section and a GP convolution particle filter given in Chapter 5 [1] can be used. Alternatively, Kalman filter based approach, given in this work, can be adopted using a modified spatial covariance kernel as proposed in [26].

Likelihood function of a single measurement

The likelihood function is derived for the i^{th} measurement. Refer to Fig. 7.2a and consider the following vectors:

$$x_i^{obj,c} = \bar{x}_i^{sen,c} - x^{sen,c}, \tag{7.18}$$

$$y^{sen,c} = \bar{y}_i^{sen,c} - y^{sen,c}, \tag{7.19}$$

where $a_1 - a_2$ represents the vector difference of a_2 from a_1, $(\bar{x}_i^{sen,c}, \bar{y}_i^{sen,c})$ represents the coordinates of the i^{th} measurement and $(x^{sen,c}, y^{sen,c})$ represents the coordinates of the IRP. Assuming a noise free environment and using vector algebra the measurement vectors are related to the IRP as given below:

$$\bar{x}_i^{sen,c} = x^{sen,c} + x_i^{obj,c} = x^{sen,c} + \bar{r}_i^{obj,p}\cos(\bar{\theta}_i^{obj,p}), \tag{7.20}$$

$$\bar{y}_i^{sen,c} = y^{sen,c} + y_i^{obj,c} = y^{sen,c} + \bar{r}_i^{obj,p}\sin(\bar{\theta}_i^{obj,p}), \tag{7.21}$$

$$z_i^{sen,c} = \boldsymbol{p} + \mu_i^{GP}\bar{\boldsymbol{\zeta}}_i, \tag{7.22}$$

where $(\bar{r}_i^{obj,p}, \bar{\theta}_i^{obj,p})$ represent the i^{th} measurement predicted coordinates, $\boldsymbol{z}_i^{sen,c}$ represents the i^{th} sensor measurement vector, $\boldsymbol{p} = [\bar{x}_i^{sen,c}, \bar{y}_i^{sen,c}]^T$ represents the coordinates of the IRP, $\mu_i^{GP} = \bar{r}_i^{obj,p}$ represents the mean of the STGP model at the i^{th} measurement angle and $\bar{\boldsymbol{\zeta}}_i = [\cos(\bar{\theta}_i^{obj,p}), \sin(\bar{\theta}_i^{obj,p})]^T$ represents the i^{th} measurement transformation vector mean. $\bar{r}_i^{obj,p}$ is determined using the STGP model prediction and has an associated error represented by the STGP covariance matrix. $\bar{\theta}_i^{obj,p}$ is calculated using coordinates transform between the sensor and the target frames (Appendix A) and has an associated uncertainty for the noisy measurement case, represented by the pdf $p(\tilde{r}, \tilde{\theta})$. The sensor and the target frames and coordinates superscripts are omitted from the right hand side of the measurement equation and the time step subscript is added from here on for clarity. The measurement equation with the noise terms is given below:

$$\boldsymbol{z}_{i,k}^{sen,c} = \boldsymbol{p}_k + (\bar{\boldsymbol{\zeta}}_{i,k} + \boldsymbol{e}_{\zeta_{i,k}})(\mu_{i,k}^{GP} + e_i^{GP}) + \boldsymbol{e}_{i,k}, \tag{7.23}$$

where $e_i^{GP} \sim \mathcal{N}(0, c_{i,k}^{GP})$ represents the error in the GP prediction on the i^{th} noisy input angle, $c_{i,k}^{GP}$ represents the corresponding error variance, $\boldsymbol{e}_{i,k} \sim \mathcal{N}(0, \boldsymbol{R}_{i,k}^{sen,c})$ represents the i^{th} measurement noise vector and $\boldsymbol{R}_{i,k}^{sen,c}$ represents the corresponding sensor error covariance matrix. The pdf of the i^{th} transformed vector $\tilde{\boldsymbol{\zeta}}_{i,k} = \left[\cos(\tilde{\theta}_{i,k}), \sin(\tilde{\theta}_{i,k})\right]^T$ (Appendix B) is approximated to a Gaussian $\tilde{\boldsymbol{\zeta}}_{i,k} \sim \mathcal{N}(\bar{\boldsymbol{\zeta}}_{i,k}, \boldsymbol{R}_{i,k}^{\tilde{\zeta}})$ where

$$\bar{\boldsymbol{\zeta}}_{i,k} = \begin{bmatrix} \mu_{\mathcal{C}_{i,k}} \\ \mu_{\mathcal{S}_{i,k}} \end{bmatrix}, \boldsymbol{R}_{i,k}^{\tilde{\zeta}} = \begin{bmatrix} \sigma^2_{\mathcal{C}_{i,k}} & \sigma^2_{\mathcal{C}_{i,k}\mathcal{S}_{i,k}} \\ \sigma^2_{\mathcal{S}_{i,k}\mathcal{C}_{i,k}} & \sigma^2_{\mathcal{S}_{i,k}} \end{bmatrix}. \tag{7.24}$$

The i^{th} measurement equation can be written as:

$$\begin{aligned} \boldsymbol{z}_{i,k}^{sen,c} &= \boldsymbol{p}_k + \bar{\boldsymbol{\zeta}}_{i,k}\mu_{i,k}^{GP} + \boldsymbol{e}_{\zeta_{i,k}}\mu_{i,k}^{GP} + \bar{\boldsymbol{\zeta}}_{i,k}e_i^{GP} + \boldsymbol{e}_{\zeta_{i,k}}e_i^{GP} + \boldsymbol{e}_{i,k} \\ &= \boldsymbol{p}_k + \bar{\boldsymbol{\zeta}}_{i,k}\mu_{i,k}^{GP} + \boldsymbol{e}_{i,k}^1 + \boldsymbol{e}_{i,k}^2 + \boldsymbol{e}_{i,k}^3 + \boldsymbol{e}_{i,k} = \boldsymbol{p}_k + \bar{\boldsymbol{\zeta}}_{i,k}\mu_{i,k}^{GP} + \boldsymbol{v}_{i,k}, \end{aligned} \tag{7.25}$$

where $\boldsymbol{v}_{i,k}$ represents the cumulative measurement error vector consisting of four error vector components $\boldsymbol{e}_{i,k}$, $\boldsymbol{e}_{i,k}^1$, $\boldsymbol{e}_{i,k}^2$ and $\boldsymbol{e}_{i,k}^3$. The components of the noise term $\boldsymbol{v}_{i,k}$ are derived below

$$\boldsymbol{e}_{i,k}^1 \sim \mathcal{N}(0, (\mu_{i,k}^{GP})^2 \boldsymbol{R}_{i,k}^{\tilde{\zeta}}) = \mathcal{N}(0, \boldsymbol{R}_{i,k}^{\bar{G}\tilde{\zeta}}), \tag{7.26}$$

$$\boldsymbol{e}_{i,k}^2 \sim \mathcal{N}(0, \bar{\boldsymbol{\zeta}}_{i,k} c_{i,k}^{GP} \bar{\boldsymbol{\zeta}}_{i,k}^T) = \mathcal{N}(0, \boldsymbol{R}_{i,k}^{\bar{\zeta}\tilde{G}}), \tag{7.27}$$

$$\boldsymbol{e}^{3}_{i,k} \sim \mathcal{N}(0, \boldsymbol{R}^{\tilde{\zeta}\tilde{G}}_{i,k}), \tag{7.28}$$

$$\boldsymbol{R}^{\tilde{\zeta}\tilde{G}}_{i,k} = \text{diag}\left(\frac{\sigma^2_{\mathcal{C}_{i,k}} c^{GP}_{i,k}}{2\pi(\sigma^2_{\mathcal{C}_{i,k}} + c^{GP}_{i,k})^2}, \frac{\sigma^2_{\mathcal{S}_{i,k}} c^{GP}_{i,k}}{2\pi(\sigma^2_{\mathcal{S}_{i,k}} + c^{GP}_{i,k})^2}\right), \tag{7.29}$$

where $\boldsymbol{R}^{\bar{G}\tilde{\zeta}}_{i,k}$,$\boldsymbol{R}^{\bar{\zeta}\tilde{G}}_{i,k}$ and $\boldsymbol{R}^{\tilde{\zeta}\tilde{G}}_{i,k}$ (see Appendix C for derivation of $\boldsymbol{R}^{\tilde{\zeta}\tilde{G}}_{i,k}$) represent the noise covariance matrices corresponding to the error terms $\boldsymbol{e}^{1}_{i,k}$, $\boldsymbol{e}^{2}_{i,k}$ and $\boldsymbol{e}^{3}_{i,k}$, respectively, and diag$(\cdot)$ represents a diagonal matrix. The sum of independent Gaussian random variables is a Gaussian given below:

$$\boldsymbol{v}_{i,k} \sim \mathcal{N}(0, \boldsymbol{\Lambda}_{i,k}), \quad \boldsymbol{\Lambda}_{i,k} = \boldsymbol{R}^{\bar{G}\tilde{\zeta}}_{i,k} + \boldsymbol{R}^{\bar{\zeta}\tilde{G}}_{i,k} + \boldsymbol{R}^{\tilde{\zeta}\tilde{G}}_{i,k} + \boldsymbol{R}^{sen,c}_{i,k}, \tag{7.30}$$

where $\boldsymbol{\Lambda}_{i,k}$ represents the i^{th} measurement noise covariance matrix. The likelihood function is given below:

$$p(\tilde{\boldsymbol{z}}^{sen,c}_{i,k}|\boldsymbol{x}_k) = \mathcal{N}(\Upsilon_{i,k}, \boldsymbol{\Lambda}_{i,k}), \tag{7.31}$$

$$\Upsilon_{i,k} = \tilde{\boldsymbol{z}}^{sen,c}_{i,k} - (\boldsymbol{p}_k + \bar{\boldsymbol{\zeta}}_{i,k}\mu^{GP}_{i,k}). \tag{7.32}$$

Likelihood Function for Multiple Measurements

In this section, the multiple measurement likelihood is given for L_k measurements using the single measurement likelihood. The measurement equation is given below:

$$\tilde{\boldsymbol{z}}^{sen,c}_{k} = \boldsymbol{H}(\tilde{\boldsymbol{z}}^{obj,p}_{k})\boldsymbol{x}_k + \boldsymbol{v}_k, \tag{7.33}$$

where $\boldsymbol{H}(\tilde{\boldsymbol{z}}^{obj,p}_{k})$ represents the measurement function and is given below:

$$\boldsymbol{H}(\tilde{\boldsymbol{z}}^{obj,p}_{k}) = \boldsymbol{H}_1(\tilde{\boldsymbol{z}}^{obj,p}_{k})\boldsymbol{C}_1(\tilde{\boldsymbol{z}}^{obj,p}_{k})\boldsymbol{C}_2, \tag{7.34}$$

where $\boldsymbol{H}_1, \boldsymbol{C}_1$ and $\boldsymbol{C}_2$ represent the sub-functions of $\boldsymbol{H}$. The matrix multiplication of $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ with the state vector $\boldsymbol{x}_k$ gives a matrix consisting of the IRP states and the prediction of the target's extent at the angles defined by L_k measurements with respect to the IRP. Subsequent multiplication with $\boldsymbol{H}_1$ performs the coordinate frame conversion of the predicted measurements from polar local to Cartesian local and further to Cartesian global. These matrices and the measurement noise are:

$$\boldsymbol{H}_1(\tilde{\boldsymbol{z}}_k^{obj,p}) = \begin{bmatrix} \boldsymbol{H}_2 & \bar{\boldsymbol{\zeta}}_{1,k} & \underline{\boldsymbol{o}}_2 & \cdots & \underline{\boldsymbol{o}}_2 \\ \boldsymbol{H}_2 & \underline{\boldsymbol{o}}_2 & \bar{\boldsymbol{\zeta}}_{2,k} & \cdots & \underline{\boldsymbol{o}}_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{H}_2 & \underline{\boldsymbol{o}}_2 & \underline{\boldsymbol{o}}_2 & \cdots & \bar{\boldsymbol{\zeta}}_{L_k,k} \end{bmatrix}, \tag{7.35}$$

$$\boldsymbol{H}_2 = \begin{bmatrix} 1 & 0 & \bar{\boldsymbol{o}}_{(n^I-2)} \\ 0 & 1 & \bar{\boldsymbol{o}}_{(n^I-2)} \end{bmatrix}, \tag{7.36}$$

$$\boldsymbol{C}_1(\tilde{\boldsymbol{z}}_k^{obj,p}) = \begin{bmatrix} \boldsymbol{I}_{n^I} & \boldsymbol{O}_{n^I \times n^E} \\ \boldsymbol{O}_{L_k \times n^I} & \boldsymbol{C}_{\bar{\boldsymbol{\theta}}_k \boldsymbol{\theta}} + \frac{\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\theta}}_k} \boldsymbol{C}''_{\bar{\boldsymbol{\theta}}_k \boldsymbol{\theta}}}{2} \end{bmatrix}, \tag{7.37}$$

$$\bar{\boldsymbol{\theta}}_k = \begin{bmatrix} \bar{\theta}_{1,k} & \bar{\theta}_{2,k} & \cdots & \bar{\theta}_{L_k,k} \end{bmatrix}^T, \tag{7.38}$$

$$\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\theta}}_k} = \operatorname{diag}(\sigma^2_{\tilde{\theta}_{1,k}}, \sigma^2_{\tilde{\theta}_{2,k}}, \cdots, \sigma^2_{\tilde{\theta}_{L_k,k}}), \tag{7.39}$$

$$\boldsymbol{C}_2 = \begin{bmatrix} \boldsymbol{I}_{n^I} & \boldsymbol{O}_{n^I \times n^E} \\ \boldsymbol{O}_{n^E \times n^I} & \boldsymbol{C}_{\boldsymbol{\theta\theta}}^{-1} \end{bmatrix}, \boldsymbol{C}_{\boldsymbol{\theta\theta}} = \boldsymbol{K}_{\boldsymbol{\theta\theta}} \otimes \boldsymbol{I}_{\frac{n^E}{B}}, \tag{7.40}$$

$$\boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{\Lambda}_k = \boldsymbol{\Omega}_k^{\bar{G}\tilde{\zeta}} + \boldsymbol{\Omega}_k^{\bar{\zeta}\tilde{G}} + \boldsymbol{\Omega}_k^{\tilde{\zeta}\tilde{G}} + \boldsymbol{\Omega}_k^{\tilde{\boldsymbol{z}}^G}), \tag{7.41}$$

$$\boldsymbol{\Omega}_k^{\bar{G}\tilde{\zeta}} = \operatorname{blkdiag}\left(\boldsymbol{R}_{1,k}^{\bar{G}\tilde{\zeta}} \quad \boldsymbol{R}_{2,k}^{\bar{G}\tilde{\zeta}} \quad \cdots \quad \boldsymbol{R}_{L_k,k}^{\bar{G}\tilde{\zeta}} \right), \tag{7.42}$$

$$\boldsymbol{\Omega}_k^{\bar{\zeta}\tilde{G}} = \operatorname{blkdiag}\left(\boldsymbol{R}_{1,k}^{\bar{\zeta}\tilde{G}} \quad \boldsymbol{R}_{2,k}^{\bar{\zeta}\tilde{G}} \quad \cdots \quad \boldsymbol{R}_{L_k,k}^{\bar{\zeta}\tilde{G}} \right), \tag{7.43}$$

$$\boldsymbol{\Omega}_k^{\tilde{\zeta}\tilde{G}} = \operatorname{blkdiag}\left(\boldsymbol{R}_{1,k}^{\tilde{\zeta}\tilde{G}} \quad \boldsymbol{R}_{2,k}^{\tilde{\zeta}\tilde{G}} \quad \cdots \quad \boldsymbol{R}_{L_k,k}^{\tilde{\zeta}\tilde{G}} \right), \tag{7.44}$$

$$\boldsymbol{\Omega}_k^{sen,c} = \operatorname{blkdiag}\left(\boldsymbol{R}_{1,k}^{sen,c} \quad \boldsymbol{R}_{2,k}^{sen,c} \quad \cdots \quad \boldsymbol{R}_{L_k,k}^{sen,c} \right), \tag{7.45}$$

where $\boldsymbol{O}_{m \times n}$ represents an m by n zero matrix, $\bar{\boldsymbol{o}}_m$ represents an m-dimensional zero row vector, $\underline{\boldsymbol{o}}_m$ represents an m-dimensional zero column vector and blkdiag$[\cdot]$ represents a block diagonal matrix. The measurements are assumed independent which gives the structure of $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\theta}}_k}$,$\boldsymbol{\Omega}_k^{\bar{G}\tilde{\zeta}}$, $\boldsymbol{\Omega}_k^{\bar{\zeta}\tilde{G}}$, $\boldsymbol{\Omega}_k^{\tilde{\zeta}\tilde{G}}$ and $\boldsymbol{\Omega}_k^{sen,c}$ as block diagonal. The GP expression appearing in $\boldsymbol{C}_1(\boldsymbol{z}_k^{obj,p})$ and $\boldsymbol{C}_2$ are derived in Section 7.3.6. The multiple measurements likelihood function is given below:

$$p(\tilde{\boldsymbol{z}}_k^{sen,c} | \boldsymbol{x}_k) = \mathcal{N}(\Upsilon_k, \boldsymbol{\Lambda}_k), \tag{7.46}$$

$$\Upsilon_k = \tilde{\boldsymbol{z}}_k^{sen,c} - \left([\boldsymbol{p}_k]_{\times L_k} + \bar{\boldsymbol{\zeta}}_k \odot \left(\boldsymbol{\mu}_k^{GP} \otimes \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right) \right), \tag{7.47}$$

where $\bar{\boldsymbol{\zeta}}_k = [\bar{\zeta}_{1,k}, \cdots, \bar{\zeta}_{L_k,k}]^T$ and $\boldsymbol{\mu}_k^{GP} = [\mu_{1,k}^{GP}, \cdots, \mu_{L_k,k}^{GP}]^T$.

7.3.6 GP Prediction at Noisy Input Locations

The input locations $\bar{\boldsymbol{\theta}}_k$ in (7.38) are corrupted by the sensor noise. This gives a non-Gaussian posterior, which is approximated to a Gaussian. The GP prediction given in (2.18)–(2.19) is valid for noise-free inputs. The GP prediction for noisy training input locations and non-noisy predicted locations is derived in [39]. In (7.33), the GP prediction is required at noisy locations using data of non-noisy input locations. This has been derived in [99, 100] for different covariance kernels. Exact first and second moments of the posterior are derived for linear or Gaussian covariance kernels. For remaining covariance kernels (like the spatial covariance kernel), using a Taylor series expansion, the approximate moments are derived. For a given input with distribution $\tilde{\boldsymbol{\theta}}_k \sim \mathcal{N}(\bar{\boldsymbol{\theta}}_k, \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\theta}}_k})$, the predictive mean and covariance are given below:

$$\begin{aligned}
\boldsymbol{\mu}_k^{GP} &= \boldsymbol{\mu}(\tilde{\boldsymbol{\theta}}_k) + \frac{1}{2}\sum_{i=1}^{B} \beta_i Tr[\boldsymbol{C}''_{\bar{\boldsymbol{\theta}}_k \boldsymbol{\theta}_i} \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\theta}}_k}],\\
\boldsymbol{C}_k^{GP} &= \boldsymbol{C}(\tilde{\boldsymbol{\theta}}_k) + Tr\Big[\Big(\frac{1}{2}\boldsymbol{C}''(\tilde{\boldsymbol{\theta}}_k) + \boldsymbol{\mu}'(\tilde{\boldsymbol{\theta}}_k)\boldsymbol{\mu}'(\tilde{\boldsymbol{\theta}}_k)^T\Big)\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\theta}}_k}\Big],
\end{aligned}$$

where $\boldsymbol{\mu}_k^{GP}$ and $\boldsymbol{C}_k^{GP}$ represent, respectively, the mean and covariance of the GP prediction at the noisy input angle measurements, $\boldsymbol{\mu}(\tilde{\boldsymbol{\theta}}_k)$ and $\boldsymbol{C}(\tilde{\boldsymbol{\theta}}_k)$ represent the noise-free GP prediction mean and covariance, respectively, and $Tr[\cdot]$ is the trace function. The terms on the right side of the summation in both equations can be seen as the correction of the noise free GP mean and covariance values. These are explained in the following equations:

$$\boldsymbol{\beta} = C_{\boldsymbol{\theta}\boldsymbol{\theta}}^{-1}\boldsymbol{x}_k^E, \quad \boldsymbol{\mu}'(\tilde{\boldsymbol{\theta}}_k) = \boldsymbol{C}'_{\bar{\boldsymbol{\theta}}_k\boldsymbol{\theta}}\boldsymbol{C}_{\boldsymbol{\theta}\boldsymbol{\theta}}^{-1}\boldsymbol{x}_k^E, \tag{7.48}$$

$$\boldsymbol{C}''(\tilde{\boldsymbol{\theta}}_k) = \boldsymbol{C}''_{\bar{\boldsymbol{\theta}}_k\bar{\boldsymbol{\theta}}_k} - \boldsymbol{C}''_{\bar{\boldsymbol{\theta}}_k\boldsymbol{\theta}}\boldsymbol{C}_{\boldsymbol{\theta}\boldsymbol{\theta}}^{-1}\boldsymbol{C}''_{\boldsymbol{\theta}\bar{\boldsymbol{\theta}}_k}, \tag{7.49}$$

$$\boldsymbol{C}''_{\bar{\boldsymbol{\theta}}_k\boldsymbol{\theta}} = \boldsymbol{\Delta}^2_{\bar{\boldsymbol{\theta}}_k\bar{\boldsymbol{\theta}}_k}(\boldsymbol{K}_{\bar{\boldsymbol{\theta}}_k\boldsymbol{\theta}}) \otimes \begin{bmatrix} 1 & \bar{\boldsymbol{o}}_{\left(\frac{n^E}{B}-1\right)} \end{bmatrix}, \tag{7.50}$$

$$\boldsymbol{C}''_{\bar{\boldsymbol{\theta}}_k\bar{\boldsymbol{\theta}}_k} = \boldsymbol{\Delta}^2_{\bar{\boldsymbol{\theta}}_k\bar{\boldsymbol{\theta}}_k}(\boldsymbol{K}_{\bar{\boldsymbol{\theta}}_k\bar{\boldsymbol{\theta}}_k}) \otimes \begin{bmatrix} 1 & \bar{\boldsymbol{o}}_{\left(\frac{n^E}{B}-1\right)} \end{bmatrix}, \tag{7.51}$$

$$\boldsymbol{C}'_{\bar{\boldsymbol{\theta}}_k\boldsymbol{\theta}} = \boldsymbol{\Delta}_{\bar{\boldsymbol{\theta}}_k}(\boldsymbol{K}_{\bar{\boldsymbol{\theta}}_k\boldsymbol{\theta}}) \otimes \begin{bmatrix} 1 & \bar{\boldsymbol{o}}_{\left(\frac{n^E}{B}-1\right)} \end{bmatrix}, \tag{7.52}$$

$$\boldsymbol{\Delta}_{\boldsymbol{\theta}}(\boldsymbol{K}_{\boldsymbol{\theta}\boldsymbol{\theta}'}) = \begin{bmatrix} \frac{\partial k(\theta,\theta'_1)}{\partial \theta}\Big|_{\theta=\theta_1} & \cdots & \frac{\partial k(\theta,\theta'_{N'})}{\partial \theta}\Big|_{\theta=\theta_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial k(\theta,\theta'_1)}{\partial \theta}\Big|_{\theta=\theta_N} & \cdots & \frac{\partial k(\theta,\theta'_{N'})}{\partial \theta}\Big|_{\theta=\theta_N} \end{bmatrix}, \tag{7.53}$$

$$\boldsymbol{\Delta}^2_{\boldsymbol{\theta}\boldsymbol{\theta}'}(\boldsymbol{K}_{\boldsymbol{\theta}\boldsymbol{\theta}'}) = \begin{bmatrix} \frac{\partial^2 k(\theta,\theta')}{\partial \theta \partial \theta'}\Big|_{\substack{\theta=\theta_1 \\ \theta'=\theta'_1}} & \cdots & \frac{\partial^2 k(\theta,\theta')}{\partial \theta \partial \theta'}\Big|_{\substack{\theta=\theta_1 \\ \theta'=\theta'_{N'}}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 k(\theta,\theta')}{\partial \theta \partial \theta'}\Big|_{\substack{\theta=\theta_N \\ \theta'=\theta'_1}} & \cdots & \frac{\partial^2 k(\theta,\theta')}{\partial^2 \theta \partial \theta'}\Big|_{\substack{\theta=\theta_N \\ \theta'=\theta'_{N'}}} \end{bmatrix}, \tag{7.54}$$

where $\boldsymbol{\Delta}_{\boldsymbol{\theta}}(\cdot)$ and $\boldsymbol{\Delta}^2_{\boldsymbol{\theta}\boldsymbol{\theta}'}$ represent the first and second differential of the corresponding noise free GP covariance matrices for an N-dimensional and N'-dimensional input vectors $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, respectively.

7.3.7 CoO Parameter Estimates

The parameters of the CoO kinematics are the position and the higher order time derivatives of the position. These parameters are calculated from the estimated shape (polygon) in the sensor frame at each time step. Consider $\{(x_1^{\hat{P}}, y_1^{\hat{P}}), \cdots, (x_B^{\hat{P}}, y_B^{\hat{P}})\}$ represents the coordinates of the estimated polygon. The positional [101] (x_k^C, y_k^C) and velocity $(\dot{x}_k^C, \dot{y}_k^C)$ parameters are determined as given below:

$$A = \frac{1}{2}\sum_{i=1}^{B}(x_i^{\hat{P}} y_{i+1}^{\hat{P}} - x_{i+1}^{\hat{P}} y_i^{\hat{P}}), \tag{7.55}$$

$$x_k^C = \frac{1}{6A}\sum_{i=1}^{B}(x_i^{\hat{P}} + x_{i+1}^{\hat{P}})(x_i^{\hat{P}} y_{i+1}^{\hat{P}} - x_{i+1}^{\hat{P}} y_i^{\hat{P}}), \tag{7.56}$$

$$y_k^C = \frac{1}{6A}\sum_{i=1}^{B}(y_i^{\hat{P}} + y_{i+1}^{\hat{P}})(x_i^{\hat{P}} y_{i+1}^{\hat{P}} - x_{i+1}^{\hat{P}} y_i^{\hat{P}}), \tag{7.57}$$

$$\dot{x}_k^C = \frac{x_k^C - x_{k-1}^C}{T}, \quad \dot{y}_k^C = \frac{y_k^C - y_{k-1}^C}{T}, \tag{7.58}$$

where A represents the area of the polygon.

7.3.8 Real-time Inference

The inference can be done using an STGP batch regression. As most of the ETT applications require real-time processing, the state estimation is done recursively. A real-time recursive filter equivalent to a full GP regression has also been proposed in [26, 27]. The mathematical equivalence of a full GP regression is a smoother rather than a filter [97]. Given a model of the form (7.14), (7.15) and (7.33) a recursive (nonlinear) Kalman filtering and smoothing solution is developed, to filter and smooth the states at each time step. In high nonlinearity scenarios, advance nonlinear filtering and smoothing methods such as sequential Monte Carlo, Markov chain Monte Carlo are preferred [102].

The processing time of the smoother increases with time and the computation becomes non-real time as more measurement samples are reported. This is due to the increase in the data size with time. A fixed lag RTSS is proposed for real-time smoothing by restricting the size of the data according to the available computational resources. It is further proposed to set the lag of the RTSS equal to as long as the states are correlated in time. In short, the quasi-realtime inference is achieved using a fixed lag RTSS with a lag value set equal to l_t.

7.4 Extended Target Tracking using Whittle-Matèrn Temporal Covariance

This section demonstrates the proposed method using an example approach.

7.4.1 Extent Evolution Model

Spatial Covariance Kernel

The extent spatial correlation is modelled using a periodic spatial covariance kernel (2.15) [39]. The kernel is illustrated in Fig. 7.3 and is given below:

$$k_\theta^E(\theta, \theta') = \sigma_m^2 \exp\left(-\frac{2\sin^2\left(\frac{\theta-\theta'}{p}\right)}{l_\theta^2}\right) + \sigma_c^2, \tag{7.59}$$

where $\sigma_m^2, \sigma_c^2, p$ and l_θ are hyperparameters. σ_m^2 controls the correlation magnitude, σ_c^2 is the prior radial variance, p is periodicity and l_θ controls the correlation length-scale. This kernel is generic and can be used for various real-world extended targets.

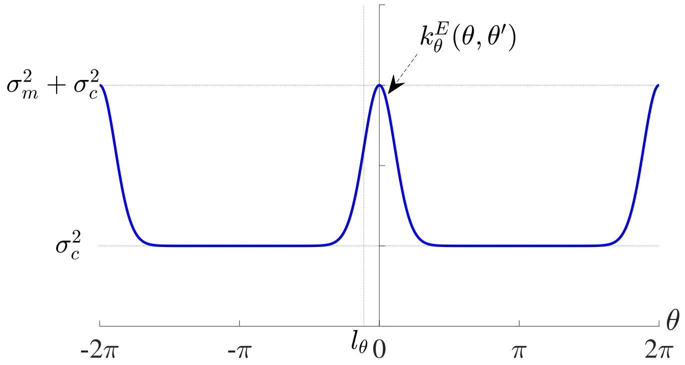


Fig. 7.3 **Spatial (Periodic) covariance kernel**.

Given $\vartheta = \cos\left(\frac{2\varpi}{p}\right)$ and $\varpi = \bar{\theta} - \theta'$, the derivatives of the covariance kernel are given below:

$$\frac{d}{d\bar{\theta}}\left[k_\theta(\bar{\theta},\theta')\right] = -\frac{\sigma_m^2 \exp(\frac{\vartheta-1}{4l_\theta^2}) \sin\left(\frac{2\varpi}{p}\right)}{2l_\theta^2 p}, \tag{7.60}$$

$$\frac{d}{d\bar{\theta}}\left[k_\theta^{'}(\bar{\theta},\theta')\right] = -\frac{\sigma_m^2 \exp\,(\frac{\vartheta-1}{4l_\theta^2})(4l_\theta^2\vartheta + \vartheta^2) - 1)}{4l_\theta^4 p^2}, \tag{7.61}$$

$$\frac{\partial}{\partial\bar{\theta}\partial\theta'}\left[k_\theta(\bar{\theta},\theta')\right] = \frac{\sigma_m^2}{l_\theta^8} \exp\left(\frac{\cos(\varpi) - 1.0}{l_\theta^2}\right)\Big(\frac{-\cos(2\varpi)}{2} + \frac{\cos(4\varpi)}{8} \tag{7.62}$$

$$- \frac{3l_\theta^2\cos(\varpi)}{2} + l_\theta^6\cos(\varpi) + \frac{3l_\theta^2\cos(3\varpi)}{2} + \frac{7l_\theta^4\cos(2\varpi)}{2} - \frac{l_\theta^4}{2} + \frac{3}{8}\Big). \tag{7.63}$$

Temporal Covariance Kernel

A Whittle-Matèrn temporal covariance kernel (2.13) [103, 104] is chosen to model the temporal correlation of the extent and is given below:

$$k_t^E(t,t') = \sigma_t^2\frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}}{l_t}(t'-t)\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}}{l_t}(t'-t)\right). \tag{7.64}$$

The corresponding spectral density $S(\omega_t)$ is given below:

$$S(\omega_t) = \sigma_t^2\frac{2\pi^{\frac{1}{2}}\Gamma(\nu+\frac{1}{2})}{\Gamma(\nu)}\lambda^{2\nu}(\lambda^2+\omega_t^2)^{-(\nu+\frac{1}{2})}, \tag{7.65}$$

where ω_t represents frequency and $\lambda = \frac{\sqrt{2\nu}}{l_t}$. As the spectral density is a function of ω_t^2, a stable transfer function $G(\iota\omega_t) = (\lambda + \iota\omega)^{-(p+1)}$ can be obtained after spectral factorization of the given kernel where $p = \nu - \frac{1}{2}$ and:

$$q_t = \frac{2\sigma_t^2 \pi^{\frac{1}{2}} \lambda^{(2p+1)} \Gamma(p+1)}{\Gamma(p+\frac{1}{2})}, \tag{7.66}$$

where q_t is the spectral density of the white noise process driving the temporal evolution of the states.

Remark 1 *Choosing the order of the Whittle-Matern covariance function, $\nu = \frac{1}{2}$, yields the Ornstein-Uhlenbeck function [105]. This in turn has the same state-space representation as the model used in GP-EKF [26]. Hence, the GP-EKF can be seen as a special case of the method proposed herein.*

Extent State Space Model

The system matrix and the noise effect vector of the corresponding state space model for $\nu = \frac{5}{2}$ are derived in [106] and given below:

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\lambda^3 & -3\lambda^2 & -3\lambda \end{pmatrix}, \quad \boldsymbol{L} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}. \tag{7.67}$$

Using above a multidimensional discrete time state space model for B keypoints is derived and given below:

$$\boldsymbol{x}_k^E = \boldsymbol{F}^E \boldsymbol{x}_{k-1}^E + \boldsymbol{w}_k^E, \tag{7.68}$$

$$\boldsymbol{F}^E = \boldsymbol{I}_B \otimes e^{\boldsymbol{A}T}, \quad \boldsymbol{w}_k^E \sim \mathcal{N}(0, \boldsymbol{Q}^E(\boldsymbol{\theta}, \boldsymbol{\theta}'; T)), \tag{7.69}$$

$$\boldsymbol{Q}^E(\boldsymbol{\theta}, \boldsymbol{\theta}'; T) = \boldsymbol{C}_{\boldsymbol{\theta}\boldsymbol{\theta}}[\boldsymbol{I}_B \otimes \tilde{\boldsymbol{Q}}(T)], \tag{7.70}$$

$$\tilde{\boldsymbol{Q}}(T) = \int_0^T \boldsymbol{F}^E(T-\tau) \boldsymbol{L} q_t \boldsymbol{L}^T \boldsymbol{F}^E(T-\tau)^T d\tau. \tag{7.71}$$

The elements of $\tilde{\boldsymbol{Q}}(T)$ are derived in Appendix D.

7.4.2 IRP Kinematics Model

The IRP kinematics are modelled using a nearly constant velocity (NCV) (see Section 3.4.1) [63] motion model as given below:

$$\boldsymbol{x}_k^I = \boldsymbol{F}^I \boldsymbol{x}_{k-1}^I + \boldsymbol{w}_k^I, \tag{7.72}$$

$$\boldsymbol{F}^I = \mathrm{diag}(\boldsymbol{F}^{NCV}, \boldsymbol{F}^{NCV}), \quad \boldsymbol{w}_k^I \sim \mathcal{N}(0, \boldsymbol{Q}^I), \tag{7.73}$$

$$\boldsymbol{Q}^I = \mathrm{diag}(q_x^{NCV} \boldsymbol{Q}^{NCV}, q_y^{NCV} \boldsymbol{Q}^{NCV}), \tag{7.74}$$

where q_x and q_y represent the process noise variances in x and y coordinates respectively.

7.4.3 State Vector

The corresponding state vectors are given below:

$$\boldsymbol{x}_k^I = \begin{bmatrix} x_k & \dot{x}_k & y_k & \dot{y}_k \end{bmatrix}^T, \tag{7.75}$$

$$\boldsymbol{x}_k^E = \begin{bmatrix} r_k^1 & \dot{r}_k^1 & \ddot{r}_k^1 & \cdots & r_k^B & \dot{r}_k^B & \ddot{r}_k^B \end{bmatrix}^T, \tag{7.76}$$

where the location of the IRP is represented by x_k, y_k and the velocity of the IRP is represented by $\dot{x}_k, \dot{y}_k$. The extent states consist of B radial values from the IRP and its first and second time derivatives. A block diagram of the proposed method is given in Fig. 7.4.

7.5 Performance Validation Methodology

The performance of the proposed method is validated by tracking of targets in simulated and real data using the example approach. The estimates of the example approach are compared with the GP-EKF estimates [26] over 100 Monte Carlo runs for the simulated experiments. The performance evaluation parameters are the positional and velocity root mean square errors (RMSE) of the CoO, the mean shape precision P_μ and the mean shape recall R_μ. These are defined below:

$$RMSE_a = \sqrt{\frac{1}{K}\sum_{j=1}^{K}\frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}}(a_j^i - \hat{a}_j^i)^2}, \tag{7.77}$$

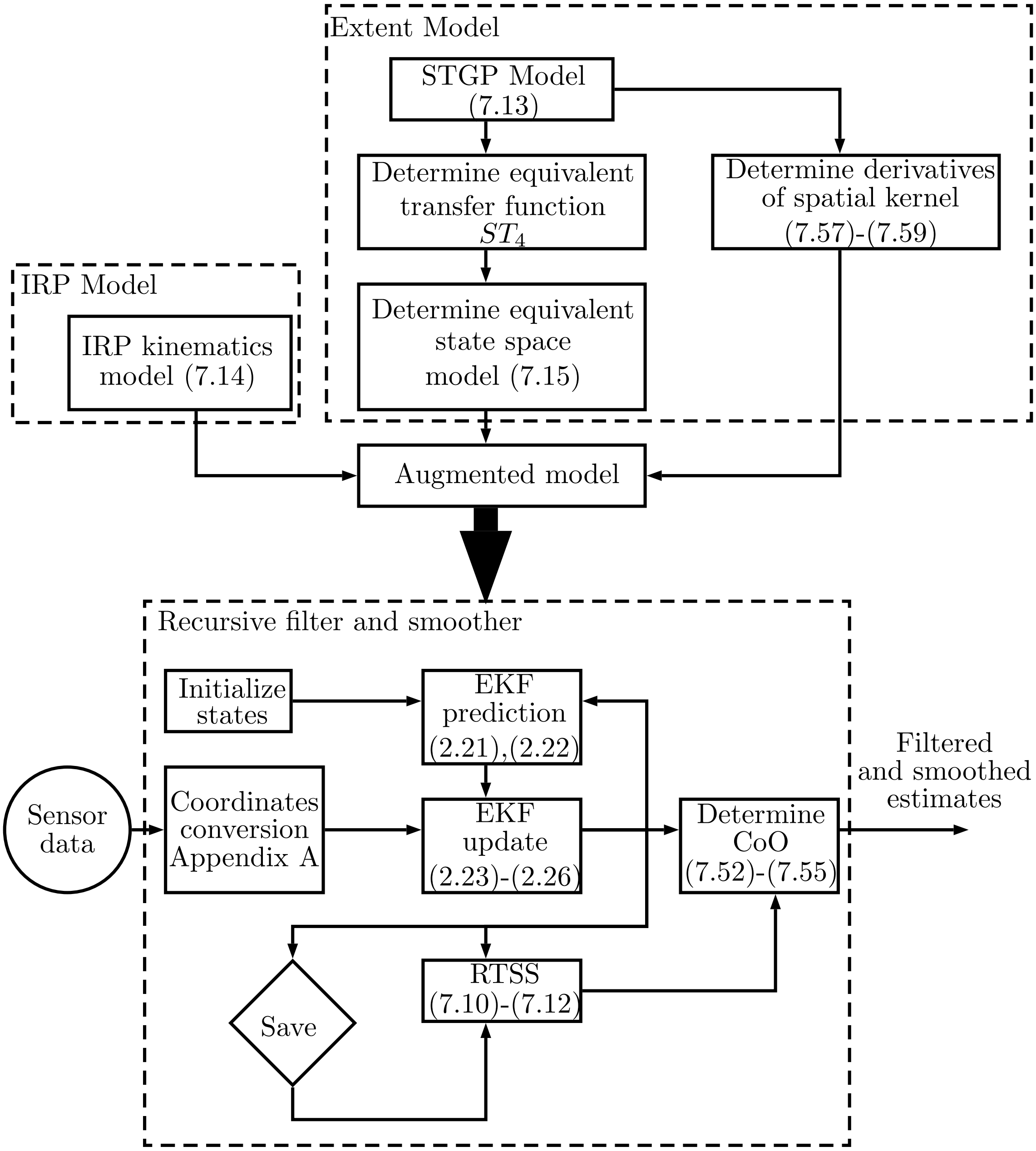


Fig. 7.4 **Proposed method**. The figure shows the proposed method. The top half (tail of bold inverted arrow) of the figure shows the modelling part whereas the bottom half shows the recursive filtering and smoothing solution. The diamond shape represents a memory storage, required by the RTSS.

$$R_{\mu} = \frac{1}{K}\sum_{j=1}^{K}\frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}}\frac{Area(\mathcal{T}_j^i \cap \mathcal{E}_j^i)}{Area(\mathcal{T}_j^i)}, \tag{7.78}$$

$$P_{\mu} = \frac{1}{K}\sum_{j=1}^{K}\frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}}\frac{Area(\mathcal{T}_j^i \cap \mathcal{E}_j^i)}{Area(\mathcal{E}_j^i)}, \tag{7.79}$$

where $RMSE_a$ represents the RMSE of the parameter a, a_j^i represents the true and $\hat{a}_j^i$ represents the estimated value, $\mathcal{T}_j^i$ represents the true shape, $\mathcal{E}_j^i$ represents the estimated shape, $\cap$ represents the intersection of two star-convex polygons and $Area(\mathcal{P})$ represents the area of the polygon $\mathcal{P}$. The recall specifies how much of the true shape has been recalled while the precision evaluates the false (not belonging to true target) area. These parameters have been used to evaluate estimators in computer vision for rectangular targets estimation problems [93]. The percentage improvement compared to GP-EKF is also given in the results section. If $RMSE_a$, R_μ or P_μ of the GP-EKF is represented by vector $\boldsymbol{b}$ and those of STGP-EKF and STGP-RTSS by $\boldsymbol{c}$, then the corresponding percentage improvement $\boldsymbol{d}$ and the mean percentage improvement $\boldsymbol{d}_\mu$ are given below. A positive value indicates the example STGP approach performs better and vice-versa:

$$\boldsymbol{d} = \frac{\boldsymbol{b}-\boldsymbol{c}}{\boldsymbol{b}}, \quad \boldsymbol{d}_{\mu} = \frac{\boldsymbol{d}}{K} \times 100. \tag{7.80}$$

7.6 Evaluation on Simulated Scenarios

The IRP motion model of the simulated target and the estimators is NCV with matched process noise variance $q_x^{NCV} = q_y^{NCV} = 1$. Five different shape evolutions are simulated using two shape models active at different time samples for $K = 250$ time samples. These are the Singer acceleration model [64] and a constant shape model. The shape of the target does not change and the time derivatives of the radial states are zero when the constant shape model is active. The Singer model is active for the time samples in the range $k = [(1-50), (80-130), (180-230)]$. The parameters of the Singer model are manoeuvre variance $\sigma_a^2 = 12\ m^2/s^4$ and manoeuvre time constant $\tau_m = 1\ s$. The shape model and the parameters for simulation are different from the model in the filter and smoother. The switching and mismatched shape models further validate the robustness of the proposed method. The simulated shape models are explained in Table 7.1. The number of keypoints is $B = 24$, the sample time is $T = \frac{1}{30}\ s$, the spatial length-scale is $l_\theta = 15°$, the prior radial variance is $\sigma_c^2 = 1\ m^2$, the spatial correlation magnitude variance is $\sigma_m^2 = 1\ m^2$, the periodicity is $p = 2$, the temporal length-scale is $l_t = 2\ s$, the temporal correlation magnitude is $\sigma_t^2 = 1\ m^2$. The GP-EKF forgetting factor is tuned to $\alpha_{drw} = 0.001$. The sensor error standard deviations are $\sigma_r = 0.25\ m$ for the range and $\sigma_\theta = 0.25°$ for the angle. The number of measurements is Poisson distributed with mean set equal to 20. The measurements are located randomly over the contour of the target using a uniform distribution.

Table 7.1 Non-rigid shape models

Non-rigid object model	Description
S1 Triangle	A triangular object randomly increases and decreases in size.
S2 Quadrilateral	A quadrilateral object randomly increases and decreases in size.
S3 Regular Dodecagon	An almost circular object (regular 12-gon) randomly increases and decreases in size.
S4 Axis-symmetric Dodecagon	A 12-gon adapting random axis-symmetric shapes.
S5 Asymmetric Dodecagon	A dodecagon adapting random asymmetric shapes.

7.6.1 Results on Simulated Data

The RMSE values and the percentage improvement from 100 Monte Carlo runs for the five scenarios are given in Tables 7.2 and 7.3. The tables show that the performance of the STGP-EKF and STGP-RTSS is improved in all five cases. Fig. 7.5 shows the tracking snapshots of a single simulation run of the five scenarios at selected time steps. The true CoO (red plus) and the shape (red solid line) along with the corresponding estimates are also presented in the figure. It can be observed that except for the S3 the shape estimates of the example approach are improved as compared to the GP-EKF [26]. It can again be observed that the GP-EKF shape estimates are less accurate as compared to both the STGP-EKF and the STGP-RTSS estimates except for S3 (simplest shape model), where they are comparable.

Table 7.2 Simulated data RMSE of STGP-EKF and STGP-RTSS

States	**Methods**	**Shape Models**				
		S1	**S2**	**S3**	**S4**	**S5**
x (m)	STGP-EKF	0.23	0.12	0.08	0.14	0.12
	STGP-RTSS	0.11	0.06	0.04	0.07	0.07
y (m)	STGP-EKF	0.34	0.14	0.09	0.16	0.13
	STGP-RTSS	0.17	0.06	0.04	0.09	0.07
$\dot{x}$ (m/s)	STGP-EKF	1.11	0.84	0.75	0.91	0.90
	STGP-RTSS	0.35	0.29	0.25	0.29	0.35
$\dot{y}$ (m/s)	STGP-EKF	1.07	0.82	0.74	0.89	0.90
	STGP-RTSS	0.35	0.29	0.25	0.30	0.34
P	STGP-EKF	0.98	0.99	1.00	0.99	0.99
	STGP-RTSS	0.97	0.99	0.99	0.99	0.99
R	STGP-EKF	0.96	0.98	0.99	0.98	0.98
	STGP-RTSS	0.98	0.99	0.99	0.99	0.99

Table 7.3 Mean percentage improvement of STGP-EKF and STGP-RTSS (Simulated data)

States	**Methods compared**	**Shape Models**				
		S1	**S2**	**S3**	**S4**	**S5**
x	STGP-EKF	85.81	86.87	79.85	83.49	85.76
	STGP-RTSS	92.09	93.47	89.66	90.89	91.47
y	STGP-EKF	77.91	84.74	73.82	83.90	84.17
	STGP-RTSS	87.55	92.47	87.15	91.50	90.86
$\dot{x}$	STGP-EKF	91.98	90.23	61.90	88.40	87.96
	STGP-RTSS	97.08	96.99	87.53	96.42	95.53
$\dot{y}$	STGP-EKF	91.79	90.43	62.69	88.43	87.85
	STGP-RTSS	96.60	96.90	87.40	96.50	95.51
P	STGP-EKF	33.27	14.71	2.01	9.33	8.51
	STGP-RTSS	32.55	14.12	1.70	8.90	8.13
R	STGP-EKF	18.02	11.21	0.69	7.20	6.59
	STGP-RTSS	19.62	12.28	1.52	8.20	7.56

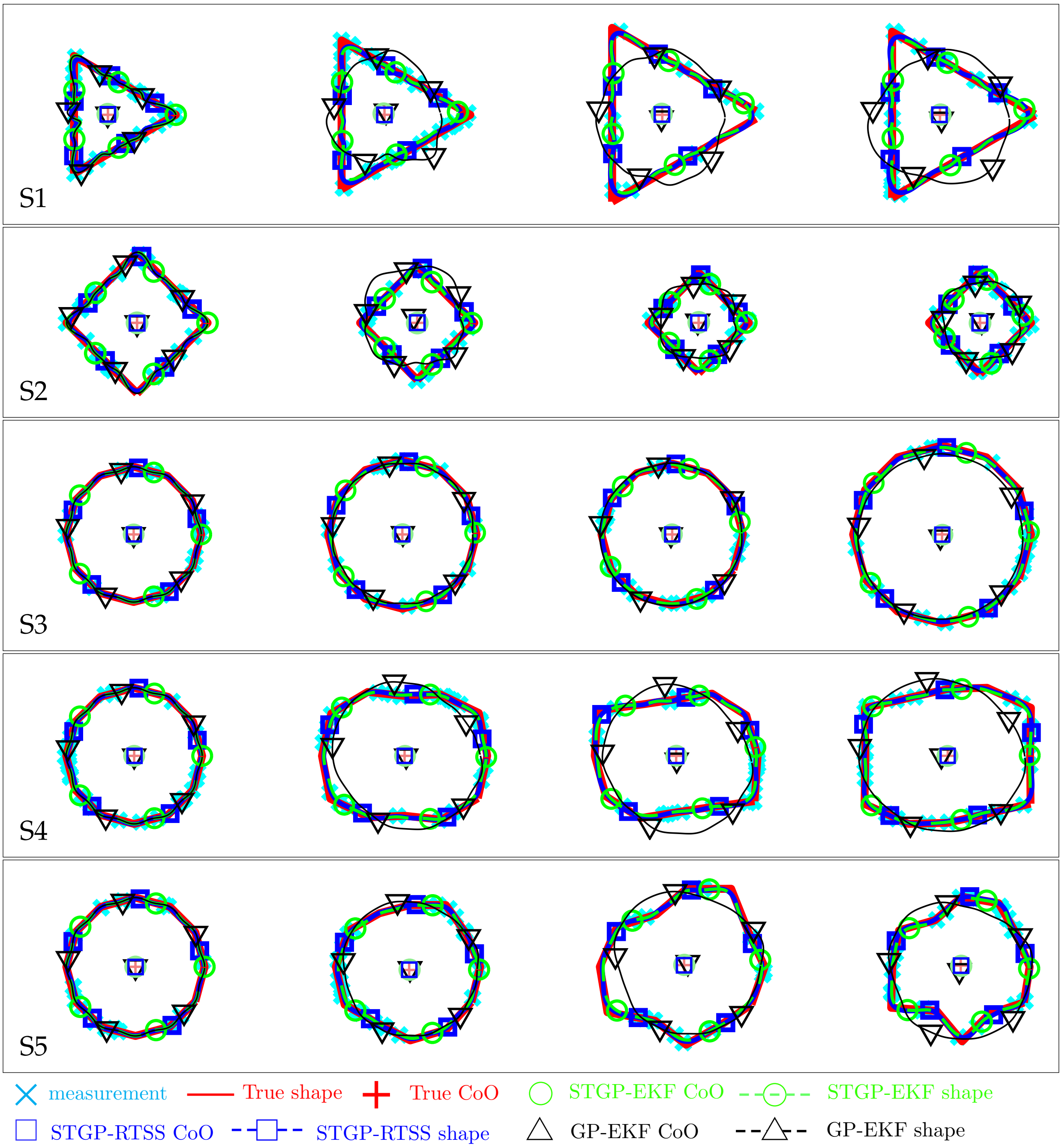


Fig. 7.5 **Tracking snapshots of simulated data**. The figure shows snapshots at at $k = 1, 50, 150, 230$ of the five different shape evolutions.

7.6.2 Effect of the STGP-RTSS Lag Value

The performance of the fixed-lag smoother is evaluated using the shape model S5. The performance is evaluated at different lag values for 100 Monte Carlo runs. The smoother lag k_s is chosen less than, equal to and more than the true temporal correlation length-scale l_t. The results are given in Fig. 7.6.

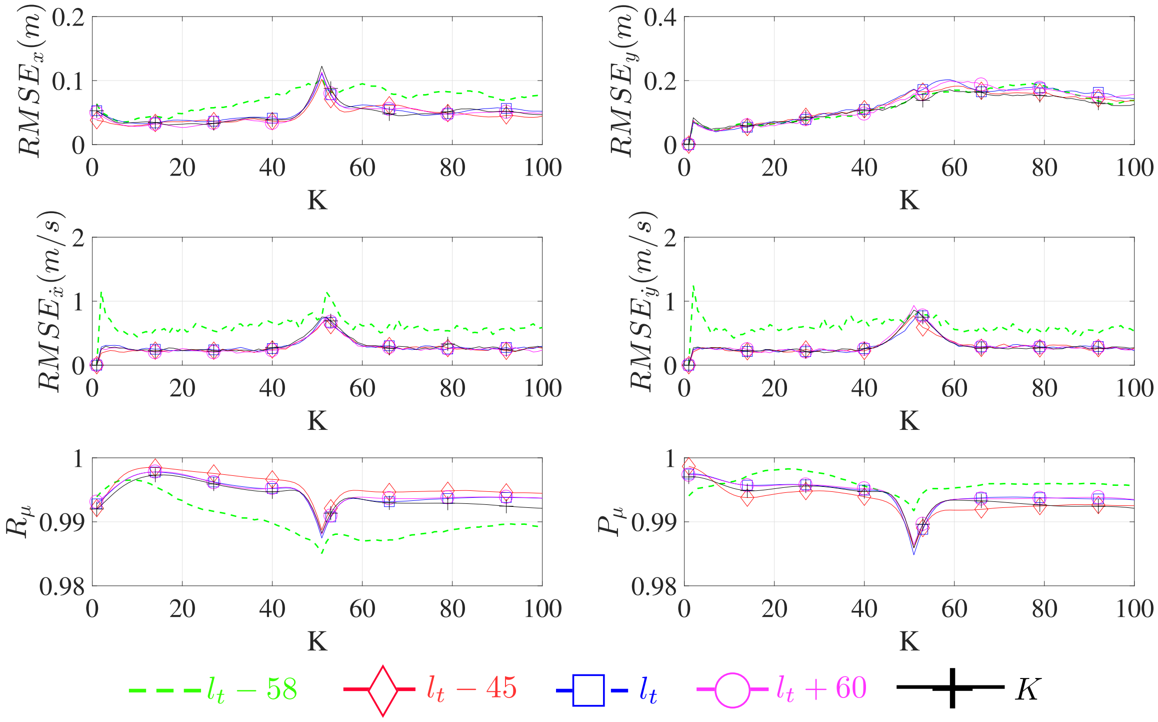


Fig. 7.6 **STGP-RTSS performance at different lag values**. The figure shows the comparison of smoother performance at different lag values compared to the true temporal correlation l_t.

It can be observed that the performance of the smoother is degraded for $k_s < l_t$. However, the smoother performance is comparable for the cases $k_s >= l_t$. The best performance is given by the smoother with lag $k_s = K$ that is a full STGP regression. Keeping in mind the computational advantage gained by keeping the lag smaller, as proposed, the $k_s = l_t$ is a reasonable trade-off value for the smoother lag. The peaks in the graphs are observed at time samples when the shape model switches between the Singer and the constant model.

7.6.3 Computational Complexity

The computational complexity of the STGP-EKF and STGP-RTSS scale as $\mathcal{O}(L_k^3 B + L_k^2 B^3)$ and $\mathcal{O}(k_s B^3)$, respectively. The empirical results with respect to the number of keypoints B are shown in Fig. 7.7. The extent state vector of the STGP model is three times the size of the GP-EKF vector. Hence, the STGP-EKF and the STGP-EKF-RTSS require more processing time. As B increases, the processing time of the STGP-EKF-RTSS rises at a much faster rate compared to the filters due to the RTSS recursion. The empirical results with respect to the number of measurements L_k are shown in Fig. 7.8. The computational cost of the STGP-EKF increases at a higher rate as compared to the GP-EKF. The STGP-EKF-RTSS computational cost is not dependent on the L_k and hence the plot follows a similar slope to STGP-EKF with a vertical shift equal to the time required for RTSS recursion. The empirical results with respect to the smoother lag k_s are shown in Fig. 7.9. The computational complexity of the filter is independent of k_s while that of RTSS increases exponentially with the increase in the lag.

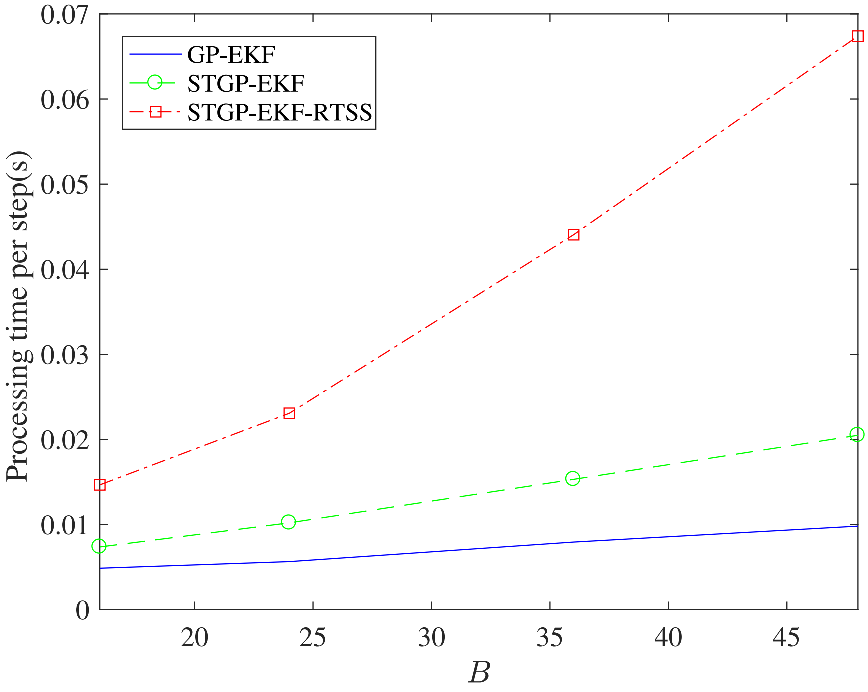


Fig. 7.7 **Effect of B on the processing time**. The figure shows a comparison of time taken per time sample by increasing B.

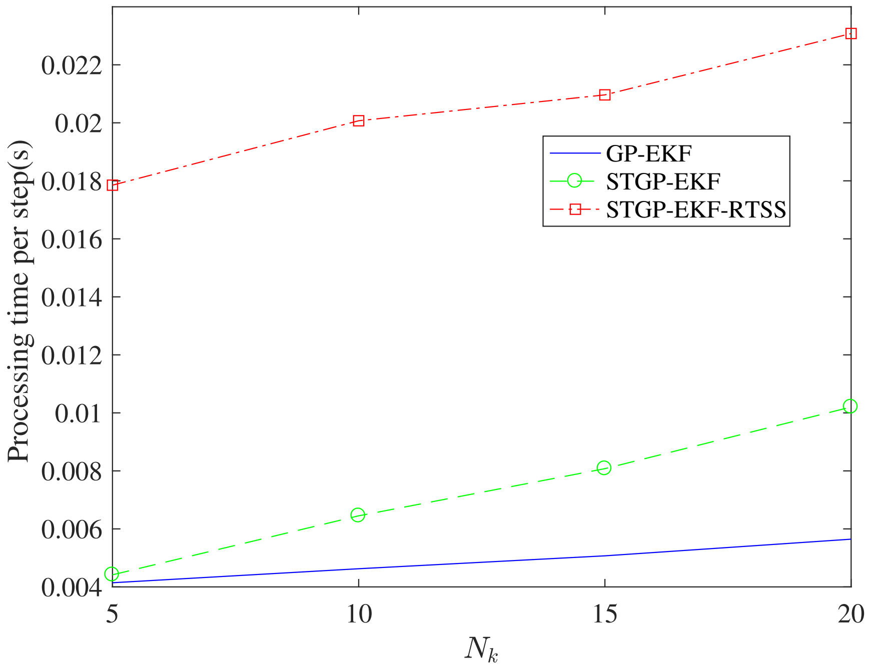


Fig. 7.8 **Effect of the** L_k **on the processing time**. The figure shows a comparison of time taken per time sample by increasing L_k.

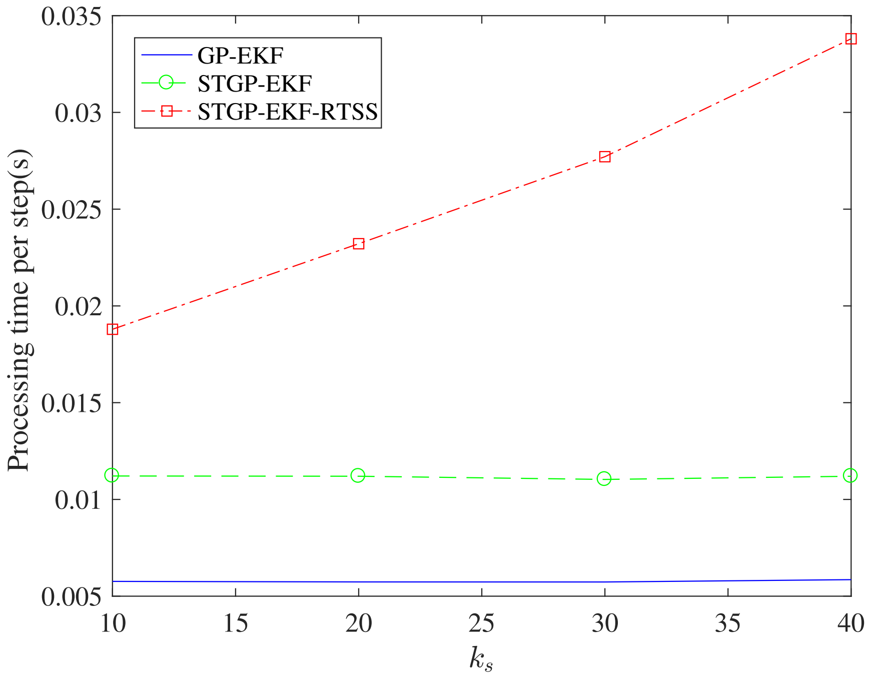


Fig. 7.9 **Effect of the** k_s **on the processing time**. The figure shows a comparison of time taken per time sample by increasing k_s.

The program was run on MATLAB R2016b on a Windows 10 (64 bit) Desktop computer installed with an Intel(R) Core(TM) i5-6500 CPU @ 3.20 GHz (4 CPUs) and 8 GB RAM. The number of keypoints B and the smoother lag k_s are the model parameters and can be managed during the design phase. The number of keypoints, B, can be decreased in the model according to the available computational resources. At the end of each time-step, the target shape can be constructed as per the requirement using the standard GP prediction (2.18) and (2.19). If the target shape is constructed at B_0 angles, the increase in computational expense due to this operation is $B_0 B$. Similarly, k_s can be reduced according to the available computational resources. The third variable, L_k, is dependent on the sensor, the target and other environmental conditions. The processing time can be further reduced through faster code implementation in C++.

7.7 Evaluation on Real Data Experiments

In this section, the example approach of the proposed method is evaluated on real data presented in [107]. This is a thermal video data of pedestrians and vehicles sampled at 10Hz obtained using a fixed camera in an open environment. Three different video samples are chosen for evaluation which are a motorcycle, a rickshaw and a pedestrian. The rickshaw appears as a regular rigid target, the motorcycle as an irregular rigid target and the pedestrian as an irregular non-rigid target. The ground truth data is not available and is manually generated by marking the target contour (as precisely as possible) in each frame, calculating the CoO location in each frame and the CoO kinematics are determined using the CoO locations of consecutive frames. The video frames are pre-processed using frame differencing and median filtering to generate contour measurements. The following parameters are changed for the real data experiments; $B = 48$, $l_t = 1\ s$, $\sigma_m^2 = 30\ pixel^2$ and $q_x^{NCV} = q_y^{NCV} = 50^2\ pixel^2$ for the STGP-EKF / STGP-RTSS and $B = 48$, $\sigma_m^2 = 2\ pixel^2$ and $q_x^{NCV} = q_y^{NCV} = 10^2\ pixel^2$ for the GP-EKF.

7.7.1 Results on Real Data

The RMSE and the percentage improvement of all three scenarios are given in Tables 7.4 and 7.5.

Table 7.4 Real data RMSE of STGP-EKF and STGP-RTSS

States	**Methods**	**Scenarios**		
		Rickshaw	**Motorcycle**	**Pedestrian**
x	STGP-EKF	1.68	1.74	1.63
(p)	STGP-RTSS	1.11	1.05	1.47
y	STGP-EKF	2.31	1.50	3.95
(p)	STGP-RTSS	0.84	1.08	4.00
$\dot{x}$	STGP-EKF	9.13	11.43	6.31
(p/s)	STGP-RTSS	7.79	7.87	5.51
$\dot{y}$	STGP-EKF	8.35	10.43	14.87
(p/s)	STGP-RTSS	6.08	8.19	13.41
P	STGP-EKF	0.99	0.96	0.83
	STGP-RTSS	0.97	0.93	0.76
R	STGP-EKF	0.81	0.81	0.84
	STGP-RTSS	0.89	0.89	0.91

Table 7.5 Mean percentage improvement of STGP-EKF and STGP-RTSS (Real data)

States	**Methods compared**	**Scenarios**		
		Rickshaw	**Motorcycle**	**Pedestrian**
x	STGP-EKF	30.37	89.18	89.56
	STGP-RTSS	54.19	93.45	90.58
y	STGP-EKF	-22.16	88.29	43.96
	STGP-RTSS	55.82	91.53	43.32
$\dot{x}$	STGP-EKF	27.11	82.52	91.11
	STGP-RTSS	37.78	87.96	92.23
$\dot{y}$	STGP-EKF	37.84	83.04	64.81
	STGP-RTSS	54.75	86.69	68.26
P	STGP-EKF	7.58	75.81	135.06
	STGP-RTSS	5.95	71.67	115.54
R	STGP-EKF	-8.94	16.23	0.20
	STGP-RTSS	0.60	27.90	9.23

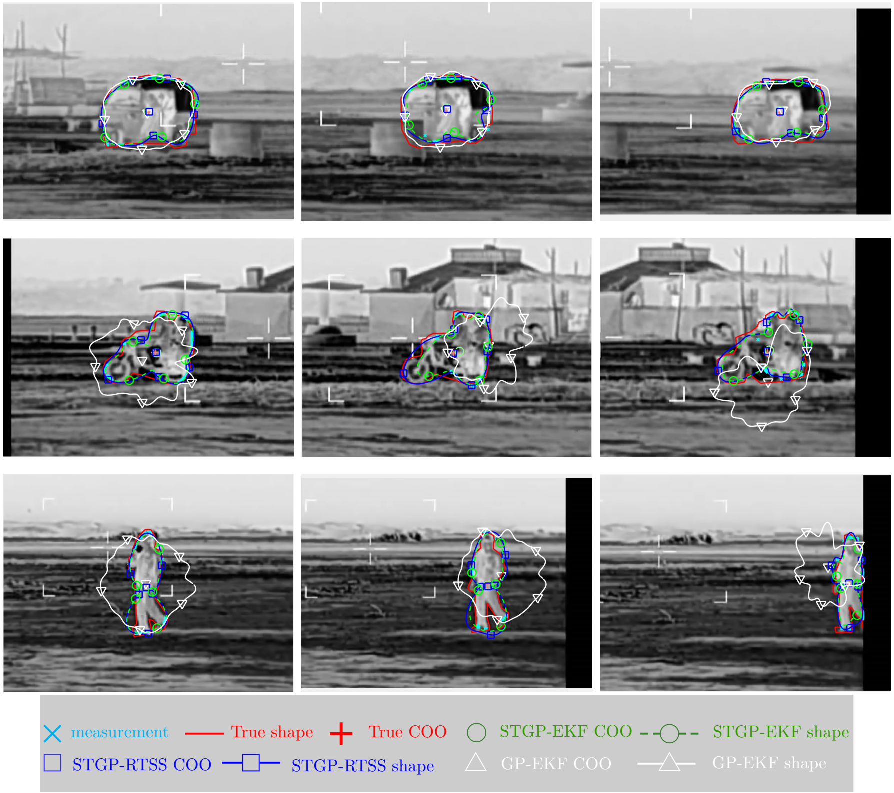


Fig. 7.10 **Tracking snapshots of real data**.

It can be observed that the performance of the example STGP approach is comparable to the reference (GP-EKF) approach while tracking a regularly shaped rigid target (rickshaw). As observed in the simulated experiments, there is a significant improvement in performance while tracking irregularly shaped targets, especially when the target shape is also changing (pedestrian). The snapshots at three different samples are given in Fig. 7.10. The figure shows snapshots at selected time samples of the three scenarios, that is rickshaw (top), motorcyclist (middle) and pedestrian (bottom). The ground truth and the estimates from the STGP-EKF, STGP-RTSS and GP-EKF are also shown. It can be observed that the shape estimates of the STGP based models are improved as compared to the GP-EKF. The shape estimates (especially the precision) are significantly improved for non-rigid targets i.e. the pedestrian.

7.8 Conclusions

This chapter proposes a generic method and presents a step by step derivation of a tracking approach for a non-rigid extended/group target using spatio-temporal

Gaussian process. The non-rigid target changes its shape with time. The spatial and temporal dependence of the extent is modelled using a product of covariance kernels based Gaussian process model, called the spatio-temporal Gaussian process model. The batch processing of the Gaussian process is solved using real-time filtering and smoothing approaches. The STGP model is converted to an equivalent state space spatio-temporal model. The shape model is augmented with the target kinematics model. The augmented state space model is estimated using Kalman filtering and smoothing methods. The extended or group target likelihood is derived for the new model by considering all assumed noises including the spatial GP input noise (ignored in all previous star-convex shape based approaches [26, 29]).

The proposed generic method is evaluated using example spatial (periodic) and temporal (Matérn) covariance kernels. The target kinematics are modelled using a nearly constant velocity model (Section 3.4.1). The example approach is compared to state-of-the-art extended and group target tracking approach [26]. Real and simulated data are considered for performance comparison. The real data consists of measurements created from thermal image data. The performance improvement to track a non-rigid real target (pedestrian) compared to the reference approach is up to 43% in position, 68% in velocity, 10% in the recall and 115% in the precision. The simulated data is created using a matched model for the centre of target kinematics and the spatial GP. The temporal changes in the shape are simulated using Singer model, which is different from the proposed and the referenced approach, to validate the robustness. Improved tracking efficiency is demonstrated using simulated data with more than 90% improvement in the accuracy in position, 95% in velocity and 7% in the shape for the tracking of an asymmetric non-rigid target.

For complicated nonlinear scenarios, advanced nonlinear filters and smoothers can be derived for the same model using similar steps. Being a general model, it can be applied to solve various real-world problems. The model can also be extended to 3D scenarios.

Chapter 8

A Gaussian Process Regression Approach for Point Target Tracking

Abstract

Target tracking performance relies on the match between the tracker motion model and the unknown target dynamics. The performance of these model-based trackers degrades when there is a mismatch between the model and the target motion. In this chapter, a Gaussian process based approach, namely, Gaussian process motion tracker (GPMT) is proposed. The Gaussian process framework is flexible and can represent an infinite number of motion modes. The evaluation of the proposed approach is performed on challenging scenarios and is compared with popular single and multiple-model based approaches. The results show high accuracy of the predicted and filtered target position and velocity over challenging manoeuvre scenarios.

8.1 Introduction

This chapter proposes a Gaussian process approach for tracking the kinematics of a point target. A wealth of research has been done on time series estimation using the GP [39, 106]. However, the GP approach has not been widely studied by the target tracking community for estimating the target motion, especially for highly manoeuvring targets. A GP based trajectory estimation has been proposed for simultaneous localization and mapping [108, 109], where the target kinematics are not considered as highly manoeuvrable. A novel approach for extended target tracking using GPs has been proposed (see Section 4.3.5) [26]. In this work and other similar works (in Chapters 5, 6 and 7) [1–3], the target shape estimation is proposed using a GP model whereas the target (centre/average) motion is filtered using the model based approaches proposed for PTT. An overlapping mixture of GPs (OMGP) is proposed in [37] to solve the data association problem arising in the PTT. To the best of our knowledge, all previous approaches for the estimation of the target kinematics are model based. Hence, for the first time, this chapter proposes a data-driven approach for the target kinematics estimation. A Gaussian process motion tracker (GPMT) is proposed in this chapter and it represents the target kinematics as a GP regression (see Section 2.2) [39].

A Gaussian process motion tracker (GPMT), proposed in this chapter, is a data driven approach based on the GP regression (2.18)and (2.19) [39]. The GPMT models the target motion as a nonlinear function of time using a GP prior over this unknown function. The mean of the GP represents the mean of the function matched to the target dynamics. Since the GP is a distribution over functions, an infinite number of functions or models selection can be achieved using the GPMT. The GPMT uses the available data to select the model and estimate the target state based on the chosen model. A GP, being a batch regression approach, is not suitable for real-time temporal systems. The GPMT assumes that the model selection depends on the training data in the near past, only. Hence, the batch regression problem is reduced and the real-time implementation is achieved, for the known hyperparameters case. All model-based approaches are recursive where the prediction step is necessary for filtering and vice-versa. In contrast, the GPMT does not require such a two-step process. The estimation and the prediction processes run independently of each other. Hence, when only estimates at the current time moment are required, the prediction process can be omitted.

The GPMT gives the location estimates. In most applications, the consideration of higher order time derivatives of the location is also important. This can be achieved by using the derivative of the GP [39]. A first order time derivative extension of the GPMT is presented in this chapter for estimating the target velocity. The same concept can be extended to determine the higher order derivatives.

The rest of the chapter is structured as follows. Sections 8.2.1 and 8.2.2 provide a brief overview of the state space models used in MTT and the motion models. The theoretical background of the derivative of the GP is covered in Section 8.2.3. The proposed approach is explained in Section 8.3 and the first order extension of the proposed model is given in Section 8.3.2. The performance evaluation is given in Section 8.4 followed by conclusions.

8.2 Background Knowledge

8.2.1 State Space Model for Multiple Target Tracking

This section gives a brief overview of the state space model used in MTT. The MTT algorithms deal with data association and maintain the trajectories of multiple targets by an appropriately chosen estimation approach. The assignment of measurements to respective target trajectories is key for achieving accurate results. A generic state-space model of the target dynamics and the sensor model is assumed as given in (3.48) and (3.49), respectively.

8.2.2 Overview of Target Motion Models and Filtering Methods

This section gives a brief overview of target motion models. A comprehensive survey of the motion models can be found in the three survey papers [63, 110, 51]. The models have been categorized and studied as non multiple-model (non-MM) [63], decision-based methods [110] and multiple-model methods (MM) [51]. Their practical implementation has been discussed extensively in [56].

The non-MM motion models can be categorized from simple to complex models based on the assumption made on the coordinate coupling and the temporal correlations as shown in Fig. 8.1 [63]. The '+'and 'x'represent, respectively, the presence and absence of the model assumption e.g. the left most box represents models that assume no coordinate coupling or time correlation. The model complexity increases from left to right. The simplest models assume that the target motion is uncoupled across coordinates and uncorrelated in time. The acceleration and jerk[1] are assumed to be a white noise process in nearly constant velocity (NCV) [56] and nearly constant acceleration (NCA) [111] models, respectively. The velocity and acceleration are assumed almost constant in the NCV and NCA, respectively.

[1] The time derivative of acceleration is called a jerk.

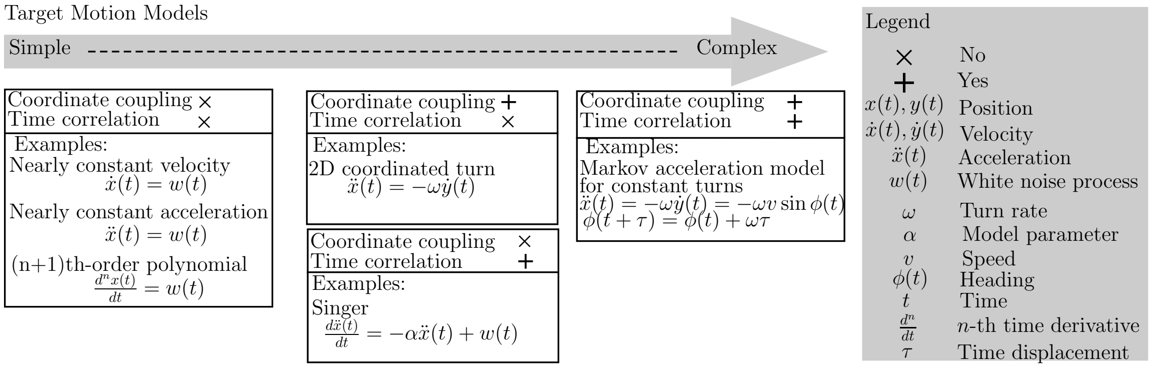


Fig. 8.1 **Classification of motion models for target tracking**. This figure shows the classification of motion models based on the assumptions of the coordinate coupling and time correlations. Example models and their respective differential equations are also given.

The $(n+1)$-th order polynomial model is achieved by assuming the n-th position derivative to be white noise [111]. The above-mentioned models assume zero coupling among the coordinates. This assumption is relaxed in the nearly coordinated turn (NCT) [111, 56] based models, which in addition to the coupling assume an almost constant forward speed and turn rate. Such models provide better estimates during target manoeuvres. The performance is degraded, in comparison to the NCV and NCA, when the target follows a linear motion. Since the true target dynamics is a continuous process, the motion parameters are correlated in time. The time correlation is considered in relatively complex models such as Singer [64] and jerk models [112]. The coordinate coupled version of the Singer model is also proposed as n-th order Markov model [113]. The target acceleration based models provide better estimates in the presence of both the position and the velocity measurements [56]. These models apply to the two dimensional (2D) tracking case directly and are extended to the three dimensional (3D) case as well. Often the Kalman filter [45] is applied when there is no measurement data association filter.

Two other groups of methods - the decision-based and the MM methods [110, 51], use a bank of filters for the state estimation. The decision-based methods choose a single filter to estimate the state at each time sample. On the other hand, the MM filters combine the filters to give the state estimates. These have been classified into three generations [51], the first generation [52, 53] methods, the second generation namely interacting MM (IMM)) [54, 55]) and the third generation referred to as the variable structure IMM (VSIMM) [57]. Considering the computational complexity and the performance, the interacting multiple-model (IMM) is shown to be the most cost-effective method. The IMM, also called Fixed Grid IMM (FGIMM), has been successfully applied in various tracking systems [56].

8.2.3 Derivative Gaussian Process

The derivative of a GP is also a GP [39]. The GP regression using derivative measurements has been proposed in [114]. In typical radar tracking applications, these derivative measurements are not available. The GP inference of a function and its derivatives using the observations is proposed in [115]. In what follows we present the new approach, using the observations only. It requires the GP covariance kernel to be as many times differentiable as the order of the desired derivative process. Next, we describe a second order derivative GP which is in the heart of the developed approach.

Consider the GP model (2.1) and (2.16). The joint probability density function of the training vector $\boldsymbol{f}(\boldsymbol{u})$, the unknown test vector $\boldsymbol{f}(\boldsymbol{u}^\star)$ and its higher order derivatives, $\boldsymbol{f}'(\boldsymbol{u}^\star)$, $\boldsymbol{f}''(\boldsymbol{u}^\star)$ and so on, is also Gaussian. The mean and the covariance matrix of the first-order derivative of Gaussian process using a squared exponential covariance kernel are given below [115]:

$$\mathbb{E}\left[\boldsymbol{f}'(\boldsymbol{u}^\star)\right] = \delta_{\boldsymbol{u}^\star}(\boldsymbol{m}(\boldsymbol{u}^\star)) + \boldsymbol{\Delta}_{\boldsymbol{u}^\star}(\boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}})\left(\boldsymbol{K}_{\boldsymbol{uu}} + \sigma^2\boldsymbol{I}_N\right)^{-1}(\boldsymbol{z} - \boldsymbol{m}(\boldsymbol{u})), \tag{8.1}$$

$$\mathbb{C}\left[\boldsymbol{f}'(\boldsymbol{u}^\star)\right] = \boldsymbol{\Delta}^2_{\boldsymbol{u}^\star\boldsymbol{u}^\star}(\boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}^\star}) - \boldsymbol{\Delta}_{\boldsymbol{u}^\star}(\boldsymbol{K}_{\boldsymbol{u}^\star\boldsymbol{u}})\left(\boldsymbol{K}_{\boldsymbol{uu}} + \sigma^2\boldsymbol{I}_N\right)^{-1}\boldsymbol{\Delta}_{\boldsymbol{u}^\star}(\boldsymbol{K}_{\boldsymbol{uu}^\star}), \tag{8.2}$$

$$\delta_{\boldsymbol{u}}(\boldsymbol{m}(\boldsymbol{u})) = \begin{bmatrix} \left.\frac{\partial m(u)}{\partial u}\right|_{u=u_1} & \cdots & \left.\frac{\partial m(u)}{\partial u}\right|_{u=u_N} \end{bmatrix}^T, \tag{8.3}$$

$$\boldsymbol{\Delta}_{\boldsymbol{u}}(\boldsymbol{K}_{\boldsymbol{uu}'}) = \begin{bmatrix} \left.\frac{\partial k(u,u'_1)}{\partial u}\right|_{u=u_1} & \cdots & \left.\frac{\partial k(u,u'_{N'})}{\partial u}\right|_{u=u_1} \\ \vdots & \ddots & \vdots \\ \left.\frac{\partial k(u,u'_1)}{\partial u}\right|_{u=u_N} & \cdots & \left.\frac{\partial k(u,u'_{N'})}{\partial u}\right|_{u=u_N} \end{bmatrix}, \tag{8.4}$$

$$\boldsymbol{\Delta}^2_{\boldsymbol{uu}'}(\boldsymbol{K}_{\boldsymbol{uu}'}) = \begin{bmatrix} \left.\frac{\partial^2 k(u,u')}{\partial u \partial u'}\right|_{\substack{u=u_1 \\ u'=u'_1}} & \cdots & \left.\frac{\partial^2 k(u,u')}{\partial u \partial u'}\right|_{\substack{u=u_1 \\ u'=u'_{N'}}} \\ \vdots & \ddots & \vdots \\ \left.\frac{\partial^2 k(u,u')}{\partial u \partial u'}\right|_{\substack{u=u_N \\ u'=u'_1}} & \cdots & \left.\frac{\partial^2 k(u,u')}{\partial^2 u \partial u'}\right|_{\substack{u=u_N \\ u'=u'_{N'}}} \end{bmatrix}, \tag{8.5}$$

where $\delta_{\cdot}(\cdot)$ denotes the derivative of the GP mean vector and $\boldsymbol{\Delta}_{\cdot}(\cdot)$, $\boldsymbol{\Delta}^2_{\cdot\cdot}(\cdot)$ represent, respectively, the first and second order derivatives of the GP covariance matrix.

8.3 Gaussian Process Motion Tracker

The GPMT relies on past measurements to filter the current and predict the future states. The model is based upon the following assumptions:

T1 Cross-coordinates coupling is weak enough to be ignored.

T2 Coordinate auto-correlation is available in time.

T3 The temporal correlation with points in the distant past is weak. These distant points are ignored while training of the GP model.

T4 A single measurement per target per sample is received.

T5 The measurement noise is an i.i.d. process.

One of the most commonly observed target manoeuvre is the horizontal turn modelled using the coordinated turn model given in Section 3.4.4. The x and y coordinates are correlated during a coordinated turn. According to the assumption **T1**, the coordinates are assumed mutually uncorrelated. The effect of ignoring this coupling in the proposed GPMT approach is not been studied in this work. This coupling can be introduced in GPMT using coupled GP [116]. The proposed GPMT approach tracks a target in 2D and can be extended to track in 3D straightforwardly.

8.3.1 Two Dimensional Gaussian Process Motion Tracker

This section presents the two dimensional GPMT. The unknown nonlinear functions of the target position are modelled in the x and y Cartesian coordinates using a GP. The model is given below:

$$z^x = f^x(t) + v^x, \quad f^x \sim GP^x(0, k^x(t,t')), \quad v^x \sim \mathcal{N}(0, \sigma_x^2), \tag{8.6}$$

$$z^y = f^y(t) + v^y, \quad f^y \sim GP^y(0, k^y(t,t')), \quad v^y \sim \mathcal{N}(0, \sigma_y^2), \tag{8.7}$$

where f^x and f^y denote, respectively, the non-linear latent functions in x and y coordinates, GP^x and GP^y represent, respectively, the modelled GP in x and y domains, t is the (input) time domain parameter, z represents the measurement and

v is the zero mean Gaussian measurement noise variable with variance σ^2. We adopt a squared exponential covariance kernel (see (2.10)) [39] for the two GPs. Other kernels can also be explored depending upon the application (see Chapter 9). From here on, the prediction and filtering for the x-coordinate is described. The y and other coordinates are filtered and predicted in a similar way. A typical radar and sonar report measurements in polar coordinates. In such scenarios, the process and measurement models, (8.6) and (8.7), are modelled in the polar coordinates. An alternate approach can be to calculate the measurement pdf in Cartesian coordinates [65, 117]. The performance may be degraded in the latter case as the cross-correlation among the x and y coordinates is ignored in (8.6) and (8.7).

The GPMT considers the d most recent measurement samples, also called depth of the tracker, for the position prediction and filtering as given below:

$$\begin{aligned}
\tilde{\mu}^x_k &= \boldsymbol{K}_{\bar{t}\boldsymbol{t}_k}[\boldsymbol{K}_{\boldsymbol{t}_k\boldsymbol{t}_k} + \sigma_x^2\boldsymbol{I}_d]^{-1}\boldsymbol{z}^x_{\boldsymbol{t}_k}, && (8.8)\\
(\tilde{\phi}^x_k)^2 &= \boldsymbol{K}_{\bar{t}\bar{t}} - \boldsymbol{K}_{\bar{t}\boldsymbol{t}_k}[\boldsymbol{K}_{\boldsymbol{t}_k\boldsymbol{t}_k} + \sigma_x^2\boldsymbol{I}_d]^{-1}\boldsymbol{K}^T_{\bar{t}\boldsymbol{t}_k}, && (8.9)\\
\hat{\mu}^x_k &= \boldsymbol{K}_{\bar{t}\boldsymbol{t}'_k}[\boldsymbol{K}_{\boldsymbol{t}'_k\boldsymbol{t}'_k} + \sigma_x^2\boldsymbol{I}_d]^{-1}\boldsymbol{z}^x_{\boldsymbol{t}'_k}, && (8.10)\\
(\hat{\phi}^x_k)^2 &= \boldsymbol{K}_{\bar{t}\bar{t}} - \boldsymbol{K}_{\bar{t}\boldsymbol{t}'_k}[\boldsymbol{K}_{\boldsymbol{t}'_k\boldsymbol{t}'_k} + \sigma_x^2\boldsymbol{I}_d]^{-1}\boldsymbol{K}^T_{\bar{t}\boldsymbol{t}'_k}, && (8.11)
\end{aligned}$$

where $\tilde{\cdot}$ and $\hat{\cdot}$ represent, respectively, the predicted and filtered variables, μ^x and $(\phi^x)^2$ are, respectively, the x-position mean and variance, $\bar{t} = k$, $\boldsymbol{t}_k = [k-d, k-d+1, \cdots, k-1]^T$, $\boldsymbol{t}'_k = [k-d+2, k-d+3, \cdots, k]^T$ and $\boldsymbol{z}_{\boldsymbol{a}}$ represents the measurement vector consisting of samples corresponding to the time vector $\boldsymbol{a}$.

The GPMT described above tracks targets that exhibit a wide set of trajectories with a fixed set of hyperparameters. A wider set of trajectories can be tracked by learning the hyperparameters at each time sample. In this work, the hyperparameter learning is proposed by maximizing the likelihood as given in Section 2.3. This approach can be computationally complex depending upon the optimization method and is not applicable in real-time applications. An online learning approach, proposed in [40], can be adopted for real-time implementation. The GPMT with online learning is proposed in Chapter 10. In this Chapter, the focus is on the performance comparison of the proposed model-free offline learning GPMT with model-based approaches.

Traditional point target kinematics methods are used for the state estimation of the extended and group targets. The GPMT proposed in this chapter is however not directly applicable to the kinematics state estimation of the extended or group targets. The input domain of the GPMT is time and according to the assumption **T4** multiple measurements are not allowed. Multiple measurements at a given time sample create duplicate training data for the GPMT. A GP does not allow duplicate measurement data as the GP covariance matrix in the GP regression (2.18) and (2.19) becomes singular. Methods are available to avoid such situations e.g. merging or removing the duplicate data, using noise term to make matrix invertible etc. Using these alternatives will disregard the spatial information contained in the multiple measurements, which is vital for the kinematics tracking of the extended targets. One possible solution could be to use a spatial model to represent the multiple measurements from the extended target at a each time sample by a single (pesudo) measurement. The GPMT processes these pseudo measurements for the kinematics state estimation.

8.3.2 First Order 2D GPMT

The 2D GPMT, given in Section 8.3.1, provides both the predicted and filtered position coordinates of the target. The first-order time derivatives of the position coordinates can be determined, based on the the derivations from [114] and [115] as given in Section 8.2.3, using the first order 2D GPMT (FO-GPMT). These are given below:

$$\tilde{\mu}_k^{\dot{x}} = \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}_k}) \left(\boldsymbol{K}_{\boldsymbol{t}_k\boldsymbol{t}_k} + \sigma_x^2 \boldsymbol{I}_d\right)^{-1} \boldsymbol{z}_{\boldsymbol{t}_k}^x, \tag{8.12}$$

$$(\tilde{\mu}_k^{\dot{\phi}})^2 = \boldsymbol{\Delta}_{\bar{t}\bar{t}}^2(\boldsymbol{K}_{\bar{t}\bar{t}}) - \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}_k}) \left(\boldsymbol{K}_{\boldsymbol{t}_k\boldsymbol{t}_k} + \sigma_x^2 \boldsymbol{I}_d\right)^{-1} \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\boldsymbol{t}_k\bar{t}}), \tag{8.13}$$

$$\hat{\mu}_k^{\dot{x}} = \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}'_k}) \left(\boldsymbol{K}_{\boldsymbol{t}'_k\boldsymbol{t}'_k} + \sigma_x^2 \boldsymbol{I}_d\right)^{-1} \boldsymbol{z}_{\boldsymbol{t}'_k}^x, \tag{8.14}$$

$$(\hat{\mu}_k^{\dot{\phi}})^2 = \boldsymbol{\Delta}_{\bar{t}\bar{t}}^2(\boldsymbol{K}_{\bar{t}\bar{t}}) - \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}'_k}) \left(\boldsymbol{K}_{\boldsymbol{t}'_k\boldsymbol{t}'_k} + \sigma_x^2 \boldsymbol{I}_d\right)^{-1} \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\boldsymbol{t}'_k\bar{t}}), \tag{8.15}$$

where $\mu^{\dot{x}}$ and $(\mu^{\dot{\phi}})^2$ denote, respectively, the mean and variance of the first order derivatives with respect to the position coordinates.

8.4 Performance Validation

The GPMT performance is validated over five challenging manoeuvring scenarios for 10000 Monte Carlo runs. The proposed approach is compared with three model-based filters, namely NCV-KF, FGIMM and Singer-KF models (see Sections 3.3 and 3.4) [63]. The root mean square error (RMSE) of the position and the velocity are considered as the performance comparison parameters. A performance grade $\in \{1, 2, 3, 4\}$ is assigned to each filter for each RMSE value of each scenario. The grade value of 1 is assigned to the best and 4 to the worst, out of the four compared approaches. The five scenarios are explained below:

S1 **Uniform motion**. The target velocity is constant. This scenario matches the NCV-KF and the FGIMM filters.

S2 **Coordinated turns matched**. The target motion is modelled using the NCT ($25°/s$ for 8 s) and NCV motion models. This scenario is matched to the FGIMM filter.

S3 **Coordinated turns mismatched**. This is similar to the scenario **S2** but the NCT ($12°/s$ for 20 s) model is not matched to the FGIMM. The scenario is not matched to any filters.

S4 **Singer matched**. The target motion is modelled using a Singer acceleration model with maximum possible acceleration set to $a_{max} = 50\ m/s^2$, probability of no-acceleration $P_0 = 0.4$, probability of maximum acceleration $P_{max} = 0.1$ and manoeuvre time constant $\tau_m = 8\ s$.

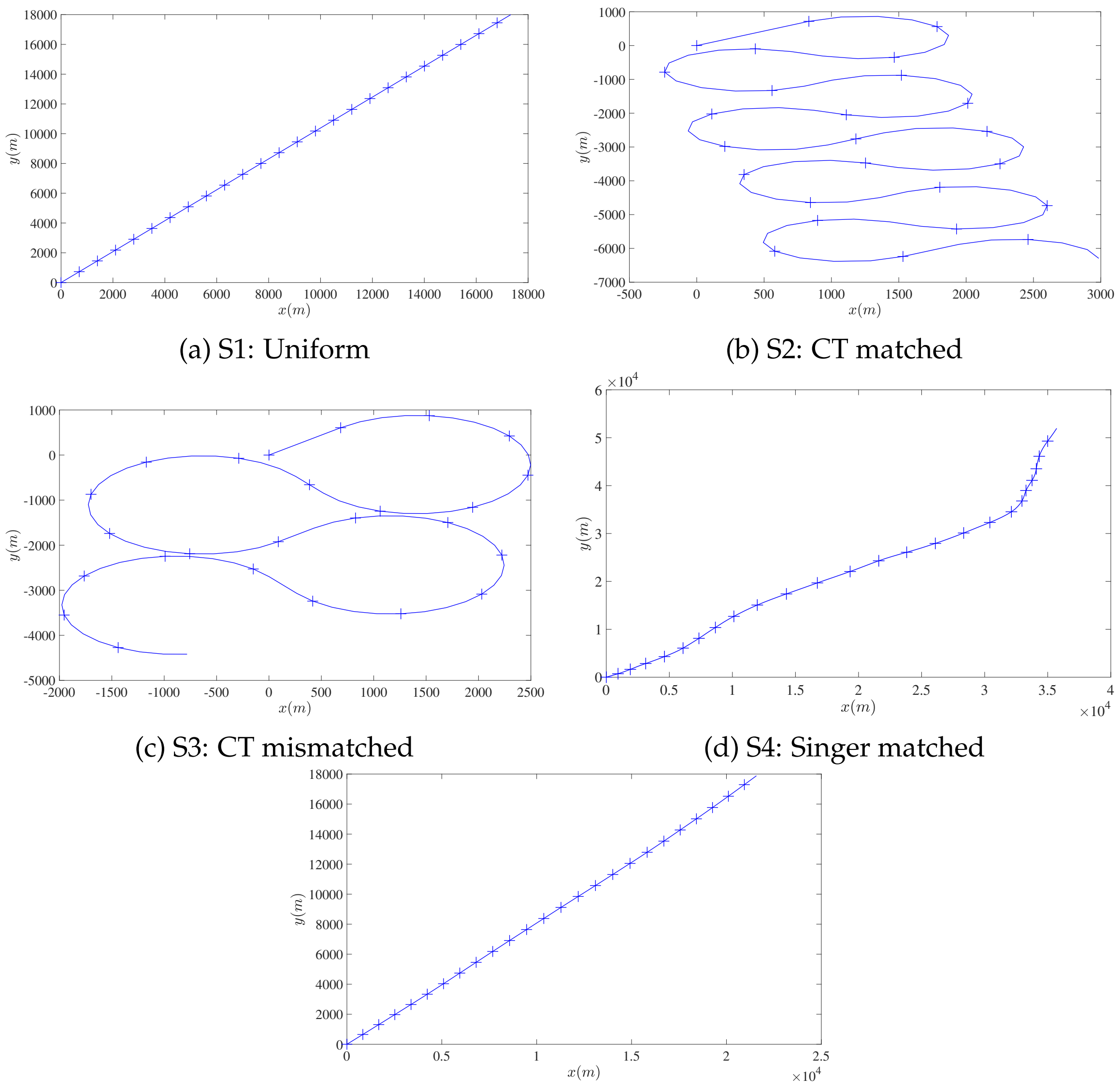
S5 **Singer mismatched**. This scenario is similar to the scenario **S4** with following changes in the parameters, $a^2_{max} = 2\ m/s^2$, $P_0 = 0.6$ and $\tau_m = 25\ s$. This scenario is not matched to any filter.

The initial target velocity in each coordinate is chosen randomly between 150 m/s and 250 m/s and the target maintains uniform motion for the initial 5 s in all scenarios. The total duration is 100 s and the measurement noise standard deviation is $\sigma_x = \sigma_y = 25\ m$. A sample trajectory of each scenario is shown in Fig. 8.2. A 15% initialization noise is added to all the model-based filters.

8.4.1 Filter Parameters

- **NCV-KF**. The process noise variance is 500 m^2/s^2. The noise variance value is chosen high to prevent the filter from diverging during the scenarios involving sharp manoeuvres i.e. **S2**, **S3** and **S4**.

- **Singer-KF**. The model parameters are chosen the same as in the scenario **S4**.



(a) S1: Uniform

(b) S2: CT matched

(c) S3: CT mismatched

(d) S4: Singer matched

(e) S5: Singer mismatched

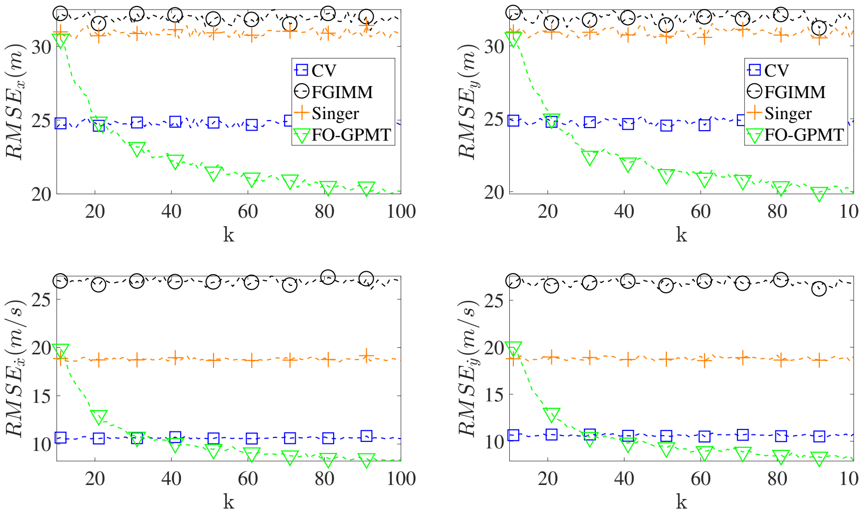
Fig. 8.2 **Sample trajectory**. The figure shows a sample trajectory for each scenario. The trajectory starts from origin (0,0) in all scenarios.

- **FGIMM**. The fixed grid is modelled using a single NCV model and two NCT models. The rate of turns of the NCT models are set to $\{-25, 25\}^\circ/s$. The Markov transition probability of the same mode is set to 0.7 and for changing the mode is 0.15, the initial model probability vector is $\{0.15, 0.7, 0.15\}$ and the process noise variance is set to $7.57 \times 10^{-8}\ m^2/s^2$ for each model.

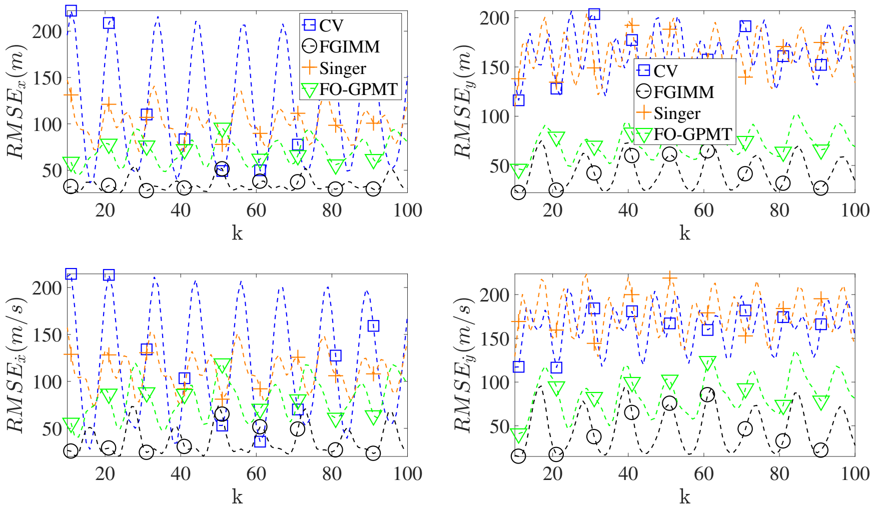
- **FO-GPMT**. The tracker depth is set to $d = 10$ samples.

8.4.2 Results

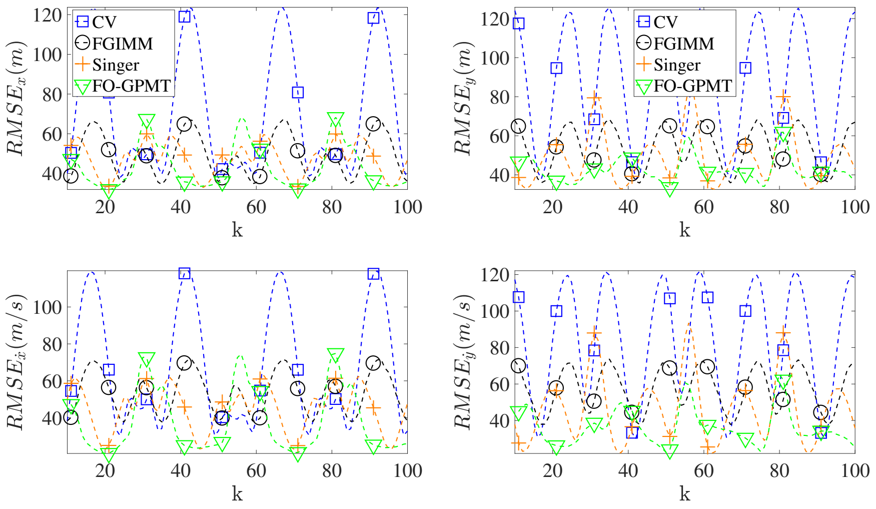
The graphical and numerical comparisons of the predicted states are given in Fig. 8.3 and Table 8.1, respectively. The comparison of the filtered states is given in the Fig. 8.4 and Table 8.2. The cell value of NaN means that the corresponding filter diverged. It can be observed that both the prediction and filtering accuracies of the proposed approach are better than the model based approaches in the mismatched scenarios, **S3** and **S5**. In matched scenarios, it is second best to the matched filter. The performance gain of the proposed method as compared to the model based approaches is less in the filtering step as compared to the performance gain in the prediction step. This improvement in the model based approaches in the filtering step, as compared to their respective prediction step, can be attributed to the Kalman filter rather than to the performance of the underlying model. Considering all five scenarios, the proposed approach performs best due to its adaptability.



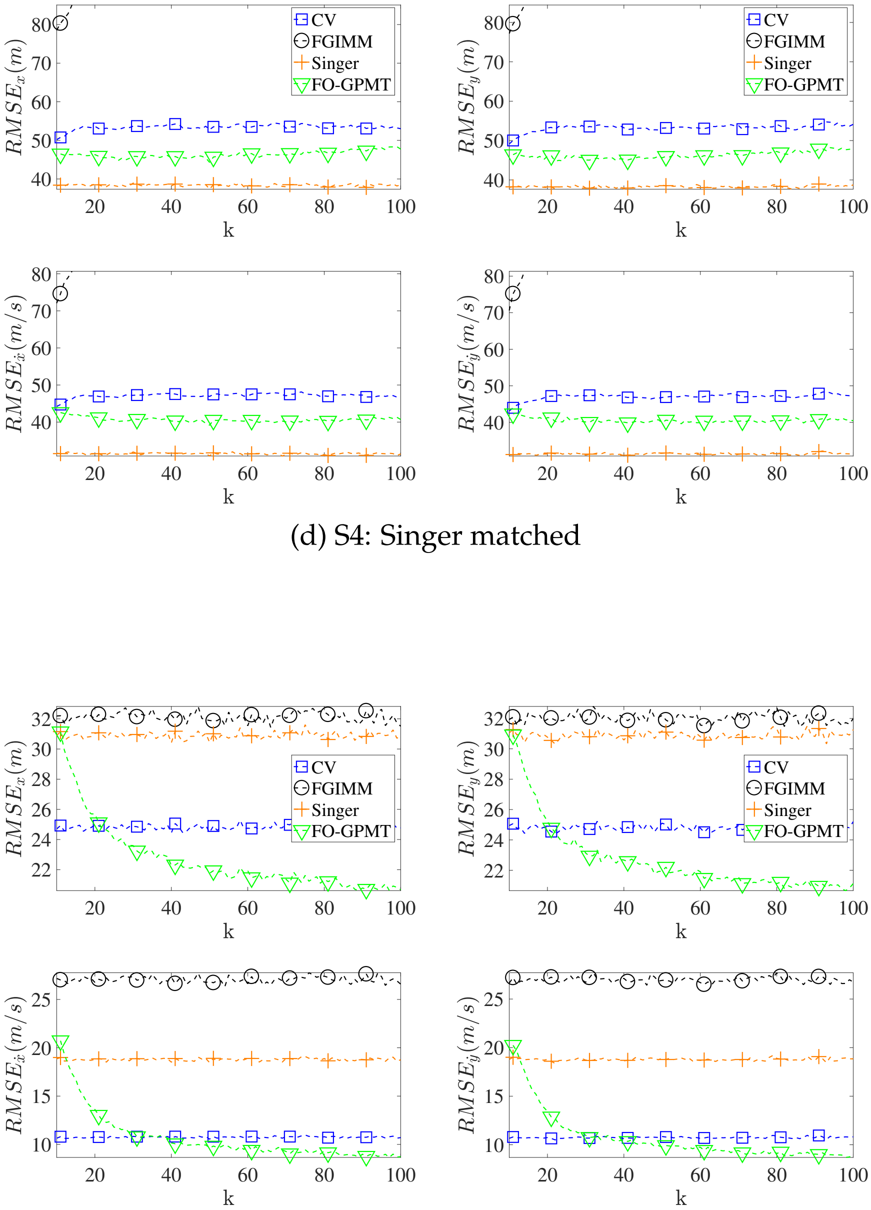
(a) S1: Uniform



(b) S2: CT matched



(c) S3: CT mismatched



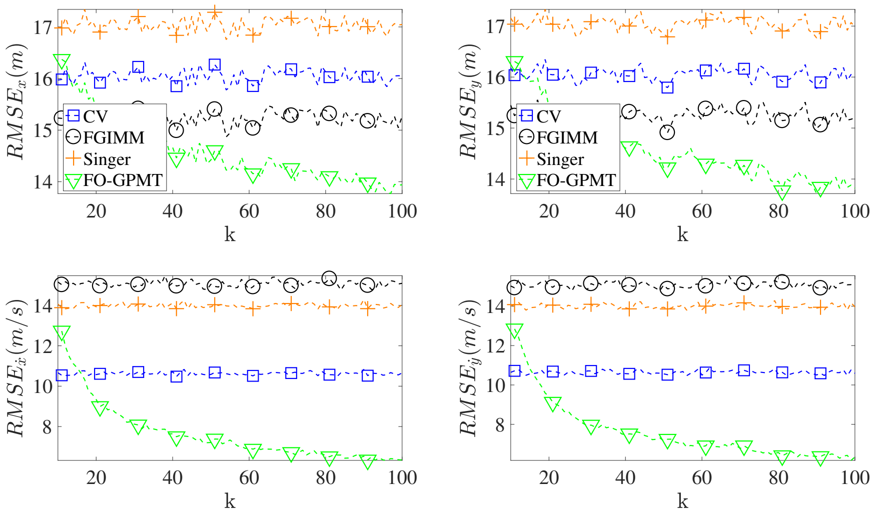
(d) S4: Singer matched

(e) S5: Singer mismatched

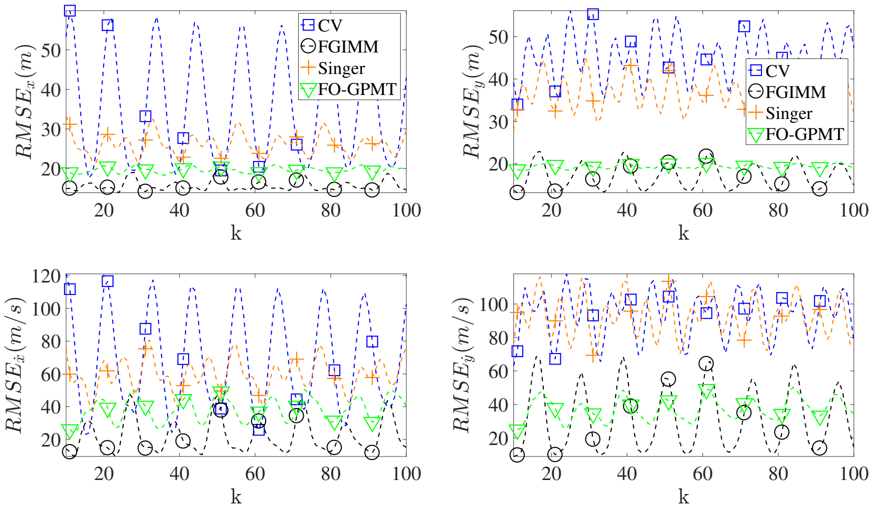
Fig. 8.3 **Prediction performance**. The figure shows the prediction performance results of 10000 Monte Carlo runs for the five scenarios.

Table 8.1 Predicted mean errors

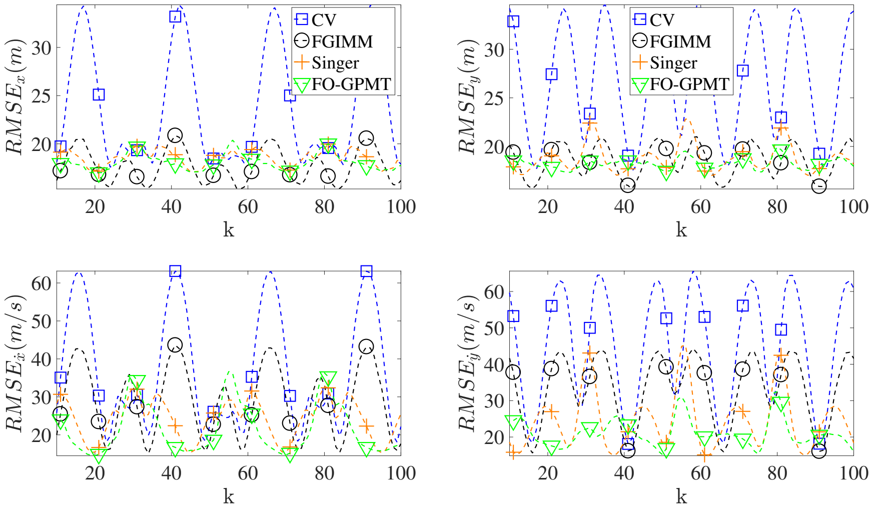
		CV	Singer	FGIMM	FO-GPMT
S1 **CV**	x	25	31	32	22
	$\dot{x}$	11	19	27	10
	y	25	31	32	22
	$\dot{y}$	11	19	27	10
S2 **CT matched**	x	115	98	35	70
	$\dot{x}$	113	107	37	81
	y	159	155	43	72
	$\dot{y}$	160	171	47	86
S3 **CT mismatched**	x	69	46	49	43
	$\dot{x}$	64	44	53	38
	y	86	49	54	43
	$\dot{y}$	82	47	57	36
S4 **Singer matched**	x	52	38	NAN	46
	$\dot{x}$	46	31	NAN	41
	y	52	38	NAN	46
	$\dot{y}$	46	31	NAN	41
S5 **Singer mismatched**	x	25	31	32	23
	$\dot{x}$	11	19	27	11
	y	25	31	32	23
	$\dot{y}$	11	19	27	11



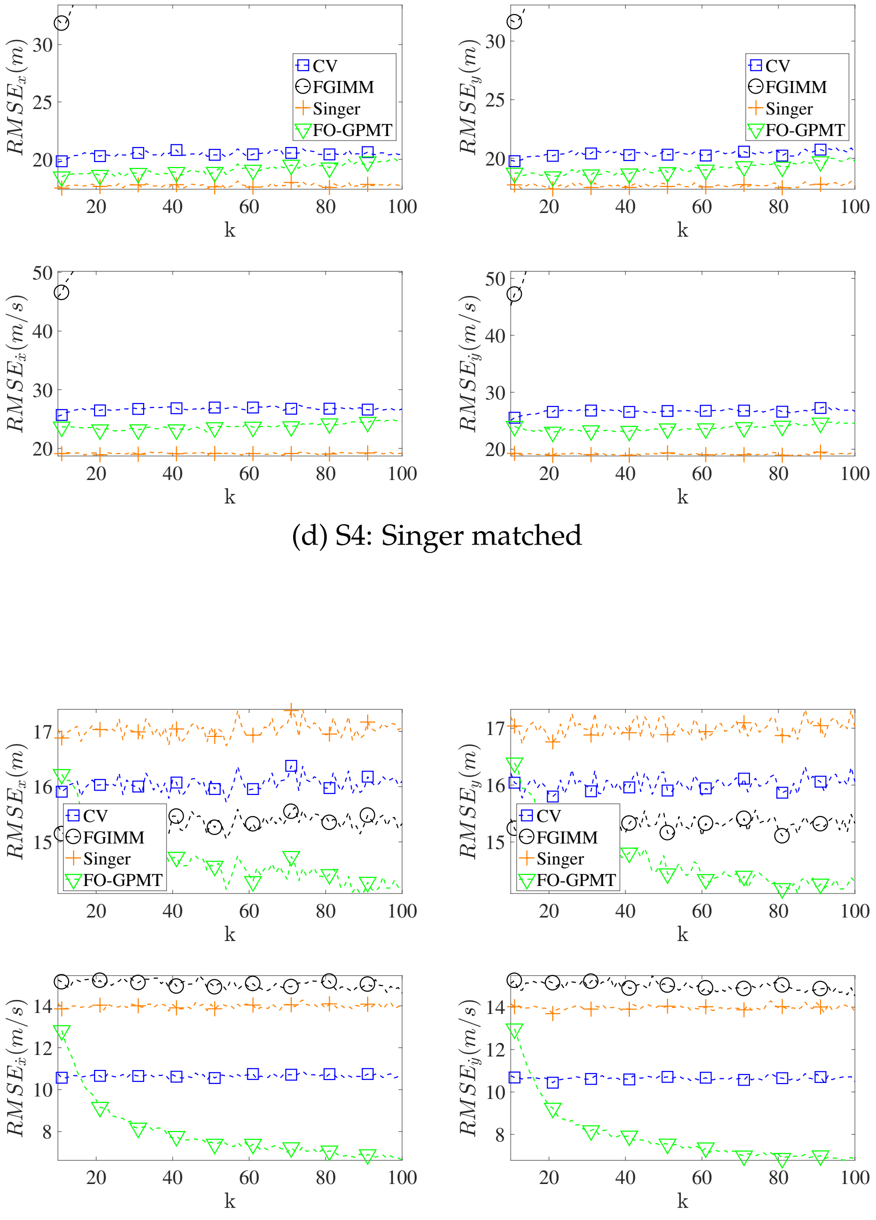
(a) S1: Uniform



(b) S2: CT matched



(c) S3: CT mismatched



(d) S4: Singer matched

(e) S5: Singer mismatched

Fig. 8.4 **Filtering performance**. The figure shows the filtering performance results of 10000 Monte Carlo runs for the five scenarios.

Table 8.2 Filtered mean errors

		CV	**Singer**	**FGIMM**	**FO-GPMT**
S1 **CV**	x	16	17	15	15
	$\dot{x}$	11	14	15	8
	y	16	17	15	15
	$\dot{y}$	11	14	15	8
S2 **CT matched**	x	35	26	16	20
	$\dot{x}$	66	58	22	38
	y	45	36	17	19
	$\dot{y}$	92	89	32	37
S3 **CT mismatched**	x	23	18	18	18
	$\dot{x}$	36	24	29	22
	y	27	19	18	18
	$\dot{y}$	45	25	33	21
S4 **Singer matched**	x	20	18	NAN	19
	$\dot{x}$	26	19	NAN	24
	y	20	18	NAN	19
	$\dot{y}$	26	19	NAN	24
S5 **Singer mismatched**	x	16	17	15	15
	$\dot{x}$	11	14	15	8
	y	16	17	15	15
	$\dot{y}$	11	14	15	8

The performance grades of the proposed approach are given in Table 8.3.

Table 8.3 Performance grades of the proposed approach

	Filtering Grades				**Prediction Grades**			
Scenario	x	y	v_x	v_y	x	y	v_x	v_y
S1	1	1	1	1	1	1	1	1
S2	2	2	2	2	2	2	2	2
S3	1	1	1	1	1	1	1	1
S4	2	2	2	2	2	2	2	2
S5	1	1	1	1	1	1	1	1

The NCV and the FGIMM filters are expected to give grade 1 performance for the scenario **S1**. However, the proposed approach performs better than both the matched filters. The FGIMM and Singer filters perform best for scenarios **S2** and **S4**, respectively, as the simulated scenario is exactly matched to the motion model of the filter. The performance of the FGIMM is worse for the mismatched scenarios and it even diverges for the scenario **S4**. The Singer based filter performs worst for the uniform motion based scenario i.e. **S1**. This motion is one of the common modes in many tracking applications. For example, the aerial targets tracking systems are expected to track airliners which are mostly moving under nearly constant motion.

8.5 Conclusions

This chapter proposes a model-free approach, using Gaussian process, for point target kinematics prediction and filtering. A one-dimensional Gaussian process model, called Gaussian process motion tracker, relates the spatial output to the temporal input for filtering of the target position. The Gaussian process model trains on past measurements to represent the training data using a prior over an infinite distribution of functions. The hyperparameters are also learned by maximizing the likelihood. The trained and learned GP model predicts and filters the target position at the new input (time). Unlike popular state-of-the-art approaches, the proposed approach does not require initialization and is not prone to initialization errors. Moreover, state prediction and filtering run independently. The prediction and filtering of the time derivative of the target position is proposed using a derivative of the Gaussian process. The proposed approach can also be extended for the prediction and filtering of higher dimensions and the higher order derivatives.

The evaluation of the proposed model-free approach is done by comparing the performance against model-based approaches. Various challenging manoeuvring scenarios are considered for performance comparison. The root mean square errors based tracking comparison shows that the proposed approach gives an overall best performance in all considered scenarios.

The proposed learning is computationally complex and cannot be done in real-time. In the future, work will be done on an online learning based GPMT. We will also focus on evaluation of the uncertainty propagation using different covariance kernels theoretically. A square exponential covariance kernel based GPMT is proposed in this chapter. Other covariance kernels will also be explored for the point target tracking.

Chapter 9

On the Impact of Different Kernels and Training Data on a Gaussian Process Approach for Tracking

Abstract

The application of multiple target tracking algorithms has exponentially increased during the last two decades. Recently, model-free approaches, such as Gaussian process regression and convolutional neural networks, have been developed for target tracking. This chapter presents a simulation-based study on the practical aspects of a very promising and recently proposed Gaussian process approach, namely the Gaussian process motion tracker (see Chapter 8) [4]. The chapter also provides design guidelines on the various aspects of the above-mentioned tracking approach by comparing the rational quadratic and Matérn based GPMT with the originally proposed squared exponential GPMT of Chapter 8. The study shows that the squared exponential GPMT performs best on constant velocity and Singer model based target trajectories. The rational quadratic GPMT performs best on the coordinated turn trajectories. The accuracy of the Matérn GPMT improves with an increasing number of training data. The accuracies of rational quadratic GPMT and squared exponential GPMT decreases with increasing data on Singer model based trajectory. Lastly, it is shown that all three GPMT approaches are robust to the changing measurement noise variances.

9.1 Introduction

Historically, model-based approaches have been applied to solving target tracking problems. Recently, machine-learning based model-free methods have been proposed either as a complete solution [37] or in a hybrid setup [27, 2]. Hybrid methods combine model-based and model-free methods for target tracking. Various machine learning methods have been proposed in the literature for different artificially intelligent systems. In the field of target tracking, deep learning and Gaussian process based methods have become popular recently. Gaussian process methods have been recently proposed as an efficient solution to different target tracking problems [37, 27, 3].

In this chapter, various aspects of the Gaussian process motion tracker (GPMT) proposed in Chapter 8 [4] are studied. The GPMT provides point target state estimation in the presence of unknown target dynamics and measurement noise. The measurement to target assignment is assumed known. The GPMT employs the Gaussian process in a regression setting for the target state estimation. The Chapter 8 [4] does not discuss some important aspects of the approach. These include the choice of the covariance kernel, the robustness of the approach to the measurement noise model and effect of the training data on the proposed approach. A simulation based study on the above-mentioned aspects of the GPMT is presented in this chapter in an attempt to highlight the strength of the GPMT approach.

The rest of the chapter is structured as follows. The study methodology is given in Section 9.2. The choice of the covariance kernel, the impact of the training data and the robustness of the GPMT to the measurement noise are studied in Sections 9.3, 9.4 and 9.5, respectively. The studies are followed by conclusions.

9.2 Study Methodology and Test Scenarios

The GPMT proposed in Chapter 8 is a point target tracking approach. Three simulation based studies on various aspects of the GPMT are presented in this chapter. The simulation-based studies are based upon the target scenarios and the evaluation methodology described in this section. The root mean square error (RMSE) of the target predicted position is chosen as the performance measure. A comprehensive database of the point target trajectories is not publicly available. Hence, the target trajectories are generated using the three most commonly used point target dynamics models. These are the nearly constant velocity (NCV) (see Section 3.4.1) [63], the nearly coordinated turn (NCT) (see Section 3.4.4) [63] and the Singer acceleration models (see Section 3.4.3) [64].

The sampling time is set to $T = 1\,s$, the total samples are $K = 100$, the measurement noise standard deviation is set to $\sigma = 25\,m$, the probability of detection is set to $p_d = 1$, the initial target velocity is randomly chosen in the limits $150\,m/s \leq v_0 \leq 250\,m/s$, the process noise variances of the NCV and the NCT models are set to $q^{NCV} = q^{NCT} = 1e - 12\,m^2/s^2$, the turn rate is set to $\omega = 15°/s$, the manoeuvre variance is set to $\sigma_a^2 = 168.75\,m^2$ and the manoeuvre sojourn time is set to $\tau_m = 8\,s$. The coordinated turn model based scenario switches between the CV and the CT models. The sojourn time of the CT based manoeuvre is 8 s. The results are computed over 1000 Monte Carlo runs.

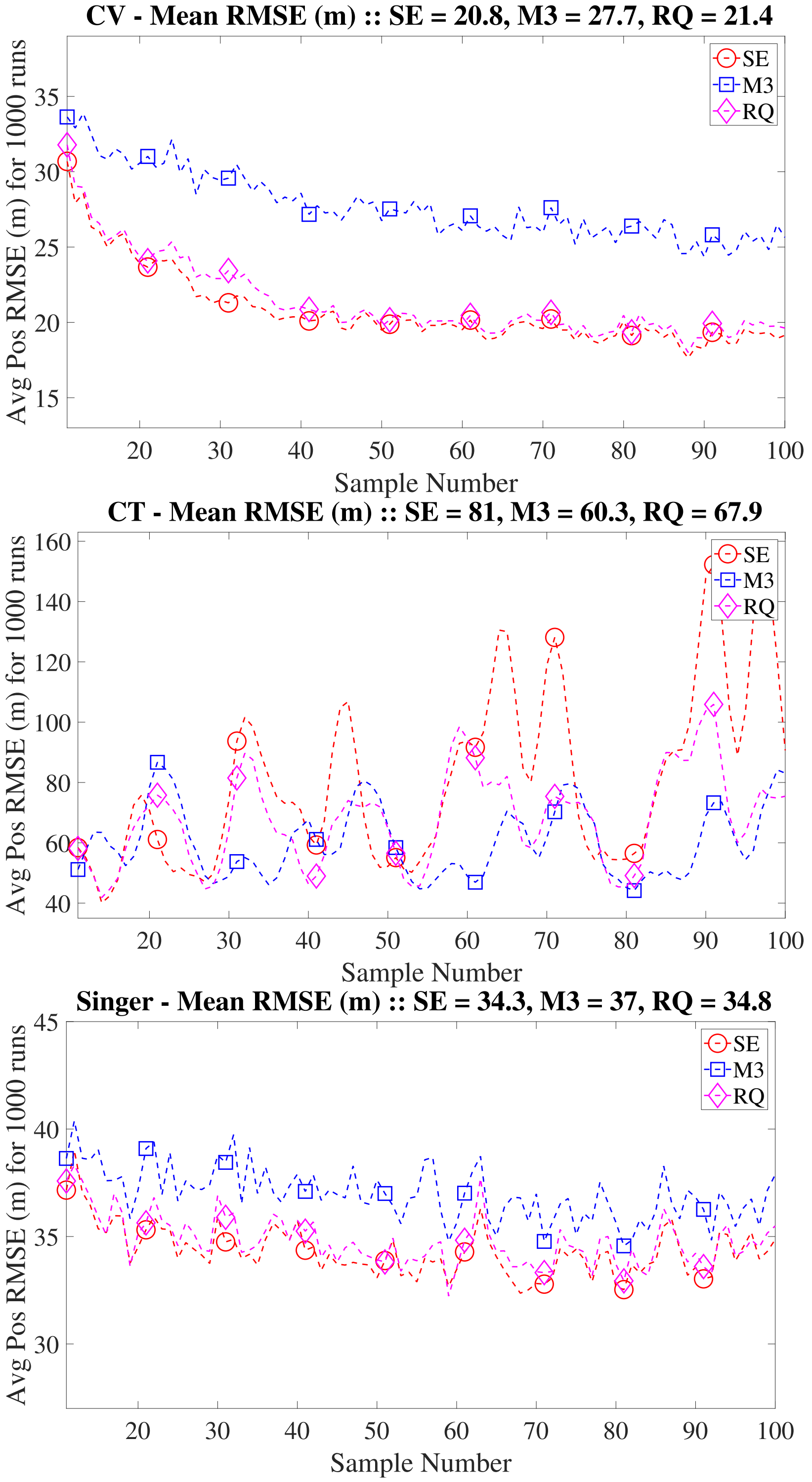
9.3 Choice of Covariance Kernels

The GPMT is proposed using a squared exponential (SE) covariance kernel in Chapter 8 [4]. This kernel is infinitely differentiable and it tracks all the higher order derivatives of the position. The kernel is, however, too smooth as compared to the real target dynamics. In this section, a simulation-based study is performed to compare the performance of the different covariance kernels. The two new kernels chosen in this study are the rational quadratic (RQ) (see (2.11)) and the Matérn (with $\nu = \frac{3}{2}$) kernels (see (2.13)). The RQ covariance kernel is smoother as compared to the Matérn kernel with $\nu = \frac{3}{2}$. The results are given in Fig. 9.1. The three plots correspond to three different target models which are the NCV (top), the NCT (middle) and the Singer (bottom).

It can be observed that the SE and the RQ perform better than the Matérn for the NCV and Singer target dynamics models based trajectories. However, the Matérn kernel outperforms both of them for the NCT model based trajectory. The RQ based GPMT performs slightly worse as compared to the Matérn kernel. The performance of the SE based GPMT is significantly poor and could be a bad choice for this type of trajectory. Based on the above study, the following recommendations are made:

R1 For the NCV and Singer based scenarios, the SE based GPMT should be chosen.

R2 For applications involving target trajectories based on all three models, the RQ based GPMT is the preferred choice.


CV - Mean RMSE (m) :: SE = 20.8, M3 = 27.7, RQ = 21.4

CT - Mean RMSE (m) :: SE = 81, M3 = 60.3, RQ = 67.9

Singer - Mean RMSE (m) :: SE = 34.3, M3 = 37, RQ = 34.8

Fig. 9.1 **Comparison of covariance kernels**. This figure show results of the positional prediction using three different variants of the GPMT based on different covariance kernels.

9.4 Effect of the Training Data

The parameter d of the GPMT controls the size of the training data. The evaluation of the GPMT in Chapter 8 [4] is done by setting it as $d = 10$, that is, the 10 most recent measurements are considered for the training of the model. In this section, the performance of the GPMT for the different values of d is studied for the three kernels. The results are given in Figs. 9.2, 9.3 and 9.4. The three different values of the parameter are $d = 10, 15, 20$.

It can be observed, in Fig. 9.2, that the accuracy of the SE based GPMT increases with the increase in the training data for the NCV model based trajectories. For the remaining two scenarios, the accuracy decreases. It can be observed, in Fig. 9.3, that the accuracy of the RQ based GPMT increases with the increase in the training data for the NCV and the NCT model based trajectories. The performance degrades with the increase in the training data for the Singer model. In Fig, 9.4, it can be observed that the accuracy of the Matérn based GPMT increases with the increasing training data. Based on the above results, it is recommended to use a Matérn kernel based GPMT when the training data size is important for the application.

9.5 Robustness to Measurement Noise Model

The measurement noise variance can be set as a hyperparameter of the GPMT [4] and learned recursively from the training data. In this way, the GPMT model is robust to the measurement noise variance. This section provides a simulation based study on the performance degradation of the GPMT with the increasing noise variance. The noise standard deviation is chosen as $\sigma = [25, 50, 75, 100]\ m$. The percentage increase in the standard deviation of the noise with respect to $\sigma = 25\ m$ is 100%, 200% and 300%. The percentage degradation of the three kernels for the assumed scenarios is given in Table 9.1. It can be observed that although the accuracy decreases with the increase in the noise, the filters do not diverge. Moreover, no prior information regarding the change in the noise variance is input to the GPMT in different considered scenarios. This is achieved by setting the noise variance as a hyperparameter and learning it from the training data. All the considered GPMT approaches are hence adaptive to the changing measurement noise variance.

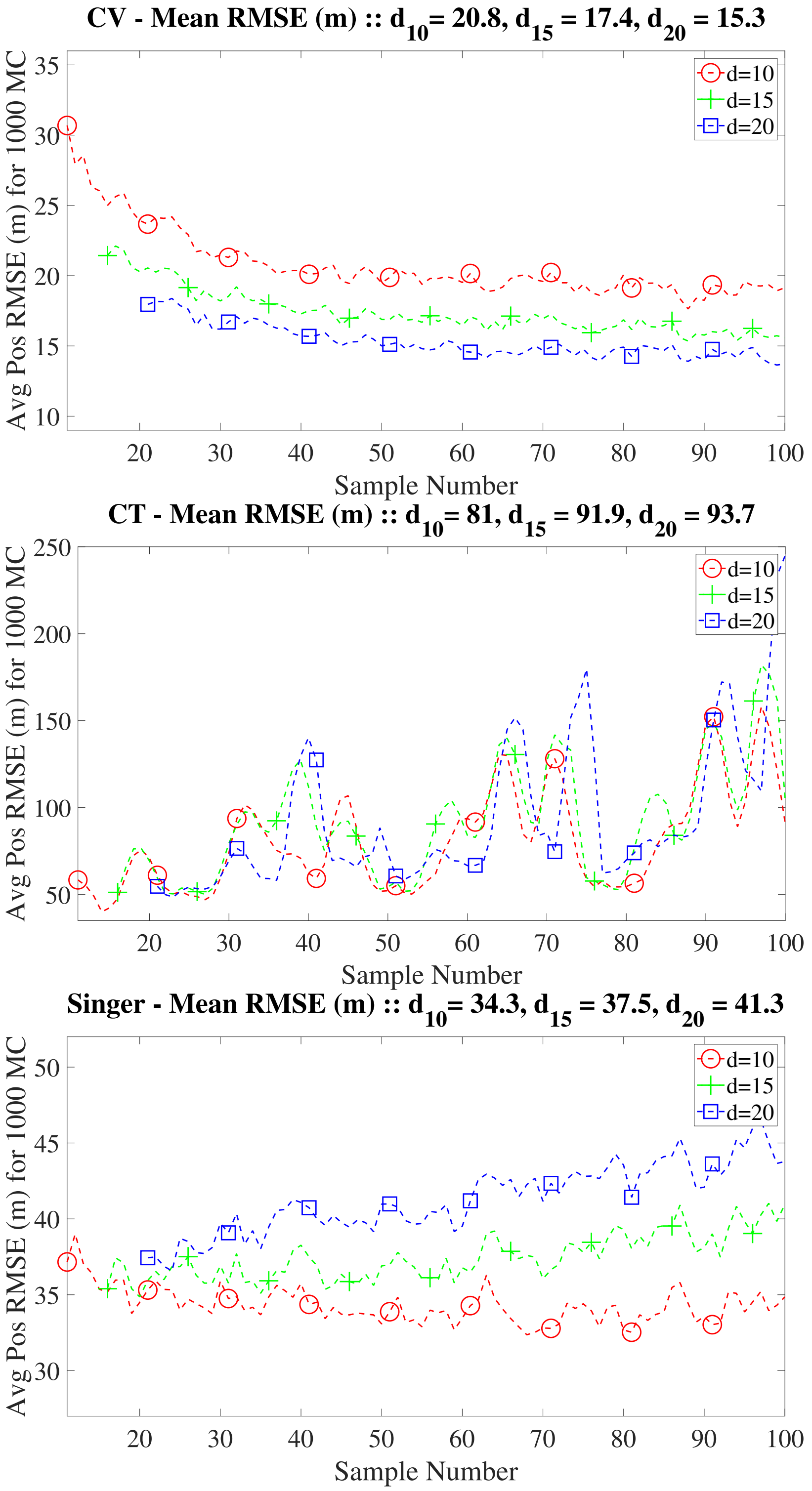


Fig. 9.2 **Effect of the training data on the SE kernel**. This figure shows the results for the three scenarios, as explained in Fig.9.1.

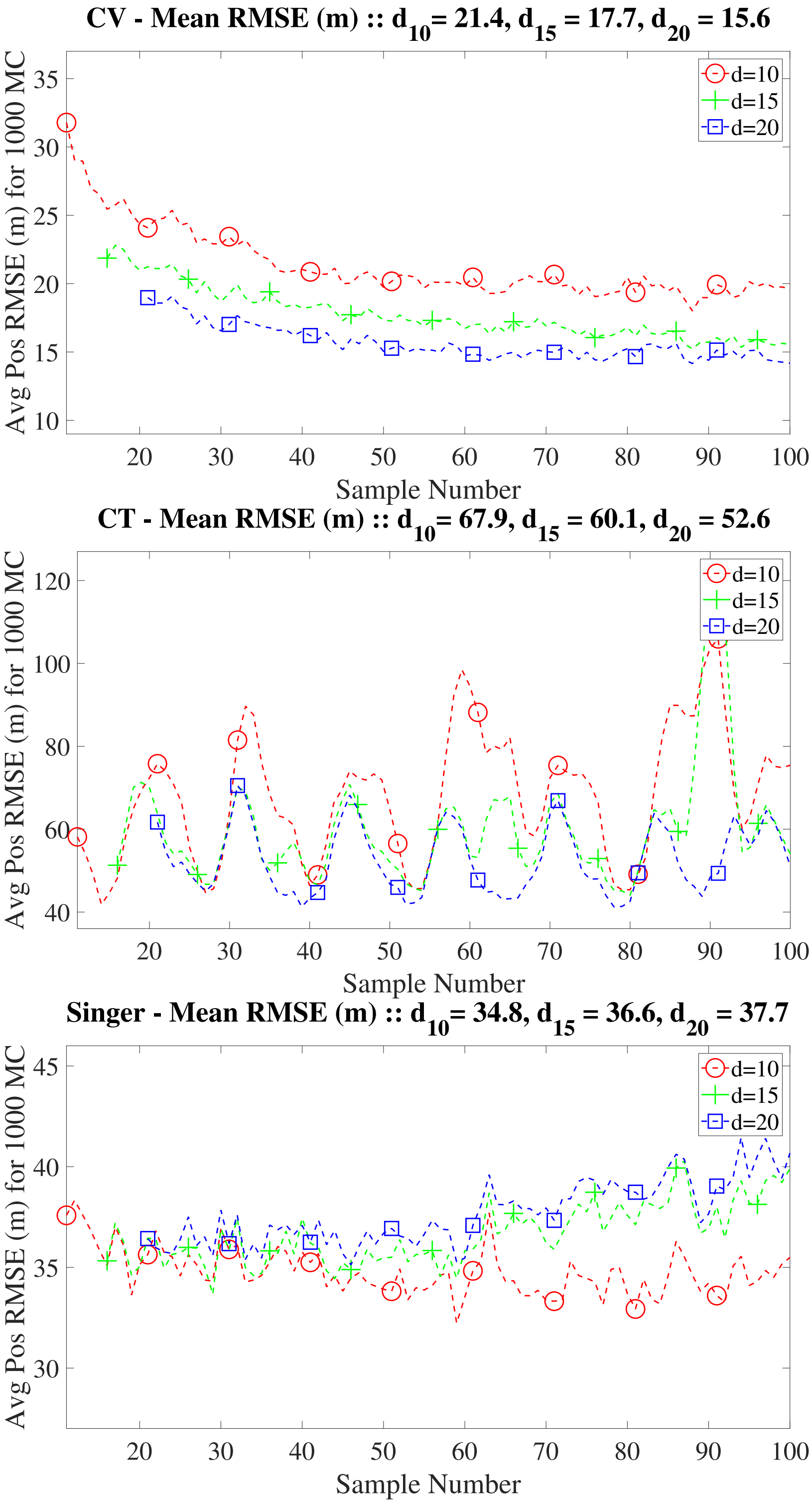
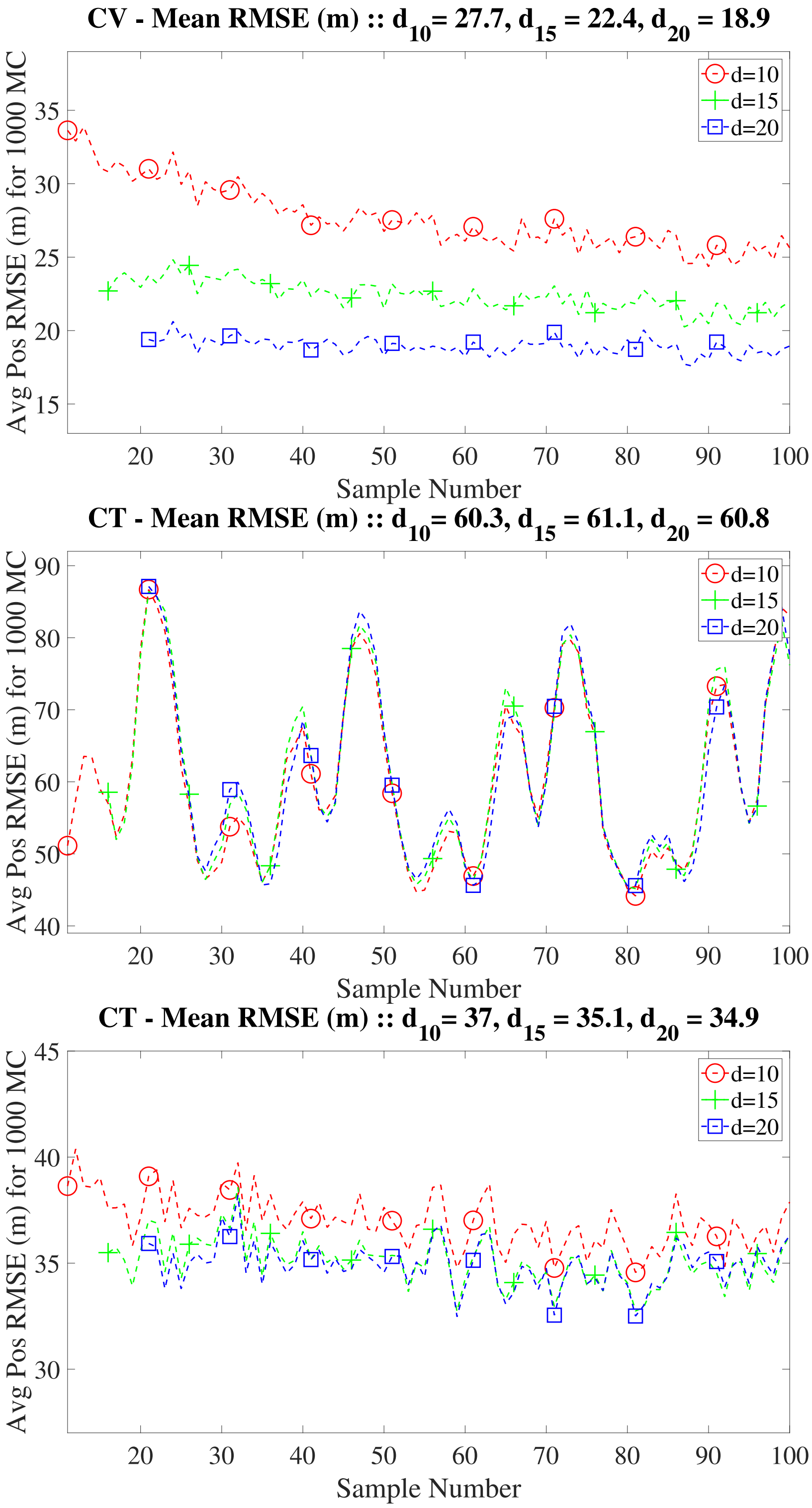


Fig. 9.3 **Effect of the training data on the RQ kernel**. This figure shows the results for the three scenarios, as explained in Fig.9.1.



CV - Mean RMSE (m) :: d_{10}= 27.7, d_{15} = 22.4, d_{20} = 18.9

CT - Mean RMSE (m) :: d_{10}= 60.3, d_{15} = 61.1, d_{20} = 60.8

CT - Mean RMSE (m) :: d_{10}= 37, d_{15} = 35.1, d_{20} = 34.9

Fig. 9.4 **Effect of the training data on the Matérn with $\nu = \frac{3}{2}$ (M3) kernel**. This figure shows the results for the three scenarios, as explained in Fig.9.1.

Table 9.1 Performance degradation with increased noise variance

	NCV			**NCT**			**Singer**		
	100	***200***	***300***	***100***	***200***	***300***	***100***	***200***	***300***
SE	92	181	270	34	64	88	64	127	187
RQ	93	183	273	41	80	112	65	127	187
M3	78	150	217	49	94	132	70	131	187

9.6 Conclusions

This chapter presents a simulation based study on various aspects of the Gaussian process motion tracker (proposed in Chapter 8). The GPMT is a Gaussian process approach for point target tracking proposed originally using a squared exponential covariance kernel. In this chapter, its performance is compared with GPMT based on two other covariance kernels, namely rational quadratic (RQ) and Matérn (M3). The popular point target kinematics models are chosen for creating simulated evaluation scenarios.

The SE-GPMT performs best for NCV and Singer models based scenarios where RQ-GPMT gives best results for the NCT model based scenario. The three approaches are next evaluated based on the size of the training data. The study shows that increasing the training data size from 10 to 20 improves the accuracy of the M3-GPMT in all scenarios. The behaviour of the remaining two approaches is not consistent with the increase in the training data. Lastly, the effect of increasing the measurement noise variance is studied. The results show that all the approaches adapt to the unknown measurement noise models.

Future work will be focused on theoretical studies of the impact of uncertainties on the GPMT. Other covariance kernels will be explored. Work will also be done to theoretically verify the results of this study.

Chapter 10

A Learning Gaussian Process Approach for Manoeuvring Target Tracking and Smoothing

Abstract

Model-based approaches for target tracking and smoothing estimate the infinite number of possible target trajectories using a finite set of models. This chapter proposes a data-driven approach that represents the possible target trajectories using a distribution over an infinite number of functions. Recursive Gaussian process and derivative based Gaussian process approaches for target tracking and smoothing are developed, with online training and parameter learning. The performance evaluation over two highly manoeuvring scenarios, shows that the proposed approach provides 80% and 62% performance improvement in the position and 49% and 22% in the velocity estimation, respectively, as compared to the best model-based filter.

10.1 Introduction

A non-recursive model-free point target tracking approach, Gaussian process motion tracker (GPMT), is proposed in Chapter 8 [4]. This is a Gaussian process based approach, which predicts and filters the position and higher order position derivatives of a point target using historical measurement data. The GPMT, includes training and learning stages, models an infinite number of target trajectories. In Chapter 8, it is demonstrated to be better in tracking a wide set of target trajectories as compared to the model-based approaches. The learning stage proposed for the GPMT is not online and hence the approach cannot be implemented in real-time. In this chapter, a recursive Gaussian process* motion tracker (RGP*MT)[1] is proposed for online model-free point target kinematics filtering and smoothing. The state estimation and the learning of the Gaussian process hyperparameters are performed recursively. The GPMT relies on the measurement data for the training of the GP whereas an inducing point set representation is used as the training data for the recursive filtering and smoothing proposed in this chapter. The performance is evaluated on both highly manoeuvrable and less agile targets.

The main contributions of this chapter are given in Section 10.1.1. Background knowledge for recursive GP methods is covered in Section 10.2. The proposed tracker and smoother are presented in Section 10.3. The performance evaluation and results are given in Section 10.4, followed by conclusions.

10.1.1 Contributions

The key contributions of this work are as follows: (i) A data-driven approach is proposed for online point target tracking. The proposed approach is shown to estimate well the unknown and nonlinear target trajectory using the recursive update of the Gaussian process hyperparameters (Section 10.3.1). (ii) A model-free recursive approach is proposed for the estimation of the derivative of a Gaussian process. This approach can also be extended in the same way to recursively update the higher order derivatives of the Gaussian process. The proposed approach is demonstrated by estimating the velocity of a point target (Section 10.3.2). (iii) A recursive point target smoother is proposed for the online position and velocity smoothing (Section 10.3.3). The proposed smoother can be extended for smoothing of the higher order derivatives of the Gaussian process in the same way. (iv) The proposed filters and smoothers are shown to be robust to the measurement noise model parameters. This is achieved by augmenting the state vector with the measurement noise variance parameter. This parameter is recursively updated at each time sample (Section 10.3.4). A simulation based study is also carried out to demonstrate the robustness of the proposed approaches to the measurement noise variance (Section 10.4.6). (v) The performance evaluation of the proposed approach is done under challenging scenarios and compared with model-based approaches (Section 10.4).

[1] The $\star$ indicates that the RGP*MT is a recursive algorithm, whose hyperparameters are learnt online.

10.2 Background Knowledge

10.2.1 Recursive Gaussian Process Regression

GP regression is a powerful estimation method with high-computational training and learning stages. The computations scale as $\mathcal{O}(N^3)$ for the N-dimensional data for the training. Learning requires solving a non-convex optimization problem. Typically, an analytical solution to this optimization problem is not available. Most of the existing numerical solutions are not applicable in realtime. In [40], an on-line approach for GP regression has been proposed. Two approaches have been proposed in [40] by assuming bounded input domain. The first one addresses the computational complexity of the training process and is termed as *on-line regression* or recursive GP (RGP). The second approach, in addition to the training, provides online learning and is called *on-line learning* or recursive GP* (RGP*). The input domain is sampled at discrete points and the input vector representing the sparse grid points is called the basis vector. The mean and the covariance at the sparse grid points form the sparse GP representation. The sparse GP representation is updated using the measurements received at each time sample.

10.2.2 Recursive Gaussian Process with Online Regression

This section summarizes the recursive Gaussian process with online regression proposed in [40]. Consider the GP (2.1) and (2.16). The unknown function f is represented by a multivariate Gaussian distributed vector of N inducing points $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{u}) = [f(u_1), \cdots, f(u_N)]^T$ with an initial distribution $p_0(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{\mu}_0^f, \boldsymbol{C}_0^f)$. At each time sample, the inducing point distribution is updated using the corresponding measurements and the prior distribution, assuming known hyperparameters. The required conditional distribution $p(\boldsymbol{f}|\boldsymbol{z}_{1:k})$ cannot be updated using the GP regression (2.18) and (2.19). Firstly, because, the GP regression is a prediction method whereas a measurement update is required here. Secondly, GP regression requires complete training data whereas a recursive update of a prior distribution is to be done. Given the prior inducing points distribution $p(\boldsymbol{f}|\boldsymbol{z}_{1:k-1}) = \mathcal{N}(\boldsymbol{\mu}_{k-1}^f, \boldsymbol{C}_{k-1}^f)$, the conditional distribution of the inducing points is expanded as follows for the recursive update:

$$p(\boldsymbol{f}|\boldsymbol{z}_{1:k}) = \int c \cdot p(\boldsymbol{z}_k|\boldsymbol{f}, \bar{\boldsymbol{f}}) \cdot p(\bar{\boldsymbol{f}}, \boldsymbol{f}|\boldsymbol{z}_{1:k-1}) d\bar{\boldsymbol{f}}, \tag{10.1}$$

where $\bar{\boldsymbol{f}} = \boldsymbol{f}(\boldsymbol{u}_k^{\bar{f}})$ denotes the functional evaluation at the measurement input vector $\boldsymbol{u}_k^{\bar{f}}$ and c is a normalization constant. Integrating $\bar{\boldsymbol{f}}$ out allows maintaining a fixed number of inducing points. It is assumed that $\bar{\boldsymbol{f}}$ is conditionally independent of the past measurements $\boldsymbol{z}_{1:k-1}$ given $\boldsymbol{f}$. The inducing points are not observable and hence the conditional distribution of the measurement vector at k is independent of $\boldsymbol{f}$ given $\bar{\boldsymbol{f}}$. Using the above two assumptions, the joint prior and the conditional distribution of the measurement vector in (10.1) are written as given below:

$$p(\boldsymbol{f}|\boldsymbol{z}_{1:k}) = \int c \cdot p(\boldsymbol{z}_k|\bar{\boldsymbol{f}}) \cdot p(\bar{\boldsymbol{f}}|\boldsymbol{f}) \cdot p(\boldsymbol{f}|\boldsymbol{z}_{1:k-1}) d\bar{\boldsymbol{f}}. \tag{10.2}$$

The predictive distribution is determined as follows:

$$\begin{aligned} p(\bar{\boldsymbol{f}}|\boldsymbol{f}) \cdot p(\boldsymbol{f}|\boldsymbol{z}_{1:k-1}) &= \mathcal{N}(\tilde{\boldsymbol{\mu}}_k^{\bar{f}}, \boldsymbol{B}) \cdot \mathcal{N}(\boldsymbol{\mu}_{k-1}^{f}, \boldsymbol{C}_{k-1}^{f}) \\ &= \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_{k-1}^{f} \\ \tilde{\boldsymbol{\mu}}_k^{\bar{f}} \end{bmatrix}, \begin{bmatrix} \boldsymbol{C}_{k-1}^{f} & \boldsymbol{C}_{k-1}^{f}\boldsymbol{J}_k^T \\ \boldsymbol{J}_k\boldsymbol{C}_{k-1}^{f} & \tilde{\boldsymbol{C}}_k^{\bar{f}} \end{bmatrix} \right), \qquad (10.3) \\ \boldsymbol{J}_k &= \boldsymbol{K}_{\boldsymbol{u}_k^{\bar{f}}\boldsymbol{u}} \boldsymbol{K}_{\boldsymbol{uu}}^{-1}, \qquad (10.4) \\ \boldsymbol{B}_k &= \boldsymbol{K}_{\boldsymbol{u}_k^{\bar{f}}\boldsymbol{u}_k^{\bar{f}}} - \boldsymbol{J}_k\boldsymbol{K}_{\boldsymbol{u}\boldsymbol{u}_k^{\bar{f}}}, \qquad (10.5) \\ \tilde{\boldsymbol{\mu}}_k^{\bar{f}} &= \boldsymbol{m}(\boldsymbol{u}_k^{\bar{f}}) + \boldsymbol{J}_k(\hat{\boldsymbol{\mu}}_{k-1}^{f} - \boldsymbol{m}(\boldsymbol{u})), \qquad (10.6) \\ \tilde{\boldsymbol{C}}_k^{\bar{f}} &= \boldsymbol{B}_k + \boldsymbol{J}_k\hat{\boldsymbol{C}}_{k-1}^{f}\boldsymbol{J}_k^T, \qquad (10.7) \end{aligned}$$

where $\boldsymbol{u}$ denotes the input basis vector, $\boldsymbol{u}_k^{\bar{f}}$ represents the input vector corresponding to the k^{th} measurement vector $\boldsymbol{z}_k$, $\boldsymbol{J}_k$ and $\boldsymbol{B}_k$ are matrices derived from the GP regression (2.18) and (2.19), $\tilde{}$ represents the predicted variables and $\boldsymbol{\mu}^{\bar{f}}$ and $\boldsymbol{C}^{\bar{f}}$ represent, respectively, the estimated mean and covariance of the unknown function evaluated at the measured location vector $\boldsymbol{u}_k^{\bar{f}}$. For the measurement update step, the joint prior is factorized by conditioning over $\bar{\boldsymbol{f}}$ i.e. $p(\bar{\boldsymbol{f}}, \boldsymbol{f}|\boldsymbol{z}_{1:k-1}) = p(\bar{\boldsymbol{f}}|\boldsymbol{z}_{1:k-1}) \cdot p(\boldsymbol{f}|\bar{\boldsymbol{f}}, \boldsymbol{z}_{1:k-1})$. The updated distributions are given below:

$$\begin{aligned} p(\boldsymbol{z}_k|\bar{\boldsymbol{f}}) \cdot p(\bar{\boldsymbol{f}}|\boldsymbol{z}_{1:k-1}) \cdot p(\boldsymbol{f}|\bar{\boldsymbol{f}}, \boldsymbol{z}_{1:k-1}) &= p(\bar{\boldsymbol{f}}|\boldsymbol{z}_{1:k}) p(\boldsymbol{f}|\bar{\boldsymbol{f}}, \boldsymbol{z}_{1:k-1}) \\ &= \mathcal{N}(\bar{\boldsymbol{f}}, \sigma^2\boldsymbol{I}) \cdot \mathcal{N}(\tilde{\boldsymbol{\mu}}_k^{\bar{f}}, \tilde{\boldsymbol{C}}_k^{\bar{f}}) p(\boldsymbol{f}|\bar{\boldsymbol{f}}, \boldsymbol{z}_{1:k-1}), \qquad (10.8) \end{aligned}$$

where the distribution of $p(\boldsymbol{z}_k|\bar{\boldsymbol{f}})$ is obtained using (2.16) and $\bar{\boldsymbol{f}} = \boldsymbol{f}(\boldsymbol{u}_k^{\bar{f}})$. The Kalman filter equations (see Section 3.3.1) [45] are used to determine $p(\bar{\boldsymbol{f}}|\boldsymbol{z}_{1:k}) = \mathcal{N}(\hat{\boldsymbol{\mu}}_k^{\bar{f}}, \hat{\boldsymbol{C}}_k^{\bar{f}})$

as given below:

$$\boldsymbol{M}_k = \tilde{\boldsymbol{C}}_k^{\bar{f}}(\tilde{\boldsymbol{C}}_k^{\bar{f}} + \sigma^2 \boldsymbol{I})^{-1}, \tag{10.9}$$

$$\hat{\boldsymbol{\mu}}_k^{\bar{f}} = \tilde{\boldsymbol{\mu}}_k^{\bar{f}} + \boldsymbol{M}_k(z_k - \tilde{\boldsymbol{\mu}}_k^{\bar{f}}), \tag{10.10}$$

$$\hat{\boldsymbol{C}}_k^{\bar{f}} = \tilde{\boldsymbol{C}}_k^{\bar{f}} - \boldsymbol{M}_k \tilde{\boldsymbol{C}}_k^{\bar{f}}, \tag{10.11}$$

where $\hat{}$ represents the estimated variables σ^2 is the measurement noise variance hyperparameter and $\boldsymbol{M}_k$ is the Kalman gain. The product of Gaussians is used to calculate $p(\bar{\boldsymbol{f}}|z_{1:k})p(\boldsymbol{f}|\bar{\boldsymbol{f}}, z_{1:k-1}) = \mathcal{N}\left(\begin{bmatrix}\hat{\boldsymbol{\mu}}_k^{f}\\ \hat{\boldsymbol{\mu}}_k^{\bar{f}}\end{bmatrix}, \begin{bmatrix}\hat{\boldsymbol{C}}_k^{f} & \boldsymbol{C}_k^{f\bar{f}}\\ \boldsymbol{C}_k^{\bar{f}f} & \hat{\boldsymbol{C}}_k^{\bar{f}}\end{bmatrix}\right)$ as given below:

$$\boldsymbol{G}_k = \hat{\boldsymbol{C}}_{k-1}^{f} \boldsymbol{J}_k^T \left(\tilde{\boldsymbol{C}}_k^{\bar{f}} + \sigma^2 \boldsymbol{I}\right)^{-1}, \tag{10.12}$$

$$\hat{\boldsymbol{\mu}}_k^{f} = \hat{\boldsymbol{\mu}}_{k-1}^{f} + \boldsymbol{G}_k(z_k - \tilde{\boldsymbol{\mu}}_k^{\bar{f}}), \tag{10.13}$$

$$\hat{\boldsymbol{C}}_k^{f} = \hat{\boldsymbol{C}}_{k-1}^{f} - \boldsymbol{G}_k \boldsymbol{J}_k \hat{\boldsymbol{C}}_{k-1}^{f}, \tag{10.14}$$

$$\boldsymbol{C}_k^{f\bar{f}} = \hat{\boldsymbol{C}}_{k-1}^{f} \boldsymbol{J}_k^T \left(\tilde{\boldsymbol{C}}_k^{\bar{f}}\right)^{-1} \hat{\boldsymbol{C}}_k^{\bar{f}}, \tag{10.15}$$

where $\boldsymbol{G}_k$ is the gain matrix and $\boldsymbol{C}_k^{f\bar{f}}$ is the cross-covariance between $\boldsymbol{f}$ and $\bar{\boldsymbol{f}}$.

10.2.3 Recursive Gaussian Process with Online Learning

This section summarizes the recursive Gaussian process with online learning proposed in [40]. Consider the GP described by (2.1) and (2.16). The hyperparameters are assumed unknown and learned online within the state space framework:

$$\begin{bmatrix}\boldsymbol{f}_k\\ \boldsymbol{\theta}_k\\ \bar{\boldsymbol{f}}_k\end{bmatrix} = \boldsymbol{A}_k \begin{bmatrix}\boldsymbol{f}_{k-1}\\ \boldsymbol{\theta}_{k-1}\end{bmatrix} + \boldsymbol{\nu}_k, \tag{10.16}$$

$$\boldsymbol{y}_k = \boldsymbol{A}_k \boldsymbol{x}_{k-1} + \boldsymbol{\nu}_k, \tag{10.17}$$

$$\boldsymbol{A}_k = \begin{bmatrix}\boldsymbol{I} & \boldsymbol{O}\\ \boldsymbol{O} & \boldsymbol{I}\\ \boldsymbol{J}_k & \boldsymbol{O}\end{bmatrix}, \quad \boldsymbol{\nu}_k \sim \mathcal{N}(\boldsymbol{\mu}_k^{\nu}, \boldsymbol{C}_k^{\nu}), \tag{10.18}$$

where $\boldsymbol{x}_k$ denotes the state vector, $\boldsymbol{y}_k$ is an augmented vector composed of the state vector and the function prediction at the measured locations, $\boldsymbol{\theta}_k = [\boldsymbol{\eta}_k, \sigma_k]^T$ represents the hyperparameters vector, $\boldsymbol{\eta}_k$ denotes the hyperparameters vector of the

GP mean and covariance functions, $\boldsymbol{f}$ and $\bar{\boldsymbol{f}}$ are, respectively, the function evaluation at the inducing points and measured locations, $\boldsymbol{A}_k$ is the state update matrix, $\boldsymbol{O}$ denotes a zero vector / matrix of appropriate dimensions, $\boldsymbol{v}_k$ is the additive Gaussian noise vector with i.i.d. elements and with the following parameters:

$$\boldsymbol{\mu}_k^{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{O} \\ \boldsymbol{b}_k \end{bmatrix}, \quad \boldsymbol{C}_k^{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{B}_k \end{bmatrix}, \tag{10.19}$$

where $\boldsymbol{b}_k = \boldsymbol{m}(\boldsymbol{u}_k^{\bar{\boldsymbol{f}}}) - \boldsymbol{J}_k \boldsymbol{m}(\boldsymbol{u})$ and $\boldsymbol{J}_k$ and $\boldsymbol{B}_k$ are given by (10.4) and (10.5), respectively. The RGP* recursion begins with s sigma points selection around the mean hyperparameters vector $\hat{\boldsymbol{\mu}}_{k-1}^{\boldsymbol{\theta}}$. In [40], the unscented transform [46] is used for sigma points selection. Given an i^{th} sigma point $\boldsymbol{\theta}_k^i$ with weight w_k^i, the predicted state vector and covariance matrix corresponding to the i^{th} sigma point are given below:

$$\boldsymbol{\mu}_k^{\boldsymbol{y}_i} = \boldsymbol{A}_k \Big|_{\boldsymbol{\theta}_k^i} \begin{bmatrix} \hat{\boldsymbol{\mu}}_{k-1}^{\boldsymbol{f}} + \boldsymbol{S}_k(\boldsymbol{\theta}_k^i - \hat{\boldsymbol{\mu}}_{k-1}^{\boldsymbol{\theta}}) \\ \boldsymbol{\theta}_k^i \end{bmatrix} + \boldsymbol{\mu}_k^{\boldsymbol{v}} \Big|_{\boldsymbol{\theta}_k^i}, \tag{10.20}$$

$$\boldsymbol{C}_k^{\boldsymbol{y}_i} = \boldsymbol{A}_k \Big|_{\boldsymbol{\theta}_k^i} \begin{bmatrix} \hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{f}} - \boldsymbol{S}_k \hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{\theta f}} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \boldsymbol{A}_k^T \Big|_{\boldsymbol{\theta}_k^i} + \boldsymbol{C}_k^{\boldsymbol{v}} \Big|_{\boldsymbol{\theta}_k^i}, \tag{10.21}$$

$$\boldsymbol{S}_k = \hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{f\theta}} (\hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{\theta}})^{-1}, \tag{10.22}$$

where $\boldsymbol{\mu}_k^{\boldsymbol{y}_i}$ and $\boldsymbol{C}_k^{\boldsymbol{y}_i}$ represent, respectively, the predicted state mean and the predicted state error covariance corresponding to the i^{th} sigma point, $\hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{\theta}}$ represents the estimated covariance of the hyperparameters vector, $\hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{\theta f}}$ and $\hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{f\theta}}$ are the estimated covariance matrices between the hyperparameters vector and the sparse GP representation of $\boldsymbol{f}$ and $\cdot\Big|_{\boldsymbol{\alpha}}$ denotes the vector / matrix that is evaluated at $\boldsymbol{\alpha}$. The combined prediction variables are calculated next, as follows:

$$\boldsymbol{\mu}_k^{\boldsymbol{y}} = \sum_{i=1}^{s} w_k^i \boldsymbol{\mu}_k^{\boldsymbol{y}_i}, \tag{10.23}$$

$$\boldsymbol{C}_k^{\boldsymbol{y}} = \sum_{i=1}^{s} w_k^i ((\boldsymbol{\mu}_k^{\boldsymbol{y}_i} - \boldsymbol{\mu}_k^{\boldsymbol{y}})(\boldsymbol{\mu}_k^{\boldsymbol{y}_i} - \boldsymbol{\mu}_k^{\boldsymbol{y}})^T + \boldsymbol{C}_k^{\boldsymbol{y}_i}). \tag{10.24}$$

The state vector $\boldsymbol{x}_k$ is decomposed into an observed, $\boldsymbol{o}_k = [\sigma_k, (\bar{\boldsymbol{f}}_k)^T]^T$, and unobserved vector, $\boldsymbol{u}_k = [(\boldsymbol{f}_k)^T, \boldsymbol{\eta}_k^T]^T$. The mean vector and covariance matrix corresponding to this decomposition are:

$$\boldsymbol{\mu}_k^{\boldsymbol{y}} = \begin{bmatrix} \boldsymbol{\mu}_{k-1}^{\boldsymbol{u}} \\ \tilde{\boldsymbol{\mu}}_k^{\boldsymbol{o}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_{k-1}^{\boldsymbol{f}} \\ \boldsymbol{\mu}_{k-1}^{\boldsymbol{\eta}} \\ \mu_{k-1}^{\sigma} \\ \tilde{\boldsymbol{\mu}}_k^{\bar{\boldsymbol{f}}} \end{bmatrix}, \tag{10.25}$$

$$\boldsymbol{C}_k^{\boldsymbol{y}} = \begin{bmatrix} \boldsymbol{C}_{k-1}^{\boldsymbol{u}} & \tilde{\boldsymbol{C}}_k^{\boldsymbol{uo}} \\ \tilde{\boldsymbol{C}}_k^{\boldsymbol{ou}} & \tilde{\boldsymbol{C}}_k^{\boldsymbol{o}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{C}_{k-1}^{\boldsymbol{f}} & \boldsymbol{C}_{k-1}^{\boldsymbol{f\eta}} & \boldsymbol{C}_{k-1}^{\boldsymbol{f}\sigma} & \tilde{\boldsymbol{C}}_k^{\boldsymbol{f}\bar{\boldsymbol{f}}} \\ \boldsymbol{C}_{k-1}^{\boldsymbol{\eta f}} & \boldsymbol{C}_{k-1}^{\boldsymbol{\eta}} & \boldsymbol{C}_{k-1}^{\boldsymbol{\eta}\sigma} & \tilde{\boldsymbol{C}}_k^{\boldsymbol{\eta}\bar{\boldsymbol{f}}} \\ \boldsymbol{C}_{k-1}^{\sigma\boldsymbol{f}} & \boldsymbol{C}_{k-1}^{\sigma\boldsymbol{\eta}} & C_{k-1}^{\sigma} & \tilde{\boldsymbol{C}}_k^{\sigma\bar{\boldsymbol{f}}} \\ \tilde{\boldsymbol{C}}_k^{\bar{\boldsymbol{f}}\boldsymbol{f}} & \tilde{\boldsymbol{C}}_k^{\bar{\boldsymbol{f}}\boldsymbol{\eta}} & \tilde{\boldsymbol{C}}_k^{\bar{\boldsymbol{f}}\sigma} & \tilde{\boldsymbol{C}}_k^{\bar{\boldsymbol{f}}} \end{bmatrix}, \tag{10.26}$$

where $\boldsymbol{\mu}$ and μ represent, respectively, the mean of a vector and a scalar, $\boldsymbol{C}^{\boldsymbol{a}}$ is the covariance matrix of vector $\boldsymbol{a}$, $\boldsymbol{C}^{\boldsymbol{ab}}$ is the cross-covariance between vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, $\boldsymbol{C}^{a\boldsymbol{b}}$ and $\boldsymbol{C}^{\boldsymbol{b}a}$ denotes the covariance between scalar a and vector b and C^{a} is the variance of the scalar variable a. The measurement model (2.16) is reformulated, to make σ explicit, as given below:

$$z_k^m = f(u_k^m) + \sigma\beta_k^m, \quad \beta \sim \mathcal{N}(0,1), \tag{10.27}$$

to make σ explicit. In (10.27), β is assumed to be a random variable sampled from the standard Gaussian distribution (which is with a zero mean and covariance equal to 1) scaled by σ, z_k^m and u_k^m are, respectively, the m^{th} measurement of $\boldsymbol{z}_k$ and its corresponding input location, σ is the noise variance (hyperparameter). The β noise is assumed uncorrelated with σ. The mean and the covariances corresponding to the model (10.27) are given below:

$$\boldsymbol{\mu}_k^{\boldsymbol{z}} = \tilde{\boldsymbol{\mu}}_k^{\bar{\boldsymbol{f}}}, \tag{10.28}$$

$$\boldsymbol{C}_k^{\boldsymbol{z}} = \tilde{\boldsymbol{C}}_k^{\bar{\boldsymbol{f}}} + \mathbb{V}[\sigma] + \mathbb{E}[\sigma]^2 = \tilde{\boldsymbol{C}}_k^{\bar{\boldsymbol{f}}} + C_{k-1}^{\sigma} + (\mu_{k-1}^{\sigma})^2, \tag{10.29}$$

$$\boldsymbol{C}_k^{\boldsymbol{oz}} = \mathbb{C}[\boldsymbol{o}_k, \boldsymbol{z}_k] = \mathbb{C}[\boldsymbol{o}_k, \bar{\boldsymbol{f}}], \tag{10.30}$$

where $\mathbb{C}[\cdot]$ and $\mathbb{V}[\cdot]$ represent, respectively, a covariance and variance function. It is assumed that $\boldsymbol{o}_k$ and $\bar{\boldsymbol{f}}$ are jointly Gaussian. The KF is used to update the observed and unobserved states as given below:

$$\hat{\boldsymbol{\mu}}_k^{\boldsymbol{o}} = \tilde{\boldsymbol{\mu}}_k^{\boldsymbol{o}} + \boldsymbol{E}_k(\boldsymbol{z}_k - \boldsymbol{\mu}_k^{\boldsymbol{z}}), \tag{10.31}$$

$$\hat{\boldsymbol{C}}_k^{\boldsymbol{o}} = \tilde{\boldsymbol{C}}_k^{\boldsymbol{o}} - \boldsymbol{E}_k\boldsymbol{C}_k^{\boldsymbol{z}}\boldsymbol{E}_k^T, \tag{10.32}$$

$$\boldsymbol{\mu}_k^{\boldsymbol{u}} = \boldsymbol{\mu}_{k-1}^{\boldsymbol{u}} + \boldsymbol{L}_k(\hat{\boldsymbol{\mu}}_k^{\boldsymbol{o}} - \tilde{\boldsymbol{\mu}}_k^{\boldsymbol{o}}), \tag{10.33}$$

$$\boldsymbol{C}_k^{\boldsymbol{u}} = \boldsymbol{C}_{k-1}^{\boldsymbol{u}} + \boldsymbol{L}_k(\hat{\boldsymbol{C}}_k^{\boldsymbol{o}} - \tilde{\boldsymbol{C}}_k^{\boldsymbol{o}})\boldsymbol{L}_k^T, \tag{10.34}$$

where $\tilde{\boldsymbol{G}}_k = \boldsymbol{C}_k^{\boldsymbol{o}\bar{\boldsymbol{f}}}(\boldsymbol{C}_k^{\bar{\boldsymbol{f}}})^{-1}$ and $\boldsymbol{L}_k = \boldsymbol{C}_k^{\boldsymbol{uo}}(\boldsymbol{C}_k^{\boldsymbol{o}})^{-1}$ are the gain matrices, $\boldsymbol{\mu}^{\boldsymbol{g}}$ and $\boldsymbol{C}^{\boldsymbol{g}}$ represent, respectively, the mean and the covariance of the vector $\boldsymbol{g}$. Finally the state vector and the covariance are updated as given below:

$$\hat{\boldsymbol{\mu}}_k^{\boldsymbol{x}} = \begin{bmatrix} \hat{\boldsymbol{\mu}}_k^{\boldsymbol{u}} \\ \boldsymbol{h}\hat{\boldsymbol{\mu}}_k^{\boldsymbol{o}} \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{\mu}}_k^{\boldsymbol{f}} \\ \hat{\boldsymbol{\mu}}_k^{\boldsymbol{\theta}} \end{bmatrix}, \tag{10.35}$$

$$\hat{\boldsymbol{C}}_k^{\boldsymbol{x}} = \begin{bmatrix} \hat{\boldsymbol{C}}_k^{\boldsymbol{u}} & \boldsymbol{L}_k\hat{\boldsymbol{C}}_k^{\boldsymbol{o}}\boldsymbol{h} \\ \boldsymbol{h}^T\hat{\boldsymbol{C}}_k^{\boldsymbol{o}}\boldsymbol{L}_k^T & \boldsymbol{h}^T\hat{\boldsymbol{C}}_k^{\boldsymbol{o}}\boldsymbol{h} \end{bmatrix}, \tag{10.36}$$

where $\boldsymbol{h}^T = [1, \boldsymbol{O}_{1\times N_k}]$ and $\boldsymbol{O}_{1\times N_k}$ represents a $1 \times N_k$-dimensional zero vector and N_k is the number of measurements received at k.

10.3 The Proposed Data-Driven Recursive Tracking Approach

A Gaussian process approach for point target tracking namely GPMT is proposed in Chapter 8 [4]. The GPMT deals with two dimensional (x and y) data. It can be extended to any number of dimensions in a similar way. In target tracking, the higher order position derivatives are also estimated. A first order GPMT (FO-GPMT[2]) is also proposed in Chapter 8 [4] for the velocity estimation. The same approach can be extended for estimating other higher order derivatives.

Importance of Learning A fixed set of hyperparameters allows tracking of a limited set of target trajectories. The hyperparameters need to be learned for tracking a relatively wide set of target trajectories. The GPMT of Chapter 8 [4] is proposed using a maximum likelihood (ML) based approach for hyperparameter learning. The computational cost is high and the GPMT processing is not in real-time. A recursive GPMT (RGP*MT) is proposed in this chapter to reduce the computational time of the learning process. As a result, the RGP*MT provides real-time target state estimation. In this chapter, the GPMT is extended for online learning.

[2] The prefix FO is removed from hereon for brevity.

10.3.1 Recursive Gaussian Process Motion Tracker (RGP*MT)

The RGP*MT is an online state estimation approach for point target tracking. Consider again equation (8.6) and the derivations for the x coordinate are presented in this section. Similar derivations hold for y, and other output dimensions in a similar way. The GPMT assumptions, **T1**-**T5**, are also valid for the RGP*MT. It is further assumed that the function values at the d most recent measurement samples are represented by a set of d inducing points. This GP representation is different from the sparse GP of [40]. A bounded input domain is assumed in [40], whereas the input dimension for the proposed approach is right unbounded. Using **T3**, the input locations of the inducing points are chosen to be the same as that of the d most recent measurements and the input locations of distant past are ignored.

The target is assumed to follow an unknown nonlinear trajectory from time $t = 0$ to $t = t_{max}$. The input domain is sampled using a fixed equidistant grid also called inducing points. Unlike [40] where multiple measurements are received at random input locations, the measurements are received in a sequence starting at $t = 0$ and going towards t_{max}. Moreover, a single measurement is received per sample. In order to keep the computational complexity low and using **T3**, a smaller set of inducing points (near the measured location) is considered for the state prediction and filtering. In this way, the inducing points represent the nonlinear function in the near vicinity of the measured location. This reduced set of inducing points is then used for the functional prediction and filtering at the measured location. The processing time of the training was reduced in the same way in GPMT [4]. However, the locations of the inducing points can be chosen differently from the measured locations. This is not possible in the GPMT [4]. Another advantage is that the location of the inducing points can be learned at each measurement sample for optimal representation of the slice (of width $d + 1$) of the underlying nonlinear function. As a result, the number of inducing points can be reduced when the target motion is uniform. Conversely, when it is exhibiting sharp maneuvers, the inducing points can be increased. The whole process also employs online learning of the hyperparameters which makes the approach highly adaptive.

The inducing points are updated at each time step. Consider the prior inducing points set locations at k are represented by the d-dimensional prior input vector $\boldsymbol{t}_k = [k - d, k - d + 1, \cdots, k - 1]^T$. The inducing points store an estimate of the nonlinear function f^x in the vector $\boldsymbol{f}_k^x = \boldsymbol{f}^x(\boldsymbol{t}_k)$. The prior inducing points are the training data for the function prediction at $\boldsymbol{t}_k$ and k. The measurement z_k^x, received at k, updates the set of inducing points and the function prediction at the measured locations. The update of the function at the measured location k is the position estimate of the target. The position estimate at k, evaluated using the inducing point set and the measurement, is included in the inducing point set. The oldest, in terms of time, the inducing point is removed and a new point is added to the inducing point. As a result, the total number of inducing points is kept fixed. The new point is the function estimate at k denoted as $\bar{f}_k^x = f^x(k)$. The recursion starts at $k = d + 1$ sample. The initial distribution of the inducing points set and the hyperparameters vector is determined using first d measurement samples and is given below:

$$p(\boldsymbol{f}_d^x) = \mathcal{N}(\boldsymbol{\mu}_d^{\boldsymbol{f}^x}, \boldsymbol{C}_d^{\boldsymbol{f}^x} = \boldsymbol{K}_{\boldsymbol{t}_d \boldsymbol{t}_d} + (\sigma_d^x)^2 \boldsymbol{I}_d), \tag{10.37}$$

$$p(\boldsymbol{\theta}_d^x) = \mathcal{N}(\boldsymbol{\mu}_d^{\boldsymbol{\theta}^x}, \boldsymbol{C}_d^{\boldsymbol{\theta}^x}), \tag{10.38}$$

where $\boldsymbol{\theta}_k^x$ represents the hyperparameters vector. The prior covariance $\boldsymbol{C}_d^{\boldsymbol{f}^x}$ differs from that proposed in [40] since the inducing points are initialized and recursively learned from noisy measurements. However, the inducing points are not learned exactly due to the measurement noise and the uncertainty in the inducing points is represented with the following assumptions:

B1 The inducing points noise is modeled as an additive random noise vector whose elements are i.i.d. with variance $(\sigma_d^x)^2$.

B2 The variance of the inducing point is of the order of the measurement noise variance.

As a result of the above assumptions, (10.4) is modified to:

$$\boldsymbol{J}_k' = \boldsymbol{K}_{\boldsymbol{u}_k^{\bar{f}} \boldsymbol{u}} (\boldsymbol{K}_{\boldsymbol{uu}} + (\sigma_k^x)^2 \boldsymbol{I}_d)^{-1}. \tag{10.39}$$

The initial value of the mean vector $\boldsymbol{\mu}_d^{\boldsymbol{\theta}^x}$ is found with the maximum likelihood estimation based hyperparameters optimization with the first d measurements. The inducing point vector and the hyperparameters vector are combined to define the state vector $\boldsymbol{x}^x = [(\boldsymbol{f}^x)^T, (\boldsymbol{\theta}^x)^T]^T$ and $\boldsymbol{\theta}^x = [\sigma^x, (\boldsymbol{\eta}^x)^T]^T$. The state space model is given by (10.16) and (10.17). For the recursive update of the inducing points, the prior at k is assumed to be $p(\boldsymbol{x}_{k-1}^x | \boldsymbol{z}_{1:k-1}^x) = \mathcal{N}(\hat{\boldsymbol{\mu}}_{k-1}^{\boldsymbol{x}^x}, \hat{\boldsymbol{C}}_{k-1}^{\boldsymbol{x}^x})$ and that given the joint distribution of the inducing points at $\boldsymbol{t}_k$ and k, the likelihood of the new measurement is independent of the previous measurements. The posterior $p(\boldsymbol{x}_k^x | \boldsymbol{z}_{1:k}^x)$ can be updated using the decomposition [118] of the state into the observable

$\boldsymbol{o}_k^x = [\sigma_k^x, (\bar{\boldsymbol{f}}_k^x)^T]^T$ and unobservable $\boldsymbol{u}_k^x = [(\boldsymbol{f}_k^x)^T, (\boldsymbol{\eta}_k^x)^T]^T$ parts:

$$p(\boldsymbol{x}_k^x|\boldsymbol{z}_{1:k}^x) = \int p(\boldsymbol{u}_k^x|\boldsymbol{o}_k^x)p(\boldsymbol{o}_k^x|\boldsymbol{z}_{1:k})d\bar{\boldsymbol{f}}_k^x, \tag{10.40}$$

$$\boldsymbol{o}_k^x = [\sigma_k^x, (\bar{\boldsymbol{f}}_k^x)^T]^T, \tag{10.41}$$

$$\boldsymbol{u}_k^x = [(\boldsymbol{f}_k^x)^T, (\boldsymbol{\eta}_k^x)^T]^T, \tag{10.42}$$

where $\boldsymbol{o}_k^x$, and $\boldsymbol{u}_k^x$ are, respectively, the observed and unobserved parts.

The above inference is done using the RGP* described in Section 10.2.3. The sigma points are chosen using the constrained unscented transform [119], which is different from [40]. The constraints are applied on the hyperparameters to remain positive i.e. > 0 at all times. The RGP* recursion is adopted using the following equivalence to give the RGP* motion tracker (RGP*MT):

$$\boldsymbol{u} = \boldsymbol{t}_k, \quad \boldsymbol{u}_k^{\bar{f}} = k, \quad \hat{\boldsymbol{x}}_k = \boldsymbol{x}_k^x, \quad \bar{\boldsymbol{f}}_k = \bar{\boldsymbol{f}}_k^x, \tag{10.43}$$

$$\boldsymbol{z}_k = \boldsymbol{z}_k^x, \quad \boldsymbol{m}(k) = 0, \quad \boldsymbol{m}(\boldsymbol{t}_k) = \boldsymbol{O}_{d\times 1}, \quad \boldsymbol{J}_k = \boldsymbol{J}_k'. \tag{10.44}$$

Using (10.43) and (10.44) and the prior distribution of $\boldsymbol{x}^x$, the posterior is estimated using (10.16)-(10.36). The mean $\hat{\mu}_k^{\bar{f}^x}$ and variance $\hat{C}_k^{\bar{f}^x}$ of the function at k are given by modifying the update equations (10.35) and (10.36), as given below:

$$\hat{\mu}_k^{\bar{f}^x} = \boldsymbol{h}'^T\hat{\boldsymbol{\mu}}_k^{\boldsymbol{o}}, \quad \boldsymbol{h}'^T = \mathbf{1}_{N_k+1} - \boldsymbol{h}^T, \tag{10.45}$$

$$\hat{C}_k^{\bar{f}^x} = \boldsymbol{h}'^T\hat{\boldsymbol{C}}_k^{\boldsymbol{o}}\boldsymbol{h}', \tag{10.46}$$

$$\boldsymbol{C}_k^{\boldsymbol{f}^x\bar{f}^x} = \boldsymbol{h}'^T\hat{\boldsymbol{C}}_k^{\boldsymbol{o}}\boldsymbol{L}_k^T, \tag{10.47}$$

where $\boldsymbol{C}_k^{\boldsymbol{f}^x\bar{f}^x}$ represents the cross-covariance vector between the inducing points set and the estimate at the new input location and $\mathbf{1}_a$ is a $1 \times a$ unit vector. Let the posterior mean and covariance given by (10.35) and (10.36) are:

$$\mathring{\mu}_k^{\boldsymbol{f}^x} = \hat{\boldsymbol{\mu}}_k^{\boldsymbol{f}}, \quad \mathring{C}_k^{\boldsymbol{f}^x} = \hat{\boldsymbol{C}}_k^{\boldsymbol{f}}, \quad \mathring{\boldsymbol{C}}_k^{\boldsymbol{f}^x\bar{f}^x} = \boldsymbol{C}_k^{\boldsymbol{f}\bar{f}}. \tag{10.48}$$

Then the posterior distribution of the inducing points can be determined from this posterior mean and covariance. Let the elements of the updated inducing point set, the covariance matrix and the cross-covariance vector be represented as $\mathring{\boldsymbol{\mu}}_k^{\boldsymbol{f}_i^x}$, $\mathring{\boldsymbol{C}}_k^{\boldsymbol{f}_{i,i}^x}$ and $\mathring{\boldsymbol{C}}_k^{\boldsymbol{f}_i^x\bar{f}^x}$, respectively, where $i = \{1, 2, \cdots, d\}$. The most recent inducing point is denoted with the index d. The following operations are performed on the updated inducing points set to get the modified set and to also keep the number of elements constant:

$$\hat{\mu}_k^{f_i^x} = \begin{cases} \mathring{\mu}_k^{f_{i+1}^x} & \text{if } i = 1, 2, \cdots, d-1, \\ \hat{\mu}_k^{\bar{f}^x} & \text{if } i = d, \end{cases} \tag{10.49}$$

$$\hat{C}_k^{f_{i,j}^x} = \begin{cases} \mathring{C}_k^{f_{i+1,j+1}^x} & \text{if } i, j = \{1, \cdots, d-1\}, \\ \mathring{C}_k^{f_{i+1}^x \bar{f}^x} & \text{if } i = \{1, \cdots, d-1\}, j = d, \\ \mathring{C}_k^{f_{j+1}^x \bar{f}^x} & \text{if } i = d, j = \{1, \cdots, d-1\}, \\ \hat{C}_k^{\bar{f}^x} & \text{if } i = j = d. \end{cases} \tag{10.50}$$

In (10.49)-(10.50) the oldest inducing point is removed and a new inducing point is added, using the estimates and the corresponding cross-covariance matrix of the current time step. The state vector $\boldsymbol{x}$ defined in (10.16) and (10.17) is built using the new inducing point set and the updated hyperparameters vector. The steps from (10.16)-(10.36) are repeated for the estimates at the next sample. The RGP*MT recursion is explained in Fig. 10.1. The target trajectory (solid red line) is sampled at 8 points and the prediction is done for $k = 9$. The GPMT uses $d = 6$ measurement samples (black dots), $3 \leq k \leq 8$, as training data. It predicts the trajectory at the 9^{th} sample shown as a black empty circle. The RGP*MT uses $d = 6$ inducing points (blue encircled plus) to predict the trajectory shown as a blue plus. The RGP*MT filtering step at $k = 9$, updates the inducing points based on the new measurement sample.

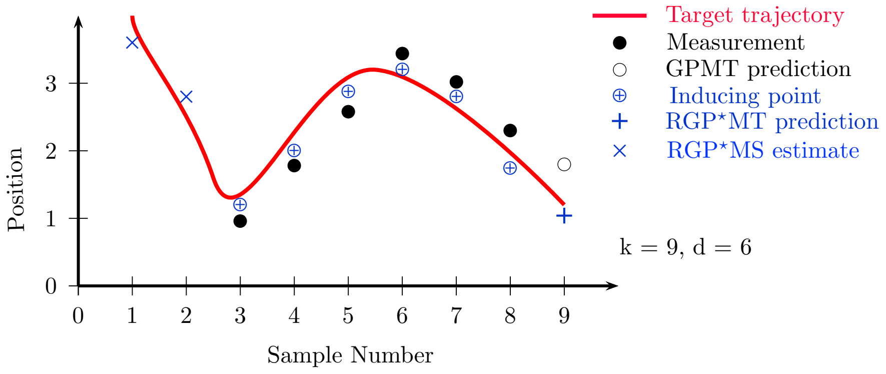


Fig. 10.1 **GPMT and RGP*MT approaches** The figure illustrates the GPMT, RGP*MT and RGP*M smoother approaches for output at $k = 9$ using a sliding window width $d = 6$.

10.3.2 Recursive Derivative of Gaussian Process* Motion Tracker

The recursion for a DGP has not been proposed in the literature. In this subsection, a recursive estimation of the first order derivative of the x position is proposed. In a similar way, the higher order derivatives of x and the time derivatives for other coordinates can be derived. This formulation is generic and can be applied to any derivative of a GP regression. Consider the GP defined in (8.6). The DGP is given below:

$$\frac{dx}{dt} = \frac{df^x(t)}{dt} = \dot{f}^x(t), \tag{10.51}$$

A derivative of a GP is also a GP [39]. As described in Section 10.3.1, the input domain is sampled using a fixed grid of inducing points. The location of the inducing points is chosen the same as for the recursive derivative of the Gaussian process* motion tracker (RDGP*MT). Consider the input vector of the inducing points is $\boldsymbol{t}_k$, the local estimates of the DGP are stored in the inducing points given below:

$$\dot{\boldsymbol{f}}^x = \dot{\boldsymbol{f}}^x(\boldsymbol{t}). \tag{10.52}$$

Consider the following initial distribution of the DGP:

$$p_d(\dot{\boldsymbol{f}}^x) = \mathcal{N}(\boldsymbol{\mu}_d^{\dot{f}^x}, \boldsymbol{C}_d^{\dot{f}^x} = \boldsymbol{K}_{\boldsymbol{t}_d \boldsymbol{t}_d} + (\sigma_d^{\dot{x}})^2 \boldsymbol{I}_d). \tag{10.53}$$

The posterior distribution is updated recursively using the Bayes law and the Chapman-Kolmogorov equations as given below:

$$p(\dot{\boldsymbol{f}}_k^x | \dot{\boldsymbol{z}}_{1:k}^x) = \frac{\int p(\dot{z}_k^x | \bar{\dot{f}}_k^x, \dot{\boldsymbol{f}}_k^x) p(\bar{\dot{f}}_k^x, \dot{\boldsymbol{f}}_k^x | \dot{\boldsymbol{z}}_{1:k-1}^x) d\bar{\dot{f}}_k^x}{c_k}, \tag{10.54}$$

where $\dot{z}$ is the measurement, $\bar{\dot{f}}_k^x$ represents the function evaluation at the measured location and c_k is the normalization constant. It is assumed that the hyperparameters have been learned during the position filtering step i.e. RGP*MT. Hence, the posterior update is independent of the hyperparameters. The DGP inducing points and the velocity estimation at time k is done recursively following the RGP formulation given in Section 10.2.1 and the following equivalence:

$$\boldsymbol{u} = \boldsymbol{t}_k, \quad \boldsymbol{u}_k^{\bar{f}} = k, \quad \boldsymbol{f}_k = \dot{\boldsymbol{f}}_k^x, \quad \bar{\boldsymbol{f}}_k = \bar{\dot{f}}_k^x, \quad \dot{m}(k) = 0,$$

$$\dot{\boldsymbol{m}}(\boldsymbol{t}) = \boldsymbol{O}_{d\times 1}, \quad \dot{z}^x_k = \frac{z^x_k - z^x_{k-1}}{T}, \quad (\dot{\sigma}^x_k)^2 = \frac{2(\sigma^x_k)^2}{T},$$

where and $\dot{z}^x_k$ is a pseudo-measurement with measurement noise variance $(\dot{\sigma}^x_k)^2$. Using the RGP recursion given in Section 10.2.2, the estimated inducing points set and the derivative of the function at k are estimated as described below:

$$\begin{aligned}
\mathring{\boldsymbol{\mu}}^{\dot{\boldsymbol{f}}^x}_k &= \hat{\boldsymbol{\mu}}^{\boldsymbol{f}}_k, \quad \mathring{\boldsymbol{C}}^{\dot{\boldsymbol{f}}^x}_k = \hat{\boldsymbol{C}}^{\boldsymbol{f}}_k, \quad \mathring{\boldsymbol{C}}^{\dot{\boldsymbol{f}}^x \dot{\bar{f}}^x}_k = \boldsymbol{C}^{\boldsymbol{f}\bar{f}}_k, \\
\hat{\mu}^{\dot{\bar{f}}^x}_k &= \hat{\boldsymbol{\mu}}^{\bar{f}}_k, \quad \hat{C}^{\dot{\bar{f}}^x}_k = \hat{\boldsymbol{C}}^{\bar{f}}_k.
\end{aligned}$$

At the end of the recursion, the inducing points are updated, similar to (10.49) and (10.50), as given below:

$$\hat{\boldsymbol{\mu}}^{\dot{\boldsymbol{f}}^x_i}_k = \begin{cases} \mathring{\boldsymbol{\mu}}^{\dot{\boldsymbol{f}}^x_{i+1}}_k & \text{if } i = 1, 2, \cdots, d-1, \\ \hat{\mu}^{\dot{\bar{f}}^x}_k & \text{if } i = d, \end{cases} \tag{10.55}$$

$$\hat{\boldsymbol{C}}^{\dot{\boldsymbol{f}}^x_{i,j}}_k = \begin{cases} \mathring{\boldsymbol{C}}^{\dot{\boldsymbol{f}}^x_{i+1,j+1}}_k & \text{if } i, j = \{1, \cdots, d-1\}, \\ \mathring{\boldsymbol{C}}^{\dot{\boldsymbol{f}}^x_{i+1}\dot{\bar{f}}^x}_k & \text{if } i = \{1, \cdots, d-1\}, j = d, \\ \mathring{\boldsymbol{C}}^{\dot{\boldsymbol{f}}^x_{j+1}\dot{\bar{f}}^x}_k & \text{if } i = d, j = \{1, \cdots, d-1\}, \\ \hat{C}^{\dot{\bar{f}}^x}_k & \text{if } i = j = d. \end{cases} \tag{10.56}$$

An alternative way, and used in Section 10.4, of determining the derivatives is from the inducing points and the learned hyperparameters of RGP*MT. The inducing points and the derivative can be written as the following multivariate Gaussian distribution [115]:

$$\begin{bmatrix} \boldsymbol{f}^x_k \\ \dot{\boldsymbol{f}}^x_k \end{bmatrix} = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{m}(\boldsymbol{t}_k) \\ \delta_{\bar{t}}(m(\bar{t})) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{t}_k\boldsymbol{t}_k} + (\sigma^x_k)^2 \boldsymbol{I}_d & \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\boldsymbol{t}_k\bar{t}}) \\ \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}_k}) & \boldsymbol{\Delta}^2_{\bar{t}\bar{t}}(K_{\bar{t}\bar{t}}) \end{bmatrix} \right), \tag{10.57}$$

The predicted and filtered distributions of the first order derivative process given the inducing points are given below:

$$\begin{aligned}
\tilde{\mu}^{\dot{\bar{f}}^x}_k &= \delta_{\bar{t}}(m(\bar{t})) + \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}_k})(\boldsymbol{K}_{\boldsymbol{t}_k\boldsymbol{t}_k} + (\sigma^x_k)^2 \boldsymbol{I}_d)^{-1}(\mathring{\boldsymbol{\mu}}^{\boldsymbol{f}^x}_k - \boldsymbol{m}(\boldsymbol{t}_k)), &(10.58)\\
\tilde{C}^{\dot{\bar{f}}^x}_k &= \boldsymbol{\Delta}^2_{\bar{t}\bar{t}}(\boldsymbol{K}_{\bar{t}\bar{t}}) - \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}_k})(\boldsymbol{K}_{\boldsymbol{t}_k\boldsymbol{t}_k} + (\sigma^x_k)^2 \boldsymbol{I}_d)^{-1}[\boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}_k})]^T, &(10.59)\\
\hat{\mu}^{\dot{\bar{f}}^x}_k &= \delta_{\bar{t}}(m(\bar{t})) + \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}'_k})(\boldsymbol{K}_{\boldsymbol{t}'_k\boldsymbol{t}'_k} + (\sigma^x_k)^2 \boldsymbol{I}_d)^{-1}(\hat{\boldsymbol{\mu}}^{\boldsymbol{f}^x}_k - \boldsymbol{m}(\boldsymbol{t}'_k)), &(10.60)
\end{aligned}$$

$$\hat{C}_k^{\dot{f}^x} = \boldsymbol{\Delta}^2_{\bar{t}\bar{t}}(\boldsymbol{K}_{\bar{t}\bar{t}}) - \boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}'_k})(\boldsymbol{K}_{\boldsymbol{t}'_k\boldsymbol{t}'_k} + (\sigma_k^x)^2\boldsymbol{I}_d)^{-1}[\boldsymbol{\Delta}_{\bar{t}}(\boldsymbol{K}_{\bar{t}\boldsymbol{t}'_k})]^T. \tag{10.61}$$

10.3.3 Recursive Gaussian Process* Motion Smoother and Recursive Derivative of Gaussian Process* Motion Smoother

This section presents the online smoothing of the GP namely Recursive Gaussian Process* Motion Smoother (RGP*MS) and the derivative of the GP also called Recursive Derivative of Gaussian Process* Motion Smoother (RDGP*MS). The training data for the GPMT prediction and filtering are recent measurements. Comparatively, the RGP*MT prediction is based on the inducing points set. The RGP*MT filtering is done using the inducing points set and the current measurement. The inducing points set is updated during each measurement update. This update, (10.33) and (10.34), is similar to a fixed-lag-recursive-smoother. The smoother lag is d measurement samples long according to (10.49) and (10.50). As shown in [40], this smoother is similar to the augmented Kalman smoother of [120], with an additional measurement update step. The $\hat{\boldsymbol{\mu}}_k^{\boldsymbol{f}_1^x}$ and $\hat{\boldsymbol{C}}_k^{\boldsymbol{f}_{1,1}^x}$ in (10.49) and (10.50) are, respectively, the smoothed position and the corresponding variance evaluated at $\boldsymbol{t}'$. Similarly, $\hat{\boldsymbol{\mu}}_k^{\dot{\boldsymbol{f}}_1^x}$ and $\hat{\boldsymbol{C}}_k^{\dot{\boldsymbol{f}}_{1,1}^x}$ in (10.55) and (10.56) are, respectively, the smoothed velocity and the corresponding variance. As for the derivative Gaussian process, an alternative way of smoothing is through the use of (10.60) and (10.61) and setting $\bar{t} = k - d$ and $\boldsymbol{t}'_k = [k, k-1, \cdots, k-d+1]^T$. The RGP*MS is also explained in Fig. 10.1. The RGP*MS estimates or the smoothed estimate (blue cross) are the inducing points that are no more part of the training data. After processing the 9th sample, the training data at $k = 3$ is removed and the data of sample $k = 9$ is added. The removed inducing point becomes the RGP*MS smoothed estimate.

10.3.4 Measurement Noise Uncertainty Analysis

The proposed RGP*MT and RGP*MS do not require prior knowledge of the measurement noise variances. This is achieved by setting the measurement variance as a hyperparameter as given in (10.41). The measurement noise variance is recursively estimated at each step. As a result, the proposed filter / smoother is proposed to be insensitive to the measurement noise modelling errors, provided the elements of the noise vector are an i.i.d. additive Gaussian random variables. The performance of the proposed approaches is studied using different measurement noise variances in Section 10.4.6 to ascertain these facts.

10.3.5 Sparsity and the Inducing Points

The RGP*MT, RGP*MS, RDGP*MT and RDGP*MS proposed in this chapter extend further the sparse GP [121], for target tracking and with studies on the impact of the noise parameters on the approaches' performance. A typical sparse GP based application has a bounded input domain. The proposed approaches in this chapter are based upon a right unbounded input domain. Secondly, a typical sparse GP based application requires an estimate across the whole input domain. In target tracking, the interest lies at the current and future time i.e. at a specific slice of the input domain. Lastly, in a typical sparse GP application, the training data is available on both sides of the test data. Conversely, in the assumed target filtering problem, the training data is not available on the right side of the input domain.

The proposed approaches introduce sparsity through the inducing points and the parameter d. The inducing points locations are proposed to be the same as those of the measurements. These can be chosen differently from the measurement locations as well. The distance between the input locations of the inducing points cannot be kept too close or too far. The optimal grid points location is proposed as a function of the lengthscale hyperparameter in [122]. Similarly, the accuracy of the proposed approaches is sensitive to the parameter d. Previously, sparse GP methods have been proposed for the automatic selection of the inducing points [123] and choice of the parameter d [124]. A comprehensive review of these methods is given in [125]. In this chapter, the parameter d is selected by trial and error approach. The automatic selection can be introduced to improve the robustness of the approach with respect to d.

10.4 Performance Validation

The performance validation of the proposed approach is done in challenging manoeuvring scenarios explained in this section.

10.4.1 Compared Approaches

The proposed approach is compared with different model-based and model-free filters described below.

(F1) **CV**. A KF with state transition modelled using a NCV [56] model. The process noise variance is set to $q_{NCV} = 10\ m^2/s^2$.

(F2) **Fixed Grid Interacting Multiple Model (FGIMM)**. An FGIMM [51] is implemented using three KFs. The grid consists of an NCV and two coordinated turn models. The turn rates are set to $\{-15, 15\}°/s$. The Markov transition probability of the same mode is set to 0.9 and for changing the mode is 0.05, the initial model probability vector is $\{0.15, 0.7, 0.15\}$ and the process noise variance is set to 26 m^2/s^2 for each model. This is an optimum process noise variance for a target moving with 200 m/sec according to [126].

(F3) **Singer**. A KF designed using a Singer [64] state transition model. The model parameters are set as follows; maximum possible acceleration is set to $a_{max} = 8\ m/s^2$, probability of no-acceleration is set to $P_0 = 0.4$, probability of maximum acceleration is set to $P_{max} = 0.1$ and manoeuvre time constant is set to $\tau_m = 8\ s$.

(F4) **GPMT, FO-GPMT**. A GPMT and FO-GPMT [4] with depth set to $d = 10\ samples$.

(F5) **CGPMT, FOCGPMT**. A constant (hyperparameters) GPMT and FOGPMT with depth set to $d = 10\ samples$. The hyperparameters are initialized using the first d measurement samples and are kept constant there after.

(F6) **RGPMT, RDGPMT**. A filter based on RGP. The learning is ML based and the depth is set to $d = 10\ samples$.

(F7) **RGP*MT, RDGP*MT & RGP*MS, RDGP*MS**. The proposed filter and smoother with depth set to $d = 10\ samples$. The smoother is denoted as **S*7**.

Filters (**F1**), (**F2**) and (**F3**) are chosen to compare the proposed approach against the model-based filtering approaches. Filter (**F4**) is chosen to study the effect of proposed changes in the training (inducing points instead of measurements) and learning (recursive instead of ML). Filter (**F6**), which adopts the same training process as the proposed approach, is introduced to study the performance of different learning approaches, ML in the filter (**F6**) versus recursive in the proposed filter (**F7**). Lastly, filter (**F5**) is introduced to study the performance degradation if the hyperparameters are not learned.

10.4.2 Test Scenarios

The above approaches are compared in the following six challenging scenarios:

(S1) **Uniform**. The target velocity is kept nearly constant. This scenario represents the commonly observed trajectory of airliners.

(S2) **Gradual coordinated Turns**. The target motion is modelled using the left and right coordinated turns (15°/s for 10 s) and the CV motion models. This scenario represents manoeuvrable targets dynamics, less agile than those in scenario (**S3)**.

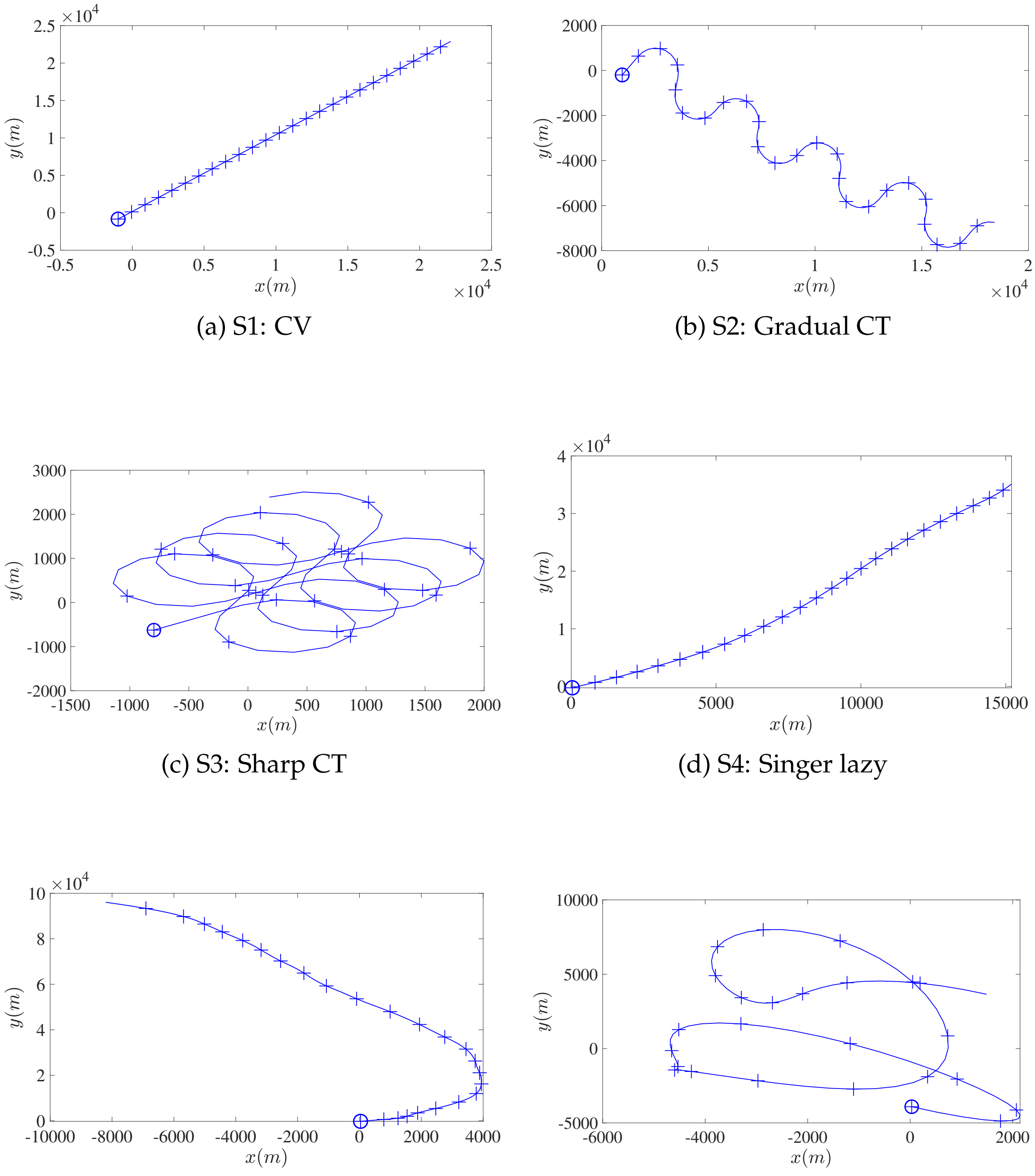
(S3) **Sharp coordinated Turns**. This is similar to scenario (**S2)** except the turn rates are set to (30°/s for 9 s). This scenario represents highly manoeuvrable targets dynamics such as military aircrafts.

(S4) **Singer lazy**. The target motion is modelled using the Singer acceleration model. The parameters are chosen the same as those of the Singer filter (**F3)**.

(S5) **Singer agile**. This scenario is similar to scenario (**S4)**, except the maximum acceleration is increased to $a_{max} = 50\ m/s^2$. As compared to scenario (**S4)**, an agile target is simulated in this test scenario.

(S6) **GP**. The target motion is modelled using two zero mean GPs, each for the x and y coordinate, respectively. The squared exponential covariance kernel is adopted for both GPs. The magnitude of variance is set to $\sigma_{gp}^2 = 10^7\ m^2$ and length-scale is set to $l_{gp} = 10\ s$.

The simulated scenarios cover a wide range of manoeuvring trajectories depicted by the aerial targets. Each scenario is matched, in terms of the structure and the parameters, to at-least one of the filters. This provides a rigorous performance evaluation for each approach. The matrix for the match and mis-match among the filters and the scenarios is depicted in Table 10.1. A sample trajectory of each scenario is shown in Fig. 10.2.



(a) S1: CV

(b) S2: Gradual CT

(c) S3: Sharp CT

(d) S4: Singer lazy

(e) S5: Singer Agile

(f) S6: GP

Fig. 10.2 **Sample trajectory**. The figure shows a sample trajectory of each scenario. The initial position is indicated by a small circle.

Table 10.1 Match Mis-match Matrix

	S1	**S2**	**S3**	**S4**	**S5**	**S6**
F1	■					
F2	■	■	▨			
F3				■	▨	
F4						▨
F5						▨
F6						▨
F7						▨
F8						▨
F9						▨

□ Mis-match ▨ Matched model ■ Matched model and parameters

The scenarios (**S5)** and (**S6)** are highly manoeuvrable target simulations and are not matched to any of the compared filters. The target velocity is initialized randomly in the range $150 \leq v_0 \leq 250\ m/s$ for model-based scenarios (**S1)** to (**S5)**. The total time of trajectory is 100 s. The measurement noise standard deviation is set to $\sigma = 25\ m$. The filters are initialized using the measurement data. The hyperparameters vector is initialized by maximizing the likelihood of first d measurement samples for filters (**F6)** and (**F7)**. The position and velocity root mean square error (RMSE) are plotted in the figures showing performance graphs. The mean-RMSE errors of the position and velocity are given in the performance tables. The RMSE and mean-RMSE in N_{MC} Monte-Carlo simulation runs are evaluated as given below:

$$rmse_q^k = \sqrt{\frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} (q_{i,k} - \hat{q}_{i,k})^2}, \tag{10.62}$$

$$mean\ rmse = \sqrt{\frac{1}{KN_{MC}} \sum_{i=1}^{N_{MC}} \sum_{k=1}^{K} (q_{i,k} - \hat{q}_{i,k})^2}, \tag{10.63}$$

where $rmse_q^k$ and $mean\ rmse$ represent, respectively, the RMSE and the mean RMSE, q_k is the true value, $\hat{q}_k$ is the filter output and K is the total number of samples.

10.4.3 Implementation Details

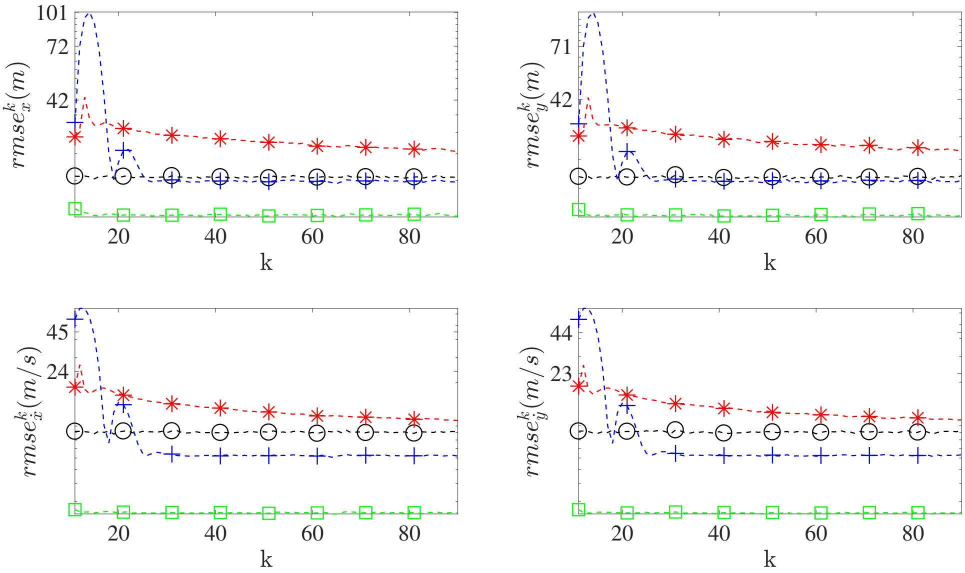
The implementation details of the proposed approach are given in this subsection. The state vector (excluding the hyperparameters) and the measurement data is scaled down at the input and up-scaled at the output of the filter. The scaling is set to $\frac{1}{70}$. The first two inducing points for the DGP are initialized to the same value. The mean hyperparameters vector is initialized using the maximum likelihood of the first d measurement samples. Let the initial hyperparameters vector is given as $\boldsymbol{\mu}_0^{\boldsymbol{\theta}} = [\sigma_m^2, l, \sigma^2]^T$, where σ_m^2 and l are the kernel variance and lengthscale hyperparameters and σ^2 is the noise variance hyperparameter. The correlation between the noise variance hyperparameter and $\boldsymbol{f}(\boldsymbol{u}_k^{\bar{f}})$ is necessary for its learning in (10.31) and (10.32). As proposed in [40], this correlation is attained by correlating the noise variance hyperparameter with the remaining hyperparameters. To achieve this, the terms in the covariance of the hyperparameters relating to the noise variance hyperparameter matrix are set to non-zero values. The initial covariance matrix is given below:

$$\boldsymbol{C}_0^{\boldsymbol{\theta}} = \begin{bmatrix} \sigma_m^2 & 0 & \frac{\sigma^2}{40} \\ 0 & 1 & \frac{\sigma^2}{40} \\ \frac{\sigma^2}{40} & \frac{\sigma^2}{40} & \frac{\sigma^2}{20} \end{bmatrix}. \tag{10.64}$$

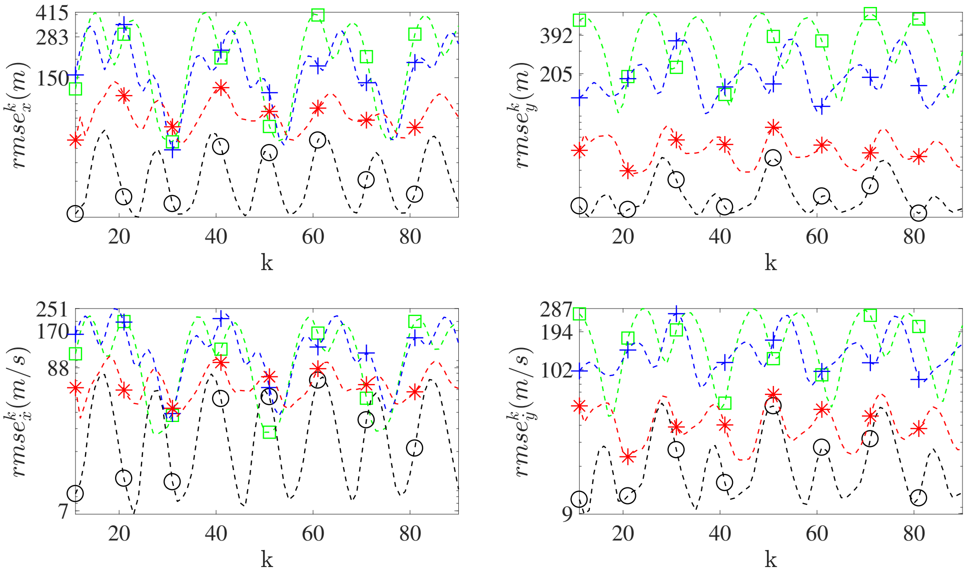
The cross-covariance matrix between the initial inducing points set and the hyperparameters is set to zero i.e. $\hat{\boldsymbol{C}}_0^{\boldsymbol{f\theta}} = \boldsymbol{O}$.

10.4.4 Results

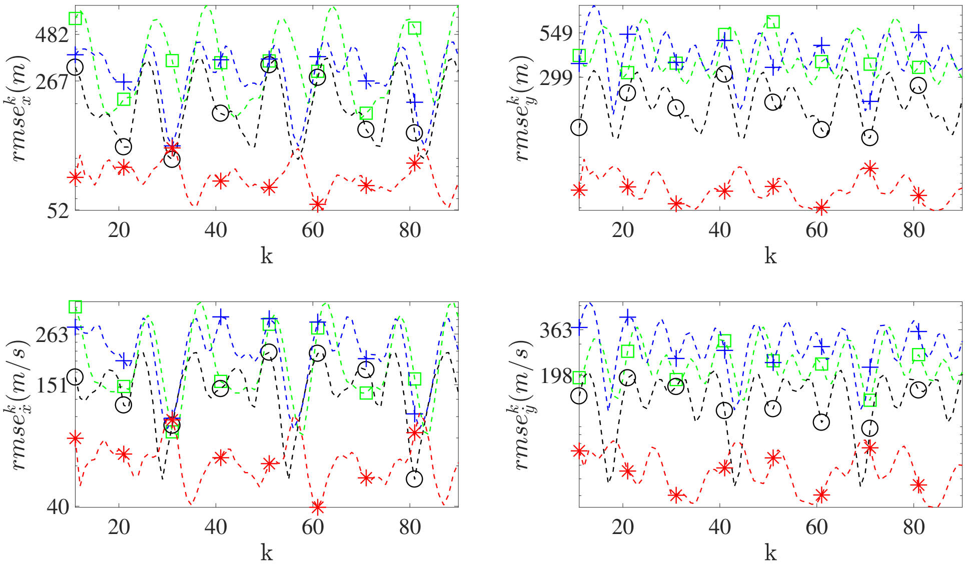
The results are obtained from 10000 Monte Carlo runs for each scenario. The accuracy of the prediction and filtering steps are evaluated separately. The graphical and numerical comparisons of the prediction process are given in Fig. 10.3 and Table 10.2, respectively. In Fig. 10.3, an incomplete plot (such as the FGIMM in S6) means that the corresponding filter sometimes diverges after that time. The Y-axis is set to log scale for readability. It can be observed that the GPMT (**F4)**, the proposed RGP*MT (**F7)** and, a variation of the proposed approach, RGPMT (**F6)** have comparable performance in all scenarios. This shows that the *performance of real-time RGP*MT, using the inducing points and online learning, is as good as ML-based nonreal-time GPMT*. The CGPMT (**F5)** performs poorly during the scenarios (**S1)** and (**S5)**. This is due to the absence of learning in this filter. The RMSE increases with time, which is not a desirable property for filters. The filter FGIMM (**F2)** sometimes diverges in scenario (**S6)**, which is also not a desirable property. The Singer KF (**F3)** performance degrades during the scenarios (**S2)**, (**S3)** and (**S6)**. The performance of the CV filter (**F1)** is not satisfactory for (**S2)**, (**S3)**, (**S5)** and (**S6)**.



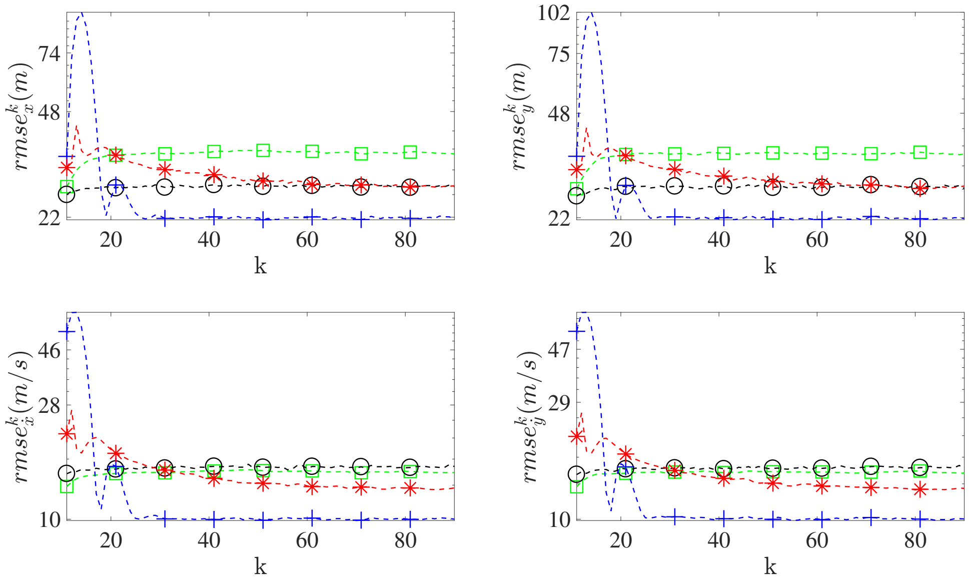
(a) S1: CV



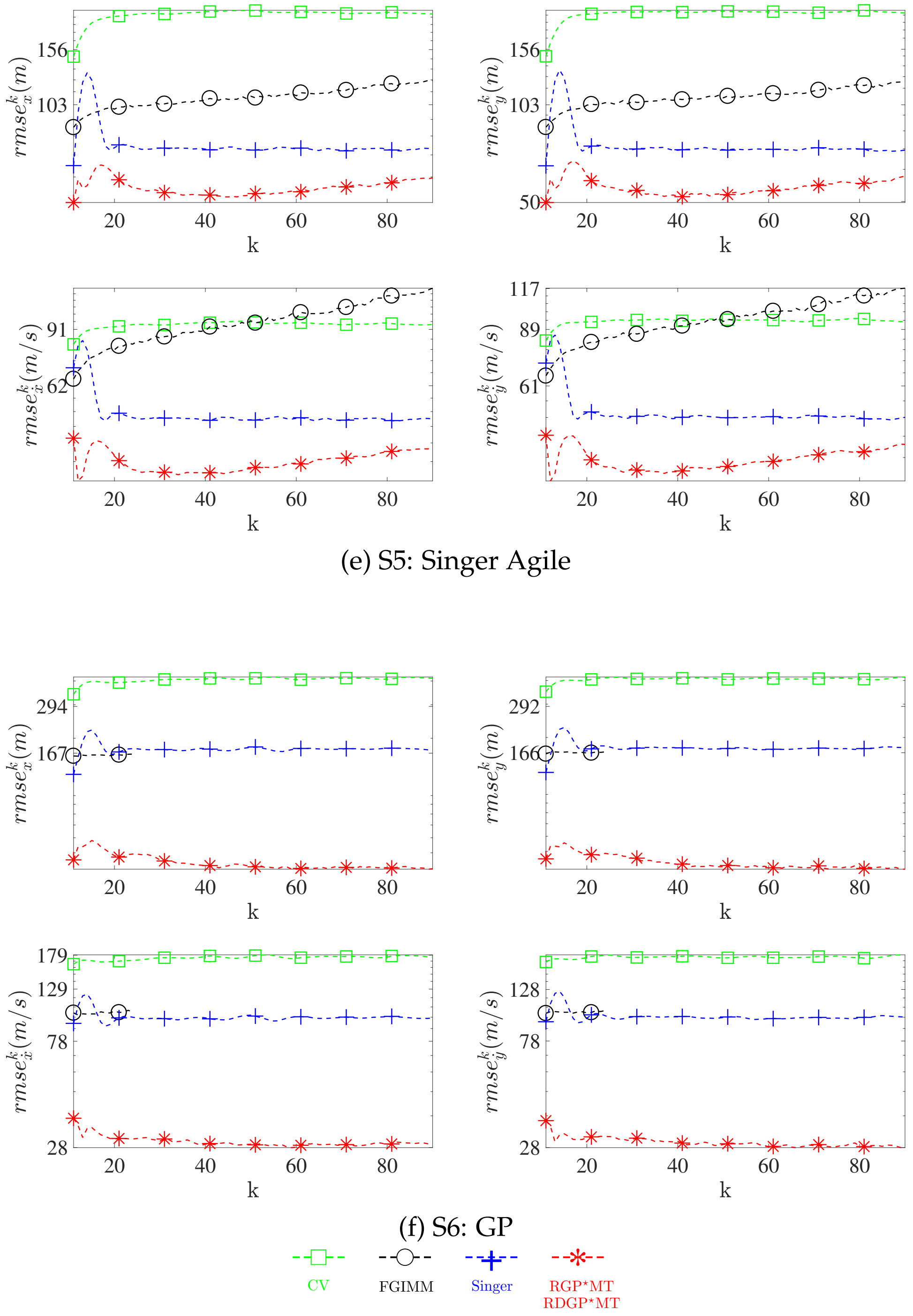
(b) S2: Gradual CT



(c) S3: Sharp CT



(d) S4: Singer lazy



(e) S5: Singer Agile

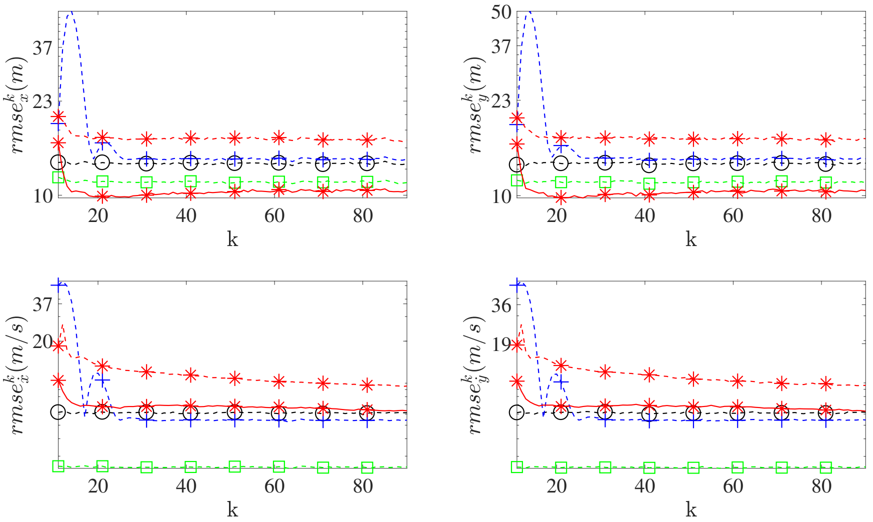
(f) S6: GP

Fig. 10.3 **Prediction RMSE**. The figure shows the prediction performance in 10000 Monte Carlo runs based on RMSE.

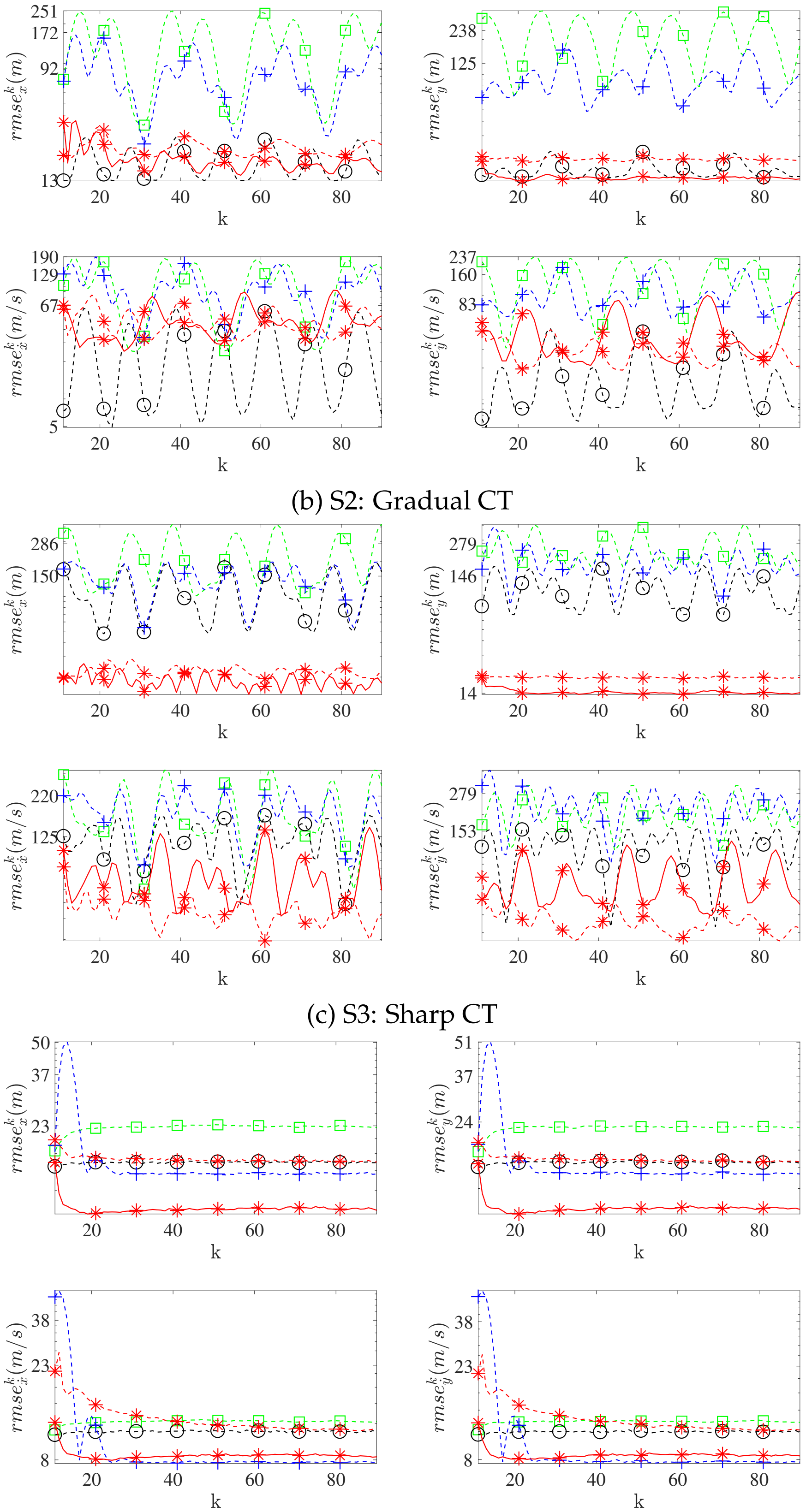
Table 10.2 Prediction Mean RMSE

		F1 **CV**	**F2** **IMM**	**F3** **CT**	**F4** **GPMT**	**F5** **CGPMT**	**F6** **RGPMT**	**F7** **RGP*MT**
S1 **Uniform**	x	13	19	24	22	41	17	28
	$\dot{x}$	3	9	10	10	31	5	13
S2 **Gradual CT**	x	230	34	181	72	113	148	87
	$\dot{x}$	123	32	130	83	119	98	67
S3 **Sharp CT**	x	365	208	303	69	71	82	79
	$\dot{x}$	203	150	224	82	79	55	66
S4 **Singer lazy**	x	35	28	27	27	45	35	30
	$\dot{x}$	15	16	13	15	35	13	15
S5 **Singer agile**	x	203	109	77	46	164	69	55
	$\dot{x}$	94	96	51	40	167	40	37
S6 **GP**	x	407	NAN	176	43	55	51	44
	$\dot{x}$	175	NAN	98	36	44	30	30

The graphical and numerical comparisons of the filtering process are given in Fig. 10.4 and Table 10.3, respectively. In Fig. 10.4, an incomplete plot (such as the FGIMM in S6) means that the corresponding filter sometimes diverges after that time. The Y-axis is set to log scale for readability. A comparison similar to the prediction process can be done. It can be concluded that the *model-based filters and constant hyperparameters based GPMT are not suitable for predicting a wide range of target trajectories*. Three highly manoeuvring and mismatched scenarios, **(S3)**, **(S5)** and **(S6)**, are considered to study the effects of the mismatch. For scenario **(S3)**, the proposed approach provides a performance improvement of $\frac{22-109}{109} \times 100 = 80\%$ in position and $\frac{45-118}{118} \times 100 = 62\%$ in velocity filtering as compared to the best model-based filter **(F2)**. For scenario **(S5)**, the proposed approach provides a performance improvement of $\frac{19-37}{37} \times 100 = 49\%$ in position and $\frac{29-37}{37} \times 100 = 22\%$ in velocity filtering as compared to the best model-based filter **(F3)**. For scenario **(S6)**, the proposed approach provides a performance improvement of $\frac{20-81}{81} \times 100 = 75\%$ in position and $\frac{27-64}{64} \times 100 = 58\%$ in velocity filtering as compared to the best model-based filter **(F3)**. The smoothing performance is also demonstrated in Fig. 10.4 and Table 10.3. It can be observed that the smoother improves upon the filtered position in all scenarios. The smoother improves the filtered velocity in all scenarios except those involving the coordinated turns. This anomaly will be investigated in the future.



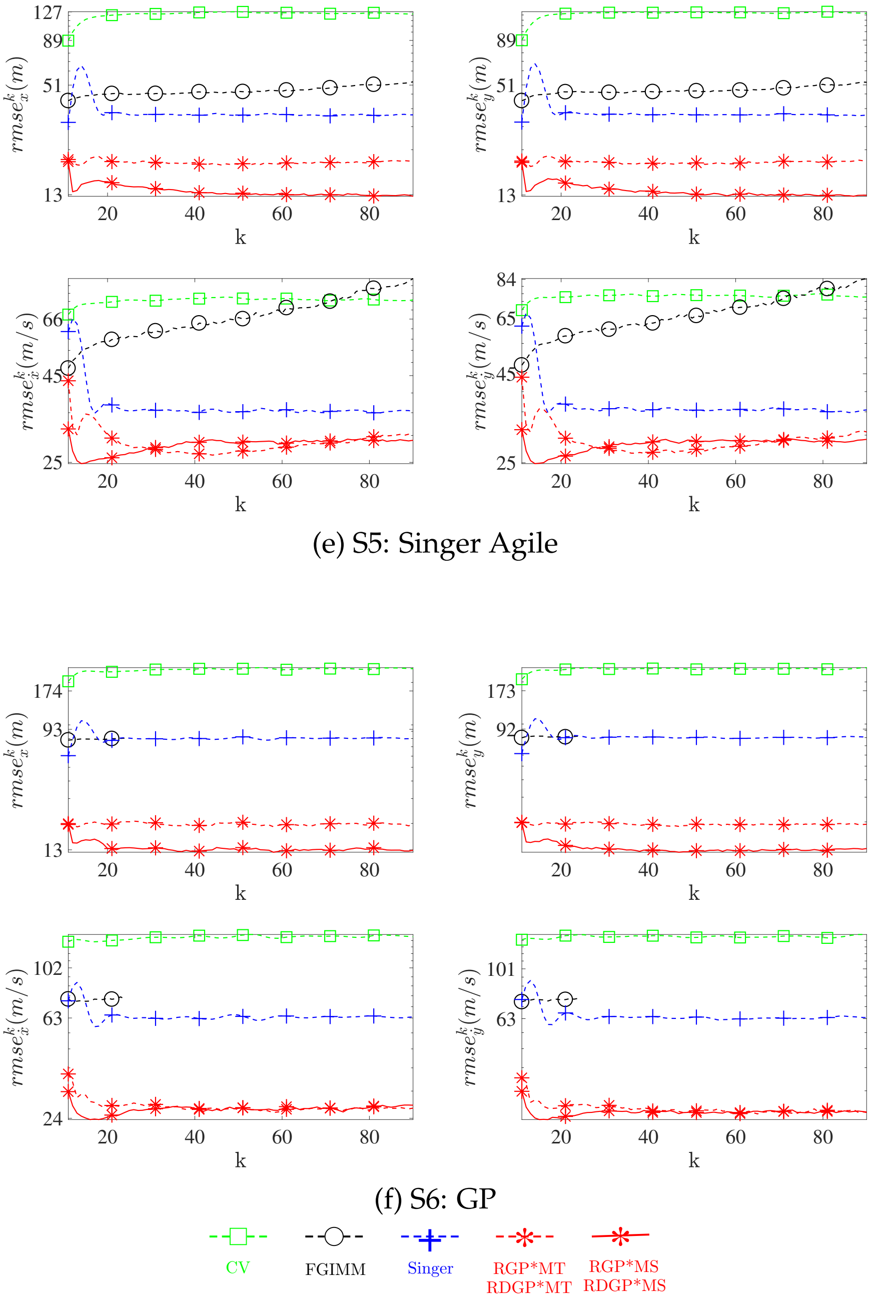
(a) S1: CV



(b) S2: Gradual CT

(c) S3: Sharp CT

(d) S4: Singer lazy



(e) S5: Singer Agile

(f) S6: GP

Fig. 10.4 **Filtering and smoothing RMSE**. The figure shows the filtering and smoothing performance in 10000 Monte Carlo runs based on RMSE.

Table 10.3 Filtering Mean RMSE

		F1 **CV**	**F2** **IMM**	**F3** **Singer**	**F4** **GPMT**	**F5** **CGPMT**	**F6** **RGPMT**	**F7** **RGP*MT**	**S*7** **RGP*MS**
S1 **Uniform**	x	11	13	16	15	19	12	17	10
	$\dot{x}$	3	6	8	8	18	5	12	7
S2 **Gradual CT**	x	140	18	83	20	30	49	23	19
	$\dot{x}$	106	25	98	38	61	87	48	48
S3 **Sharp CT**	x	221	109	139	19	21	22	22	19
	$\dot{x}$	175	118	174	38	39	47	45	73
S4 **Singer lazy**	x	23	16	17	16	20	22	17	11
	$\dot{x}$	12	11	10	10	20	13	13	8
S5 **Singer agile**	x	123	48	37	19	44	29	19	14
	$\dot{x}$	75	67	37	24	85	35	29	28
S6 **GP**	x	247	NAN	81	18	23	22	20	13
	$\dot{x}$	137	NAN	64	26	30	29	27	26

10.4.5 Processing Time

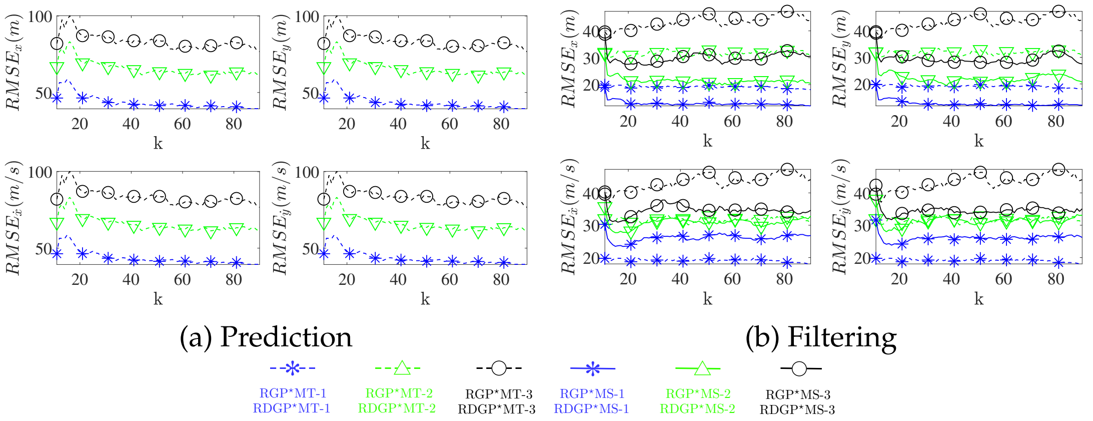
The program was run on MATLAB R2018b on a Windows 7 Home (64 bit) Laptop computer installed with an Intel(R) Core(TM) i3-M350 CPU @ 2.27GHz(2 CPUs) and 4GB RAM. The processing time of a single filter iteration averaged over 500 Monte Carlo runs is given in Table 10.4, sorted from fastest to slowest. Filters (**F4)** and (**F6)** take maximum time per iteration due to the time taken for ML-based learning. These are not suitable for real-time processing. Filters (**F1)**, (**F3)** and (**F5)** are the fastest among the compared approaches. The combined processing time for the proposed filter and smoother is around $5ms$. It is almost 10 times slower than (**F2)**. The processing time of the proposed approach can be improved further by optimization of the code. It can also be run faster by changing the platform to C++.

Table 10.4 Processing Time in Milliseconds

Method	**Time (msec)**
CGPMT + FOCGPMT	0.01
CV	0.05
Singer	0.07
FGIMM	0.5
RGP⋆MT + RDGP⋆MT + RGP*MS + RDGP*MS	5
GPMT + FO-GPMT	41
RGPMT + RDGPMT + RGPMS + RDGPMS	52

10.4.6 Impact of Increased Noise Variance on the Performance

A simulation based study on the performance degradation of the proposed approaches with increasing noise variance is given in this section for x coordinate. The GP based simulation scenario, (**S6)**, is considered for the evaluation. The results are compiled for three different measurement noise standard deviations, $\sigma_x = [25, 40, 50]$ m. The position and velocity RMSE for 5000 Monte Carlo runs of the three test scenarios are given in Figure 10.5. The left and right halves correspond, respectively, to the prediction and the filtering. The numerics after the name of the approach represents the scenarios i.e. the postfix numeric 1 corresponds to 25 m standard deviation in measurement noise while 2 and 3 represent, respectively, 40 m and 50 m standard deviations. It can be observed that the performance degrades as the measurement noise variance is increased. It is important to note that no prior information regarding the change in the noise variance is provided to the filter and the smoother. The proposed approaches adapt to the changing noise variance scenarios through recursive estimation of the variance. The filter and the smoother also do not diverge.



(a) Prediction

(b) Filtering

Fig. 10.5 **Performance with respect to measurement noise variance**. This figure shows the performance of the proposed approaches using different measurement noise variances.

The percentage degradation in the prediction, filtering and smoothing with respect to the percentage increase in the noise variance is given in Table 10.5. It can be observed that the percentage degradation in the position prediction, filtering and smoothing is comparable to the percentage increase in the noise variance e.g. a 100% increase in the noise variance degrades the position prediction by 92%. The percentage degradation in the velocity prediction, filtering and smoothing is less as compared with the percentage degradation of the filtered position, e.g. a 100% increase in the noise variance degrades the velocity prediction by 42%.

Table 10.5 Percentage Degradation

	% Increment	**% degradation**			
		Prediction		**Filtering**	
	σ_x	**pos**	**vel**	**pos**	**vel**
Filter	60	51	25	66	22
	100	92	42	127	40
Smoother	60	-	-	66	18
	100	-	-	130	33

10.5 Conclusions

This chapter proposes an online Gaussian process approach for point target tracking and smoothing. The proposed filter, recursive Gaussian process motion tracker, and smoother, recursive Gaussian process motion smoother, are highly adaptive due to recursive learning of the hyperparameters. The online tracking and smoothing of the derivative of the Gaussian process are also proposed for point target velocity estimation. The above approaches can be extended to higher dimensions and higher order derivatives in a similar way.

The performance evaluation of the proposed approaches is carried out in challenging scenarios. The popular model-based approaches are chosen for performance comparison. Six different scenarios, covering a wide range of point target trajectories, are simulated using model-based and model-free approaches. For the two highly manoeuvrable and mismatched scenarios, (**S3)** CT and (**S5)** Singer, the proposed approach provides 80% and 62% performance improvement in the position estimates and 49% and 22% in the velocity estimates, respectively, as compared to the best model-based filter (**F3)** i.e. a Singer motion model based Kalman Filter.

The flexibility provided by the proposed model-free approach strongly advocates its preference over model-based approaches for applications involving a wide set of target trajectories. In the future, the performance of the proposed approach will be studied with information measures to quantify the impact of uncertainties and using different GP covariance kernels.

Chapter 11

Conclusions

The thesis proposes Gaussian process based offline and online methods for multiple point, extended and group target tracking and smoothing. The various complexities assumed and solved using these methods are time varying unknown number of targets; measurement noise and origin uncertainty; and unknown target kinematics, shape and shape dynamics.

A Gaussian process convolution particle filter (GPCPF) is proposed for tracking of a single extended or group target in Chapter 5. A novel covariance kernel for the Gaussian process model and a convolution kernel for measurement likelihood and particle weight calculations are proposed. The proposed GPCPF approach is shown to track an arbitrarily shaped target in clutter. The proposed approach is shown to be insensitive to the statistical properties of the reflection points. The GPCPF for multiple extended or group target tracking is proposed in Chapter 6. A novel convolution kernel is proposed for multi-target likelihood calculation. A target birth/existence/death is proposed within a statistical framework for tracking the number of targets. The proposed multiple extended or group target filter is shown to track the real extended targets in the presence of various challenges such as dense clutter, occlusion, static and moving targets and low observability. The proposed approach is also shown to be insensitive to the statistical properties of the reflection points in similar targets, that is cars. The processing time of the proposed approach can be improved by reducing the number of patricles, which results in slightly degraded performance.

A generic spatio-temporal Gaussian process (STGP) model is proposed for tracking and smoothing of non-rigid extended or group target in Chapter 7. The proposed method is an improvement over state-of-the-art complex shape tracking approaches, which model the target shape changes as a random walk [90] or a damped random walk [26]. The proposed STGP model relies on a product of spatial and temporal kernel for shape tracking. A state space form of the shape model is derived from the product of covariance kernels based STGP and augmented with the kinematics model. The augmented model is solved using an extended Kalman filter and Rauch-Tung-Striebel smoother. A step by step example approach, based on the proposed method, is described and compared with state-of-the-art Gaussian process approach [26]. The example approach of the proposed method is shown to improve tracking on real and simulated non-rigid extended targets.

A Gaussian process approach for kinematics estimation of point targets is proposed in Chapter 8. The processing time of the proposed Gaussian process motion tracker (GPMT) is improved using a sliding window based online training. The GPMT relies on historic measurement data, rather than a motion model, and hyperparameter learning for prediction and filtering of the target position. A first order GPMT is also proposed for the prediction and filtering of the derivative of the position, using the derivative of the Gaussian process. The proposed approach can be extended to higher dimensions and higher order derivatives in a similar way. The performance evaluation is done in challenging scenarios and compared with popular model-based approaches. The root mean square errors based comparison shows that the proposed approach performs overall best in all compared scenarios. A simulation based study on various aspects of the GPMT is given in Chapter 9. A rational quadratic (RQ) and Matérn (M3) covariance kernel based GPMT are compared with the proposed squared exponential (SE) kernel based GPMT of Chapter 8. The study shows that the SE-GPMT performs best on constant velocity and Singer model based target trajectories. The RQ-GPMT performs best on the coordinated turn trajectories. The accuracy of the M3-GPMT improves with the increasing number of training data. The accuracies of RQ-GPMT and SE-GPMT decreases with increasing data on the Singer model based trajectory. Lastly, it has been shown that all three GPMT approaches are robust to the changing measurement noise variances.

The learning of hyperparameters for the GPMT proposed in Chapter 8 is not online and the approach is computationally expensive. The processing time is improved using a recursive approach for learning. A recursive Gaussian process motion tracker (RGP*MT) and smoother (RGP*MS) are, respectively, proposed for point target kinematics tracking and smoothing in Chapter 10. The proposed approaches use a grid of inducing points on the recent past to predict and filter the target position. The adaptability of the approach is improved through the online learning of the hyperparameters. The target velocity tracking and smoothing are proposed using the recursive derivative of the Gaussian process motion tracker (RDGP*MT) and smoother (RDGP*MS). Two different approaches are proposed for tracking of the derivative, first is based on pseudo-measurement and the second on the actual measurement. The proposed approaches are compared with popular model-based and different variants of the Gaussian process based approaches. The root mean square error based analysis shows that the proposed approach is comparable in performance to the GPMT, proposed in Chapter 8, and overall better than the model-based approaches. The processing time of the GPMT is 8 times higher than the proposed approach using a test system. The proposed approach is also shown to be robust to the changes in the measurement noise variance.

11.1 Future Work

This thesis proposes Gaussian process based methods for multiple point and extended or group target tracking. Given below are recommendations for future work, based upon the research conducted for writing of this thesis:

- The GPCPF for multiple extended or group targets tracking is shown to track arbitrarily shaped multiple targets in clutter (see Chapter 6). The proposed approach can be improved for the handling of closely moving targets. This can be achieved by improving the clustering approach proposed in Chapter 6 from DBSCAN to measurement set partitioning methods proposed in [127]. The multiple target likelihood needs to be derived according to the most likely set partitions.

- The STGP method for extended or group target tracking and smoothing are shown to track non-rigid targets efficiently in Chapter 7. The proposed approach can be extended to tracking of multiple extended or group targets using various data association methods. A Gaussian process based data association approach for extended or group targets tracking, similar to the overlapping mixture of Gaussian process proposed for point targets [37], is the recommended direction.

- The STGP method, proposed in Chapter 7, assumes known and fixed hyperparameters. The adaptability of the method can be improved through online learning of the hyperparameters. The online recursive Gaussian process [40] is one of the methods that can be used for online learning.

- A simulation based study on various aspects of the GPMT is given in Chapter 9. The experimental results can be validated using theoretical derivations in the future. Moreover, more complex covariance kernels based GPMT can be studied such as using the sum and product of the kernels. A similar study can be done on the first order (FO)-GPMT.

- The model free approaches RGP*MT and RGP*MS, proposed in Chapter 10, are shown to be better than model-based approaches. Both approaches rely on the inducing point sets for tracking and smoothing. In future, work can be done on an automatic selection of the number and location of inducing points set. Various methods of automatic selection of the inducing points for sparse Gaussian processes can be exploited.

- Work can also be done on deriving the uncertainty bounds of RGP*MT and RGP*MS. The filter consistency, an important aspect of the target tracking methods, can be evaluated.

- The online hyperparameter learning proposed in RGP*MT and RGP*MS uses constrained unscented Kalman filter for sigma point calculations. Other Sigma point algorithms can also be researched for improving performance.

This thesis shows that the Gaussian process can be effectively employed to solve various multiple point and extended or group target tracking problems. It also shows the promising future of these methods and how they are poised to replace the model-based approaches. Further work on the above proposed methods, as recommended in this chapter, will mature these methods and strengthen their suitability for practical systems.

References

[1] Waqas Aftab, Allan De Freitas, Mahnaz Arvaneh, and Lyudmila Mihaylova. A Gaussian process approach for extended object tracking with random shapes and for dealing with intractable likelihoods. In *Proceedings of the 22nd International Conference on Digital Signal Processing (DSP)*, pages 1–5, London, UK, 2017. IEEE.

[2] Waqas Aftab, Allan De Freitas, Mahnaz Arvaneh, and Lyudmila Mihaylova. A Gaussian process convolution particle filter for multiple extended objects tracking with non-regular shapes. In *Proceedings of the 21st International Conference on Information Fusion (FUSION)*, pages 1–8, Cambridge, UK, 2018. IEEE.

[3] Waqas Aftab, Roland Hostettler, Allan De Freitas, Mahnaz Arvaneh, and Lyudmila Mihaylova. Spatio-temporal Gaussian process models for extended and group object tracking with irregular shapes. *IEEE Transactions on Vehicular Technology*, 68(3):2137–2151, 2019.

[4] Waqas Aftab and Lyudmila Mihaylova. A Gaussian process regression approach for point target tracking. In *Proceedings of the 22nd International Conference on Information Fusion (FUSION)*, pages 1–8, Ottawa, Canada, 2019. IEEE.

[5] Waqas Aftab and Lyudmila Mihaylova. On the impact of different kernels and training data on a Gaussian process approach for tracking. In *Proceedings of the 23rd International Conference on Information Fusion (FUSION)*, South Africa, 2020. IEEE. In press.

[6] Waqas Aftab and Lyudmila Mihaylova. A learning Gaussian process approach for manoeuvring target tracking and smoothing. *IEEE Transactions on Aerospace and Electronic Systems*, 2020. In press.

[7] L Mihaylova and W Aftab. Recent advances in extended and group objects tracking. In *Proceedings of the Artificial Intelligence for Military Multisensor Fusion Engines(STO-MP-SET-262)*, pages 14.1–14.12. IEEE, 2019.

[8] Jeffrey K Uhlmann. Algorithms for multiple-target tracking. *American Scientist*, 80(2):128–141, 1992.

[9] Abdollah Arasteh, Bijan Vosoughi Vahdat, and Reza Salman Yazdi. Multi-target tracking of human spermatozoa in phase-contrast microscopy image sequences using a hybrid dynamic Bayesian network. *Scientific reports*, 8(1): 5068, 2018.

[10] Jaebum Choi, Simon Ulbrich, Bernd Lichte, and Markus Maurer. Multi-target tracking using a 3D-Lidar sensor for autonomous vehicles. In *Proceedings of the International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 881–886, The Hague, Netherlands, 2013. IEEE.

[11] Felix Kunz, Dominik Nuss, Jürgen Wiest, Hendrik Deusch, Stephan Reuter, Franz Gritschneder, Alexander Scheel, Manuel Stübler, Martin Bach, Patrick Hatzelmann, et al. Autonomous driving at Ulm University: A modular, robust, and sensor-independent fusion approach. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 666–673. IEEE, 2015.

[12] Christian Lundquist, Umut Orguner, and Fredrik Gustafsson. Extended target tracking using polynomials with applications to road-map estimation. *IEEE Transactions on Signal Processing*, 59(1):15–26, 2010.

[13] Andreas Kräußling. *Intelligent Techniques and Tools for Novel System Architectures*, chapter Tracking Extended Moving Objects with a Mobile Robot, pages 513–530. Springer Berlin Heidelberg, 2008.

[14] Viktor Edman, Maria Andersson, Karl Granstrom, and Fredrik Gustafsson. Pedestrian group tracking using the GM-PHD filter. In *Proceedings of the 21st European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE, 2013.

[15] Brandon A Jones, Daniel S Bryant, Ba-Tuong Vo, and Ba-Ngu Vo. Challenges of multi-target tracking for space situational awareness. In *Proceedings of the 18th International Conference on Information Fusion (Fusion)*, pages 1278–1285. IEEE, 2015.

[16] Seiichi Sato, Masafumi Hashimoto, Manabu Takita, Kiyokazu Takagi, and Takashi Ogawa. Multilayer LiDAR-based pedestrian tracking in urban environments. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 849–854. IEEE, 2010.

[17] François Septier, Avishy Carmi, and Simon Godsill. Tracking of multiple contaminant clouds. In *Proceedings of the 12th International Conference on Information Fusion*, pages 1280–1287. IEEE, 2009.

[18] Weishi Chen, Hong Liu, Sha Hu, and Huansheng Ning. Group tracking of flock targets in low-altitude airspace. In *Proceedings of the IEEE Ninth International Symposium on Parallel and Distributed Processing with Applications Workshops*, pages 131–136. IEEE, 2011.

[19] Jiawei Yi, Yunyan Du, Fuyuan Liang, and Chenghu Zhou. An auto-tracking algorithm for mesoscale eddies using global nearest neighbor filter. *Limnology and Oceanography: Methods*, 15(3):276–290, 2017.

[20] Inseok Hwang, Hamsa Balakrishnan, Kaushik Roy, Jaewon Shin, Leonidas Guibas, and Claire Tomlin. Multiple-target tracking and identity management. In *SENSORS, 2003 IEEE*, volume 1, pages 36–41, Toronto, Canada, 2003. IEEE.

[21] Karl Granström, Antonio Natale, Paolo Braca, Giovanni Ludeno, and Francesco Serafino. Gamma Gaussian inverse Wishart probability hypothesis density for extended target tracking using X-band marine radar data. *IEEE Transactions on Geoscience and Remote Sensing*, 53(12):6617–6631, 2015.

[22] Florian Faion, Marcus Baum, and Uwe D Hanebeck. Tracking 3D shapes in noisy point clouds with random hypersurface models. In *Proceedings of the 15th International Conference on Information Fusion*, pages 2230–2235. IEEE, 2012.

[23] Radford M Neal. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pages 29–53. Springer, 1996.

[24] Lyudmila Mihaylova, Avishy Y Carmi, François Septier, Amadou Gning, Sze Kim Pang, and Simon Godsill. Overview of Bayesian sequential Monte Carlo methods for group and extended object tracking. *Digital Signal Processing*, 25:1–16, 2014.

[25] Karl Granstrom, Marcus Baum, and Stephan Reuter. Extended object tracking: Introduction, overview and applications. *ISIF Journal of Advances in Information Fusion*, 12(2):139–174, 2017.

[26] Niklas Wahlström and Emre Özkan. Extended target tracking using Gaussian processes. *IEEE Transactions on Signal Processing*, 63(16):4165–4178, 2015.

[27] Tobias Hirscher, Alexander Scheel, Stephan Reuter, and Klaus Dietmayer. Multiple extended object tracking using Gaussian processes. In *Proceedings of the 19th International Conference on Information Fusion*, pages 868–875, 2016.

[28] Marcus Baum, Benjamin Noack, and Uwe D Hanebeck. Mixture random hypersurface models for tracking multiple extended objects. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 3166–3171. IEEE, 2011.

[29] Marcus Baum and Uwe D Hanebeck. Random hypersurface models for extended object tracking. In *Proceedings of the IEEE International Symposium on Signal Processing and Information Technology*, pages 178–183. IEEE, 2009.

[30] Joost N Kok, EJ Boers, Walter A Kosters, Peter Van der Putten, and Mannes Poel. Artificial intelligence: Definition, trends, techniques, and cases. *Artificial Intelligence*, 1, 2009.

[31] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24, 2007.

[32] Cyril Voyant, Gilles Notton, Soteris Kalogirou, Marie-Laure Nivet, Christophe Paoli, Fabrice Motte, and Alexis Fouilloy. Machine learning methods for solar radiation forecasting: A review. *Renewable Energy*, 105:569–582, 2017.

[33] Maria De Marsico, Alfredo Petrosino, and Stefano Ricciardi. Iris recognition through machine learning techniques: A survey. *Pattern Recognition Letters*, 82: 106–115, 2016.

[34] Ariruna Dasgupta and Asoke Nath. Classification of machine learning algorithms. *International Journal of Innovative Research in Advanced Engineering (IJIRAE)*, 3(3):6–11, 2016.

[35] Chang Gao, Hongwei Liu, Shenghua Zhou, Hongtao Su, Bo Chen, Junkun Yan, and Kuiying Yin. Maneuvering target tracking with recurrent neural networks for radar application. In *Proceedings of the International Conference on Radar (RADAR)*, pages 1–5. IEEE, 2018.

[36] Debasis Sengupta and Ronald A Iltis. Neural solution to the multitarget tracking data association problem. *IEEE Transactions on Aerospace and Electronic Systems*, 25(1):96–108, 1989.

[37] Miguel Lázaro-Gredilla, Steven Van Vaerenbergh, and Neil D Lawrence. Overlapping mixtures of Gaussian processes for the data association problem. *Pattern Recognition*, 45(4):1386–1395, 2012.

[38] Farid Amoozegar. Neural-network-based target tracking state-of-the-art survey. *Optical Engineering*, 37(3):836–846, 1998.

[39] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[40] Marco F Huber. Recursive Gaussian process: On-line regression and learning. *Pattern Recognition Letters*, 45:85–91, 2014.

[41] N Peach. Bearings-only tracking using a set of range-parameterised extended Kalman filters. *Proceedings of the IEE Control Theory and Applications*, 142(1): 73–80, 1995.

[42] Yaakov Bar-Shalom, Peter K Willett, and Xin Tian. *Tracking and data fusion*. YBS publishing Storrs, CT, USA:, 2011.

[43] Yaakov Bar-Shalom, Thiagalingam Kirubarajan, and Cenk Gokberk. Tracking with classification-aided multiframe data association. *IEEE Transactions on Aerospace and Electronic systems*, 41(3):868–878, 2005.

[44] Andrew H Jazwinski. *Stochastic processes and filtering theory*. Courier Corporation, 2007.

[45] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[46] Simon J Julier, Jeffrey K Uhlmann, and Hugh F Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of the 1995 American Control Conference-ACC'95*, volume 3, pages 1628–1632. IEEE, 1995.

[47] Simon J Julier and Jeffrey K Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[48] Eric A Wan and Rudolph Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. IEEE, 2000.

[49] Yi Xu and Li Liping. Single observer bearings-only tracking with the unscented Kalman filter. In *Proceedings of the International Conference on Communications, Circuits and Systems (IEEE Cat. No. 04EX914)*, volume 2, pages 901–905. IEEE, 2004.

[50] Fredrik Gustafsson and Gustaf Hendeby. Some relations between extended and unscented Kalman filters. *IEEE Transactions on Signal Processing*, 60(2): 545–555, 2011.

[51] X Rong Li and Vesselin P Jilkov. Survey of maneuvering target tracking. Part V. Multiple-model methods. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1255–1321, 2005.

[52] D Magill. Optimal adaptive estimation of sampled stochastic processes. *IEEE Transactions on Automatic Control*, 10(4):434–439, 1965.

[53] Demetrios Lainiotis. Optimal adaptive estimation: Structure and parameter adaption. *IEEE Transactions on Automatic Control*, 16(2):160–170, 1971.

[54] Henk AP Blom. A sophisticated tracking algorithm for atc surveillance data. In *Proceedings of the International Radar Conference*, pages 393–398, Paris, France, 1984.

[55] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[56] Samuel Blackman and Robert Popoli. Design and analysis of modern tracking systems(book). *Norwood, MA: Artech House, 1999.*, 1999.

[57] X Rong Li and Yaakov Bar-Shakm. Mode-set adaptation in multiple-model estimators for hybrid systems. In *Proceedings of the American Control Conference*, pages 1794–1799. IEEE, 1992.

[58] VPS Naidu, Girija Gopalaratnam, and N Shanthakumar. Three model IMM-EKF for tracking targets executing evasive maneuvers. In *Proceedings of the 45th AIAA Aerospace Sciences Meeting and Exhibit*, page 1204, 2007.

[59] Rafael Toledo-Moreo, Miguel A Zamora-Izquierdo, Benito Ubeda-Minarro, and Antonio F Gómez-Skarmeta. High-integrity IMM-EKF-based road vehicle navigation with low-cost GPS/SBAS/INS. *IEEE Transactions on Intelligent Transportation Systems*, 8(3):491–511, 2007.

[60] Gu Xiaodong, Yuan Zhiyong, and Zhou Hao. Bearings-only tracking of maneuvering target based on IMM-UKF algorithm. *Journal of Data Acquisition and Processing*, 24(S 1):88–91, 2009.

[61] Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *Proceedings of the IEE F (Radar and Signal Processing)*, volume 140, pages 107–113. IET, 1993.

[62] Xuedong Wang, Tiancheng Li, Shudong Sun, and Juan Corchado. A survey of recent advances in particle filters and remaining challenges for multitarget tracking. *Sensors*, 17(12):2707, 2017.

[63] X Rong Li and Vesselin P Jilkov. Survey of maneuvering target tracking. Part I. Dynamic models. *IEEE Transactions on Aerospace and Electronic systems*, 39(4): 1333–1364, 2003.

[64] Robert A Singer. Estimating optimal tracking filter performance for manned maneuvering targets. *IEEE Transactions on Aerospace and Electronic Systems*, AES-6(4):473–483, 1970.

[65] Zhansheng Duan, Chongzhao Han, and X Rong Li. Comments on "unbiased converted measurements for tracking". *IEEE Transactions on Aerospace and Electronic Systems*, 40(4):1374, 2004.

[66] Oliver E Drummond. Tracking clusters and extended objects with multiple sensors. In *Proceedings of the Signal and Data Processing of Small Targets 1990*, volume 1305, page 362. International Society for Optics and Photonics, 1990.

[67] Milton J Waxman and Oliver E Drummond. A bibliography of cluster (group) tracking. In *Proceedings of the Signal and Data Processing of Small Targets*, volume 5428, pages 551–560. International Society for Optics and Photonics, 2004.

[68] STEVEN Bordonaro, PETER Willett, YB Shalom, T Luginbuhl, and M Baum. Extended object tracking with exploitation of range rate measurements. *ISIF Journal of Advances in Information Fusion*, 12(2), 2017.

[69] Marcus Baum, Florian Faion, and Uwe D Hanebeck. Modeling the target extent with multiplicative noise. In *Proceedings of the 15th International Conference on Information Fusion*, pages 2406–2412, 2012.

[70] Allan De Freitas, Lyudmila Mihaylova, Amadou Gning, Donka Angelova, and Visakan Kadirkamanathan. Autonomous crowds tracking with box particle filtering and convolution particle filtering. *Automatica*, 69:380–394, 2016.

[71] Donka Angelova, Lyudmila Mihaylova, Nikolay Petrov, and Amadou Gning. A convolution particle filtering approach for tracking elliptical extended objects. In *Proceedings of the 16th International Conference on Information Fusion*, pages 1542–1549, 2013.

[72] J. W. Koch. Bayesian approach to extended object and cluster tracking using random matrices. *IEEE Transactions on Aerospace and Electronic Systems*, 44(3): 1042–1059, July 2008. ISSN 0018-9251. doi: 10.1109/TAES.2008.4655362.

[73] Marcus Baum and Uwe D Hanebeck. Extended object tracking with random hypersurface models. *IEEE Transactions on Aerospace and Electronic Systems*, 50 (1):149–159, 2014.

[74] Jian Lan and X Rong Li. Tracking of maneuvering non-ellipsoidal extended object or target group using random matrix. *IEEE Transactions on Signal Processing*, 62(9):2450–2463, 2014.

[75] Botao Lei, Cuiyun Li, and Hongbing Ji. Nonlinear maneuvering non-ellipsoidal extended object tracking using random matrix. In *Proceedings of the 20th International Conference on Information Fusion*, pages 1–6. IEEE, 2017.

[76] Karl Granström, Stephan Reuter, Daniel Meissner, and Alexander Scheel. A multiple model PHD approach to tracking of cars under an assumed rectangular shape. In *Proceedings of the 17th International Conference on Information Fusion*, pages 1–8. IEEE, 2014.

[77] Karl Granström, Antonio Natale, Paolo Braca, Giovanni Ludeno, and Francesco Serafino. PHD extended target tracking using an incoherent X-band radar: Preliminary real-world experimental results. In *Proceedings of the 17th International Conference on Information Fusion*, pages 1–8. IEEE, 2014.

[78] Luo-jia Chi, Xin-xi Feng, and Lu Miao. Generalized labeled multi-Bernoulli extended target tracking based on Gaussian process regression. In *Proceedings of the MATEC Web of Confs.*, volume 176, page 01017. EDP Sciences, 2018.

[79] Emre Özkan, Niklas Wahlström, and Simon J Godsill. Rao-blackwellised particle filter for star-convex extended target tracking models. In *Proceedings of the 19th Conference on Information Fusion*, pages 1193–1199, 2016.

[80] Gemine Vivone and Paolo Braca. Joint probabilistic data association tracker for extended target tracking applied to x-band marine radar data. *IEEE Journal of Oceanic Engineering*, 41(4):1007–1019, 2016.

[81] Monika Wieneke and Wolfgang Koch. A PMHT approach for extended objects and object groups. *IEEE Transactions on Aerospace and Electronic Systems*, 48(3): 2349–2370, 2012.

[82] Kevin Gilholm, Simon Godsill, Simon Maskell, and David Salmond. Poisson models for extended target and group tracking. In *Proceedings of the Optics & Photonics 2005*, pages 59130R–59130R, 2005.

[83] Allan De Freitas, Lyudmila Mihaylova, Amadou Gning, Marek Schikora, Martin Ulmke, Donka Angelova, and Wolfgang Koch. A box particle filter method for tracking multiple extended objects. *IEEE Transactions on Aerospace and Electronic Systems*, 55(4):1640–1655, 2018.

[84] Karl Granstrom, Christian Lundquist, Fredrik Gustafsson, and Umut Orguner. Random set methods: Estimation of multiple extended objects. *IEEE Robotics & Automation Magazine*, 21(2):73–82, 2014.

[85] Ronald Mahler. PHD filters for nonstandard targets, I: Extended targets. In *Proceedings of the 12th International Conference on Information Fusion*, pages 915–921. IEEE, 2009.

[86] Michael Beard, Stephan Reuter, Karl Granström, Ba-Tuong Vo, Ba-Ngu Vo, and Alexander Scheel. A generalised labelled multi-Bernoulli filter for extended multi-target tracking. In *Proceedings of the 18th International Conference on Information Fusion (Fusion)*, pages 991–998, 2015.

[87] Karl Granström, Stephan Renter, Maryam Fatemi, and Lennart Svensson. Pedestrian tracking using Velodyne data–Stochastic optimization for extended object tracking. In *Proceedings of the Intelligent Vehicles Symposium (IV), 2017 IEEE*, pages 39–46. IEEE, 2017.

[88] Michael Feldmann, Dietrich Franken, and Wolfgang Koch. Tracking of extended objects and group targets using random matrices. *IEEE Transactions on Signal Processing*, 59(4):1409–1420, 2011.

[89] Marcus Baum, Vesa Klumpp, and Uwe D Hanebeck. A novel Bayesian method for fitting a circle to noisy points. In *Proceedings of the 13th Conference on Information Fusion*, pages 1–6, 2010.

[90] Marcus Baum and Uwe D Hanebeck. Shape tracking of extended objects and group targets with star-convex rhms. In *Proceedings of the 14th International Conference on Information Fusion*, pages 1–8. IEEE, 2011.

[91] Vivien Rossi and Jean-Pierre Vila. Nonlinear filtering in discrete time: A particle convolution approach. *Annales de l'Institut de Statistique de l'Université de Paris*, 50(3):71–102, 2006.

[92] Jun S Liu and Rong Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

[93] Christian Wolf and Jean-Michel Jolion. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *International Journal of Document Analysis and Recognition (IJDAR)*, 8(4):280–296, 2006.

[94] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the KDD*, volume 96, pages 226–231, 1996.

[95] Sze Kim Pang, Jack Li, and Simon J Godsill. Detection and tracking of coordinated groups. *IEEE Transactions on Aerospace and Electronic Systems*, 47(1): 472–502, 2011.

[96] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The Kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[97] Simo Sarkka, Arno Solin, and Jouni Hartikainen. Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering. *IEEE Signal Processing Magazine*, 30(4):51–61, 2013.

[98] Simo Särkkä and Jouni Hartikainen. Infinite-dimensional Kalman filtering approach to spatio-temporal Gaussian process regression. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 993–1001, 2012.

[99] Agathe Girard, Carl Edward Rasmussen, Joaquin Quinonero Candela, and Roderick Murray-Smith. Gaussian process priors with uncertain inputs-application to multiple-step ahead time series forecasting. *Advances in Neural Information Processing Systems*, pages 545–552, 2003.

[100] Agathe Girard and Roderick Murray-Smith. Gaussian processes: Prediction at a noisy input and application to iterative multiple-step ahead forecasting of time-series. *Lecture Notes in Computer Science*, 3355:158–184, 2005.

[101] Paul Bourke. Calculating the area and centroid of a polygon. *Swinburne University of Technology*, 7, 1988.

[102] Fred Daum. Nonlinear filters: beyond the Kalman filter. *IEEE Aerospace and Electronic Systems Magazine*, 20(8):57–69, 2005.

[103] Mark S Handcock and Michael L Stein. A Bayesian analysis of kriging. *Technometrics*, 35(4):403–410, 1993.

[104] P Guttorp and T Gneiting. On the Whittle-Matérn correlation family. *National Research Center for Statistics and the Environment-Technical Report Series, Seattle, Washington*, NRCSE TR(80):1–8, 2005.

[105] George E Uhlenbeck and Leonard S Ornstein. On the theory of the brownian motion. *Physical Review*, 36(5):823, 1930.

[106] Jouni Hartikainen and Simo Särkkä. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *IEEE International Workshop on Machine Learning for Signal Processing*, pages 379–384. IEEE, 2010.

[107] Aparna Akula, Ripul Ghosh, Satish Kumar, and HK Sardana. Moving target detection in thermal infrared imagery using spatiotemporal information. *JOSA A*, 30(8):1492–1501, 2013.

[108] Tim D Barfoot, Chi Hay Tong, and Simo Särkkä. Batch continuous-time trajectory estimation as exactly sparse Gaussian process regression. In *Proceedings of Robotics: Science and Systems*. Citeseer, 2014.

[109] Jing Dong, Byron Boots, and Frank Dellaert. Sparse Gaussian processes for continuous-time trajectory estimation on matrix lie groups. *arXiv preprint arXiv:1705.06020*, 2017.

[110] X Rong Li and Vesselin P Jilkov. Survey of maneuvering target tracking: decision-based methods. In *Proceedings of the Signal and Data Processing of Small Targets 2002*, volume 4728, pages 511–535. International Society for Optics and Photonics, 2002.

[111] Yaakov Bar-Shalom and Xiao-Rong Li. Estimation and tracking- Principles, techniques, and software. *Norwood, MA: Artech House, Inc.*, 1993.

[112] Kishore Mehrotra and Pravas R Mahapatra. A jerk model for tracking highly maneuvering targets. *IEEE Transactions on Aerospace and Electronic Systems*, 33 (4):1094–1105, 1997.

[113] James P Helferty. Improved tracking of maneuvering targets: The use of turn-rate distributions for acceleration modeling. *IEEE Transactions on Aerospace and Electronic Systems*, 32(4):1355–1361, 1996.

[114] Phillip Boyle. *Gaussian processes for regression and optimisation*. PhD thesis, Victoria University of Wellington, Wellington, New Zealand, 2007.

[115] Peter S Swain, Keiran Stevenson, Allen Leary, Luis F Montano-Gutierrez, Ivan BN Clark, Jackie Vogel, and Teuta Pilizota. Inferring time derivatives including cell growth rates using Gaussian processes. *Nature Communications*, 7:13766, 2016.

[116] Ognjen Rudovic, Maja Pantic, and Ioannis Patras. Coupled Gaussian processes for pose-invariant facial expression recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1357–1369, 2013.

[117] Steven V Bordonaro, Peter Willett, and Yaakov Bar-Shalom. Unbiased tracking with converted measurements. In *Proceedings of the Radar Conference (RADAR)*, pages 0741–0745. IEEE, 2012.

[118] Frederik Beutler, Marco F Huber, and Uwe D Hanebeck. Gaussian filtering using state decomposition methods. In *Proceedings of the 12th International Conference on Information Fusion*, pages 579–586. IEEE, 2009.

[119] Rambabu Kandepu, Lars Imsland, and Bjarne A Foss. Constrained state estimation using the unscented Kalman filter. In *Proceedings of the 16th Mediterranean Conference on Control and Automation*, pages 1453–1458. IEEE, 2008.

[120] Steven Reece and Stephen Roberts. An introduction to Gaussian processes for the Kalman filter expert. In *Proceedings of the 13th International Conference on Information Fusion*, pages 1–9. IEEE, 2010.

[121] Alex J Smola and Peter L Bartlett. Sparse greedy Gaussian process regression. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 619–625, 2001.

[122] Clas Veibäck, Jonatan Olofsson, Tom Rune Lauknes, and Gustaf Hendeby. Learning target dynamics while tracking using Gaussian processes. *IEEE Transactions on Aerospace and Electronic Systems*, 56(4):2591–2602, 2020.

[123] Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1257–1264, 2006.

[124] Hyoung-Moon Kim, Bani K Mallick, and CC Holmes. Analyzing nonstationary spatial data using piecewise Gaussian processes. *Journal of the American Statistical Association*, 100(470):653–668, 2005.

[125] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems*, 2020. In press.

[126] A Munir and DP Atherton. Maneuvring target tracking using different turn rate models in the interacting multiple model algorithm. In *Proceedings of the 34th IEEE Conference on Decision and Control*, volume 3, pages 2747–2751. IEEE, 1995.

[127] Hongyan Zhu, Pandeng Zhang, and Tingting Ma. Research on measurement set partitioning method for tracking multiple extended targets. In *Proceedings of the 27th Chinese Control and Decision Conference (2015 CCDC)*, pages 4099–4104. IEEE, 2015.

[128] Ben H Cantrell. Description of an alpha-beta filter in Cartesian coordinates. Technical report, Naval research lab Washington, D.C., 1973.

[129] Simon J Julier and Jeffrey K Uhlmann. Consistent debiased method for converting between polar and Cartesian coordinate systems. In *Proceedings of the Acquisition, Tracking, and Pointing XI*, volume 3086, pages 110–122. International Society for Optics and Photonics, 1997.

[130] Tiancheng Li, Juan M Corchado, Javier Bajo, Shudong Sun, and Juan F De Paz. Effectiveness of Bayesian filters: An information fusion perspective. *Information Sciences*, 329:670–689, 2016.

[131] Mo Longbin, Song Xiaoquan, Zhou Yiyu, Sun Zhong Kang, and Yaakov Bar-Shalom. Unbiased converted measurements for tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):1023–1027, 1998.

[132] Steven Bordonaro, Peter Willett, and Yaakov Bar-Shalom. Decorrelated unbiased converted measurement Kalman filter. *IEEE Transactions on Aerospace and Electronic Systems*, 50(2):1431–1444, 2014.

Appendix A

Sensor Measurements Coordinate Conversions

This Appendix presents the coordinate converted measurement pdfs as derived in [128]. It presents a geometrical approximation to an i.i.d. Gaussian pdf which undergoes the following transformations; polar to Cartesian, translation, and then back from the translated Cartesian to polar. Another approximate approach is presented using unscented transforms for the above mentioned transformations in [129]. In the Appendix of [128], the author derives the pdf for polar to Cartesian case using mathematical identities, which is exact. A biased conversion degrades the filter performance[130]. The unbias coordinate conversion has also been proposed namely unbiased converted measurement (UCM) in [131], for polar to Cartesian conversion. In [65], an incompatibility in the UCM derivation was highlighted and removed. The corrected conversion was named modified UCM (MUCM), which was later verified through experiments in [117, 132]. The MUCM conversion is exactly same as proposed in the Appendix of [128]. In this work, we use the geometric approximate conversion proposed in [128], as the approximation is valid for low sensor noise, which is often the case in ETT applications.

The sensor measurement noise is modeled i.i.d. Gaussian with variances $\sigma^2_{\tilde{\psi}_{i,k}}$ and $\sigma^2_{\tilde{\phi}_{i,k}}$. The sensor measurement pdf in polar global frame is $\tilde{\boldsymbol{z}}^{sen,p}_{i,k} = \mathcal{N}(\boldsymbol{\mu}^{sen,p}_{i,k}, \boldsymbol{R}^{sen,p}_{i,k})$ with $\tilde{\boldsymbol{z}}^{sen,p}_{i,k} = [\tilde{\psi}_{i,k}, \tilde{\phi}_{i,k}]^T$, $\boldsymbol{\mu}^{sen,p}_{i,k} = [\bar{\psi}_{i,k}, \bar{\phi}_{i,k}]^T$ and $\boldsymbol{R}^{sen,p}_{i,k} = \mathrm{diag}(\sigma^2_{\tilde{\psi}_{i,k}}, \sigma^2_{\tilde{\phi}_{i,k}})$. The corresponding sensor measurement pdf in Cartesian global frame is $\tilde{\boldsymbol{z}}^{sen,c}_{i,k} = \mathcal{N}(\boldsymbol{\mu}^{sen,c}_{i,k}, \boldsymbol{R}^{sen,c}_{i,k})$ with

$$\tilde{\boldsymbol{z}}^{sen,c}_{i,k} = [\tilde{x}_{i,k}, \tilde{y}_{i,k}]^T, \quad \boldsymbol{\mu}^{sen,c}_{i,k} = [\bar{x}_{i,k}, \bar{y}_{i,k}]^T, \quad \boldsymbol{R}^{sen,c}_{i,k} = \begin{bmatrix} \sigma^2_{\tilde{x}_{i,k}} & \sigma^2_{\tilde{x}_{i,k}\tilde{y}_{i,k}} \\ \sigma^2_{\tilde{y}_{i,k}\tilde{x}_{i,k}} & \sigma^2_{\tilde{y}_{i,k}} \end{bmatrix}, \tag{A.1}$$

$$\bar{x}_{i,k} = \lambda_b \cos \bar{\phi}_{i,k}, \quad \bar{y}_{i,k} = \lambda_b \sin \bar{\phi}_{i,k}, \quad \lambda_b = \exp\left(\frac{-\tilde{\phi}_{i,k}^2}{2}\right) \bar{\psi}_{i,k}, \tag{A.2}$$

$$\sigma_{\tilde{x}_{i,k}}^2 = \frac{1}{2}(\bar{\psi}_{i,k}^2 + \sigma_{\tilde{\psi}_{i,k}}^2)[1 + \cos(2\bar{\phi}_{i,k}) \exp(-2\sigma_{\tilde{\phi}_{i,k}}^2)] - \exp(\sigma_{\tilde{\phi}_{i,k}}^2)\bar{\psi}_{i,k}^2 \cos^2 \bar{\phi}_{i,k} \tag{A.3}$$

$$\sigma_{\tilde{y}_{i,k}}^2 = \frac{1}{2}(\bar{\psi}_{i,k}^2 + \sigma_{\tilde{\psi}_{i,k}}^2)[1 - \cos(2\bar{\phi}_{i,k}) \exp(-2\sigma_{\tilde{\phi}_{i,k}}^2)] - \exp(\sigma_{\tilde{\phi}_{i,k}}^2)\bar{\psi}_{i,k}^2 \sin^2 \bar{\phi}_{i,k} \tag{A.4}$$

$$\sigma_{\tilde{x}_{i,k}\tilde{y}_{i,k}}^2 = \frac{1}{2}(\bar{\psi}_{i,k}^2 + \sigma_{\tilde{\psi}_{i,k}}^2)[\sin(2\bar{\phi}_{i,k}) \exp(-2\sigma_{\tilde{\phi}_{i,k}}^2)] - \exp(\sigma_{\tilde{\phi}_{i,k}}^2)\bar{\psi}_{i,k}^2 \cos \bar{\phi}_{i,k} \sin \bar{\phi}_{i,k} \tag{A.5}$$

Suppose a $[-x_k, -y_k]^T$ translation is applied to $\tilde{\boldsymbol{z}}_{i,k}^{sen,c}$ to obtain $\tilde{\boldsymbol{z}}_{i,k}^{obj,c} = [\tilde{x}_{i,k}^t, \tilde{y}_{i,k}^t]^T$ where $\tilde{x}_{i,k}^t = \bar{x}_{i,k} - x_k + \nu_{\tilde{x}_{i,k}} = \breve{x}_{i,k} + \nu_{\tilde{x}_{i,k}}$, $\tilde{y}_{i,k}^t = \bar{y}_{i,k} - y_k + \nu_{\tilde{y}_{i,k}} = \breve{y}_{i,k} + \nu_{\tilde{y}_{i,k}}$ and $\boldsymbol{\nu}_{i,k} = [\nu_{\tilde{x}_{i,k}}, \nu_{\tilde{y}_{i,k}}]^T = \mathcal{N}(0, \boldsymbol{R}_{i,k}^{sen,c})$ represents measurement noise. The measurement pdf after converting the translated vector to the polar coordinates is approximated to a Gaussian $\tilde{\boldsymbol{z}}_{i,k}^{obj,p} = \mathcal{N}(\boldsymbol{\mu}_{i,k}^{obj,p}, \boldsymbol{R}_{i,k}^{obj,p})$ with $\tilde{\boldsymbol{z}}_{i,k}^{obj,p} = [\tilde{r}_{i,k}, \tilde{\theta}_{i,k}]^T$, $\boldsymbol{\mu}_{i,k}^{obj,p} = [\bar{r}_{i,k}, \bar{\theta}_{i,k}]^T$ and $\boldsymbol{R}_{i,k}^{obj,p} = \begin{bmatrix} \sigma_{\tilde{r}_{i,k}}^2 & \sigma_{\tilde{r}_{i,k}\tilde{\theta}_{i,k}}^2 \\ \sigma_{\tilde{\theta}_{i,k}\tilde{r}_{i,k}}^2 & \sigma_{\tilde{\theta}_{i,k}}^2 \end{bmatrix}$ where

$$\sigma_{\tilde{r}_{i,k}}^2 = \sigma_{\tilde{x}_{i,k}}^2 \cos^2(\bar{\theta}_{i,k}) + \varrho_{i,k} + \sigma_{\tilde{y}_{i,k}}^2 \sin^2(\bar{\theta}_{i,k}) \tag{A.6}$$

$$\sigma_{\tilde{\theta}_{i,k}}^2 = \frac{\sigma_{\tilde{x}_{i,k}}^2 \sin^2(\bar{\theta}_{i,k}) - \varrho_{i,k} + \sigma_{\tilde{y}_{i,k}}^2 \cos^2(\bar{\theta}_{i,k})}{\bar{r}_{i,k}^2} \tag{A.7}$$

$$\varrho_{i,k} = 2\sigma_{\tilde{x}_{i,k}\tilde{y}_{i,k}} \cos(\bar{\theta}_{i,k}) \sin(\bar{\theta}_{i,k}) \tag{A.8}$$

$$\rho_{\tilde{r}_{i,k}\tilde{\theta}_{i,k}} = \frac{(-\sigma_{\tilde{x}_{i,k}}^2 + \sigma_{\tilde{y}_{i,k}}^2)\sin(2\bar{\theta}_{i,k}) + 2\sigma_{\tilde{x}_{i,k}\tilde{y}_{i,k}}\cos(2\bar{\theta}_{i,k})}{2\sigma_{\tilde{r}_{i,k}}\sigma_{\tilde{\theta}_{i,k}}\bar{r}_{i,k}} \tag{A.9}$$

$$\sigma_{\tilde{r}_{i,k}\tilde{\theta}_{i,k}} = \sigma_{\tilde{\theta}_{i,k}\tilde{r}_{i,k}} = \rho_{\tilde{r}_{i,k}\tilde{\theta}_{i,k}}\sigma_{\tilde{r}_{i,k}}\sigma_{\tilde{\theta}_{i,k}}, \quad \bar{r}_{i,k} = \sqrt{\breve{x}_{i,k}^2 + \breve{y}_{i,k}^2}, \quad \bar{\theta}_{i,k} = \tan^{-1}\left(\frac{\breve{y}_{i,k}}{\breve{x}_{i,k}}\right) \tag{A.10}$$

The above conversions are approximate and this approximation is valid in the central and near central regions. If the angular error is $\sigma_{\tilde{\phi}} = 0.5°$, then the approximation becomes invalid at $10\sigma_{\tilde{\phi}}$. Similarly, if $\frac{\sigma_{\tilde{\psi}}}{\bar{\psi}} = 0.01$, then 5% error occurs at $5\sigma_{\tilde{\psi}}$. The sensor errors in the ETT applications are generally lower and the above approximation remains valid.

Appendix B

Probability Density Function of $\tilde{\zeta}_{i,k}$

Given $\boldsymbol{\zeta}_{i,k} = \Big[\cos(\tilde{\theta}_{i,k}), \sin(\tilde{\theta}_{i,k})\Big]^T$, the Gaussian approximation of the pdf of $\boldsymbol{\zeta}_{i,k}$ is derived in this Appendix. Suppose, a cosine transformation is applied to a standard normal distribution $\beta \sim \mathcal{N}(0, \sigma^2_\beta)$. According to Euler's formula $\exp(\iota\beta) = \cos\beta + \iota\sin\beta$ and $\mathbb{E}[\exp(\iota\beta)] = \exp\Bigg(-\frac{\sigma^2_\beta}{2}\Bigg)$ where $\mathbb{E}[\cdot]$ represents the mathematical expectation operator. Also $\mathbb{E}[e^{\iota\beta}] = \mathbb{E}[\cos\beta + \iota\sin\beta] = \mathbb{E}[\cos\beta] + \iota\mathbb{E}[\sin\beta]$. As a result, the real and imaginary parts can be equated as $\Re\{\mathbb{E}[e^{\iota\beta}]\} = \exp\Bigg(-\frac{\sigma^2_\beta}{2}\Bigg) = \mathbb{E}[\cos\beta]$ and $\Im\{\mathbb{E}[e^{\iota\beta}]\} = 0 = \mathbb{E}[\sin\beta]$, respectively, where $\Re\{.\}$ and $\Im\{.\}$ represent the real and imaginary parts of the variable. Now consider cosine and sine transformations applied to $\tilde{\theta}_{i,k} \sim \mathcal{N}(\bar{\theta}_{i,k}, \sigma^2_{\tilde{\theta}_{i,k}})$ with $\sigma^2_{\tilde{\theta}_{i,k}} = \sigma^2_\beta$, $\mathcal{C}_{i,k} = \cos(\tilde{\theta}_{i,k})$ and $\mathcal{S}_{i,k} = \sin(\tilde{\theta}_{i,k})$. Given that $\beta = \tilde{\theta}_{i,k} - \bar{\theta}_{i,k}$, the mean and variances are approximated as follows:

$$\mu_{\mathcal{C}_{i,k}} = \mathbb{E}[\cos(\tilde{\theta}_{i,k})] = e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}} \cos\bar{\theta}_{i,k}, \tag{B.1}$$

$$\mu_{\mathcal{C}^2_{i,k}} = \mathbb{E}[\cos^2(\tilde{\theta}_{i,k})] = \frac{1}{2} + \frac{1}{2} e^{-2\sigma^2_{\tilde{\theta}_{i,k}}} \cos 2\bar{\theta}_{i,k}, \tag{B.2}$$

$$\sigma^2_{\mathcal{C}_{i,k}} = \mathbb{E}[\cos^2\tilde{\theta}_{i,k}] - (\mathbb{E}[\tilde{\theta}_{i,k}])^2 = \frac{1}{2} + \frac{1}{2} e^{-2\sigma^2_{\tilde{\theta}_{i,k}}} \cos 2\bar{\theta}_{i,k} - e^{-\sigma^2_{\tilde{\theta}_{i,k}}} \cos^2\bar{\theta}_{i,k}, \tag{B.3}$$

$$\mu_{\mathcal{S}_{i,k}} = e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}} \sin\bar{\theta}_{i,k}, \tag{B.4}$$

$$\sigma^2_{\mathcal{S}_{i,k}} = \frac{1}{2} - \frac{1}{2} e^{-2\sigma^2_{\tilde{\theta}_{i,k}}} \cos 2\bar{\theta}_{i,k} - e^{-\sigma^2_{\tilde{\theta}_{i,k}}} \sin^2\bar{\theta}_{i,k}, \tag{B.5}$$

$$\sigma^2_{\mathcal{C}_{i,k}\mathcal{S}_{i,k}} = \sigma^2_{\mathcal{S}_{i,k}\mathcal{C}_{i,k}} = \mathbb{E}[\{\cos(\tilde{\theta}_{i,k}) - \mathbb{E}(\cos(\tilde{\theta}_{i,k}))\} \times \{\sin(\tilde{\theta}_{i,k}) - \mathbb{E}(\sin(\tilde{\theta}_{i,k}))\}] = 0, \tag{B.6}$$

where $\mu_{\mathcal{C}_{i,k}}$ and $\mu_{\mathcal{S}_{i,k}}$ represent the mean, $\sigma^2_{\mathcal{C}_{i,k}}$ and $\sigma^2_{\mathcal{S}_{i,k}}$ represent the variances and $\sigma^2_{\mathcal{C}_{i,k}\mathcal{S}_{i,k}}$ and $\sigma^2_{\mathcal{S}_{i,k}\mathcal{C}_{i,k}}$ represent the covariances. Using above, the pdf can be approximated to a Gaussian $\tilde{\boldsymbol{\zeta}}_{i,k} \sim \mathcal{N}(\bar{\boldsymbol{\zeta}}_{i,k}, \boldsymbol{R}^{\tilde{\zeta}}_{i,k})$ where:

$$\bar{\boldsymbol{\zeta}}_{i,k} = \begin{bmatrix} \mu_{\mathcal{C}_{i,k}} \\ \mu_{\mathcal{S}_{i,k}} \end{bmatrix}, \boldsymbol{R}^{\tilde{\zeta}}_{i,k} = \begin{bmatrix} \sigma^2_{\mathcal{C}_{i,k}} & \sigma^2_{\mathcal{C}_{i,k}\mathcal{S}_{i,k}} \\ \sigma^2_{\mathcal{S}_{i,k}\mathcal{C}_{i,k}} & \sigma^2_{\mathcal{S}_{i,k}} \end{bmatrix}. \tag{B.7}$$

The approximation is valid in central and near central regions as explained in the Appendix A.

Appendix C

Product of Two Gaussians

The product of two Gaussians, required in the likelihood derivation, is presented in this Appendix. The product of two Gaussian random variables $\alpha = f_{m_1,\sigma_1^2}(x) \sim \mathcal{N}(x; m_1, \sigma_1^2)$ and $\beta = f_{m_2,\sigma_2^2}(x) \sim \mathcal{N}(x; m_2, \sigma_2^2)$ is given below:

$$\alpha \times \beta = f_{m_1,\sigma_1^2}(x) \times f_{m_2,\sigma_2^2}(x) = C_c f_{m_c,\sigma_c^2}(x), \tag{C.1}$$

$$C_c = f_{m_1,\sigma_1^2+\sigma_2^2}(m_2) = f_{m_2,\sigma_1^2+\sigma_2^2}(m_1), \tag{C.2}$$

$$m_c = \frac{1}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}}\Big(\frac{m_1}{\sigma_1^2} + \frac{m_2}{\sigma_2^2}\Big), \quad \sigma_c = \frac{1}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}}. \tag{C.3}$$

Also, for a random variable X and a constant c

$$\mathbb{E}[c.X] = c.\mathbb{E}[X], Var[c.X] = c^2 Var[X], \tag{C.4}$$

where $\mathbb{E}[\cdot]$ is expectation operator and $Var[\cdot]$ is variance. Using above two Gaussian operations, the pdf of the product of two Gaussians, $\boldsymbol{e}_{\zeta_{i,k}} \sim \mathcal{N}(0, \boldsymbol{R}_{i,k}^{\tilde{\zeta}})$ and $e_i^{GP} \sim \mathcal{N}(0, c_{\tilde{\theta}_{i,k}\tilde{\theta}_{i,k}})$ is determined as follows:

$$\boldsymbol{e}_{\zeta_{i,k}} e_i^{GP} = a\mathcal{N}(\boldsymbol{\mu}_{i,k}^{\tilde{\zeta}\tilde{G}}, \boldsymbol{R}_{i,k}^{\tilde{\zeta}\tilde{G}}), \tag{C.5}$$

$$\boldsymbol{\mu}_{i,k}^{\tilde{\zeta}\tilde{G}} = 0, \tag{C.6}$$

$$\boldsymbol{R}_{i,k}^{\tilde{\zeta}\tilde{G}} = \begin{bmatrix} \frac{\sigma^2_{\mathcal{C}_{i,k}} c_{\tilde{\theta}_{i,k}\tilde{\theta}_{i,k}}}{2\pi(\sigma^2_{\mathcal{C}_{i,k}} + c_{\tilde{\theta}_{i,k}\tilde{\theta}_{i,k}})^2} & 0 \\ 0 & \frac{\sigma^2_{\mathcal{S}_{i,k}} c_{\tilde{\theta}_{i,k}\tilde{\theta}_{i,k}}}{2\pi(\sigma^2_{\mathcal{S}_{i,k}} + c_{\tilde{\theta}_{i,k}\tilde{\theta}_{i,k}})^2} \end{bmatrix}. \tag{C.7}$$

Appendix D

Elements of $\tilde{Q}(T)$

The elements of the process noise covariance matrix are derived in this Appendix. Let

$$\tilde{\boldsymbol{Q}}(T) = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix}, \tag{D.1}$$

$$e^{\boldsymbol{A}T} = e^{-\varphi} \begin{bmatrix} \frac{(\varphi^2+2\varphi+2)}{2} & T(\varphi+1) & \frac{T^2}{2} \\ \frac{-\varphi^2\lambda}{2} & (-\varphi^2+\varphi+1) & \frac{-(T(\varphi-2))}{2} \\ \frac{\varphi\lambda^2(\varphi-2)}{2} & \varphi\lambda(\varphi-3) & \frac{(\varphi^2-4\varphi+2)}{2} \end{bmatrix} = \begin{bmatrix} F^E_{11} & F^E_{12} & F^E_{13} \\ F^E_{21} & F^E_{22} & F^E_{23} \\ F^E_{31} & F^E_{32} & F^E_{33} \end{bmatrix}, \tag{D.2}$$

where $\varphi = T\lambda$. The elements of $\boldsymbol{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}'; Ts)$ are derived below:

$$q_{11} = \int_0^T F^E_{13} F^E_{13} d\tau = -\frac{e^{-2\varphi}(16\varphi^4 + 32\varphi^3 + 48\varphi^2 + 48\varphi + 24) - 24}{128\lambda^5}, \tag{D.3}$$

$$q_{12} = \int_0^T F^E_{13} F^E_{23} d\tau = \frac{T^4 e^{-2\varphi}}{8} \tag{D.4}$$

$$q_{13} = \int_0^T F^E_{13} F^E_{33} d\tau = \frac{e^{-2\varphi}(-2\varphi^4 + 4\varphi^3 + 2\varphi^2 + 2\varphi + 1) - 1}{16\lambda^3} \tag{D.5}$$

$$q_{21} = q_{12}, \quad q_{31} = q_{13}, \quad q_{32} = q_{23} \tag{D.6}$$

$$q_{22} = \int_0^T F^E_{23} F^E_{23} d\tau = -\frac{e^{-2\varphi}(2\varphi^4 - 4\varphi^3 + 2\varphi^2 + 2\varphi + 1) - 1}{16\lambda^3}$$

$$q_{23} = \int_0^T F^E_{23} F^E_{33} d\tau = \frac{T^2 e^{-2\varphi}(\varphi - 2)^2}{8} \tag{D.7}$$

$$q_{33} = \int_0^T F^E_{33} F^E_{33} d\tau = -\frac{e^{-2\varphi}(2\varphi^4 - 12\varphi^3 + 22\varphi^2 - 10\varphi + 3) - 3}{16\lambda} \tag{D.8}$$

Appendix E

Jacobian of $\mathcal{H}(\tilde{z}_k^{obj,p})$

The Jacobian of the measurement function $\boldsymbol{H}_k$ is derived in this Appendix.

$$\boldsymbol{H}_k = \frac{d\boldsymbol{\mathcal{H}}(\tilde{\boldsymbol{z}}_k^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k}\bigg|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_{k|k-1}} = \left[\frac{d\boldsymbol{\mathcal{H}}(\tilde{\boldsymbol{z}}_{1,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k} \quad \dots \quad \frac{d\boldsymbol{\mathcal{H}}(\tilde{\boldsymbol{z}}_{N_k,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k}\right]^T_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_{k|k-1}}, \tag{E.1}$$

where

$$\frac{d\boldsymbol{\mathcal{H}}(\tilde{\boldsymbol{z}}_{i,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k} = \left[\frac{d\boldsymbol{h}_a(\tilde{\boldsymbol{z}}_{i,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k} \quad \frac{d\boldsymbol{h}_b(\tilde{\boldsymbol{z}}_{i,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k}\right]^T, \tag{E.2}$$

$$\frac{d\boldsymbol{h}_i(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k} = \left[\frac{d\boldsymbol{h}_i(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{dx_k}, \frac{d\boldsymbol{h}_i(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\dot{x}_k}, \frac{d\boldsymbol{h}_i(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{dy_k},\right.$$

$$\left.\frac{d\boldsymbol{h}_i(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\dot{y}_k}, \frac{d\boldsymbol{h}_i(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k^E}\right], i = a, b \tag{E.3}$$

$$\boldsymbol{h}_a(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k = x_k + e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}\cos(\bar{\theta}_{j,k})\tilde{c}_{j,k}, \quad \tilde{c}_{j,k} = \tilde{\boldsymbol{d}}_{j,k}\boldsymbol{x}_k^E, \tag{E.4}$$

$$\boldsymbol{h}_b(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k = y_k + e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}\sin(\bar{\theta}_{j,k})\tilde{c}_{j,k}, \quad \tilde{\boldsymbol{d}}_{j,k} = \left[\boldsymbol{C}_{\bar{\theta}_{j,k}\boldsymbol{\theta}} + \frac{\sigma^2_{\tilde{\theta}_{j,k}}\boldsymbol{C}''_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{2}\right]\boldsymbol{C}^{-1}_{\boldsymbol{\theta\theta}}. \tag{E.5}$$

The derivatives are calculated below:

Derivatives w.r.t x_k

The derivatives with respect to x_k are derived below.

$$\frac{d\boldsymbol{h}_a(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{dx_k} = 1 + \left[\frac{de^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}}{dx_k}\cos(\bar{\theta}_{j,k})\tilde{c}_{j,k} + e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}\left(\frac{d\cos(\bar{\theta}_{j,k})}{dx_k}\tilde{c}_{j,k} + \cos(\bar{\theta}_{j,k})\frac{d\tilde{c}_{j,k}}{dx_k}\right)\right], \tag{E.6}$$

$$\frac{d\bar{\boldsymbol{h}}_b(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{dx_k} = \frac{de^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}}{dx_k}\sin(\bar{\theta}_{j,k})\tilde{c}_{j,k} + e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}\left(\frac{d\sin(\bar{\theta}_{i,k})}{dx_k}\tilde{c}_{j,k} + \sin(\bar{\theta}_{i,k})\frac{d\tilde{c}_{j,k}}{dx_k}\right), \tag{E.7}$$

$$\frac{d\cos(\bar{\theta}_{j,k})}{dx_k} = \frac{(y_k - \bar{y}_{j,k})^2}{(x_k - \bar{x}_{j,k})^3\left(\frac{(y_k-\bar{y}_{j,k})^2}{(x_k-\bar{x}_{j,k})^2} + 1\right)^{\frac{3}{2}}}, \tag{E.8}$$

$$\frac{d\sin(\bar{\theta}_{j,k})}{dx_k} = \frac{-(y_k - \bar{y}_{j,k})}{(x_k - \bar{x}_{j,k})^2\left(\frac{(y_k-\bar{y}_{j,k})^2}{(x_k-\bar{x}_{j,k})^2} + 1\right)^{\frac{3}{2}}}, \tag{E.9}$$

$$\frac{d\tilde{c}_{j,k}}{dx_k} = \left[\frac{d\boldsymbol{C}_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{dx_k} + \frac{\sigma^2_{\tilde{\theta}_{j,k}}}{2}\frac{d\boldsymbol{C}''_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{dx_k} + \frac{\boldsymbol{C}''_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{2}\frac{d\sigma^2_{\tilde{\theta}_{j,k}}}{dx_k}\right]\boldsymbol{C}^{-1}_{\boldsymbol{\theta\theta}}\boldsymbol{x}^E_k, \tag{E.10}$$

$$\frac{d\boldsymbol{C}_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{dx_k} = \left[\frac{dk^E_\theta(\bar{\theta}_{j,k},\theta^{1'})}{dx_k} \quad \dots \frac{dk^E_\theta(\bar{\theta}_{j,k},\theta^{B'})}{dx_k}\right], \tag{E.11}$$

$$\frac{d\boldsymbol{C}''_{\boldsymbol{\theta}_i\boldsymbol{\theta}^E}}{dx_k} = \left[\frac{dk^{E''}_\theta(\bar{\theta}_{j,k},\theta^{1'})}{dx_k} \quad \dots \frac{dk^{E''}_\theta(\bar{\theta}_{j,k},\theta^{B'})}{dx_k}\right], \tag{E.12}$$

$$\frac{dk^E_\theta(\bar{\theta}_{j,k},\theta^{i'})}{dx_k} = \frac{\sigma^2_f}{l^2_\theta p}e^{-\frac{\sin^2\left(\frac{\bar{\theta}_{j,k}-\theta^{i'}}{p}\right)}{2l^2_\theta}}\cos\left(\frac{\bar{\theta}_{j,k} - \theta^{i'}}{p}\right)\left[-\frac{(y_k - \bar{y}_{j,k})}{(x_k - \bar{x}_{j,k})^2(\frac{(y_k-\bar{y}_{j,k})^2}{(x_k-\bar{x}_{j,k})^2} + 1)}\right], \tag{E.13}$$

$$\frac{dk^{E''}_\theta(\bar{\theta}_{j,k},\theta^{i'})}{dx_k} = \frac{2\theta^i - 2\tan^{-1}(\frac{y_k-\bar{y}_{i,k}}{x_k-\bar{x}_{i,k}})}{p}, \tag{E.14}$$

$$\frac{d\sigma^2_{\tilde{\theta}_{j,k}}}{dx_k} = \frac{d\frac{\sigma^2_{\tilde{x}_{j,k}}\sin^2(\bar{\theta}_{j,k}) - \varrho_{j,k} + \sigma^2_{\tilde{y}_{j,k}}\cos^2(\bar{\theta}_{j,k})}{\bar{\psi}^2_{j,k}}}{dx_k}. \tag{E.15}$$

Derivatives w.r.t y_k

The derivatives with respect to y_k are derived below.

$$\frac{d\boldsymbol{h}_a(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{dy_k} = \frac{de^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}}{dy_k}\cos(\bar{\theta}_{j,k})\tilde{c}_{j,k} + e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}\left(\frac{d\cos(\bar{\theta}_{j,k})}{dy_k}\tilde{c}_{j,k} + \cos(\bar{\theta}_{j,k})\frac{d\tilde{c}_{j,k}}{dy_k}\right), \tag{E.16}$$

$$\frac{d\boldsymbol{h}_b(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{dy_k} = 1 + \left(\frac{de^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}}{dy_k}\sin(\bar{\theta}_{j,k})\tilde{c}_{j,k} + e^{-\frac{\sigma^2_{\tilde{\theta}_{i,k}}}{2}}\left(\frac{d\sin(\bar{\theta}_{j,k})}{dy_k}\tilde{c}_{j,k} + \sin(\bar{\theta}_{j,k})\frac{d\tilde{c}_{j,k}}{dy_k}\right)\right), \tag{E.17}$$

$$\frac{d\cos(\bar{\theta}_{j,k})}{dy_k} = -\frac{2y_k - 2\bar{y}_{j,k}}{2(x_k - \bar{x}_{j,k})^2\left(\frac{(y_k-\bar{y}_{j,k})^2}{(x_k-\bar{x}_{j,k})^2} + 1\right)^{\frac{3}{2}}}, \tag{E.18}$$

$$\frac{d\sin(\bar{\theta}_{j,k})}{dy_k} = \frac{1}{(x_k - \bar{x}_{j,k})\left(\frac{(y_k-\bar{y}_{j,k})^2}{(x_k-\bar{x}_{j,k})^2} + 1\right)^{\frac{3}{2}}}, \tag{E.19}$$

$$\frac{d\tilde{c}_{j,k}}{dy_k} = \left[\frac{d\boldsymbol{C}_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{dy_k} + \frac{\sigma^2_{\tilde{\theta}_{j,k}}}{2}\frac{d\boldsymbol{C}''_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{dy_k} + \frac{\boldsymbol{C}''_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{2}\frac{d\sigma^2_{\tilde{\theta}_{j,k}}}{dy_k}\right]\boldsymbol{C}_{\boldsymbol{\theta\theta}}^{-1}\boldsymbol{x}_k^E, \tag{E.20}$$

$$\frac{d\boldsymbol{C}_{\bar{\theta}_{j,k}\boldsymbol{\theta}}}{dy_k} = \left[\frac{dk_\theta^E(\bar{\theta}_{j,k},\theta^{1'})}{dy_k} \quad \dots \frac{dk_\theta^E(\bar{\theta}_{j,k},\theta^{B'})}{dy_k}\right], \tag{E.21}$$

$$\frac{d\boldsymbol{C}''_{\boldsymbol{\theta}_i\boldsymbol{\theta}^E}}{dy_k} = \left[\frac{dk_\theta^{E''}(\bar{\theta}_{j,k},\theta^{1'})}{dy_k} \quad \dots \frac{dk_\theta^{E''}(\bar{\theta}_{j,k},\theta^{B'})}{dy_k}\right], \tag{E.22}$$

$$\frac{dk_\theta^E(\bar{\theta}_{j,k},\theta^{i'})}{dy_k} = \frac{\sigma_f^2}{l_\theta^2 p}e^{-\frac{\sin^2\left(\frac{\bar{\theta}_{j,k}-\theta^{j'}}{p}\right)}{2l_\theta^2}}\cos\left(\frac{\bar{\theta}_{j,k}-\theta^{j'}}{p}\right)\left[\frac{1}{(x_k - \bar{x}_{j,k})\left(\frac{(y_k-\bar{y}_{j,k})^2}{(x_k-\bar{x}_{i,k})^2} + 1\right)}\right], \tag{E.23}$$

$$\frac{dk_\theta^{E''}(\bar{\theta}_{j,k},\theta^{i'})}{dy_k} = \frac{2\theta^j - 2\tan^{-1}\left(\frac{y_k-\bar{y}_{j,k}}{x_k-\bar{x}_{j,k}}\right)}{p}, \tag{E.24}$$

$$\frac{d\sigma^2_{\tilde{\theta}_{j,k}}}{dy_k} = \frac{d\frac{\sigma^2_{\tilde{x}_{j,k}} \sin^2(\bar{\theta}_{j,k}) - \varrho_{j,k} + \sigma^2_{\tilde{y}_{j,k}} \cos^2(\bar{\theta}_{j,k})}{\bar{\psi}^2_{j,k}}}{dy_k}. \tag{E.25}$$

Derivatives w.r.t $\dot{x}_k$, $\dot{y}_k$

The derivatives with respect to $\dot{x}_k$ and $\dot{y}_k$ are derived below.

$$\frac{d\boldsymbol{h}_a(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\dot{x}_k} = \frac{d\boldsymbol{h}_b(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\dot{x}_k} = 0, \tag{E.26}$$

$$\frac{d\boldsymbol{h}_a(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\dot{y}_k} = \frac{d\boldsymbol{h}_b(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\dot{y}_k} = 0. \tag{E.27}$$

Derivatives w.r.t x_k^E

The derivatives with respect to x_k^E are derived below.

$$\frac{d\boldsymbol{h}_a(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k^E} = \cos(\bar{\theta}_{j,k})\tilde{\boldsymbol{d}}_{j,k}, \tag{E.28}$$

$$\frac{d\boldsymbol{h}_b(\tilde{\boldsymbol{z}}_{j,k}^{obj,p})\boldsymbol{x}_k}{d\boldsymbol{x}_k^E} = \sin(\bar{\theta}_{j,k})\tilde{\boldsymbol{d}}_{j,k}. \tag{E.29}$$