NEGATIVE INFORMATION FUSION FOR GAUSSIAN PROCESS BASED THREE-DIMENSIONAL EXTENDED TARGET TRACKING

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

BY

CEM GÜRKAN SÜR

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

MAY 2021

Approval of the thesis:

NEGATIVE INFORMATION FUSION FOR GAUSSIAN PROCESS BASED THREE-DIMENSIONAL EXTENDED TARGET TRACKING

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

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

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

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

Examining Committee Members:

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

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

Assist. Prof. Dr. Elif Vural
Electrical and Electronics Engineering Department, METU _____

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

Assist. Prof. Dr. Osman Serdar Gedik
Computer Engineering, AYBU _____

Date: **10.05.2021**

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

Name, Surname: Cem Gürkan Sür

Signature :

ABSTRACT

NEGATIVE INFORMATION FUSION FOR GAUSSIAN PROCESS BASED THREE-DIMENSIONAL EXTENDED TARGET TRACKING

Sür, Cem Gürkan
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assoc. Prof. Dr. Emre Özkan

May 2021, 114 pages

Extended target tracking refers to the estimation of extent of a target as well as its position, kinematics, and orientation. In this thesis, we compare performances of Gaussian process based extended target tracking methods. Additionally, we propose a method that uses negative information fusion in three-dimensional point cloud data to enhance extent estimates of the target. Wide-ranging simulations are carried out to demonstrate the performance of the proposed algorithm. All simulations are carried out on a modular environment to be able to easily integrate different scenarios and compare the performances of the different methods. Results are obtained with both simulated and real data to attain better performance comparisons.

Keywords: Extended Target Tracking, Gaussian Processes, Negative Information Fusion

ÖZ

GAUSS SÜREÇLERİ BAZLI ÜÇ BOYUTLU GENİŞLETİLMİŞ HEDEF TAKİBİ İÇİN NEGATİF BİLGİ BİRLEŞTİRMESİ

Sür, Cem Gürkan
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Emre Özkan

Mayıs 2021 , 114 sayfa

Genişletilmiş hedef takibi hedef boyutlarının yanı sıra konum, hız, yönelim bilgilerinin de bir arada kestirimine denir. Bu tezde, Gauss süreçleri tabanlı genişletilmiş hedef takibi metotlarının performansları karşılaştırılmaktadır. Buna ek olarak, negatif bilgi birleştirmesinin üç boyutlu nokta bulutu verilerinden daha iyi boyut kestirimi yapılmasında kullanılması hakkında bir metot önerilmektedir. Önerilen algoritmanın performansını göstermek için geniş kapsamlı benzetimler gerçekleştirilmiştir. Yapılan tüm benzetimler modüler bir ortamda gerçekleştirilmiş olup, farklı senaryoların kolayca entegre edilebilmesi ve farklı metotların performanslarının kolay bir şekilde karşılaştırılması hedeflenmiştir. Hem temsili hem de gerçek veri setinden alınan sonuçlarla algoritmanın performans karşılaştırması daha iyi bir şekilde elde edilmiştir.

Anahtar Kelimeler: Genişletilmiş Hedef Takibi, Gauss Süreçleri, Negatif Bilgi Birleştirmesi

To my all beloved ones

ACKNOWLEDGMENTS

I would like to express my sincere thanks and gratitude to my supervisor Assoc. Prof. Dr. Emre Özkan for his complete guidance, advice and help throughout this thesis.

I would like to express my appreciation to ASELSAN Inc. for providing me a peaceful working environment and my colleagues in ASELSAN Inc. for their great support.

I would like to thank Turkish Scientific and Technological Research Council (TÜBİTAK) for their financial support (2210-A) during my M.Sc. studies.

I also would like to thank Murat Kumru for his valuable ideas and help throughout this study. His help was priceless.

I would like to express my special thanks to my family for their endless support and unconditional love. Especially, I am very grateful to my dear mother Hülya for her perpetual boost throughout my education. I would not be successful without her.

Lastly, I have to thank love of my life Berna for her endless patience, care and love. This work would not be completed without her support.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vi

ACKNOWLEDGMENTS viii

TABLE OF CONTENTS ix

LIST OF TABLES xiv

LIST OF FIGURES xvi

LIST OF ABBREVIATIONS xix

CHAPTERS

1 INTRODUCTION 1

1.1 Motivation and Problem Definition 1

1.2 Proposed Methods and Models 2

1.3 Topics Studied in the Thesis 2

1.4 The Outline of the Thesis 3

2 BACKGROUND 5

2.1 Extended Target Tracking 5

2.1.1 Introduction 5

2.1.2 Extended Target Modelling 5

2.1.2.1 Target State Modelling 6

2.1.2.2 Measurement Modelling 7
2.1.2.3 Shape Modelling 7
2.1.2.4 Target Dynamics 8
2.1.3 Prevalent Methods in Extended Target Tracking 8
2.1.3.1 Random Matrix Models 8
2.1.3.2 Star Convex Shape Based Models 9
Random Hypersurface Models: 10
Gaussian Process Model: 10
2.2 Gaussian Processes 11
2.2.1 Gaussian Process Regression 12
2.2.2 Recursive Gaussian Process Regression 13
2.3 Negative Information Fusion 15
3 INFERENCE TECHNIQUES 17
3.1 Kalman Filter 17
3.1.1 Introduction 17
3.1.2 Kalman Filter Equations 18
3.1.2.1 Time Update (Prediction) 18
3.1.2.2 Measurement Update 19
3.2 Kalman Smoother 20
3.3 Extended Kalman Filter 22
3.3.1 Introduction 22
3.3.2 Extended Kalman Filter Equations 22
3.3.2.1 Time Update (Prediction) 23

3.3.2.2 Measurement Update 23
3.4 Extended Kalman Smoother 24
3.5 Unscented Kalman Filter 24
3.5.1 Unscented Transformation 25
3.5.2 Unscented Kalman Filter Equations 26
3.6 Unscented Kalman Smoother 27
4 EXTENDED TARGET TRACKING USING GAUSSIAN PROCESSES . . 31
4.1 ETTGP 2D Algorithm 31
4.1.1 Introduction 31
4.1.2 Extent Model for 2D Objects 32
4.1.3 GP Modelling for Target Extent 33
4.1.3.1 Mean Function 33
4.1.3.2 Covariance Function 34
4.1.4 State Space Model 35
4.1.4.1 Extent Model 36
4.1.4.2 Process Model 37
4.1.4.3 Measurement Model 37
4.1.4.4 Negative Measurement Model 39
Pseudo Measurements for Angular Constraint 41
Pseudo Measurements for Radial Constraint 43
4.1.5 Inference 44
4.1.6 UKF Ng. ETTGP 2D Algorithm's Overview 45
4.2 ETTGP 3D Algorithm 48

4.2.1 Introduction 48
4.2.2 Extent Model for 3D Objects 48
4.2.3 GP Modelling for Object Extent 48
4.2.3.1 Mean Function 49
4.2.3.2 Covariance Function 50
4.2.4 State Space Model 51
4.2.4.1 Extent Model 51
4.2.4.2 Process Model 52
Rotational Motion Model 53
4.2.4.3 Measurement Model 57
4.2.4.4 Negative Measurement Model 59
Pseudo Measurements for Angular Constraint 60
Pseudo Measurements for Radial Constraint 62
4.2.5 Inference 64
4.2.6 UKF Ng. ETTGP 3D Algorithm Overview 64
5 SIMULATION RESULTS AND DISCUSSIONS 67
5.1 ETTGP 2D Simulation Results and Discussions 67
5.1.1 ETTGP 2D Framework 68
5.1.2 Performance Measures 70
5.1.3 Computation Time 71
5.1.4 Results and Discussions 72
Individual Effects of Angular and Radial Constraints: 81
5.2 ETTGP 3D Simulation Results and Discussions 84

5.2.1 ETTGP 3D Framework 84

5.2.2 Performance Measures 86

5.2.3 Computation Time 87

5.2.4 Results and Discussions for Simulated Data 88

Individual Effects of Angular and Radial Constraints: 95

5.2.5 Results and Discussions for Real Data 98

6 CONCLUSIONS 109

REFERENCES 111

LIST OF TABLES

TABLES

Table 5.1 Simulation configurations for ETTGP 2D algorithms 69

Table 5.2 Parameter set used in ETTGP 2D simulations 70

Table 5.3 Performances of ETTGP 2D algorithms for scenario $O_2 - M_3 - S_1$ 72

Table 5.4 Performances of ETTGP 2D algorithms for scenario $O_2 - M_3 - S_2$ 76

Table 5.5 Performances of ETTGP 2D algorithms for scenario $O_2 - M_2 - S_1$ 79

Table 5.6 Performances of ETTGP 2D algorithms for scenario $O_2 - M_2 - S_2$ 79

Table 5.7 Performances of ETTGP 2D algorithms for scenario $O_3 - M_2 - S_1$ 80

Table 5.8 Performances of ETTGP 2D algorithms for scenario $O_3 - M_2 - S_2$ 80

Table 5.9 Performances of ETTGP 2D algorithms for scenario $O_3 - M_3 - S_1$ 80

Table 5.10 Performances of ETTGP 2D algorithms for scenario $O_3 - M_3 - S_2$ 81

Table 5.11 Individual effects of angular and radial constraints on the performances for scenario $O_2 - M_3 - S_2$ 81

Table 5.12 Simulation configurations for ETTGP 3D algorithms 85

Table 5.13 Parameter set used in ETTGP 3D simulations 86

Table 5.14 Performances of ETTGP 3D algorithms for scenario $O_2 - M_2 - S_2$ 95

Table 5.15 Performances of ETTGP 3D algorithms for scenario $O_2 - M_3 - S_2$ 95

Table 5.16 Individual effects of angular and radial constraints on the performances for scenario $O_2 - M_3 - S_2$ 97

LIST OF FIGURES

FIGURES

Figure 2.1 Approaches for shape modelling 8

Figure 2.2 Negative information example 16

Figure 3.1 Demonstration of the difference between filtering and smoothing 20

Figure 3.2 Visualization of different smoothing types 21

Figure 4.1 Representation of 2D shapes with radial function $r = f(\theta)$. (a) Target shape, (b) Radial function 32

Figure 4.2 Covariance function vs angle 35

Figure 4.3 Representation of a single measurement in global and local coordinate frames in ETTGP 2D algorithm 38

Figure 4.4 Demonstration of different extent realizations 40

Figure 4.5 Demonstration of angular constraints imposed by negative information 41

Figure 4.6 Demonstration of pseudo angular measurement generation . . . 42

Figure 4.7 Demonstration of constraints imposed by negative information . 44

Figure 4.8 Demonstration of pseudo radial measurements generation 45

Figure 4.9 Overview of UKF Ng. ETTGP 2D Algorithm 47

Figure 4.10 Representation of 3D shapes with radial function $r = f(\theta, \phi)$ in spherical coordinates 49

Figure 4.11 Representation of a single measurement in spherical coordinate system w.r.t. global and local coordinate frames in ETTGP 3D algorithm 58

Figure 4.12 Demonstration of radial constraint imposed by negative information 60

Figure 4.13 Demonstration of angular constraints imposed by negative information 61

Figure 4.14 Demonstration of pseudo angular measurement generation . . . 63

Figure 4.15 Overview of UKF Ng. ETTGP 3D Algorithm 66

Figure 5.1 Schematic of ETTGP 2D framework 69

Figure 5.2 A sample run from $O_2 - M_3 - S_1$ Scenario 73

Figure 5.3 Orientation estimation in $O_2 - M_3 - S_1$ Scenario 73

Figure 5.4 IoU results in $O_2 - M_3 - S_1$ Scenario 74

Figure 5.5 ETTGP 2D a sample run from scenario $O_2 - M_3 - S_2$ 75

Figure 5.6 Orientation estimation in scenario $O_2 - M_3 - S_2$ 75

Figure 5.7 IoU estimations in scenario $O_2 - M_3 - S_2$ 76

Figure 5.8 ETTGP 2D a sample run from scenario $O_3 - M_3 - S_2$ 77

Figure 5.9 Orientation estimation in scenario $O_3 - M_3 - S_2$ 78

Figure 5.10 IoU estimations in scenario $O_3 - M_3 - S_2$ 78

Figure 5.11 Individual effects of angular and radial constraints in scenario $O_2 - M_3 - S_2$ 82

Figure 5.12 Orientation estimations in scenario $O_2 - M_3 - S_2$ 83

Figure 5.13 IoU estimations in scenario $O_2 - M_3 - S_2$ 83

Figure 5.14 Schematic of ETTGP 3D Framework 85

Figure 5.15 ETTGP 3D sample run from scenario $O_2 - M_2 - S_2$ 89

Figure 5.16 ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$ 90

Figure 5.17 ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$ from (Top view) 91

Figure 5.18 ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$ (Right view) 92

Figure 5.19 ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$ (Front view) 93

Figure 5.20 ETTGP 3D IoU estimations in scenario $O_2 - M_2 - S_2$ 94

Figure 5.21 ETTGP 3D IoU estimations in scenario $O_2 - M_3 - S_2$ 94

Figure 5.22 Individual effects of angular and radial constraints in scenario $O_2 - M_3 - S_2$ 96

Figure 5.23 IoU estimations in scenario $O_2 - M_3 - S_2$ 97

Figure 5.24 Visualization of Kitti data set scenario 1 98

Figure 5.25 Visualization of Kitti data set scenario 2 99

Figure 5.26 Visualization of Kitti data set scenario 3 100

Figure 5.27 ETTGP 3D Kitti scenario 1 sample run 102

Figure 5.28 ETTGP 3D Kitti scenario 2 sample run 103

Figure 5.29 ETTGP 3D Kitti scenario 3 sample run 104

Figure 5.30 ETTGP 3D Kitti scenario 3 sample run (Top view) 105

Figure 5.31 ETTGP 3D Kitti scenario 3 sample run (Right view) 106

Figure 5.32 ETTGP 3D Kitti scenario 3 sample run (Front view) 107

LIST OF ABBREVIATIONS

ABBREVIATIONS

ETT	Extended Target Tracking
GP	Gaussian Processes
ETTGP	Extended Target Tracking using Gaussian Processes Algorithm
KF	Kalman Filter
EKF	Extended Kalman Filter
UKF	Unscented Kalman Filter
UKF Ng.	Unscented Kalman Filter with Negative Information Fusion
EKS	Extended Kalman Smoother
UKS	Unscented Kalman Smoother
UKS Ng.	Unscented Kalman Smoother with Negative Info. Fusion
IoU	Intersection Over Union
MC	Monte Carlo
IID	Independent Identically Distributed
RM	Random Matrix
RHF	Random Hypersurface
CV	Constant Velocity
CT	Coordinated Turn
CA	Constant Acceleration

SYMBOLS

B	Input matrix
F	State transition matrix
f	Dynamic Model Function
H	Output matrix
h	Measurement Model Function
$\mathbf{x}_k$	State at time k
$\mathbf{u}_k$	Input at time k
$\mathbf{q}_k$	Process noise at time k
$\mathbf{z}_k$	Measurement at time k
$\mathbf{r}_k$	Measurement noise at time k
Q	Process noise covariance
R	Measurement noise covariance
$\hat{\mathbf{x}}_{k+1\|k}$	Predicted (a priori) state estimate at time k
$P_{k+1\|k}$	Predicted (a priori) state covariance at time k
$\tilde{\mathbf{y}}_{k+1}$	Innovation (measurement) residual at time k
$P_{x_k y_k}$	Cross covariance at time k
S_k	Innovation covariance at time k
K_k	Kalman gain at time k
$\hat{\mathbf{x}}_{k+1\|k+1}$	Updated (a posteriori) state estimate at time k+1
$P_{k+1\|k+1}$	Updated (a posteriori) state covariance at time k+1
$\mathbf{m}_k^s$	Smoothed state estimate at time k
P_k^s	Smoothed state covariance estimate at time k
C_k	Smoother gain at time k
I	Identity matrix

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

Target tracking has many applications in both military and civil sectors such as air defense, surveillance, robotics etc. Independent of where target tracking is used, main motivation is estimating the trajectory of a target better using some sensor measurements. In target tracking applications, point object assumption is very common. Point object assumption essentially means that at most one measurement per scan from any target can be obtained from a sensor. This assumption is quite valid when resolution of the sensors are comparable with target size. However, sensors' resolution increased more and more in recent years. Furthermore, available computational power has also increased over years. Thus, algorithms that make use of multiple measurements from a target per scan are increasingly used. By utilizing multiple measurements, one can obtain more information from the target, such as its size, shape etc. This information can be further used for classification purposes. Extended Target Tracking (ETT) (also known as Extended Object Tracking (EOT) in some fields) algorithms are used for the estimation of the extent of the object jointly with the kinematics based on the assumption of multiple measurements.

First methods for ETT assumed a known geometric shape to represent the target extent, such as a stick, rectangle or a circle [1, 2, 3]. Random matrix (RM) approach (a.k.a. elliptical extend method) proposed by Koch [4] gained traction between these methods. Then, star convex shape based random hypersurface model (RHF) was proposed. Random hypersurface models are firstly used for extended target tracking applications in [5] by Baum. A new random hypersurface model, namely Gaussian

process (GP) model, was introduced in [6]. Unlike the random matrix model which represents shapes by ellipses, these models represented shapes by any star convex shape. A detailed literature survey of the ETT can be found in [7].

In this thesis, the main aim is to integrate the Negative Information Fusion concept with Extended Target Tracking with Gaussian Process (ETTGP) Algorithms to obtain more robust and accurate extent estimates.

1.2 Proposed Methods and Models

In this thesis, estimation of the target extent is achieved by the utilization of Gaussian Processes. Gaussian Processes is a common tool for machine learning applications [8]. Integration of Gaussian Processes to ETT problem enables us to estimate random star convex shaped objects as well as better accuracy compared to conventional methods such as RM and RHF models. The resulting algorithm is called ETTGP. Several nonlinear Kalman filters and smoothers are utilized for inference in this study such as Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), Extended Kalman Smoother (EKS), and Unscented Kalman Smoother (UKS). Firstly, two-dimensional version of the ETTGP with Negative Information Fusion algorithm (ETTGP 2D) is derived step by step. Then, a three-dimensional version of ETTGP with Negative Information Fusion algorithm (ETTGP 3D) is proposed.

1.3 Topics Studied in the Thesis

Topics studied in the thesis are as follows:

- Utilization of ETTGP algorithms enabled the estimation of the extent of random star shaped convex shapes along with the state kinematics in both 2D and 3D.
- Performances of different filters and smoothers such as EKF, UKF, EKS, UKS in ETTGP 2D and 3D algorithms are compared.

- Fusion of negative information with ETTGP algorithms enabled us to have better state estimates.
- A method for ETTGP 3D with negative information fusion is proposed. Results of the proposed algorithm are compared with the existing ETTGP 3D algorithms.

1.4 The Outline of the Thesis

Chapter 2 contains the necessary background. In the first part of Chapter 2 general ETT problem in the literature is discussed. After that, Gaussian processes is covered in detail. Then, negative information concept and benefits of negative information fusion are explained. Inference techniques used in the thesis are explained in Chapter 3. ETTGP algorithms for 2D and 3D are derived in Chapter 4. Simulation results along with discussions are given in Chapter 5. Lastly, conclusions are given in Chapter 6.

CHAPTER 2

BACKGROUND

2.1 Extended Target Tracking

2.1.1 Introduction

Extended target tracking simply refers to the estimation of the extent of a target along with its position, kinematics, and orientation. There are many well-established methods in the extended target tracking literature, such as random matrix models, proposed firstly in [4], and star convex shape based models proposed firstly in [5]. In this section, basics of extended target tracking will be explained. Common definitions and concepts are summarized at first. Then, target state, measurement, shape, and target dynamics are discussed concisely. At last, prevalent methods for extended target tracking are briefly introduced.

2.1.2 Extended Target Modelling

Extended target modelling comprised of state representation, measurement modelling and target dynamics as in standard target tracking application. However, it additionally requires shape (or extent) modeling. In this section, our aim is to present basics of extended target modeling.

In real life, every object, or target as referred in this study, has a spatial extent. Each target we try to represent has an area for 2D applications or volume for 3D applications. Nevertheless, there is a very widespread assumption called point target assumption in target tracking studies. This assumption stems from the fact that for

many applications resolution of the sensor is not sufficiently high compared to the target size to attain multiple measurements at a scan. However, resolution of the sensors increases as the technology constantly evolves and progresses. As resolution of the sensors becomes comparable to the target size, obtaining multiple measurements at a scan becomes possible. To summarize, point target tracking and extended target tracking can be defined as follows:

- **Point Target Tracking:** Target generates at most one, i.e., there might be missed detections, measurement at a scan. The resolution of the sensor is not sufficiently high compared to the target size.
- **Extended Target Tracking:** Target generates multiple measurements at a scan. In other words, resolution of the sensor is sufficiently high compared to the target size.

Having described the difference between point and extended target tracking, let us examine how to model an extended target tracking problem in the subsequent sections.

2.1.2.1 Target State Modelling

Extended target state models usually contain position, velocity, direction, and extent, i.e., size and shape, of the target direction. Thus, extended target state can simply be divided to three as follows:

- **Position:** Target position in 2D (x, y) or 3D (x, y, z).
- **Kinematics:** Parameters related to the motion of the target, e.g., orientation, velocity, acceleration etc.
- **Extent:** Parameters related to the size and shape of the target.

Parameters utilized in extended target state depends highly on the application and method. For instance, one angle is enough to represent orientation in a 2D extended target tracking problem. However, in 3D orientation (or pose) of the object should be represented with more angles.

2.1.2.2 Measurement Modelling

In extended target tracking applications, we assume that multiple measurements at each scan of the sensor is generated. Let $Z = \left\{\mathbf{z}^{(i)}\right\}_{i=1}^{n}$ denote the measurements collected at a scan. Extended target measurements are modelled by the conditional distribution, $p(Z|\mathbf{x})$, given state $\mathbf{x}$. This distribution is referred to as extended target measurement likelihood in general. Aim of the measurement modelling is to express both the number of measurements and distribution of the measurements over the extent of the target.

2.1.2.3 Shape Modelling

Shape modelling in extended target tracking depends highly on the specific application. Complexity of the tracking problem and shape of the target we wish to track affect shape modelling approach. These approaches can be summarized as follows:

1. Easiest approach is to neglect the extent information and using all available measurements to update a point target. This method is useful if estimating only the position and kinematics of the target is sufficient for the application.

2. Second approach is assuming a basic geometric shape such as stick, rectangle, ellipse etc. This approach is useful when there is a sound prior knowledge about the shape of the target. If the expected shape of the target is known, these representations come in handy.

3. Most advanced approach is to have a flexible model which is able to represent a variety of shapes. This method is the most accurate of all, but it is very complex compared to other approaches.

These approaches are visualized in Fig. 2.1.

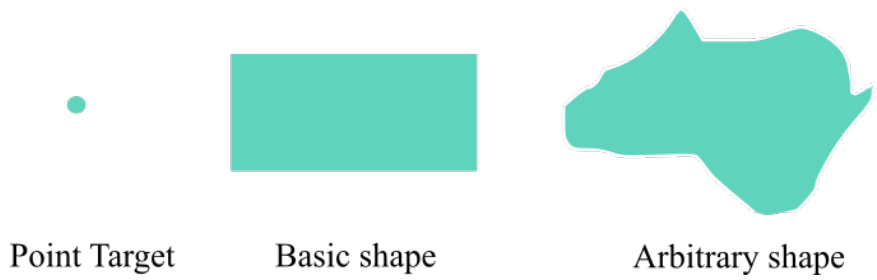

Figure 2.1: Approaches for shape modelling

2.1.2.4 Target Dynamics

Target dynamics defines how the extended target state changes over time. For a moving target, position, kinematics of the target, i.e., velocity, acceleration etc., and changes in the extent over time that define the motion of the extended target can be considered as target dynamics. In general, dynamics of the position and kinematics of the target can be represented with standard models in point target tracking. Constant velocity (CV), constant acceleration (CA), nearly constant velocity model, and coordinated turn (CT) are few examples of these common methods. We assume that target size and shape does not change over time if the target is a rigid object. Hence, there is no dynamics involved. However, when the target makes a turn, the orientation of the extent changes. Thus, orientation of the object should also be estimated.

2.1.3 Prevalent Methods in Extended Target Tracking

Widespread models used in extended target tracking algorithms is briefly explained in this section.

2.1.3.1 Random Matrix Models

The random matrix (RM) model is first introduced in [4]. RM model extended target state with a combination of target kinematics, $\mathbf{x}_k$, and extent matrix X_k. $\mathbf{x}_k$ vector contains target position, velocity, acceleration etc. whereas X_k is an $n \times n$ matrix that represent extent of the target where n denotes dimension of the target. For a 2D target $n = 2$, whereas for a 3D target $n = 3$. X_k is assumed to be symmetric

and positive-definite which implies target extent is represented by an ellipse. Each measurement is assumed to be independent of one another. Measurement likelihood for a single measurement, $\mathbf{z}_{k,l} \in \mathbb{R}^m$, is modelled as

$$p(\mathbf{z}_{k,l}|\mathbf{x}_k, X_k) = \mathcal{N}(\mathbf{z}_{k,l};\ C_k\mathbf{x}_k, zX_k + R), \tag{2.1}$$

where C_k denotes the $n \times m$ measurement matrix, R denotes the measurement noise covariance matrix and z is a scaling factor. Posterior distribution of $\mathbf{x}_k$ can be represented as

$$\begin{aligned} p(\mathbf{x}_k, X_k|Z^k) &\approx p(\mathbf{x}_k|Z^k)p(X_k|Z^k), \\ &= \mathcal{N}(\mathbf{x}_k;\ m_{k|k}, P_{k|k}) \times \mathcal{IW}(X_k;\ \nu_{k|k}, V_{k|k}), \end{aligned} \tag{2.2}$$

where estimated kinematics, $\mathbf{x}_k$, is Gaussian with mean $m_{k|k}$ and covariance $P_{k|k}$. Inverse Wishart distribution with scalar degrees of freedom, ν, and parameter matrix V is denoted with $\mathcal{IW}(X_k;\ \nu_{k|k}, V_{k|k})$.

2.1.3.2 Star Convex Shape Based Models

Random hypersurface models are firstly used for extended target tracking applications in [5]. A new random hypersurface model, namely Gaussian process model, is introduced in [6]. Unlike the random matrix model which represent shapes by ellipses, these models represent shapes by any star convex shape. Definition of star convex shapes is presented in Sec. 4.1.2. Both of these models have the following attributes:

- Extent of the target is expressed parametrically.
- Gaussian distribution is utilized to express uncertainties of the parameters in the state vector.
- Nonlinear Kalman filters, e.g., EKF, UKF, are utilized for estimation.

Random Hypersurface Models: Star convex shapes are represented via Fourier series expansion in Random hypersurface models. A radius function, $f(\mathbf{p}_k, \phi)$, is used for parametrisation where $\mathbf{p}_k$ denotes shape parameter vector and ϕ denotes the angle from the center of the target to the extent. A finite dimensional $\mathbf{p}_k$ can be expressed by Fourier series expansion [9]

$$\begin{aligned} f(\mathbf{p}_k, \phi) &= \mathbf{p}_k R(\phi), \quad \text{where} \\ \mathbf{p}_k &= \left[a_k^{(0)},\ a_k^{(1)},\ b_k^{(1)},\ \ldots,\ a_k^{(N)},\ b_k^{(N)}\right]^\top, \\ \mathbf{R}(\phi) &= \left[\frac{1}{2},\ \sin(\phi),\ \ldots,\ \cos(N\phi),\ \sin(N\phi)\right], \end{aligned} \tag{2.3}$$

where N is the number of Fourier coefficients. State vector $\mathbf{x}_k$ can be expressed as

$$\mathbf{x}_k = \left[\mathbf{p}_k^\top,\ \mathbf{c}_k^\top,\ \mathbf{v}_k^\top\right]^\top, \tag{2.4}$$

where $\mathbf{c}_k$ denotes position of the center of the target and $\mathbf{v}_k$ denotes kinematics. A general measurement equation can be given as

$$\mathbf{z}_k = f(\mathbf{p}_k, \phi_k) + \mathbf{c}_k + \mathbf{v}_k. \tag{2.5}$$

Having derived the measurement equation in (2.5), a nonlinear filter such as EKF, UKF can be applied for measurement update.

Gaussian Process Model: Star convex shapes are represented via Gaussian process in this model unlike Fourier series expansion used in random hypersurface method. GP model is covered extensively in the following section.

2.2 Gaussian Processes

A Gaussian process (GP) is a stochastic process, i.e., a generalization of a probability distribution to functions, that specifies a probability distribution in the function space. In essence, Gaussian processes are generalizations of Gaussian probability distributions [8].

A GP model is expressed as

$$\begin{aligned} f(u) &\sim \mathcal{GP}(\mu(u), k(u, u')), \\ \mu(u) &= \mathbb{E}[f(u)], \\ k(u, u') &= \mathbb{E}[(f(u) - \mu(u))(f(u') - \mu(u'))^\top], \end{aligned} \tag{2.6}$$

where $\mu(u)$ is mean, $k(u, u')$ is the covariance function (kernel) and u is the input of the function. One of the important characteristics of the GP is that it can be uniquely defined by the mean and covariance function.

A meaningful way to interpret GP is to think about it as a collection of random variables, for which any finite number of them have a joint Gaussian distribution that is consistent with the defined mean and kernel.

The joint distribution of the function evaluations, $f(u_1), ..., f(u_N)$, at the inputs, $u_1, ..., u_N$, are given as

$$\begin{bmatrix} f(u_1) \\ \vdots \\ f(u_N) \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, K), \quad \text{where} \tag{2.7}$$

$$\boldsymbol{\mu} = \begin{bmatrix} \mu(u_1) \\ \vdots \\ \mu(u_N) \end{bmatrix}, \text{ and } K = \begin{bmatrix} k(u_1, u_1) & \dots & k(u_1, u_N) \\ \vdots & & \vdots \\ k(u_N, u_1) & \dots & k(u_N, u_N) \end{bmatrix}.$$

2.2.1 Gaussian Process Regression

Gaussian process regression is a non-parametric method for learning of a regression function given some noisy observations. It is commonly used for learning unknown functions using some training data. Consider the following measurement model

$$z = f(u) + e, \quad \text{where} \quad e \sim \mathcal{N}(0, R), \tag{2.8}$$

where z is a noisy measurement, u is the training input and e is the measurement noise. Our aim is to find out the function values, $\mathbf{f} \triangleq [f(u_1^{\mathbf{f}}) \ \dots \ f(u_{N^{\mathbf{f}}}^{\mathbf{f}})]^\top$, at desired test inputs, $\mathbf{u}^{\mathbf{f}} \triangleq [u_1^{\mathbf{f}} \ \dots \ u_{N^{\mathbf{f}}}^{\mathbf{f}}]^\top$, given a set of measurements, $\mathbf{z} \triangleq [z_1 \ \dots \ z_N]^\top$, and respective inputs $\mathbf{u} \triangleq [u_1 \ \dots \ u_N]^\top$.

GP model given in (2.7) alongside the measurement model given in (2.8) leads to joint distribution

$$\begin{bmatrix} \mathbf{z} \\ \mathbf{f} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(\mathbf{u},\mathbf{u}) + I_N \otimes R & K(\mathbf{u},\mathbf{u}^{\mathbf{f}}) \\ K(\mathbf{u}^{\mathbf{f}},\mathbf{u}) & K(\mathbf{u}^{\mathbf{f}},\mathbf{u}^{\mathbf{f}}) \end{bmatrix}\right), \tag{2.9}$$

$$\text{where } K(\mathbf{u},\mathbf{u}^{\mathbf{f}}) = \begin{bmatrix} k(u_1, u_1^{\mathbf{f}}) & \dots & k(u_1, u_{N^{\mathbf{f}}}^{\mathbf{f}}) \\ \vdots & & \vdots \\ k(u_N, u_1^{\mathbf{f}}) & \dots & k(u_N, u_{N^{\mathbf{f}}}^{\mathbf{f}}) \end{bmatrix}.$$

I_N is defined as $N \times N$ identity matrix, and $\otimes$ is the Kronecker product. Mean function is set to zero for simplicity. Derivation of the GP regression for an arbitrary mean function is simple from this specific case [8].

The conditional distribution $p(\mathbf{f}|\mathbf{z})$ is given as

$$\begin{aligned} p(\mathbf{f}|\mathbf{z}) &\sim \mathcal{N}(A\mathbf{z}, P), \quad \text{where} \\ A &= K(\mathbf{u}^{\mathbf{f}},\mathbf{u})K_y^{-1}, \\ P &= K(\mathbf{u}^{\mathbf{f}},\mathbf{u}^{\mathbf{f}}) - K(\mathbf{u}^{\mathbf{f}},\mathbf{u})K_y^{-1}K(\mathbf{u},\mathbf{u}^{\mathbf{f}}), \\ K_y &= K(\mathbf{u},\mathbf{u}) + I_N \otimes R. \end{aligned} \tag{2.10}$$

2.2.2 Recursive Gaussian Process Regression

In order to perform GP regression, complete measurement vector, $\mathbf{z}$, and the covariance matrix, K_y, is needed. In target tracking applications, the main aim is to obtain the posterior density $p(\mathbf{f}|z_{1:k})$ at time k with measurements obtained sequentially. Thus, GP regression does not offer an answer for this online estimation problem. Furthermore, the computational complexity of GP regression method increases cubically with the number of measurements that makes it inappropriate for online applications. Hence, one needs an efficient recursive algorithm that will update the posterior density with only new measurements.

In order to unravel this problem, an approximation of the GP is proposed in [10, 11]. In this method, GP model is represented at a finite collection of basis points. Another recursive method that is similar to [10] is presented in [6]. We will utilize the method in [6] in this study.

To reiterate, aim of this method is to derive a formulation for the posterior density $p(\mathbf{f}|z_{1:N})$ which can be applied recursively. Therefore, the posterior can be represented as follows by iterative application of the Bayes' law

$$\begin{aligned} p(\mathbf{f}|z_{1:N}) &\propto p(z_N|\mathbf{f}, z_{1:N-1})p(\mathbf{f}|z_{1:N-1}), \\ &\propto \cdots \underbrace{p(z_k|\mathbf{f}, z_{1:k-1}) \cdots p(\mathbf{f})}_{p(\mathbf{f}|z_{1:k})}. \end{aligned} \tag{2.11}$$

In this step, we assume that $\mathbf{f}$ provides the sufficient statistics for z_k. With this assumption, z_k conditioned on $\mathbf{f}$ are independent of all the previous measurements

$$p(z_k|\mathbf{f}, z_{1:k-1}) \approx p(z_k|\mathbf{f}). \tag{2.12}$$

Note that this assumption is exact if the inputs of z_k are a subset of the inputs of $\mathbf{f}$. Aside from that, if the distance between the inputs of z_k and $\mathbf{f}$ are sufficiently small in comparison to the length scale of the kernel, this approximation is quite reasonable.

With the aforementioned assumptions, it is now possible to use recursive Bayesian inference for $\mathbf{f}$ given measurements, measurement likelihood and the initial prior densities. Joint distribution of the measurement z_k and $\mathbf{f}$ is obtained by using the GP definition in (2.9) as follows

$$\begin{bmatrix} z_k \\ \mathbf{f} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(u_k, u_k) + R & K(u_k, \mathbf{u}^{\mathbf{f}}) \\ K(\mathbf{u}^{\mathbf{f}}, u_k) & K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}) \end{bmatrix}\right). \tag{2.13}$$

Thus, the joint distribution in (2.13) alongside (2.10) leads to the following likelihood and prior densities

$$\begin{aligned} p(z_k|\mathbf{f}) &= \mathcal{N}(z_k; H_k^{\mathbf{f}}\mathbf{f}, R_k^{\mathbf{f}}), \\ p(\mathbf{f}) &= \mathcal{N}(\mathbf{0}, P_0^{\mathbf{f}}), \quad \text{where} \\ H_k^{\mathbf{f}} &= H^{\mathbf{f}}(u_k) = K(u_k, \mathbf{u}^{\mathbf{f}})[K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}})]^{-1}, \\ R_k^{\mathbf{f}} &= R^{\mathbf{f}}(u_k) = k(u_k, u_k) + R \\ &\quad - K(u_k, \mathbf{u}^{\mathbf{f}})[K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}})]^{-1} K(\mathbf{u}^{\mathbf{f}}, u_k), \\ P_0^{\mathbf{f}} &= K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}). \end{aligned} \tag{2.14}$$

2.3 Negative Information Fusion

Negative information phrase refers to the case that there is no object detected within a sensor's field of view although a detection is expected. Possible reasons of negative information are as follows:

- The object may be out of range.
- The object may be occluded by another object.
- The measurement may be false due to sensor failure.

Examples for the first two negative information cases above are visualized in Fig. 2.2. In this scenario, vehicles have a sensor that can obtain multiple measurements per scan. Blue cones in Fig. 2.2 show the field of view of the sensors. There are two vehicles, T_1 and T_2, in field of view of the V_1, thus two sets of measurements are obtained. For V_2, there are no objects to be detected as can be clearly seen from the figure. Thus, we have negative information. In V_3 case, measurements for T_2 is not obtained since T_2 is occluded by T_1.

The first two cases must be distinguished from the last one to be able to employ negative information correctly. An accurate sensor model is a necessity for utilizing negative information. One needs to model the measurement process and sensor characteristics precisely so that false negatives does not cause misinformation and lead to inaccurate results. Then, carefully obtained negative information can be fused with measurements in a Bayesian structure to obtain improved estimates. Lack of measurements from a sensor conveys information about where the target cannot be. One can obtain better extent estimates by combining positive and negative measurements considering the sensor to the target geometry.

Negative information in sensor data fusion is discussed with some examples in [12, 13]. In [14], negative information is utilized in Markov localization. Negative information fusion is used in star-convex shaped target tracking in [15]. A novel method utilizing angular and radial constraints to represent negative information is proposed in [16].

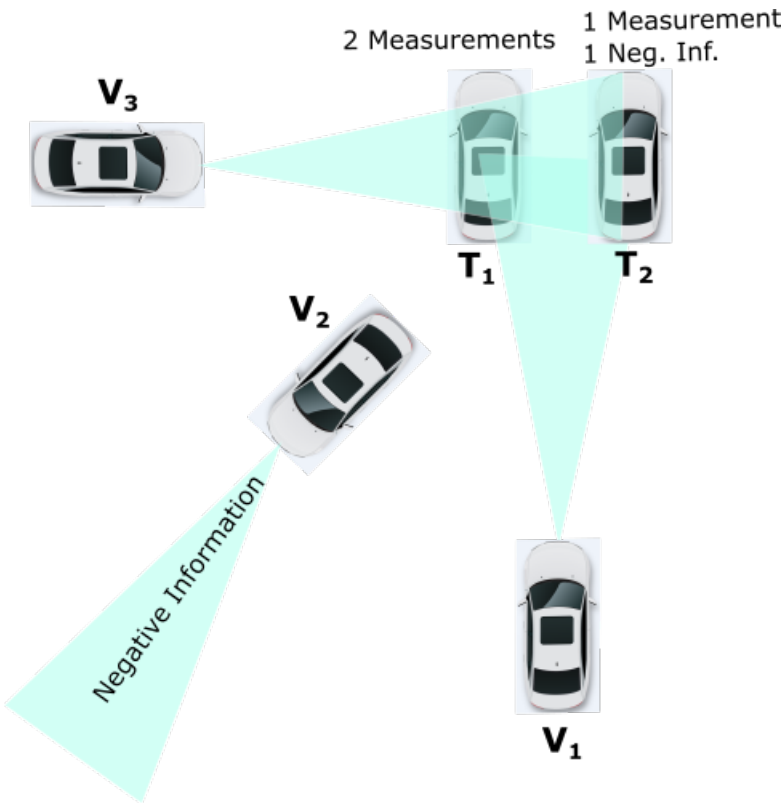

Figure 2.2: Negative information example

CHAPTER 3

INFERENCE TECHNIQUES

Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) are used in ETTGP 2D and ETTGP 3D algorithms for inference. Before explaining the details of the ETTGP algorithms, it is appropriate to exhibit the basics of these filters. In order to establish the basis for EKF and UKF, Kalman filter is explained firstly. It is paramount to understand KF since EKF and UKF are nonlinear extensions of it. Moreover, Kalman smoother (KS), Extended Kalman smoother (EKS), Unscented Kalman smoother (UKS) are discussed briefly in this chapter.

3.1 Kalman Filter

3.1.1 Introduction

Rudolf Emil Kalman published his prominent paper [17] that describes a recursive solution to the linear filtering problem in 1960. Due to its some important properties, which we will discuss later, Kalman filter was used extensively ever since. It provides a way to compute states of a process along with covariances efficiently. One of the important properties of the Kalman filter is that it minimizes mean squared error. It assumes the state-space model is linear with additive Gaussian process and measurement noises. With these assumptions, Kalman filter is an optimal filter.

This introduction includes a description and some discussion of the Kalman filter equations. It is important to explain the intricacies of the Kalman filter since other filters used in this study (EKF, UKF etc.) are also based on Kalman filter theory.

3.1.2 Kalman Filter Equations

In List of Abbreviations, notation for Kalman filter is given. The state-space model of the system can be written as

$$\begin{aligned} \mathbf{x}_{k+1} &= F\mathbf{x}_k + B\mathbf{u}_{k+1} + \mathbf{q}_k, \\ \mathbf{z}_{k+1} &= H\mathbf{x}_{k+1} + \mathbf{r}_{k+1}, \end{aligned} \tag{3.1}$$

where $\mathbf{q}_k$ (process noise) and $\mathbf{r}_{k+1}$ (measurement noise) are i.i.d. (independent identically distributed) and defined as

$$\begin{aligned} \mathbf{q}_k &= \mathcal{N}(0, Q), \\ \mathbf{r}_{k+1} &= \mathcal{N}(0, R), \end{aligned} \tag{3.2}$$

where Q is the process noise covariance and R is the measurement noise covariance.

Kalman filter uses this state-space model to predict the states for the next time interval. Furthermore, it uses the measurements to correct the prediction to obtain better estimates. This process continues recursively. Kalman filter algorithm is mainly divided into two steps, namely time update (prediction) and measurement update steps. In the following subsections, time update and measurement update steps are discussed.

3.1.2.1 Time Update (Prediction)

First one is the time update (prediction) step, which uses previously estimated states and the linear state-space model to predict the next state along with the state estimate covariance

$$\begin{aligned} \hat{\mathbf{x}}_{k+1|k} &= F\mathbf{x}_{k+1|k} + B\mathbf{u}_{k+1}, \\ P_{k+1|k} &= FP_{k|k}F^T + Q. \end{aligned} \tag{3.3}$$

3.1.2.2 Measurement Update

Second one is the measurement update step, that use the current measurements along with assumed statistics of the measurement noise of the model to correct the state estimate. In the measurement update step, innovation covariance and Kalman gain are calculated that are used in updating the mean and covariance of the posterior density of the states. Measurement update equations are given as

$$\begin{aligned}
\tilde{\mathbf{y}}_{k+1} &= \mathbf{z}_{k+1} - H\hat{\mathbf{x}}_{k+1|k}, \\
S_{k+1} &= HP_{k+1|k}H^T + R, \\
K_{k+1} &= P_{k+1|k}H^T S_{k+1}^{-1}, \\
\hat{\mathbf{x}}_{k+1|k+1} &= \hat{\mathbf{x}}_{k+1|k} + K_{k+1}\tilde{\mathbf{y}}_{k+1}, \\
P_{k+1|k+1} &= (I - K_{k+1}H)P_{k+1|k}.
\end{aligned} \tag{3.4}$$

These two steps are repeated for every sample: $k = 1, 2, \ldots, K$.

In order to have a deep understanding of Kalman filter, let's investigate innovation (measurement) residual and Kalman gain concepts.

Innovation residual is the difference between the true measurement ($\mathbf{z}_k$) and estimated measurement ($C\hat{\mathbf{x}}_{k|k-1}$). This residual is multiplied with the Kalman gain (K_k) and added to the predicted estimate ($\hat{\mathbf{x}}_{k|k-1}$) to correct the state estimates. Thus, we can interpret that innovation residual acts as a correction mechanism to compensate for the differences between predicted and true measurements.

Kalman gain acts as a correction factor in the update step. Its equation may not seem very intuitive. Let us look at each term of the equation. S_k is the estimated covariance of the measurements. If S_k is large, it means that measurements are not very reliable. Thus, one can expect as S_k increases K_k should decrease. It means that correction due to the measurements will be small since measurements are not trusted. $P_{k|k-1}$ is the estimated state covariance. If $P_{k|k-1}$ is large, it means that states can vary a lot. Hence, we need bigger K_k to correct the estimates using measurements. The last term, C^T, is needed for the transition from observations to states.

3.2 Kalman Smoother

Consider the estimation of state, $E[\mathbf{x}(t-\tau)|Z_0^t]$, given measurements up to time t, Z_0^t

- If $\tau < 0$, it is called prediction.
- If $\tau = 0$, it is called filtering.
- If $\tau > 0$, it is called smoothing.

Main difference between filtering and smoothing can be visualized in Fig. 3.1. Green bars in Fig. 3.1 denote the measurements.

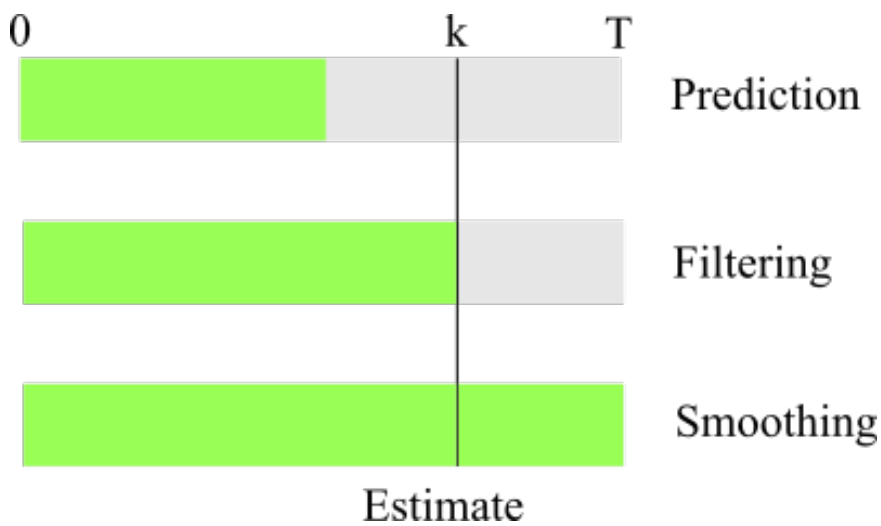

Figure 3.1: Demonstration of the difference between filtering and smoothing

Thus, smoothing (or retrodiction) can be defined as the estimation of the state at time k based on the prior measurements up to time $t > k$, $\hat{\mathbf{x}}(k|t) = E[\mathbf{x}(k)|Z^t]$. Smoothing types can be simply summarized as follows [18]:

1. **Fixed Point Smoothing:** k is fixed and $t = k+1, k+2, \ldots$
2. **Fixed Interval Smoothing:** Data interval is up to N ($t = N$) and $k = 0, 1, \ldots, N$.
3. **Fixed Lag Smoothing:** k is varying and $t = k + L$ where L is the defined lag.

Smoothing types can be visualized in Fig. 3.2.

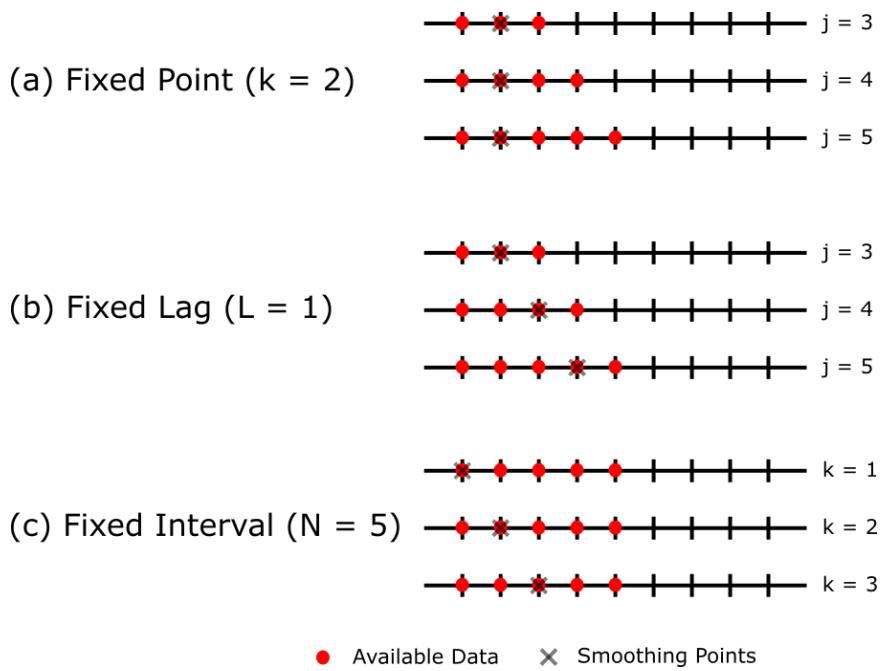

Figure 3.2: Visualization of different smoothing types

Kalman Smoother, also known as Rauch-Tung-Striebel-smoother (RTS), is introduced in 1965 [19]. Smoothing is the estimation of states at a desired time k based on the available information up to time T where $T > k$. Thus, Kalman smoother needs the whole Kalman filter estimates as an input. Kalman smoother is a backwards algorithm. Starting from the last time step, it estimates the smoothed states recursively. In List of Abbreviations, notation for Kalman smoother is given. Kalman Smoother equations [20] can be summarized as

$$
\begin{aligned}
\mathbf{m}_{k+1}^{-} &= F_k \mathbf{m}_k, \\
P_{k+1}^{-} &= F_k P_k F_k^T + Q_k, \\
C_k &= P_k F_k^T [P_{k+1}^{-}]^{-1}, \\
\mathbf{m}_k^s &= \mathbf{m}_k + C_k [\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^{-}], \\
P_k^s &= P_k + C_k [P_{k+1}^s - P_{k+1}^{-}] C_k^T.
\end{aligned}
\tag{3.5}
$$

3.3 Extended Kalman Filter

3.3.1 Introduction

Many practical problems encountered in real life are nonlinear and/or not Gaussian. In order to make use of the Kalman Filter theory, linear approximations are utilized. Extended Kalman Filter (EKF) [18] is one of these methods that exploit Taylor Series approximation for linearization in dynamic and/or measurement equation. EKF is widely used in the literature due to its effectiveness compared to its complexity. In the subsequent chapters details of EKF are given.

3.3.2 Extended Kalman Filter Equations

The state-space model of the system can be written as

$$\begin{aligned} \mathbf{x}_{k+1} &= f(\mathbf{x}_k) + \mathbf{q}_k, \\ \mathbf{z}_{k+1} &= h(\mathbf{x}_{k+1}) + \mathbf{r}_{k+1}, \end{aligned} \tag{3.6}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state vector, $\mathbf{z}_k \in \mathbb{R}^m$ is the measurement vector, $\mathbf{q}_k$ (process noise) and $\mathbf{r}_{k+1}$ (measurement noise) are i.i.d. (independent identically distributed) and defined as

$$\begin{aligned} \mathbf{q}_k &= \mathcal{N}(0, Q_k), \\ \mathbf{r}_{k+1} &= \mathcal{N}(0, R_{k+1}), \end{aligned} \tag{3.7}$$

where Q is the process noise covariance and R is the measurement noise covariance. For the sake of simplicity, it is assumed that there is no control input, and noises are additive with zero mean.

EKF approximates the density of $\mathbf{x}_k$ given the measurements $z_{1:k}$ with a Gaussian density as

$$p(\mathbf{x}_k|z_{1:k}) \approx N(\mathbf{x}_k|\mathbf{m}_k, P_k). \tag{3.8}$$

In List of Abbreviations, notation for Extended Kalman filter is given. EKF is composed of two main parts namely time update (prediction) and measurement update steps as in Kalman filter. Details of time and measurement update steps are given in subsequent sections.

3.3.2.1 Time Update (Prediction)

In the time update step, predicted state estimate, $\hat{\mathbf{x}}_{k+1|k}$, and predicted covariance matrix $P_{k+1|k}$ are calculated as

$$\begin{aligned} \hat{\mathbf{x}}_{k+1|k} &= \mathbf{f}(\hat{\mathbf{x}}_{k|k}), \\ J_k &= \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k|k}}, \\ P_{k+1|k} &= J_k P_{k|k} J_k^\top + BQB^\top, \end{aligned} \tag{3.9}$$

where J_k is the Jacobian of the dynamic model function $\mathbf{f}(\cdot)$ evaluated at the last state estimation $\hat{\mathbf{x}}_{k|k}$.

3.3.2.2 Measurement Update

Innovation, $\tilde{\mathbf{z}}$, innovation covariance matrix, S, and Kalman gain, K, are computed in measurement update step of the EKF firstly. After that, using aforementioned variables above; updated state estimate, $\hat{\mathbf{x}}_{k+1|k+1}$, and covariance matrix, $P_{k+1|k+1}$, are calculated as

$$
\begin{aligned}
\tilde{\mathbf{z}}_{k+1} &= \mathbf{z}_{k+1} - \mathbf{h}(\hat{\mathbf{x}}_{k+1|k}), \\
H_{k+1} &= \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k+1|k}}, \\
S_{k+1} &= H_{k+1} P_{k+1|k} H_{k+1}^{\top} + R, \\
K_{k+1} &= P_{k+1|k} H_{k+1}^{\top} S_{k+1}^{-1}, \\
\hat{\mathbf{x}}_{k+1|k+1} &= \hat{\mathbf{x}}_{k+1|k} + K_{k+1} \tilde{\mathbf{z}}_{k+1}, \\
P_{k+1|k+1} &= (I - K_{k+1} H_{k+1}) P_{k+1|k}.
\end{aligned}
\tag{3.10}
$$

Time and measurement updates are repeated recursively throughout estimation.

3.4 Extended Kalman Smoother

Extended Kalman Smoother (EKS) [21] is related to Kalman Smoother in the way EKF and KF are related. F_k, state transition matrix, in Kalman smoother equations is replaced with J_k, Jacobian. Thus, EKS equations can be given simply as

$$
\begin{aligned}
\mathbf{m}_{k+1}^{-} &= f(\mathbf{m}_k, k), \\
P_{k+1}^{-} &= J_k P_k J_k^T + Q_k, \\
C_k &= P_k J_k^T [P_{k+1}^{-}]^{-1}, \\
\mathbf{m}_k^s &= \mathbf{m}_k + C_k [\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^{-}], \\
P_k^s &= P_k + C_k [P_{k+1}^s - P_{k+1}^{-}] C_k^T.
\end{aligned}
\tag{3.11}
$$

3.5 Unscented Kalman Filter

Unscented Kalman filter (UKF) is another nonlinear filtering method that is widely used. UKF was proposed by Julier et al., in 1995 [22]. UKF is widely applied to many real life problems of signal estimation and target tracking since then. UKF utilizes set of points, so-called sigma points, to approximate the nonlinear function unlike EKF's one point, i.e., mean of the Gaussian, approximation. Mean and covariance information after nonlinear transformation is transferred by Unscented Transform (UT).

Main advantage of the UKF is the ability to procure higher order moments as opposed to EKF. Furthermore, Hessian and Jacobian matrices, needed in EKF, are not used which makes it easy to implement and less error prone. In order to understand UKF properly, firstly we discuss the UT.

3.5.1 Unscented Transformation

Unscented Transformation (UT) is a method that utilized for calculating the statistics of a random variable which goes through a nonlinear transformation. UT concept is based on the idea that "It is easier to approximate a probability density than it is to approximate an arbitrary nonlinear function or transformation" [23]. Firstly, a set of sigma points are specifically chosen in a way that their mean and covariance are $\mathbf{m}$ and P. After that, nonlinear transformation is applied to each sigma point. Lastly, statistics of the transformed sigma points are calculated to have an estimate of the transformed mean and covariance.

In List of Abbreviations, notation for Unscented Kalman filter is given.

A total of $2\,n+1$ sigma points (denoted by $\mathbf{X}$) are calculated from the columns of the matrix $\eta\sqrt{P}$ as follows

$$\begin{aligned}
\mathbf{X}^{(0)} &= \mathbf{m}, \\
\mathbf{X}^{(i)} &= \mathbf{m} + [\eta\sqrt{P}]_i \quad for \;\; i = 1, \ldots, n, \\
\mathbf{X}^{(i)} &= \mathbf{m} - [\eta\sqrt{P}]_i \quad for \;\; i = n+1, \ldots, 2\,n,
\end{aligned} \tag{3.12}$$

with the weights

$$\begin{aligned}
\mathbf{W}_m^{(0)} &= \frac{\lambda}{\eta^2}, \\
\mathbf{W}_c^{(0)} &= \frac{\lambda}{\eta^2} + (1 - \alpha^2 + \beta), \\
\mathbf{W}_m^{(i)} &= \frac{1}{2\eta^2} \quad for \;\; i = 1, \ldots, 2\,n, \\
\mathbf{W}_c^{(i)} &= \frac{1}{2\eta^2} \quad for \;\; i = 1, \ldots, 2\,n,
\end{aligned} \tag{3.13}$$

where $\mathbf{W}_m$ denote weights for the mean, $\mathbf{W}_c$ denote weights for the covariance, P is a positive semi-definite matrix such that $P = SS^\top$, $\eta = \sqrt{n+\lambda}$, λ is a scaling parameter defined as $\lambda = \alpha^2(n+\kappa) - n$, and $\alpha \geq 0$, $\beta \geq 0$, κ are scalar constants.

A discrete-time nonlinear system can be described as

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{q}_k, \\ \mathbf{z}_{k+1} &= \mathbf{h}(\mathbf{x}_{k+1}, \mathbf{u}_k + 1) + \mathbf{r}_{k+1}, \end{aligned} \tag{3.14}$$

where $\mathbf{x}_k \in \mathbb{R}^n$, $\mathbf{u}_k \in \mathbb{R}^v$, and $\mathbf{z}_k \in \mathbb{R}^p$ are, respectively, state variables, inputs, and observed measurements at time step k; the estimated mean and estimation error covariance are $\mathbf{m}$ and P; $\mathbf{f}$ and $\mathbf{h}$ are vectors consisting of nonlinear state transition functions and measurement functions; $\mathbf{q}_k \sim N(0, Q_k)$ is the Gaussian process noise at time step k; $\mathbf{r}_{k+1} \sim N(0, R_{k+1})$ is the Gaussian measurement noise at time step $k+1$; and Q_k and R_{k+1} are covariances of $\mathbf{q}_k$ and $\mathbf{r}_{k+1}$.

3.5.2 Unscented Kalman Filter Equations

Simple steps for UKF can be summarized as below:

1. Compute Sigma Points
2. Assign corresponding weights to each sigma point
3. Transform sigma points through non-linear function
4. Compute the Gaussian from transformed sigma points
5. Compute mean and variance of the new Gaussian.

Assume the initial estimated mean and the initial estimation error covariance are $\mathbf{m}_0$ and P_0,

Details of UKF algorithm is given in a prediction step and an update step, as in Algorithms 1 and 2.

Algorithm 1 UKF Time Update (Prediction)

1: Sigma points calculation

$$\mathbf{X}_{k-1} = \big[\underbrace{\mathbf{m}_{k-1} \cdots \mathbf{m}_{k-1}}_{2n+1}\big] + \eta \, \big[\mathbf{0}_{n,1} \quad \sqrt{P_{k-1}} \quad -\sqrt{P_{k-1}}\big]. \tag{3.15}$$

2: Dynamic model function evaluation at the sigma points

$$\hat{\mathbf{X}}_k = \mathbf{f}(\mathbf{X}_{k-1}). \tag{3.16}$$

3: State mean prediction

$$\mathbf{m}_k^- = \sum_{i=0}^{2n} \mathbf{W}_m^{(i)} \, \hat{\mathbf{X}}_{i,k}. \tag{3.17}$$

4: Covariance prediction

$$P_k^- = \sum_{i=0}^{2n} \mathbf{W}_c^{(i)} \, (\hat{\mathbf{X}}_{i,k} - \mathbf{m}_k^-)(\hat{\mathbf{X}}_{i,k} - \mathbf{m}_k^-)^\top + Q_{k-1}. \tag{3.18}$$

3.6 Unscented Kalman Smoother

Unscented Kalman (Rauch–Tung–Striebel, RTS) smoother (UKS) is proposed in [24]. UKS approximates the density of x_k given the measurements $z_{1:T}$ with a Gaussian density as

$$p(\mathbf{x}_k | z_{1:T}) \approx N(\mathbf{x}_k | {\mathbf{x}_k}^s, {P_k}^s). \tag{3.27}$$

Given the measurements up to time T where $T > k$, UKS recursively estimate the states at time k in a backwards manner. Details of UKS algorithm is given in Algorithm 3.

Algorithm 2 UKF Measurement Update

1: Sigma points prediction

$$\mathbf{X}_k^- = \big[\underbrace{\mathbf{m}_k^- \cdots \mathbf{m}_k^-}_{2n+1}\big] + \eta \left[\mathbf{0}_{n,1} \quad \sqrt{P_k^-} \quad -\sqrt{P_k^-}\right]. \tag{3.19}$$

2: Measurement model function evaluation at the sigma points

$$\mathcal{Y}_k^- = \mathbf{h}(\mathbf{X}_k^-). \tag{3.20}$$

3: Measurement prediction

$$\mathbf{y}_k^- = \sum_{i=0}^{2n} \mathbf{W}_m^{(i)} \mathcal{Y}_{i,k}^-. \tag{3.21}$$

4: Innovation covariance estimation

$$P_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k} = \sum_{i=0}^{2n} \mathbf{W}_c^{(i)} \big(\mathcal{Y}_{i,k}^- - \mathbf{y}_k^-\big)\big(\mathcal{Y}_{i,k}^- - \mathbf{y}_k^-\big)^\top + R_k. \tag{3.22}$$

5: Cross-covariance estimation

$$P_{\mathbf{x}_k \mathbf{y}_k} = \sum_{i=0}^{2n} \mathbf{W}_c^{(i)} \big(\mathbf{X}_{i,k}^- - \mathbf{m}_k^-\big)\big(\mathcal{Y}_{i,k}^- - \mathbf{y}_k^-\big)^\top. \tag{3.23}$$

6: Kalman gain estimation

$$K_k = P_{\mathbf{x}_k \mathbf{y}_k} P_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^{-1}. \tag{3.24}$$

7: Updated (a posteriori) state estimation

$$\mathbf{m}_k = \mathbf{m}_k^- + K_k \big(\mathbf{y}_k - \mathbf{y}_k^-\big). \tag{3.25}$$

8: Updated (a posteriori) covariance

$$P_k = P_k^- - K_k P_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k} K_k^\top. \tag{3.26}$$

Algorithm 3 Unscented Kalman Smoother

1: Calculate the matrix of sigma points of $\tilde{\mathbf{x}}_k = (x_k^T, q_k^T)^T$:

$$\tilde{\mathbf{X}}_k = [\underbrace{\tilde{\mathbf{m}}_k \cdots \tilde{\mathbf{m}}_k}_{2n+1}] + \sqrt{c}\,[0_{n,1} \quad \sqrt{\tilde{P}_k} \quad -\sqrt{\tilde{P}_k}], \quad \text{where} \tag{3.28}$$

$$\tilde{\mathbf{m}}_k = \begin{pmatrix} m_k \\ 0 \end{pmatrix}, \qquad \tilde{P}_k = \begin{pmatrix} P_k & 0 \\ 0 & Q_k \end{pmatrix}. \tag{3.29}$$

2: Dynamic model function evaluation at the sigma points

$$\tilde{\mathbf{X}}_{k+1} = \mathbf{f}(\tilde{\mathbf{X}}_k^x, \tilde{\mathbf{X}}_k^q). \tag{3.30}$$

3: State mean and covariance prediction

$$\mathbf{m}_{k+1}^- = \sum_{i=0}^{2n} \mathbf{W}_m^{(i-1)}\, \tilde{\mathbf{X}}_{i,k+1}, \tag{3.31}$$

$$P_{k+1}^- = \sum_{i=0}^{2n} \mathbf{W}_c^{(i-1)}\, (\tilde{\mathbf{X}}_{i,k+1} - \mathbf{m}_{k+1}^-)(\tilde{\mathbf{X}}_{i,k+1} - \mathbf{m}_{k+1}^-)^\top. \tag{3.32}$$

4: Cross covariance calculation

$$C_{k+1} = \sum_{i=0}^{2n} \mathbf{W}_c^{(i-1)}\, (\tilde{\mathbf{X}}_{i,k+1}^x - \mathbf{m}_{k+1}^-)(\tilde{\mathbf{X}}_{i,k+1} - \mathbf{m}_{k+1}^-)^\top. \tag{3.33}$$

5: Smoother gain calculation

$$D_k = C_{k+1}[P_{k+1}^-]^{-1}. \tag{3.34}$$

6: Smoothed mean and variance estimation

$$\mathbf{m}_k^s = \mathbf{m}_k + D_k\big(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-\big), \tag{3.35}$$

$$P_k^s = P_k + D_k\big(P_{k+1}^s - P_{k+1}^-\big)D_k^T. \tag{3.36}$$

CHAPTER 4

EXTENDED TARGET TRACKING USING GAUSSIAN PROCESSES

Necessary literature for the derivations of ETTGP 2D and ETTGP 3D algorithms such as Extended Target Tracking, Gaussian Processes, Negative Information Fusion and inference techniques are extensively discussed in the previous chapters. In this chapter, all the aforementioned concepts are fused to construct ETTGP 2D and ETTGP 3D algorithms.

Firstly, brief introductions for the algorithms are given. After that, extent models for 2D and 3D objects are discussed. GP Modelling for object extents are explained in the subsequent parts. Then, state space models utilized for the algorithms are discussed. Process and measurement model parts of the ETTGP 2D and ETTGP 3D filters are described later. After that, negative measurement models for ETTGP 2D and ETTGP3D are introduced. Then, inference models are explained. Lastly, overviews of UKF Ng. ETTGP 2D and 3D Algorithms are given.

4.1 ETTGP 2D Algorithm

4.1.1 Introduction

ETTGP 2D is an extended target tracking algorithm that utilizes Recursive Gaussian process regression to learn the shape (extent) of the target alongside kinematics [6]. ETTGP 2D algorithm is flexible enough to represent many shapes without the need of parameter tuning thanks to attractive analytical properties of Gaussian processes.

4.1.2 Extent Model for 2D Objects

Targets are generally modelled as point sources that generate one measurement per scan in traditional target tracking applications. However, we will assume that sensor resolution is sufficiently high in comparison to target's size so that multiple measurements along the extent of the target are generated. Target extent can be modelled in a simple manner using simple geometric shapes (stick, rectangle, ellipse etc.). Nevertheless, we chose to describe the extent via star convex shapes to represent various shapes more accurately.

Definition: A set $S(x_k)$ is called star convex if each vector from any point to the center is contained in $S(x_k)$.

Note that convex sets are subsets of star convex sets by definition. Assuming we have a star-convex shaped target, which is not a very restrictive assumption, target extent can be represented via a radial function $r = f(\theta)$ in polar coordinates as shown in Fig. 4.1.

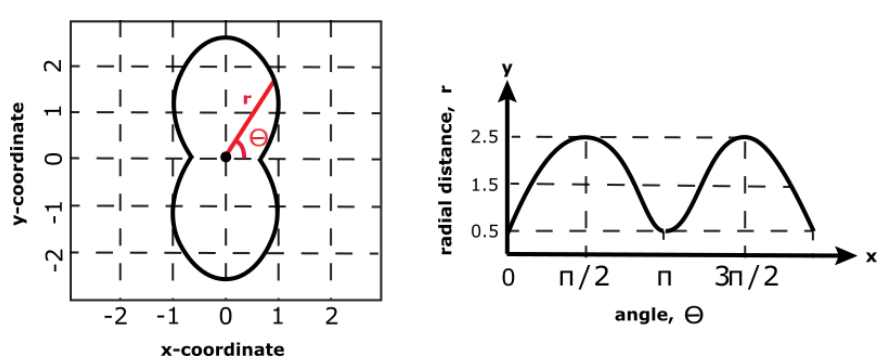

Figure 4.1: Representation of 2D shapes with radial function $r = f(\theta)$. (a) Target shape, (b) Radial function

Based on this representation, the measurement equation can be written as

$$\mathbf{z}_{k,l} = \mathbf{x}_k^c + \mathbf{p}(\theta_{k,l}) f(\theta_{k,l}) + \mathbf{e}_{k,l}, \tag{4.1}$$

where k denotes time index, $\mathbf{x}_k^c$ is the target position, $\{\mathbf{z}_{k,l}\}_{l=1}^{n_k}$ are the n_k measurements at time k, $\{\theta_{k,l}\}_{l=1}^{n_k}$ denote the angles which measurements originated, $\mathbf{e}_{k,l} \sim \mathcal{N}(0, R)$ is the measurement noise with zero mean and covariance R, and

$\mathbf{p}(\theta_{k,l})$ represents the orientation vector defined as

$$\mathbf{p}(\theta_{k,l}) \triangleq \begin{bmatrix} \cos(\theta_{k,l}) \\ \sin(\theta_{k,l}) \end{bmatrix}. \tag{4.2}$$

It is essential to emphasize that by defining star convex shape as in (4.1), $\mathbf{x}_k^c$ and $f(\theta)$ will not be unique. Same contour can be expressed by various pairs of $\mathbf{x}_k^c$ and $f(\theta)$.

Star convex shapes are used for target tracking applications firstly in [5]. In this work, unknown radial function, $f(\theta)$, is parametrized by Fourier series expansion. Fourier series expansion is a prominent choice for periodic signals. Nonetheless, it has some flaws in a stochastic setup. Thus, GP is chosen to model the radial function instead of Fourier series expansion. GP is a probabilistic model that enables the specification of the posterior density of the radial function naturally. Moreover, GP is utilized in spatial domain rather than frequency domain unlike Fourier series expansion. Uncertainty of extent representation is preserved due to local learning of the radial function. Thus, it is possible to use the uncertainty for gating and association.

In Sec. 4.1.4, unknown radial function will be augmented with target position and kinematics to derive the state space model which makes possible simultaneous estimation of extent and kinematics.

4.1.3 GP Modelling for Target Extent

Gaussian processes are extensively discussed in Sec. 2.2. In ETTGP 2D algorithm, our aim is to learn target extent online via Gaussian process. Input of the GP is chosen as polar angle, $\theta = u$, whereas output is the radius of extent, $r = y$, at the corresponding angles. Chosen mean and covariance functions of GP are explained in the following subsections.

4.1.3.1 Mean Function

The mean function, $\mu(\theta)$, is chosen to be constant but unknown for this study. $\mu(\theta) = r$ can be thought as the mean radius of the target extent.

$$f(\theta) \sim \mathcal{GP}\ (r, k(\theta, \theta')), \quad \text{where} \quad r \sim \mathcal{N}(0, \sigma_r^2). \tag{4.3}$$

Mean function in (4.3) can also be modelled as a zero mean GP by integrating out r

$$f(\theta) \sim \mathcal{GP}\ (0, k(\theta, \theta') + \sigma_r^2). \tag{4.4}$$

4.1.3.2 Covariance Function

Covariance function selection is an important aspect of GP modelling. Functions that will be learned via GP is governed by covariance function. There are various covariance functions that are utilized in the literature, however we chose to progress with the most common choice squared exponential covariance function [8]

$$k(\theta, \theta') = \sigma_f^2 e^{-\frac{|\theta-\theta'|^2}{2l^2}}. \tag{4.5}$$

σ_f^2 is the prior variance of the function amplitude and l represents the length scale of the functions we wish to learn. Squared exponential kernel results in higher correlations for $f(\theta)$ and $f(\theta')$ if angles θ and θ' are close to each other.

In order to make $\mathbf{f}(\cdot)$ periodic in terms of θ, (4.5) are updated as

$$k(\theta, \theta') = \sigma_f^2 e^{-\frac{\sin^2|\theta-\theta'|}{l^2}}. \tag{4.6}$$

Lastly, the effect of mean function described in Sec. 4.1.3.1 is added to covariance function which results in

$$k(\theta, \theta') = \sigma_f^2 e^{-\frac{\sin^2|\theta-\theta'|}{l^2}} + \sigma_r^2. \tag{4.7}$$

The derived covariance function is shown in Fig. 4.2. The covariance function has periodicity of 2π since $k(\theta, \theta') = k(\theta + 2\pi, \theta')$. Note that, radius for different angles is guaranteed to be positively correlated, and correlation increases with decreasing

angular distance. Moreover, this approach enables to model a variety of object sizes since we consider r to be a random variable.

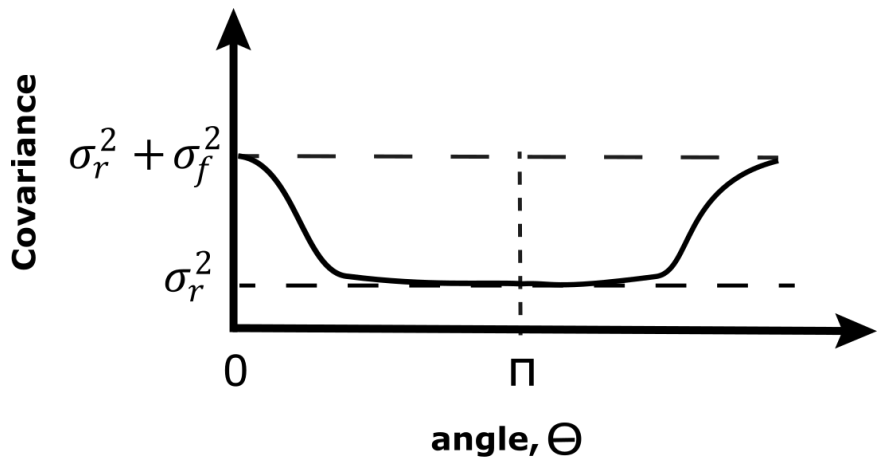

Figure 4.2: Covariance function vs angle

4.1.4 State Space Model

An augmented state space model will be derived in this section. Firstly, let us define state variables

$$\begin{aligned} \mathbf{x}_k &\triangleq [\bar{\mathbf{x}}_k^T \quad (\mathbf{x}_k{}^f)^T]^T, \quad \text{where} \\ \bar{\mathbf{x}}_k &\triangleq [(\mathbf{x}_k{}^c)^T \quad \psi_k \quad (\mathbf{x}_k{}^*)^T]^T, \end{aligned} \tag{4.8}$$

where $\bar{\mathbf{x}}_k$ denotes target state, $\mathbf{x}_k{}^f$ denotes states for extent estimation, $\mathbf{x}_k{}^c$ is the target position, ψ_k is the orientation, and $\mathbf{x}_k{}^*$ denote kinematics of the target (velocity and angular velocity). Having defined the state variables in (4.8), following augmented state space model can be written

$$\begin{aligned} \mathbf{x}_{k+1} &= F\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q_k), \\ \mathbf{z}_{k,l} &= \mathbf{h}_{k,l}(\mathbf{x}_k) + \mathbf{e}_{k,l}, \quad \mathbf{e}_{k,l} \sim \mathcal{N}(\mathbf{0}, R_{k,l}), \\ \mathbf{x}_0 &\sim \mathcal{N}(\boldsymbol{\mu}_0, P_0). \end{aligned} \tag{4.9}$$

Details for representation of target state and extent is presented in the subsequent sections.

4.1.4.1 Extent Model

Structure derived in (2.14) allows us to perform recursive regression on the following state space model

$$\begin{aligned} \mathbf{x}^{\mathbf{f}}_{k+1} &= \mathbf{x}^{\mathbf{f}}_{k}, \\ \mathbf{z}_k &= H^{\mathbf{f}}(\mathbf{u}_k)\, \mathbf{x}^{\mathbf{f}}_k + \mathbf{e}^{\mathbf{f}}_k, \quad \mathbf{e}^{\mathbf{f}}_k \sim \mathcal{N}(0, R^{\mathbf{f}}(u_k)), \\ \mathbf{x}^{\mathbf{f}}_0 &\sim \mathcal{N}(\mathbf{0}, P^{\mathbf{f}}_0), \end{aligned} \tag{4.10}$$

where $\mathbf{x}^f_k = \mathbf{f} = [f(u^{\mathbf{f}}_1) \ \dots \ f(u^{\mathbf{f}}_{N^{\mathbf{f}}})]^\top$ is the extent state. Furthermore, assuming that the extent of the target changes over time, this state space model can be modified as follows to incorporate dynamical behavior

$$\mathbf{x}^{\mathbf{f}}_{k+1} = F^{\mathbf{f}}\, \mathbf{x}^{\mathbf{f}}_k \ + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(0, Q^{\mathbf{f}}), \tag{4.11}$$

with

$$F^f = e^{\gamma T} I, \quad Q^{\mathbf{f}} = (1 - e^{2\gamma T}) K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}). \tag{4.12}$$

$\gamma \geq 0$ can be considered as a forgetting factor. As γ increases importance given to former measurements decreases and vice versa. With $\gamma = 0$, all measurement have the equal importance. It is important to note that choice of dynamics, $F^{\mathbf{f}}$ and $Q^{\mathbf{f}}$, in (4.12) guarantees the stationary covariance $K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}})$ to be irrespective of γ since

$$P = F^{\mathbf{f}} P (F^{\mathbf{f}})^T + Q^{\mathbf{f}} \quad \Longrightarrow \quad P = K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}). \tag{4.13}$$

The augmented state space model in Sec. 4.1.4 is constructed with the extent model and process model in Sec. 4.1.4.2.

4.1.4.2 Process Model

Target state $\bar{\mathbf{x}}_k = [({\mathbf{x}_k}^c)^T, \quad \psi_k, \quad ({\mathbf{x}_k}^*)^T]^T$ is represented with a linear state space model as

$$\begin{aligned} \bar{\mathbf{x}}_{k+1} &= \bar{F}\bar{\mathbf{x}}_k + \bar{\mathbf{w}}_k, \quad \bar{\mathbf{w}}_k \sim \mathcal{N}(\mathbf{0}, \bar{Q}), \\ \bar{\mathbf{x}}_0 &\sim \mathcal{N}(\bar{\mu}_{\mathbf{0}}, \bar{P}_0). \end{aligned} \tag{4.14}$$

Thus, the augmented target dynamics is constructed together with the extent model in (4.11)

$$\begin{aligned} \mathbf{x}_{k+1} &= F\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q), \\ \mathbf{x}_0 &\sim \mathcal{N}(\mu_{\mathbf{0}}, P_0), \end{aligned} \tag{4.15}$$

where

$$\begin{aligned} \mathbf{x}_k &= \begin{bmatrix} \bar{\mathbf{x}}_k \\ \mathbf{x}_k^{\mathbf{f}} \end{bmatrix}, \quad F = \begin{bmatrix} \bar{F} & 0 \\ 0 & F^{\mathbf{f}} \end{bmatrix}, \quad Q = \begin{bmatrix} \bar{Q} & 0 \\ 0 & Q^{\mathbf{f}} \end{bmatrix}, \\ \boldsymbol{\mu}_0 &= \begin{bmatrix} \bar{\boldsymbol{\mu}}_0 \\ \boldsymbol{\mu}_0^{\mathbf{f}} \end{bmatrix}, \quad P_0 = \begin{bmatrix} \bar{P}_0 & 0 \\ 0 & P_0^{\mathbf{f}} \end{bmatrix}, \; P_0^{\mathbf{f}} = K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}). \end{aligned} \tag{4.16}$$

Target state dynamics are chosen as constant velocity model

$$\bar{F} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \otimes I_3, \quad \bar{Q} = \begin{bmatrix} \frac{T}{3}^3 & \frac{T}{2}^2 \\ \frac{T}{2}^2 & T \end{bmatrix} \otimes \begin{bmatrix} \sigma_q^2 & 0 & 0 \\ 0 & \sigma_q^2 & 0 \\ 0 & 0 & \sigma_\psi^2 \end{bmatrix}. \tag{4.17}$$

4.1.4.3 Measurement Model

Each measurement obtained from the extent of the target is associated with an angle in global coordinate frame, $\boldsymbol{\theta}_{k,l}^G$, depending on their relative position to the target position, $\mathbf{x}_k^c$,

$$\boldsymbol{\theta}_{k,l}^{G}(\mathbf{x}_k^c) = \angle\left(\mathbf{z}_{k,l} - \mathbf{x}_k^c\right). \tag{4.18}$$

Measurements can also be represented in local coordinate frame as follows

$$\boldsymbol{\theta}_{k,l}^{L}(\mathbf{x}_k^c, \psi_k) = \boldsymbol{\theta}_{k,l}^{G}(\mathbf{x}_k^c) - \psi_k. \tag{4.19}$$

Representations in (4.18) and (4.19) can be visualized in Fig. 4.3 where $\mathbf{x}^G/\mathbf{y}^G$ $\mathbf{x}^L/\mathbf{y}^L$ denote basis vectors of global and local coordinate frames respectively.

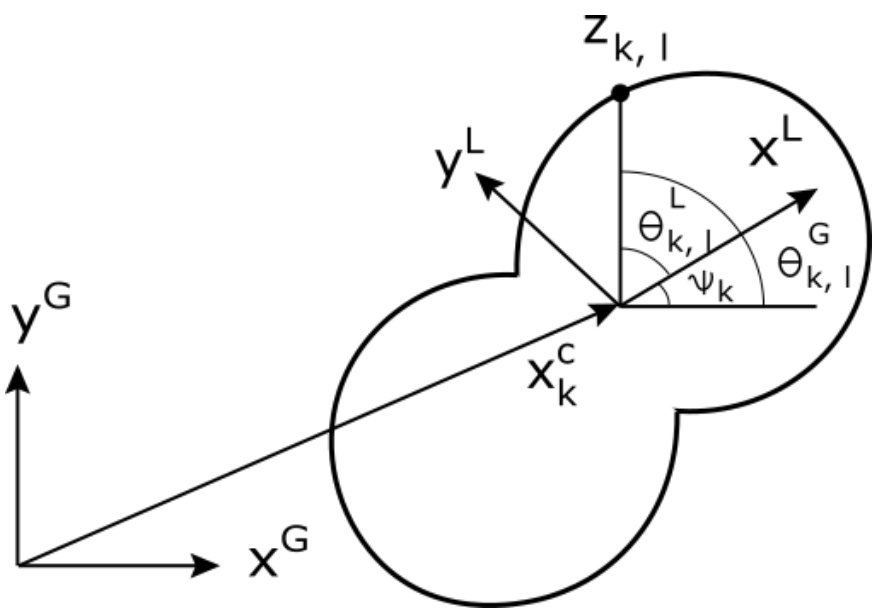

Figure 4.3: Representation of a single measurement in global and local coordinate frames in ETTGP 2D algorithm

With the angle representation in (4.19), we can simply define the relationship between target state and a single measurement as follows

$$\begin{aligned} \mathbf{z}_{k,l} &= \mathbf{x}_k^c + \mathbf{p}_{k,l}(\mathbf{x}_k^c) f\left(\boldsymbol{\theta}_{k,l}^{L}(\mathbf{x}_k^c, \psi_k)\right) + \bar{\mathbf{e}}_{k,l}, \\ \bar{\mathbf{e}}_{k,l} &\sim \mathcal{N}(\mathbf{0}, R_{k,l}). \end{aligned} \tag{4.20}$$

Orientation vector, $\mathbf{p}_{k,l}(\mathbf{x}_k^c)$, is formulated as follows by combining (4.2) and (4.18)

$$\mathbf{p}_{k,l}(\mathbf{x}_k^c) = \frac{\mathbf{z}_{k,l} - \mathbf{x}_k^c}{\|\mathbf{z}_{k,l} - \mathbf{x}_k^c\|}. \tag{4.21}$$

At this point, it is paramount to emphasize that the radial function we aim to learn using GP, $\mathbf{f}(\boldsymbol{\theta}_{k,l}^L(\mathbf{x}_k^c, \psi_k))$, takes angles in local coordinate frame as inputs. Moreover, note that $\boldsymbol{\theta}_{k,l}$ is a function of $\mathbf{x}_k^c$ and ψ_k.

Measurement equation for ETTGP algorithm can be formulated as follows by utilizing (2.14) discussed in Sec. 2.2.

$$\begin{aligned}
\mathbf{z}_{k,l} &= \mathbf{x}_k^c + \mathbf{p}_{k,l}(\mathbf{x}_k^c)\left[H^{\mathbf{f}}\left(\boldsymbol{\theta}_{k,l}^L(\mathbf{x}_k^c, \psi_k)\right)\mathbf{x}_k^{\mathbf{f}} + \mathbf{e}_{k,l}^{\mathbf{f}}\right] + \bar{\mathbf{e}}_{k,l}, \\
&= \underbrace{\mathbf{x}_k^c + \tilde{H}_l\left(\mathbf{x}_k^c, \psi_k\right)\mathbf{x}_k^{\mathbf{f}}}_{=\mathbf{h}_{k,l}(\mathbf{x}_k)} + \underbrace{\mathbf{p}_{k,l}(\mathbf{x}_k^c)\mathbf{e}_{k,l}^{\mathbf{f}} + \bar{\mathbf{e}}_{k,l}}_{=\mathbf{e}_{k,l}}, \\
&= \mathbf{h}_{k,l}(\mathbf{x}_k) + \mathbf{e}_{k,l}, \quad \mathbf{e}_{k,l} \sim \mathcal{N}(\mathbf{0}, R_{k,l}),
\end{aligned} \tag{4.22}$$

where

$$\begin{aligned}
\tilde{H}_l\left(\mathbf{x}_k^c, \psi_k\right) &= \mathbf{p}_{k,l}\, H^{\mathbf{f}}\left(\boldsymbol{\theta}_{k,l}^L(\mathbf{x}_k^c, \psi_k)\right), \\
\mathbf{h}_{k,l}(\mathbf{x}_k) &= \mathbf{x}_k^c + \tilde{H}_l\left(\mathbf{x}_k^c, \psi_k\right)\mathbf{x}_k^{\mathbf{f}}, \\
R_{k,l} &= \mathbf{p}_{k,l}\, R_{k,l}^{\mathbf{f}}\, \mathbf{p}_{k,l}^\top + R, \\
\mathbf{p}_{k,l} &= \mathbf{p}_{k,l}(\mathbf{x}_k^c), \quad R_{k,l}^{\mathbf{f}} = R^{\mathbf{f}}\left(\boldsymbol{\theta}_{k,l}^L(\mathbf{x}_k^c, \psi_k)\right).
\end{aligned} \tag{4.23}$$

Note that measurement noise in (2.14) is excluded since measurement noise, R, is already included in (4.23).

4.1.4.4 Negative Measurement Model

Negative Information Fusion concept is extensively discussed in Sec. 2.3. In this algorithm, negative measurement model introduced in [16] will be utilized. From now on, we will assume that if the target is in the range of the sensor there will be some measurements. To simply put, there will be no missed detections. By assuming so, we will incorporate negative information into the measurement model to obtain better

extent estimates. In this study, we will consider sensors that can measure distance as well as the angle from the sensor to the measurement source. These sensors, e.g., RADAR and LIDAR, are widely used in target tracking applications. We will utilize the angular and radial constraints on the measurements to derive a negative information update which sharpens the accuracy of state estimates. Note that all equations in this section is given for time instant k. For notational simplicity, index k in the variables are dropped.

Consider three realizations of the extent for given measurements in Fig. 4.4.[16] When we only pay regard to measurements, Case 1, case 2, and case 3 are all plausible since they all accurately explain $\{\mathbf{z}_l\}_{l=1}^{n}$. Nevertheless, if we account also for negative measurements it is clear that case 2 and case 3 are not feasible realizations. Assuming the target is in the field of view of the sensor, we would expect measurements scattered throughout the extent of the target. However, in case 2 one can observe that measurements are only collected from a small part of the extent. Thus, realization in case 2 is not plausible. As we will define later, it violates the angular constraint. In case 3, we can observe that measurements are obtained from the uppermost part of the extent. However, in reality we would expect exact the opposite since the uppermost part of the extent is occluded by the target itself. As we will define later, case 3 violates the radial constraint.

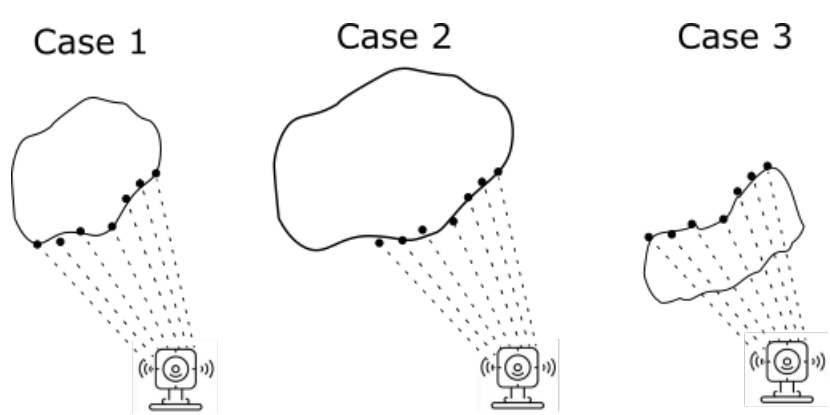

Figure 4.4: Demonstration of different extent realizations

In Fig. 4.7, angular and radial constraints are visualized. Angular constraint basically imposes limitation on the angular extent of the object whereas radial constraint restricts the radial distance between the sensor and visible parts of the target.

Pseudo Measurements for Angular Constraint : Angular extent of a target is related to minimum and maximum angles with respect to sensor position, ω_{min} and ω_{max}, as shown in Fig. 4.5. ω_{min} and ω_{max} can be simply found by the geometric relation between measurements and sensor position. Given a set of measurements, the angle from the sensor to the each measurement is calculated. Thus, minimum and maximum of these angles are ω_{min} and ω_{max} respectively. Let h_{min} and h_{max} denote the minimum and maximum angles calculated from the extent state x. ϑ_{min} and ϑ_{max} denote the noisy measurements of these quantities.

$$\begin{aligned} \vartheta_{min} &= h_{min}(\mathbf{x}) + r_{min}, \\ \vartheta_{max} &= h_{max}(\mathbf{x}) + r_{max}, \end{aligned} \tag{4.24}$$

where $\mathbf{x}$ is the state, h_{min} and h_{max} are nonlinear functions that generate minimum and maximum angles calculated from the extent, r_{min} and r_{max} are noises with $\sim \mathcal{N}(0, R_\vartheta)$. Note that pseudo angular measurements are functions of state. These pseudo angular measurements are compared with the actual minimum ad maximum angles calculated from the set of measurements.

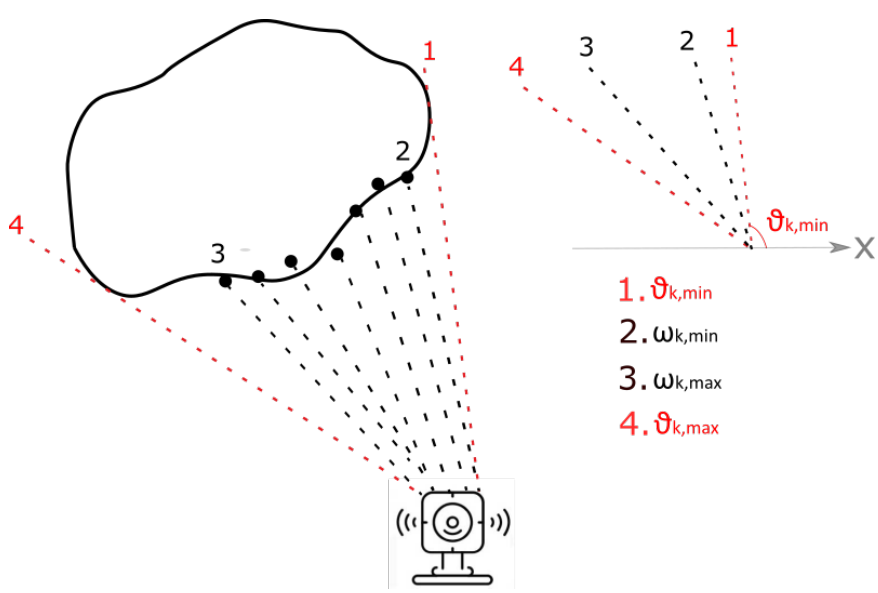

Figure 4.5: Demonstration of angular constraints imposed by negative information

Deriving analytical expressions for the nonlinear mappings are troublesome since $\mathbf{x}_k$ can represent any arbitrary shape. Hence, they will be calculated numerically.

Vectors from the sensor to the basis points $i = 1, \ldots, N$, $\mathbf{v}_i$, can be expressed as

$$\mathbf{v}_i = [x + r_i \cos(\theta_i + \psi) - s_x \quad y + r_i \sin(\theta_i + \psi) - s_y]^T, \tag{4.25}$$

where $\mathbf{s} = [s_x, \ s_y]^T$ denote the sensor coordinates. Now, we can easily find the minimum and maximum angles, h_{min} and h_{max} from h_i.

$$h_i = \arctan\left(\frac{y + r_i \sin(\theta_i + \psi) - s_y}{x + r_i \cos(\theta_i + \psi) - s_x}\right). \tag{4.26}$$

where h_i denote the nonlinear mappings for $i = 1, \ldots, N$. After that, ϑ_{min} and ϑ_{max} are used as pseudo angular measurements for additional measurement update step to improve extent estimation. This update step gets rid of the low likelihood realizations like case 2 in Fig. 4.4.

Demonstration of pseudo angular measurement generation is given in Fig. 4.6.

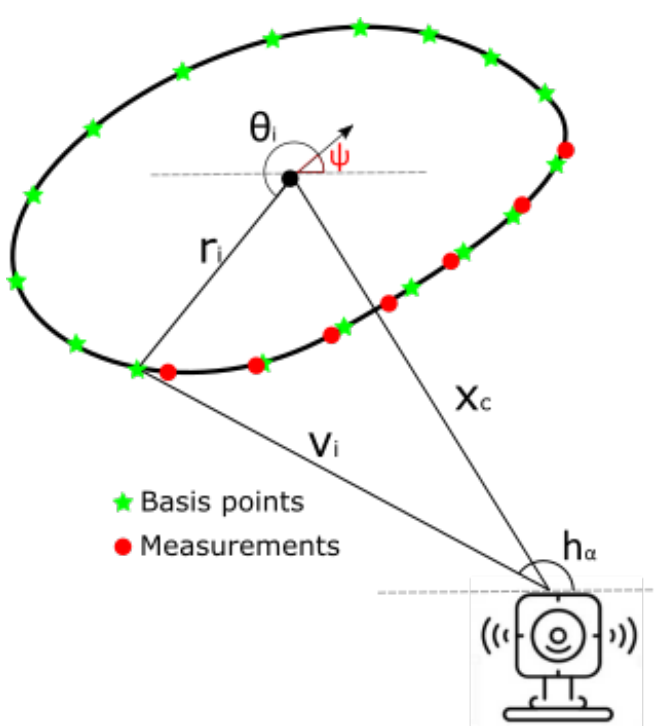

Figure 4.6: Demonstration of pseudo angular measurement generation

Pseudo Measurements for Radial Constraint : Radial distance between sensor and the target can be a filtering factor as we observed in case 3 in Fig. 4.4. In order to set a radial constraint, let us define the unit vectors, $\hat{\mathbf{u}}_l$, from sensor to the measurements $\{\mathbf{z}_l\}_{l=1}^{n}$

$$\hat{\mathbf{u}}_l = \frac{\mathbf{z}_l - \mathbf{s}}{\|\mathbf{z}_l - \mathbf{s}\|}. \tag{4.27}$$

Consider lines, g_l, that extend from $\mathbf{s}$ in the direction of $\hat{u}_l$. Each of the lines are checked to see if they intersect with the extent represented by state $\mathbf{x}$. Assume a subset of the obtained measurements, $m \leq n$, $\{\mathbf{z}_j\}_{j=1}^{m}$, intersects with the extent. For the newly formed subset of measurements, radial distance between the sensor and the measurements, d_j, can be written as

$$d_j = \|\mathbf{z}_j - \mathbf{s}\|. \tag{4.28}$$

Measurement model for the pseudo measurements that represent radial distance can be specified as

$$d_j = h_j(\mathbf{x}) + r_j, \tag{4.29}$$

where h_j denote nonlinear mappings which calculate the radial distance estimate between sensor and the extent represented by state $\mathbf{x}$ along $\hat{\mathbf{u}}_l$, d_j denote the noisy measurements of these quantities. r_j represent measurement noise for radial constraint with $\sim \mathcal{N}(0, R_d)$. Note that pseudo radial measurements are functions of state. These pseudo radial measurements are compared with the actual distances calculated from the set of measurements.

Similar to the angular constraints case, vectors from sensor to the basis points $i = 1, \ldots, N$, $\mathbf{v}_i$, can be calculated as in (4.25). From these N vectors, five closest vectors in the direction of $\hat{\mathbf{u}}_l$ are chosen for each measurement. Among these vectors, radial distance from the sensor to the basis points are calculated. The closest basis point is chosen to find h_j for each measurement $j = 1, \ldots, m$

$$h_j(\mathbf{X}) = \left| \begin{bmatrix} x + r_j \cos(\theta_j + \psi) - s_x \\ y + r_j \sin(\theta_j + \psi) - s_y \end{bmatrix} \right|. \tag{4.30}$$

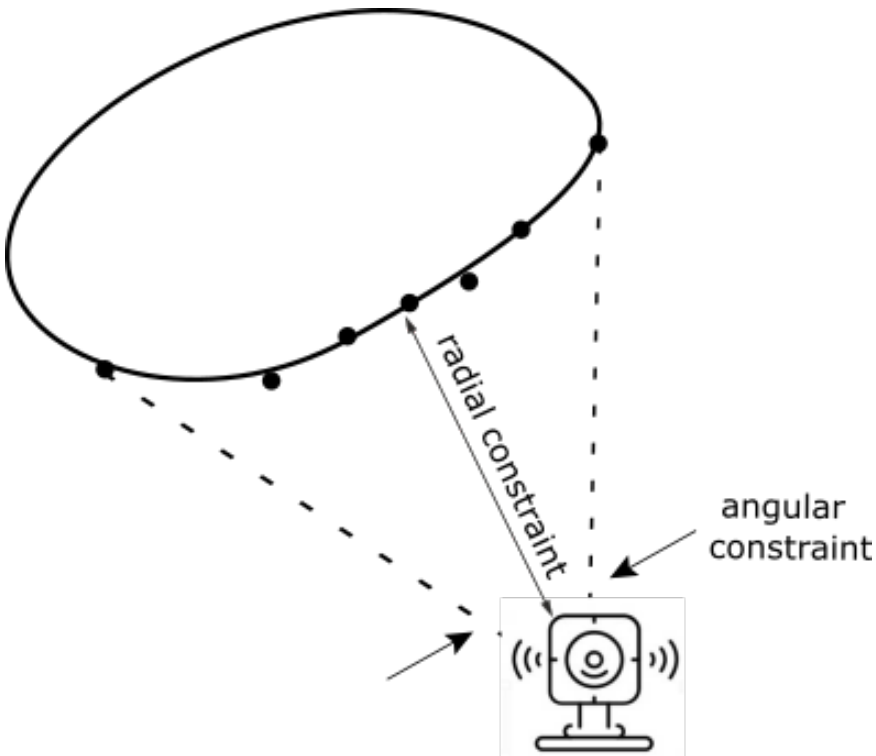

Figure 4.7: Demonstration of constraints imposed by negative information

After that, $\{d_j\}_{j=1}^{m}$ are used as pseudo measurements for additional measurement update step to improve extent estimation. This update step eliminates the low likelihood realizations like case 3 in Fig. 4.4.

Demonstration of pseudo radial measurements generation is given in Fig. 4.8.

4.1.5 Inference

Kalman filter based inference techniques can be applied to state space model formulated in (4.9) to compute posterior density of the state vector. In order to update posterior density recursively, we need to concatenate all measurements at an instant, $\{\mathbf{z}_{k,l}\}_{l=1}^{n_k}$, as follows

$$\begin{aligned} \mathbf{z}_k &= \left[\mathbf{z}_{k,1}^{\top},\ \ldots,\ \mathbf{z}_{k,n_k}^{\top}\right]^{\top}, \\ \mathbf{h}_k(\mathbf{x}_k) &= \left[\mathbf{h}_{k,1}(\mathbf{x}_k)^{\top},\ \ldots,\ \mathbf{h}_{k,n_k}^{\top}(\mathbf{x}_k)\right]^{\top}, \\ R_k &= \operatorname{diag}\left[R_{k,1},\ \ldots,\ R_{k,n_k}\right]. \end{aligned} \tag{4.31}$$

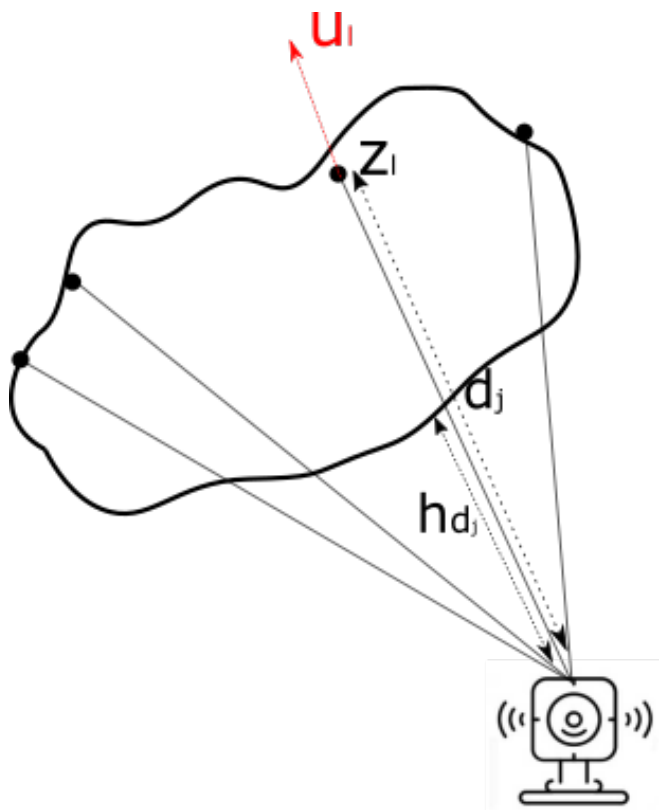

Figure 4.8: Demonstration of pseudo radial measurements generation

With this modification, we obtain the following state space model

$$
\begin{aligned}
\mathbf{x}_{k+1} &= F\mathbf{x}_k + \mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q), \\
\mathbf{z}_k &= \mathbf{h}_k(\mathbf{x}_k) + \mathbf{e}_k, \quad \mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, R_k), \\
\mathbf{x}_0 &\sim \mathcal{N}(\boldsymbol{\mu}_0, P_0).
\end{aligned} \tag{4.32}
$$

Using this structure, we can use a nonlinear filtering technique for state estimation. In this study, EKF discussed in Sec. 3.3 and UKF discussed in Sec. 3.5 are applied for inference.

4.1.6 UKF Ng. ETTGP 2D Algorithm's Overview

UKF Ng. ETTGP 2D Algorithm's overview is given in Algorithm 4. First four steps of the Algorithm 4 correspond to the UKF ETTGP 2D algorithm. Steps 6-10 correspond to the pseudo angular measurements update. Remaining steps correspond to

the pseudo radial measurements update. It is important to emphasize that, this algorithm consist of three measurement update steps with different types of measurements namely spatial measurements, pseudo angular measurements, and pseudo radial measurements. There are in total $\underbrace{n}_{Spatial} + \underbrace{2}_{Pseudo\ Angular} + \underbrace{m}_{Pseudo\ Radial}$ measurements. Algorithm 4 is visualized in Fig. 4.9.

Algorithm 4 UKF Ng. ETTGP 2D Overview

1: Perform UKF prediction as in Algorithm 1.
2: Calculate weights of sigma points as in (3.13).
3: Calculate sigma points as in (3.19).
4: Perform measurement update as in (3.20) together with (4.22) and (4.23).
5: Perform pseudo angular measurements update as in Steps 6-10.
6: Calculate minimum and maximum angles from sensor to the measurements, ω_{min} and ω_{max}.
7: Calculate weights of sigma points as in (3.13).
8: Calculate sigma points as in (3.19).
9: Calculate minimum and maximum angle estimates from the extent, ϑ_{min} and ϑ_{max}, using (4.24) and (4.26).
10: Perform measurement update with minimum and maximum angle estimates as in (3.21) − (3.26) together with (4.24) and (4.26).
11: Perform pseudo radial measurements update as in Steps 12-17.
12: Construct the measurement set for radial constraint implementation using (4.27).
13: Calculate minimum and maximum angles from sensor to the measurements, ω_{min} and ω_{max}.
14: Calculate sigma points as in (3.19).
15: Calculate weights of sigma points as in (3.13).
16: Calculate minimum radial distance estimates using (4.28) − (4.30).
17: Perform measurement update with minimum radial distance estimates as in (3.21) − (3.26) together with (4.28) − (4.30).

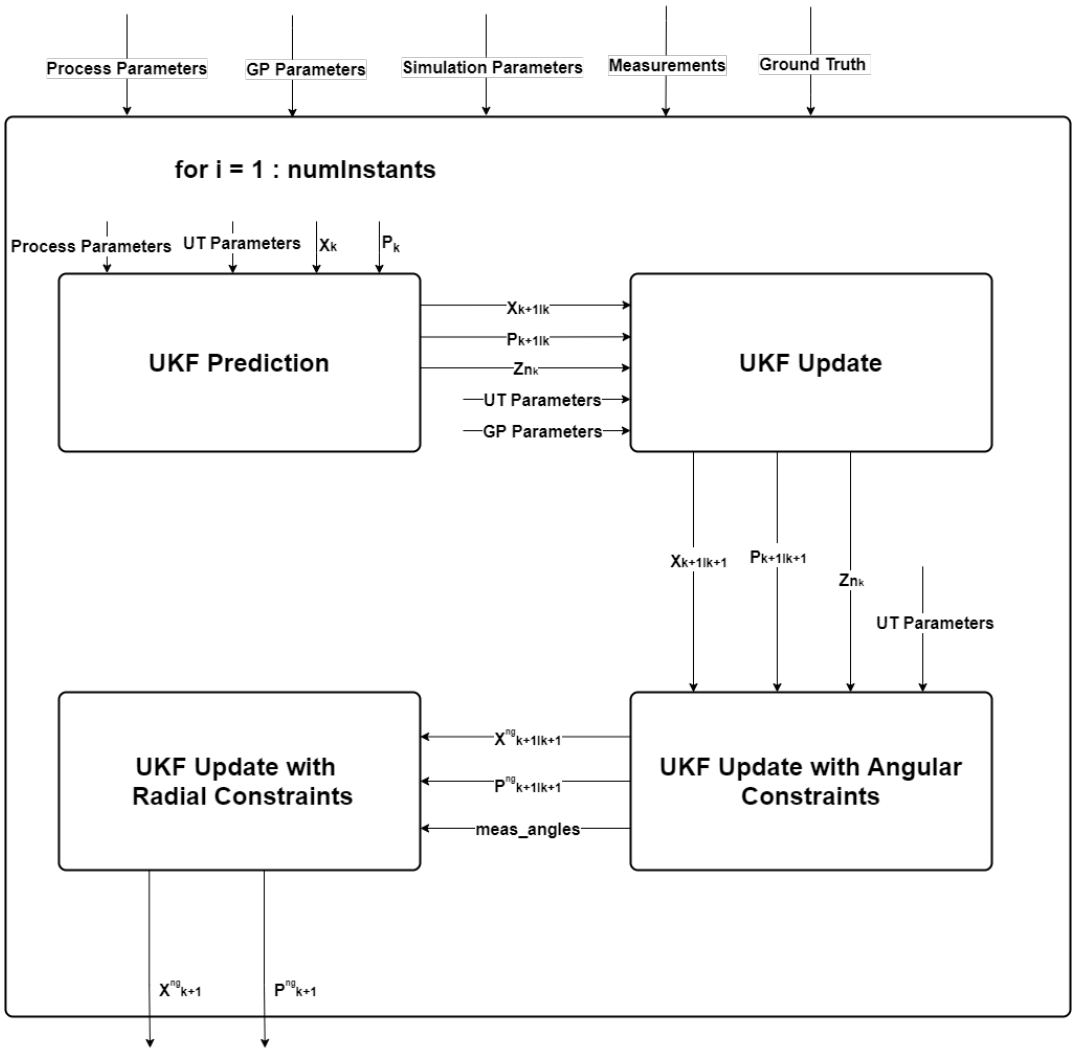

Figure 4.9: Overview of UKF Ng. ETTGP 2D Algorithm

4.2 ETTGP 3D Algorithm

4.2.1 Introduction

ETTGP 3D is an extended target tracking algorithm that utilizes Recursive Gaussian process regression to learn the shape (extent) of the target alongside kinematics [25]. ETTGP 3D algorithm is flexible enough to represent many shapes without the need of parameter tuning thanks to attractive analytical properties of Gaussian processes. In this section, we will derive a negative measurement update step to improve extent estimates.

4.2.2 Extent Model for 3D Objects

Target shape in 3D is represented using spherical coordinates with a radial function, $r = f(\theta, \phi)$, where θ and ϕ are azimuth and elevation angle respectively. Azimuth angle, $\theta \in [-\pi, \pi]$, is the angle from x-axis to y-axis whereas elevation angle, $\phi \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, is the angle from the x-y plane to the z axis as shown in Fig. 4.10. Corresponding output of the radial function, r, is the distance from the center of the target to the point on the extent specified by the pair (θ, ϕ). By this representation, we are able to define the boundary of a 3D object. It is necessary to note that star convex assumption defined in Sec. 4.1.2 is also valid for 3D extent representation. To simplify the notation, from here on out we define the spherical angle pair as $\xi \triangleq (\theta, \phi)$.

In Sec. 4.2.4, unknown radial function will be augmented with target position and kinematics to derive the state space model which makes possible simultaneous estimation of extent and kinematics.

4.2.3 GP Modelling for Object Extent

Gaussian processes are extensively discussed in Sec. 2.2. In ETTGP 3D algorithm, our aim is to learn target extent online via Gaussian process. Input of the GP is chosen as angle pair, $(\theta, \phi) = u$, in spherical coordinates whereas output is the radius of extent, $r = y$, at the corresponding angle pairs. Chosen mean and covariance

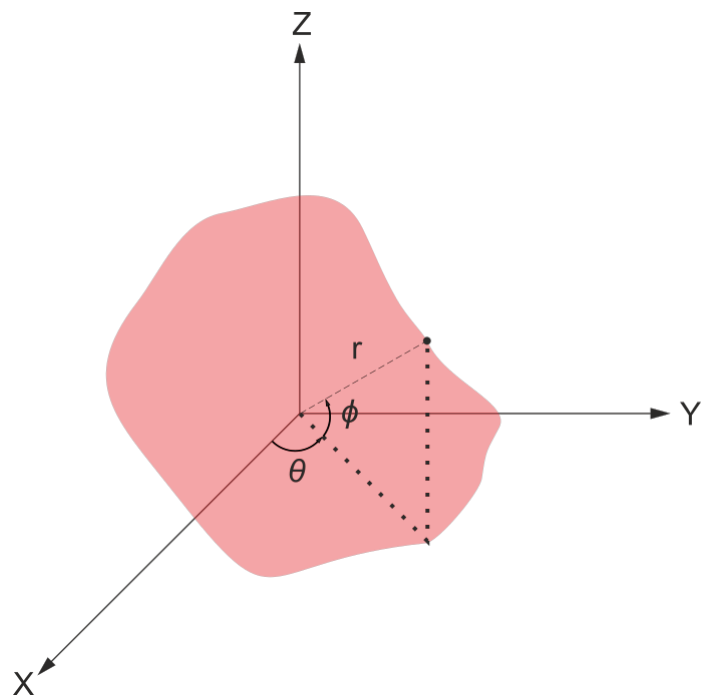

Figure 4.10: Representation of 3D shapes with radial function $r = f(\theta, \phi)$ in spherical coordinates

functions of GP are explained in the following subsections.

4.2.3.1 Mean Function

The mean function, $\mu(\xi)$, is chosen to be constant but unknown for this study. $\mu(\xi) = r$ can be thought as the mean radius of the target extent.

$$f(\xi) \sim \mathcal{GP}\ (r, k(\xi, \xi')), \quad \text{where} \quad r \sim \mathcal{N}(0, \sigma_r^2). \tag{4.33}$$

Mean function in (4.33) can also be modelled as a zero mean GP by integration out r

$$f(\xi) \sim \mathcal{GP}\ (0, k(\xi, \xi') + \sigma_r^2). \tag{4.34}$$

4.2.3.2 Covariance Function

Covariance function selection is an important aspect of GP modelling. Functions that will be learned via GP is governed by covariance function. There are many covariance functions that are utilized in the literature, however we chose to progress with the most common choice squared exponential covariance function [8].

$$k(\xi, \xi') = \sigma_f^2 e^{-\frac{d^2(\xi,\xi')}{2l^2}}, \tag{4.35}$$

σ_f^2 is the prior variance of the function amplitude and l represents the length scale of the functions we wish to learn. Squared exponential kernel results in higher correlations for $f(\xi)$ and $f(\xi')$ if angle pairs ξ and ξ' are close to each other that if they are apart as desired. Notice that unlike (4.5), Euclidean Distance is not used to represent relative proximity of two angle pairs. Euclidean Distance may lead to inaccurate results at some specific points for spherical coordinate representation. To illustrate an inaccuracy, consider angle pairs $\xi = \left(0, -\frac{\pi}{2}\right)$ and $\xi' = \left(\pi, -\frac{\pi}{2}\right)$. Both of these angle pairs represent the lower pole of a sphere, yet Euclidean distance for these two angle pairs is π. In order to quantify the relative distance between two angle pairs on a sphere, the angle of the shortest arc connecting these two pairs is a far better choice. The angle of the shortest arc between two angle pairs can be formulated as

$$\begin{aligned} d(\xi, \xi') = \arccos\big(&\cos(\phi)\cos(\phi')\cos(\theta)\cos(\theta') + \sin(\phi)\sin(\phi') \\ &+ \cos(\phi)\cos(\phi')\sin(\theta)\sin(\theta')\big). \end{aligned} \tag{4.36}$$

Lastly, the effect of mean function described in Sec. 4.2.3.1 is added to covariance function as in (4.7) to obtain final kernel.

$$\begin{aligned} k_{tot}(\xi, \xi') &= k(\xi, \xi') + \sigma_r^2, \\ &= \sigma_f^2 e^{-\frac{d^2(\xi,\xi')}{2l^2}} + \sigma_r^2. \end{aligned} \tag{4.37}$$

4.2.4 State Space Model

The augmented state space model will be derived in this section. Firstly, let us define state variables

$$\begin{aligned} \mathbf{x}_k &\triangleq [\bar{\mathbf{x}}_k^T \quad (\mathbf{x}_k{}^f)^T]^T, \quad \text{where} \\ \bar{\mathbf{x}}_k &\triangleq [(\mathbf{x}_k{}^c)^T \quad \mathbf{v}_k^T \quad \mathbf{q}_k^T]^T, \end{aligned} \tag{4.38}$$

where $\bar{\mathbf{x}}_k$ denotes target state, $\mathbf{x}_k{}^f$ denotes states for extent estimation, $\mathbf{x}_k{}^c$ is the target position, $\mathbf{v}_k$ denote velocity of the target, and $\mathbf{q}_k$ denote the unit quaternions, $\mathbf{q}_k = [q_{0k}\ q_{1k}\ q_{2k}\ q_{3k}]^\top$ where $\|\mathbf{q}_k\| = 1$, used for representing orientation of the target in 3D. Orientation representation is addressed extensively in Sec. 4.2.4.2. State space model for ETTGP 3D is given as

$$\begin{aligned} \mathbf{x}_{k+1} &= F\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q_k), \\ \mathbf{z}_{k,l} &= \mathbf{h}_{k,l}(\mathbf{x}_k) + \mathbf{e}_{k,l}, \quad \mathbf{e}_{k,l} \sim \mathcal{N}(\mathbf{0}, R_{k,l}), \\ \mathbf{x}_0 &\sim \mathcal{N}(\boldsymbol{\mu}_0, P_0). \end{aligned} \tag{4.39}$$

Details for representation of target state and extent is presented in the subsequent sections.

4.2.4.1 Extent Model

Structure derived in (2.14) allows us to perform recursive regression on the following state space model

$$\begin{aligned} \mathbf{x}_{k+1}^{\mathbf{f}} &= \mathbf{x}_k^{\mathbf{f}}, \\ \mathbf{z}_k &= H^{\mathbf{f}}(\mathbf{u}_k)\, \mathbf{x}_k^{\mathbf{f}} + \mathbf{e}_k^{\mathbf{f}}, \quad \mathbf{e}_k^{\mathbf{f}} \sim \mathcal{N}(0, R^{\mathbf{f}}(u_k)), \\ \mathbf{x}_0^{\mathbf{f}} &\sim \mathcal{N}(\mathbf{0}, P_0^{\mathbf{f}}), \end{aligned} \tag{4.40}$$

where $\mathbf{x}_k^f = \mathbf{f} = [f(u_1^{\mathbf{f}})\ \ldots\ f(u_{N^{\mathbf{f}}}^{\mathbf{f}})]^\top$ is the extent state. Furthermore, assuming

extent of the target changes over time, this state space model can be modified as follows to incorporate dynamical behavior

$$\mathbf{x}^{\mathbf{f}}_{k+1} = \mathbf{x}^{\mathbf{f}}_{k} + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q^{\mathbf{f}}_k), \tag{4.41}$$

with

$$Q^{\mathbf{f}}_k = \left(\frac{1}{\lambda} - 1\right) P^{\mathbf{f}}_{k|k}, \tag{4.42}$$

where $P^{\mathbf{f}}_{k|k}$ denotes covariance of the estimated target extent. This model keeps the same mean for predicted density with estimated density whereas covariance is scaled with a factor $\lambda < 1$ as

$$\lambda P^{\mathbf{f}}_{k+1|k} = P^{\mathbf{f}}_{k|k}. \tag{4.43}$$

This extent representation results in a maximum entropy distribution as described in [26]. Using this model to represent extent dynamics, we might correct erroneous extent estimations.

Augmented state space model in Sec. 4.2.4 is constructed with the extent model and process model in Sec. 4.2.4.2.

4.2.4.2 Process Model

Target state $\bar{\mathbf{x}}_k = [(\mathbf{x}_k{}^c)^T, \ \mathbf{v}^T_k, \ \mathbf{q}^T_k]^T$ is represented with a linear state space model as

$$\begin{aligned} \bar{\mathbf{x}}_{k+1} &= \bar{F}\bar{\mathbf{x}}_k + \bar{\mathbf{w}}_k, \quad \bar{\mathbf{w}}_k \sim \mathcal{N}(\mathbf{0}, \bar{Q}), \\ \bar{\mathbf{x}}_0 &\sim \mathcal{N}(\bar{\mu}_{\mathbf{0}}, \bar{P}_0). \end{aligned} \tag{4.44}$$

Thus, the augmented target dynamics model is constructed together with the extent model in (4.41)

$$
\begin{aligned}
\mathbf{x}_{k+1} &= F\mathbf{x}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q), \\
\mathbf{x}_0 &\sim \mathcal{N}(\mu_{\mathbf{0}}, P_0),
\end{aligned}
\tag{4.45}
$$

where

$$
\begin{aligned}
\mathbf{x}_k &= \begin{bmatrix} \bar{\mathbf{x}}_k \\ \mathbf{x}_k^{\mathbf{f}} \end{bmatrix}, \quad F = \begin{bmatrix} \bar{F} & 0 \\ 0 & F^{\mathbf{f}} \end{bmatrix}, \quad Q = \begin{bmatrix} \bar{Q} & 0 \\ 0 & Q^{\mathbf{f}} \end{bmatrix}, \\
\boldsymbol{\mu}_0 &= \begin{bmatrix} \bar{\boldsymbol{\mu}}_0 \\ \boldsymbol{\mu}_0^{\mathbf{f}} \end{bmatrix}, \quad P_0 = \begin{bmatrix} \bar{P}_0 & 0 \\ 0 & P_0^{\mathbf{f}} \end{bmatrix}, \; P_0^{\mathbf{f}} = K(\mathbf{u}^{\mathbf{f}}, \mathbf{u}^{\mathbf{f}}).
\end{aligned}
\tag{4.46}
$$

Target state dynamics are chosen as constant velocity model

$$
\begin{aligned}
\bar{F} &= \begin{bmatrix} \bar{F}_c & 0 \\ 0 & \bar{F}_q \end{bmatrix}, \quad \bar{F}_c = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \otimes I_3, \; \bar{F}_q = I_4, \\
\bar{Q} &= \begin{bmatrix} \bar{Q}_c & 0 \\ 0 & \bar{Q}_q \end{bmatrix}, \; \bar{Q}_c = \begin{bmatrix} \frac{T}{3}^3 & \frac{T}{2}^2 \\ \frac{T}{2}^2 & T \end{bmatrix} \otimes \sigma_c^2 I_3, \; \bar{Q}_q = \sigma_q^2 \, I_4.
\end{aligned}
\tag{4.47}
$$

where σ_c^2 denote variance of process noise for the target center.

Rotational Motion Model : Orientation representation in 3D is a challenging task since there are many singularities and constraints present [27]. The simplest approach would be using so-called Euler angles (yaw, pitch, roll). However, inherent singularities in this model prevent consistent representation of orientation. Unit quaternions that have four components to represent orientation unlike Euler Angles, offer an efficient solution. The unit quaternion model is free of singularities, but it necessitates a nonlinear constraint $\|\mathbf{q}_k\| = 1$, i.e., norm of the quaternions must be 1. Norm constraint in unit quaternion model calls forth for careful execution in orientation estimation.

Due to the recently mentioned reasons, we model the orientation with a different approach. In order to represent orientation, a reference orientation that uses unit quater-

nions and an error vector with three component are utilized. This representation enables us to employ and EKF to estimate error vector. Aforementioned representation is known as a Multiplicative EKF (MEKF) [28] in the literature. Using MEKF, we guarantee that the norm constraint in the unit quaternion representation is satisfied while global orientation is robustly represented.

Now, we will obtain a constant velocity (CV) model for orientation estimation. [29] is referred for unit quaternion basics.

Unit quaternions, $q \in \mathbb{R}^4$, are described as

$$\mathbf{q} \triangleq [\bar{\mathbf{q}}^\top \mathbf{q}_4]^\top = [q_0 \; q_1 \; q_2 \; q_3]^\top, \tag{4.48}$$

where $\|\mathbf{q}\| = 1$. Rotation matrix, $R_G^L(\mathbf{q})$, transforms the orientation of the target w.r.t. local frame to global frame as

$$R_G^L(\mathbf{q}) = (\mathbf{q}_4^2 - \bar{\mathbf{q}}^\top \bar{\mathbf{q}}) I_3 + 2\bar{\mathbf{q}}\bar{\mathbf{q}}^\top - 2q_4[\bar{\mathbf{q}}\times], \tag{4.49}$$

where $[\bar{\mathbf{q}}\times]$ defined as the following cross product matrix

$$[\bar{\mathbf{q}}\times] = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix}. \tag{4.50}$$

From this point on, we will utilize the following property

$$R(\mathbf{q})R(\mathbf{p}) = R(\mathbf{q} \odot \mathbf{p}), \tag{4.51}$$

where $R(\mathbf{q})$ and $R(\mathbf{p})$ denote rotation matrices with given unit quaternions $\mathbf{q}$ and $\mathbf{p}$, $\odot$ is defined as the quaternion product

$$\mathbf{q} \odot \mathbf{p} = \begin{bmatrix} q_4\bar{\mathbf{p}} + p_4\bar{\mathbf{q}} - \bar{\mathbf{q}} \times \bar{\mathbf{p}} \\ q_4 p_4 - \bar{\mathbf{q}}^\top \bar{\mathbf{p}}. \end{bmatrix} \tag{4.52}$$

The property in (4.51) comes in handy since consecutive rotations can be expressed as a quaternion product described in (4.52). Using this convenience, we can represent orientation of the target as

$$\mathbf{q} = \delta\mathbf{q}(\mathbf{a}) \odot \mathbf{q}_{ref}, \tag{4.53}$$

where $\delta\mathbf{q}(\cdot)$ denotes deviation from reference and $\mathbf{q}_{ref}$ denotes reference orientation. Rodrigues Parametrization is utilized to define $\delta\mathbf{q}(\mathbf{a})$ as

$$\delta\mathbf{q}(\mathbf{a}) = \frac{1}{\sqrt{4 + |\mathbf{a}|^2}} \begin{bmatrix} \mathbf{a} \\ 2 \end{bmatrix}, \tag{4.54}$$

where $\mathbf{a} \in \mathbb{R}^3$ is the error (or deviation) vector. Dynamics of the error vector can be written as in [30]

$$\dot{\mathbf{a}} = \left(I_3 + \frac{1}{4}\mathbf{a}\mathbf{a}^\top + \frac{1}{2}\bar{\mathbf{a}}\times \right) \boldsymbol{\omega}, \tag{4.55}$$

where angular rate of local coordinate frame w.r.t. global coordinate frame is defined as $\boldsymbol{\omega} \triangleq [\omega_x \; \omega_y \; \omega_z]^\top$. Assuming $\mathbf{a}$ takes small values, the quadratic term in (4.55) can be ignored

$$\dot{\mathbf{a}} \approx \left(I_3 + \frac{1}{2}\bar{\mathbf{a}}\times \right) \boldsymbol{\omega}. \tag{4.56}$$

A constant velocity model can be utilized in the following model based on [26]

$$\begin{bmatrix} \dot{\mathbf{a}} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \left(I_3 + \frac{1}{2}\bar{\mathbf{a}}\times \right) \boldsymbol{\omega} \\ \mathbf{0}_{3x1} \end{bmatrix} + \begin{bmatrix} 0_3 \\ I_3 \end{bmatrix} \boldsymbol{\alpha}, \tag{4.57}$$

where rotational acceleration vector is denoted as $\boldsymbol{\alpha}$. Rotational acceleration vector is modeled as zero mean white Gaussian noise with covariance

$$\begin{aligned} \mathrm{cov}[\boldsymbol{\alpha}(t), \boldsymbol{\alpha}(t')] &= \delta(t - t')\Sigma_\alpha, \\ \Sigma_\alpha &= \sigma_\alpha^2 I_3. \end{aligned} \tag{4.58}$$

Note that, the dynamic model in (4.57) is nonlinear. Thus, we need to linearize the model firstly. Consider the following equations

$$\begin{aligned} \dot{\mathbf{x}}^r &= f(\mathbf{x}^r) + B\boldsymbol{\alpha}, \quad \text{where} \\ \mathbf{x}^r &\triangleq [\mathbf{a}^\top \boldsymbol{\omega}^\top]^\top. \end{aligned} \tag{4.59}$$

Now, let us substitute Taylor series approximation of $f(\cdot)$

$$\begin{aligned} f(\mathbf{x}^r) &\approx f(\hat{\mathbf{x}}^r) + A_k^r(\mathbf{x}^r - \hat{\mathbf{x}}^r), \quad \text{where} \\ A_k^r &= \frac{d}{d\hat{x}^r} f(\mathbf{x}^r)\bigg|_{\mathbf{x}^r = \hat{\mathbf{x}}_{k|k}^r}. \end{aligned} \tag{4.60}$$

Previous posterior, $\hat{\mathbf{x}}_{k|k}^r \triangleq [\hat{\mathbf{a}}_{k|k}^\top \;\; \hat{\boldsymbol{\omega}}_{k|k}^\top]^\top$, is chosen as the linearization point since it is the best possible estimate. It is important to emphasize reference orientation is corrected by the estimated error vector after each measurement update step using quaternion product given in (4.52). Then, the error vector is reset to zero, $\hat{\mathbf{a}}_{k|k} = \mathbf{0}$, to complete the process. In UKF and UKF Ng. implementations, reference orientation correction and error vector reset steps are applied after each prediction as suggested in [31] as well as after each measurement update.Thus, (4.60) becomes

$$\begin{aligned} f(\mathbf{x}^r) &= A_k^r \mathbf{x}^r, \quad \text{where} \\ A_k^r &= \begin{bmatrix} \frac{1}{2}[-\hat{\boldsymbol{\omega}}_{k|k}\times] & I_3 \\ 0_3 & 0_3 \end{bmatrix}. \end{aligned} \tag{4.61}$$

Derived linearized system is as follows

$$\dot{\mathbf{x}}^r = A_k^r \mathbf{x}^r + B\boldsymbol{\alpha}. \tag{4.62}$$

Lastly, we discretize (4.62) to obtain a linear Gaussian model for representation of

orientation dynamics

$$
\begin{aligned}
\mathbf{x}_{k+1}^r &= F_k^r \mathbf{x}_k^r + \mathbf{w}_k^r, \quad \text{where} \\
\mathbf{w}_k^r &\sim \mathcal{N}(\mathbf{0}, Q_k^r), \\
F_k^r &= exp(A_k^r T)\mathcal{N}(\mu_{\mathbf{0}}, P_0), \\
Q_k^r &= G_k \Sigma_\alpha G_k^\top, \\
G_k &= \left(\int_0^T exp(A_k^r \tau) d\tau \right) B.
\end{aligned}
\tag{4.63}
$$

$\mathbf{x}_k^r \triangleq [\mathbf{a}_k^\top \;\; \boldsymbol{\omega}_k^\top]^\top$ denotes rotational dynamics state, F_k^r denotes dynamic model function, Q_k^r denotes process noise covariance, and T denotes sampling time. Remark that F_k^r and Q_k^r are time varying since they change with each new linearization point. Details of the matrices F_k^r and G_k^r are given in Appendix C of [25].

4.2.4.3 Measurement Model

Each measurement obtained at time k from the extent of the target is associated with an angle pair in the global coordinate frame, $\xi_{k,l}$, depending on their relative position to the target position, $\mathbf{x}_k^c$, as follows

$$
\begin{aligned}
\mathbf{z}_{k,l} &= \mathbf{x}_k^c + \mathbf{p}(\xi_{k,l}) f(\xi_{k,l}) + \bar{\mathbf{e}}_{k,l}, \\
\bar{\mathbf{e}}_{k,l} &\sim \mathcal{N}(\mathbf{0}, \bar{R}).
\end{aligned}
\tag{4.64}
$$

$\mathbf{x}_k^c$ denotes target center, $\xi_{k,l}$ denotes the spherical angle pair from which measurement originated, $\mathbf{p}(\xi_{k,l})$ denotes the unit length vector in the direction of $\xi_{k,l}$, $f(\cdot)$ denotes the radial function, and lastly $\bar{\mathbf{e}}_{k,l}$ denotes measurement noise.

It is important to emphasize that spherical angle pairs, $\xi_{k,l}$, are not readily available. They can be represented as a function of measurements, $\mathbf{z}_{k,l}$, and orientation of the target. Hence, we first need to represent measurements in the local coordinate frame. Firstly, the center of the coordinate frame is translated to $\mathbf{x}_k^c$. Then, coordinate frame is rotated according to the pose of the target. These successive transformations are presented in the following equation. These transformations are depicted in Fig. 4.11.

$$\mathbf{z}_{k,l}^{L}\left(\mathbf{x}_k^c, \mathbf{q}_k\right) = \underbrace{R_G^L(\mathbf{q}_k)}_{Rotation} \underbrace{(\mathbf{z}_{k,l} - \mathbf{x}_k^c)}_{Translation}, \tag{4.65}$$

where $R_G^L(\mathbf{q})$ defined as the rotation matrix from global to local coordinate frame using unit quaternions as

$$R_G^L(\mathbf{q}) = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & 1-2(q_1^2+q_2^2) \end{bmatrix}. \tag{4.66}$$

Note that $\mathbf{z}_{k,l}^{L}\left(\mathbf{x}_k^c, \mathbf{q}_k\right)$ will not be explicitly used in measurement update equations. However, spherical angle pairs, $\xi_{k,l}$, that will be used in measurement update equations are obtained by conversion of $\mathbf{z}_{k,l}^{L}$ into the spherical coordinates.

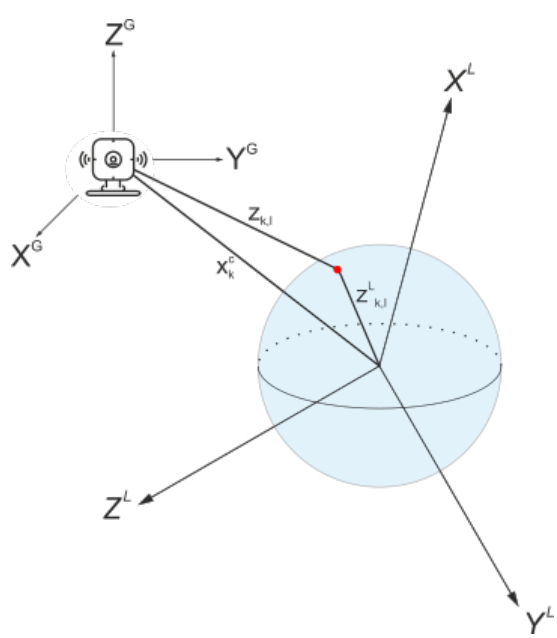

Figure 4.11: Representation of a single measurement in spherical coordinate system w.r.t. global and local coordinate frames in ETTGP 3D algorithm

With the help of local measurement representation in (4.65), we can simply define the relationship between target state and a single measurement as

$$
\begin{aligned}
\mathbf{z}_{k,l} &= \mathbf{x}_k^c + \mathbf{p}_{k,l}(\mathbf{x}_k^c) f\left(\xi_{k,l}^L(\mathbf{x}_k^c, \mathbf{q}_k)\right) + \bar{\mathbf{e}}_{k,l}, \\
\bar{\mathbf{e}}_{k,l} &\sim \mathcal{N}(\mathbf{0}, R_{k,l}),
\end{aligned} \tag{4.67}
$$

where orientation vector, $\mathbf{p}_{k,l}(\mathbf{x}_k^c)$, can be formulated as

$$
\mathbf{p}_{k,l}(\mathbf{x}_k^c) = \frac{\mathbf{z}_{k,l} - \mathbf{x}_k^c}{\|\mathbf{z}_{k,l} - \mathbf{x}_k^c\|}. \tag{4.68}
$$

Finally, measurement equations for ETTGP 3D algorithm can be formulated as follows by utilizing (2.14) discussed in Sec. 2.2.

$$
\begin{aligned}
\mathbf{z}_{k,l} &= \mathbf{x}_k^c + \mathbf{p}_{k,l}(\mathbf{x}_k^c)\left[H^{\mathbf{f}}\left(\xi_{k,l}(\mathbf{x}_k^c, \mathbf{q}_k)\right)\mathbf{x}_k^{\mathbf{f}} + \mathbf{e}_{k,l}^{\mathbf{f}}\right] + \bar{\mathbf{e}}_{k,l}, \\
&= \underbrace{\mathbf{x}_k^c + \tilde{H}_l\left(\mathbf{x}_k^c, \mathbf{q}_k\right)\mathbf{x}_k^{\mathbf{f}}}_{=\mathbf{h}_{k,l}(\mathbf{x}_k)} + \underbrace{\mathbf{p}_{k,l}(\mathbf{x}_k^c)\mathbf{e}_{k,l}^{\mathbf{f}} + \bar{\mathbf{e}}_{k,l}}_{=\mathbf{e}_{k,l}}, \\
&= \mathbf{h}_{k,l}(\mathbf{x}_k) + \mathbf{e}_{k,l}, \quad \mathbf{e}_{k,l} \sim \mathcal{N}(\mathbf{0}, R_{k,l}),
\end{aligned} \tag{4.69}
$$

where

$$
\begin{aligned}
\tilde{H}_l\left(\mathbf{x}_k^c, \mathbf{q}_k\right) &= \mathbf{p}_{k,l}\, H^{\mathbf{f}}\left(\xi_{k,l}(\mathbf{x}_k^c, \mathbf{q}_k)\right), \\
\mathbf{h}_{k,l}(\mathbf{x}_k) &= \mathbf{x}_k^c + \tilde{H}_l\left(\mathbf{x}_k^c, \mathbf{q}_k\right)\mathbf{x}_k^{\mathbf{f}}, \\
R_{k,l} &= \mathbf{p}_{k,l}\, R_{k,l}^{\mathbf{f}}\, \mathbf{p}_{k,l}^\top + R, \\
\mathbf{p}_{k,l} &= \mathbf{p}_{k,l}(\mathbf{x}_k^c), \quad R_{k,l}^{\mathbf{f}} = R^{\mathbf{f}}\left(\xi_{k,l}(\mathbf{x}_k^c, \mathbf{q}_k)\right).
\end{aligned} \tag{4.70}
$$

Note that measurement noise in (2.14) is excluded since measurement noise, R, is already included in (4.70).

4.2.4.4 Negative Measurement Model

Negative measurement model for ETTGP 3D algorithm follows similar process to ETTGP 2D algorithm which discussed extensively in Sec. 4.1.4.4. In this section, we will expand negative information fusion concept to obtain better 3D extent estimates. Similarly, we will assume that if the target is in the range of the sensor there will be

some measurements. To simply put, there will be no missed detections. By assuming so, we will incorporate negative information into the measurement model to obtain better extent estimates. In this study, we will consider sensors that can measure distance as well as angle from the sensor to the measurement source. These sensors, e.g., RADAR and LIDAR, are widely used in 3D target tracking applications. We will update the angular and radial constraints to accommodate third dimension on the measurements to derive a negative information update which sharpens the accuracy of state estimates. Note that all equations in this section is given for time instant k. For notational simplicity, index k in the variables are dropped.

In Fig. 4.12, angular and radial constraints are visualized. Angular constraint basically imposes limitation on the angular extent of the object whereas radial constraint restricts the radial distance between the sensor and visible parts of the target.

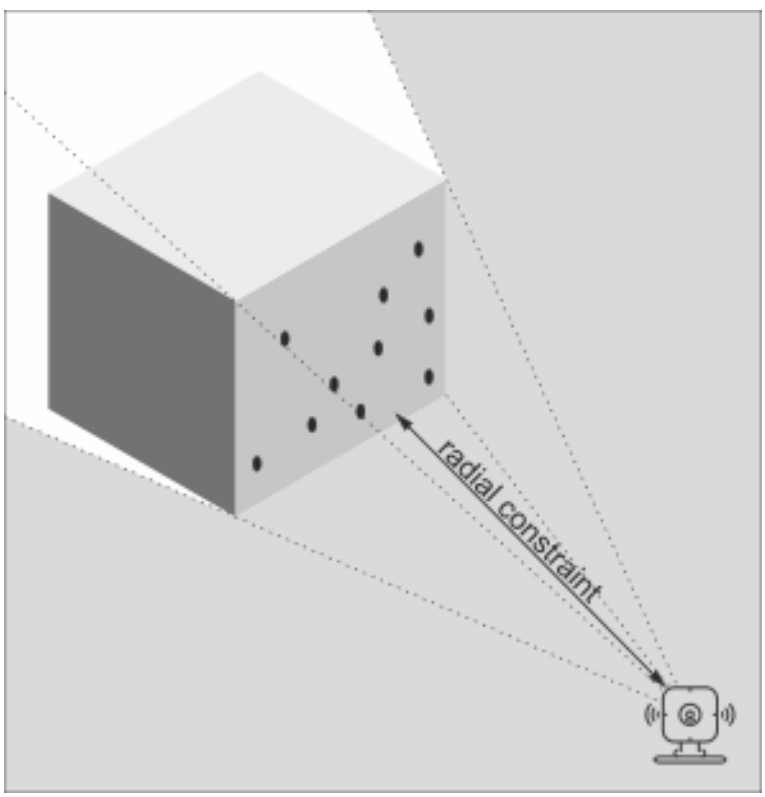

Figure 4.12: Demonstration of radial constraint imposed by negative information

Pseudo Measurements for Angular Constraint : Angular extent of a target is related to the minimum and maximum angle pairs with respect to sensor position,

$\xi_{min} \triangleq (\theta_{min}, \phi_{min})$ and $\xi_{max} \triangleq (\theta_{max}, \phi_{max})$, as shown in Fig. 4.13. ξ_{min} and ξ_{max} can be simply found by the geometric relation between measurements and sensor position. Given a set of measurements, spherical angle pairs from the sensor to each measurement is calculated. Thus, minimum and maximum of these angle pairs are ξ_{min} and ξ_{max} respectively. Let h_{min} and h_{max} denote the minimum and maximum angle pairs calculated from the extent state x. ε_{min} and ε_{max} denote the noisy measurements of these quantities.

$$\begin{aligned} \varepsilon_{min} &= h_{min}(\mathbf{x}) + r_{min}, \\ \varepsilon_{max} &= h_{max}(\mathbf{x}) + r_{max}, \end{aligned} \tag{4.71}$$

where $\mathbf{x}$ is the state, h_{min} and h_{max} are nonlinear functions that generate minimum and maximum angles calculated from the extent, r_{min} and r_{max} are noises with $\sim \mathcal{N}(0, R_\varepsilon)$. Note that pseudo angular measurements are functions of state. These pseudo angular measurements are compared with the actual maximum and minimum angle pairs calculated from the set of measurements.

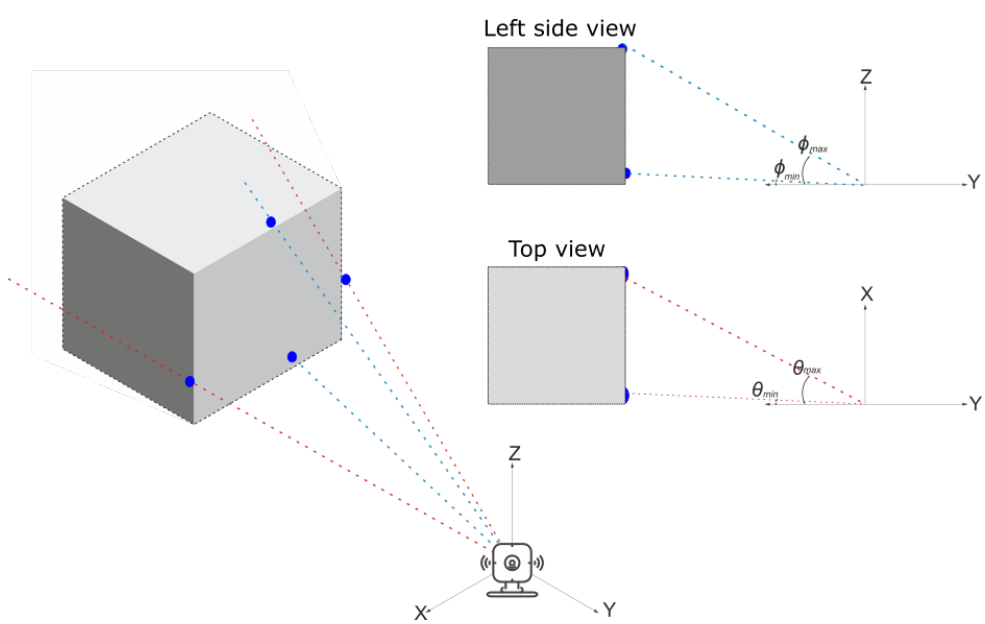

Figure 4.13: Demonstration of angular constraints imposed by negative information

Deriving analytical expressions for the nonlinear mappings are troublesome since $\mathbf{x}$ can represent any arbitrary 3D shape. Hence, they will be calculated numerically.

Vectors from sensor to the basis points $i = 1, \ldots, N$, $\mathbf{v}_i$, can be expressed as

$$
\begin{aligned}
\mathbf{v}_i \triangleq \left[v_{x,i}, v_{y,i}, v_{z,i}\right]^T &= \mathbf{x}^c + R_L^G(\mathbf{q})\mathbf{b}_i - \mathbf{s}, \quad \text{where} \\
R_L^G(\mathbf{q}) &\triangleq R_G^L(\mathbf{q})^T, \\
\mathbf{s} &\triangleq [s_x, s_y, s_z]^T, \\
\mathbf{b}_i &\triangleq [b_{i,x}, b_{i,y}, b_{i,z}]^T,
\end{aligned}
\tag{4.72}
$$

where $\mathbf{x}^c$ is the position estimate, $R_L^G(\mathbf{q})$ is the rotation matrix from local to global frame, $\mathbf{s}$ denote the sensor coordinates, $\mathbf{b}_i$ denote the ith basis angle coordinates in local coordinate frame. Now, we can easily find the minimum and maximum angle pairs calculated from the extent, h_{min} and h_{max} from h_i.

$$
h_i = \left(\arctan\left(\frac{v_{y,i}}{v_{x,i}}\right), \arctan\left(\frac{v_{z,i}}{\sqrt{v_{x,i}^2 + v_{y,i}^2}}\right)\right). \tag{4.73}
$$

where h_i denote the nonlinear mappings for $i = 1, \ldots, N$. After that, ε_{min} and ε_{max} are used as pseudo measurements for additional measurement update step to improve extent estimation. Demonstration of pseudo angular measurement generation is given in Fig. 4.14.

Pseudo Measurements for Radial Constraint : In order to set a radial constraint, let us define the unit vectors, $\hat{\mathbf{u}}_l$, from sensor to the measurements $\{\mathbf{z}_l\}_{l=1}^n$

$$
\hat{\mathbf{u}}_l = \frac{\mathbf{z}_l - \mathbf{s}}{\|\mathbf{z}_l - \mathbf{s}\|}. \tag{4.74}
$$

Consider lines, g_l, that extend from $\mathbf{s}$ in the direction of $\hat{\mathbf{u}}_l$. Each of the lines are checked to see if they intersect with the extent represented by state $\mathbf{x}$. Assume a subset of the obtained measurements, $m \leq n$, $\{\mathbf{z}_l\}_{j=1}^m$, intersects with the extent. For the newly formed subset of measurements, radial distance between the sensor and the measurements, d_j, can be expressed as

$$
d_j = \|\mathbf{z}_j - \mathbf{s}\|. \tag{4.75}
$$

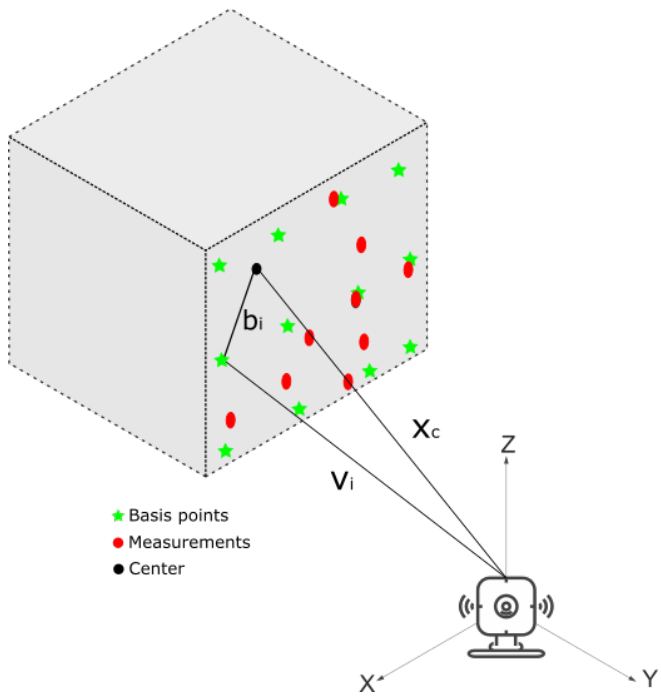

Figure 4.14: Demonstration of pseudo angular measurement generation

Measurement model for the pseudo measurements that represent radial distance can be specified as

$$d_j = h_j(\mathbf{x}) + r_j, \tag{4.76}$$

where h_j denote nonlinear mappings which calculate radial distance estimate between sensor and the extent represented by state $\mathbf{x}$ along $\hat{\mathbf{u}}_l$, d_j denote the noisy measurements of these quantities. r_j represent measurement noises with $\sim \mathcal{N}(0, R_d)$. Note that pseudo radial measurements are functions of state. These pseudo radial measurements are compared with the actual distances calculated from the set of measurements.

Similar to the angular constraints case, vectors from sensor to the basis points $i = 1, \ldots, N$, $\mathbf{v}_i$, can be calculated as in (4.72). From these N vectors, five closest vectors in the direction of $\hat{\mathbf{u}}_l$ are chosen for each measurement. Among these vectors, radial distance from the sensor to the basis points are calculated. The closest basis point is chosen to find h_j for each measurement $j = 1, \ldots, m$

$$h_j(\mathbf{x}) = \left|\mathbf{x}^c + R_L^G(\mathbf{q})\mathbf{b}_j - \mathbf{s}\right|. \tag{4.77}$$

After that, $\{d_j\}_{j=1}^{m}$ are used as pseudo measurements for additional measurement update step to improve extent estimation.

4.2.5 Inference

Inference for ETTGP 3D algorithm follows the same process as in ETTGP 2D algorithm. For the sake of completeness, the procedure is repeated in this section. Kalman filter based inference techniques can be applied to state space model formulated in (4.39) to compute the posterior density of the state vector. In order to update posterior density recursively, we need to concatenate all measurements at an instant, $\{\mathbf{z}_{k,l}\}_{l=1}^{n_k}$

$$\begin{aligned} \mathbf{z}_k &= \left[\mathbf{z}_{k,1}^{\top},\ \ldots,\ \mathbf{z}_{k,n_k}^{\top}\right]^{\top}, \\ \mathbf{h}_k(\mathbf{x}_k) &= \left[\mathbf{h}_{k,1}(\mathbf{x}_k)^{\top},\ \ldots,\ \mathbf{h}_{k,n_k}^{\top}(\mathbf{x}_k)\right]^{\top}, \\ R_k &= \text{diag}\left[R_{k,1},\ \ldots,\ R_{k,n_k}\right]. \end{aligned} \tag{4.78}$$

With this modification, we obtain the following state space model

$$\begin{aligned} \mathbf{x}_{k+1} &= F\mathbf{x}_k + \mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, Q), \\ \mathbf{z}_k &= \mathbf{h}_k(\mathbf{x}_k) + \mathbf{e}_k, \quad \mathbf{e}_k \sim \mathcal{N}(\mathbf{0}, R_k), \\ \mathbf{x}_0 &\sim \mathcal{N}(\boldsymbol{\mu}_0, P_0). \end{aligned} \tag{4.79}$$

Using this structure, we can use a nonlinear filtering technique for state estimation. In this study, EKF discussed in Sec. 3.3 and UKF discussed in Sec. 3.5 are applied for inference just like ETTGP 2D algorithm. Details of the required recursions for EKF are given in Appendices of [6].

4.2.6 UKF Ng. ETTGP 3D Algorithm Overview

UKF Ng. ETTGP 3D algorithm's overview is given in Algorithm 5. First four steps of the Algorithm 5 correspond to the UKF ETTGP 3D algorithm. Steps 6-10 correspond to the pseudo angular measurements update. Remaining steps correspond

to the pseudo radial measurements update. It is important to emphasize that, this algorithm consist of three measurement update steps with different types of measurements namely spatial measurements, pseudo angular measurements, and pseudo radial measurements. There are in total $\underbrace{n}_{Spatial} + \underbrace{2}_{PseudoAngular} + \underbrace{m}_{PseudoRadial}$ measurements. Algorithm 5 is visualized in Fig. 4.15.

Algorithm 5 UKF Ng. ETTGP 3D Overview

1: Perform UKF prediction as in Algorithm 1.
2: Calculate weights of sigma points as in (3.13).
3: Calculate sigma points as in (3.19).
4: Perform measurement update as in (3.20) together with (4.69) and (4.70).
5: Perform pseudo angular measurements update as in Steps 6-10.
6: Calculate minimum and maximum angle pairs from sensor to the measurements, ξ_{min} and ξ_{max}.
7: Calculate weights of sigma points as in (3.13).
8: Calculate sigma points as in (3.19).
9: Calculate minimum and maximum angle pair estimates from the extent, ε_{min} and ε_{max}, using (4.71) $-$ (4.73).
10: Perform measurement update with minimum and maximum angle pair estimates as in (3.21) $-$ (3.26) together with (4.71) $-$ (4.73).
11: Perform pseudo radial measurements update as in Steps 12-17.
12: Construct the measurement set for radial constraint implementation using (4.74).
13: Calculate minimum and maximum angle pairs from sensor to the measurements, ξ_{min} and ξ_{max}.
14: Calculate sigma points as in (3.19).
15: Calculate weights of sigma points as in (3.13).
16: Calculate minimum radial distance estimates using (4.75) $-$ (4.77).
17: Perform measurement update with minimum radial distance estimates as in (3.21) $-$ (3.26) together with (4.75) $-$ (4.77).

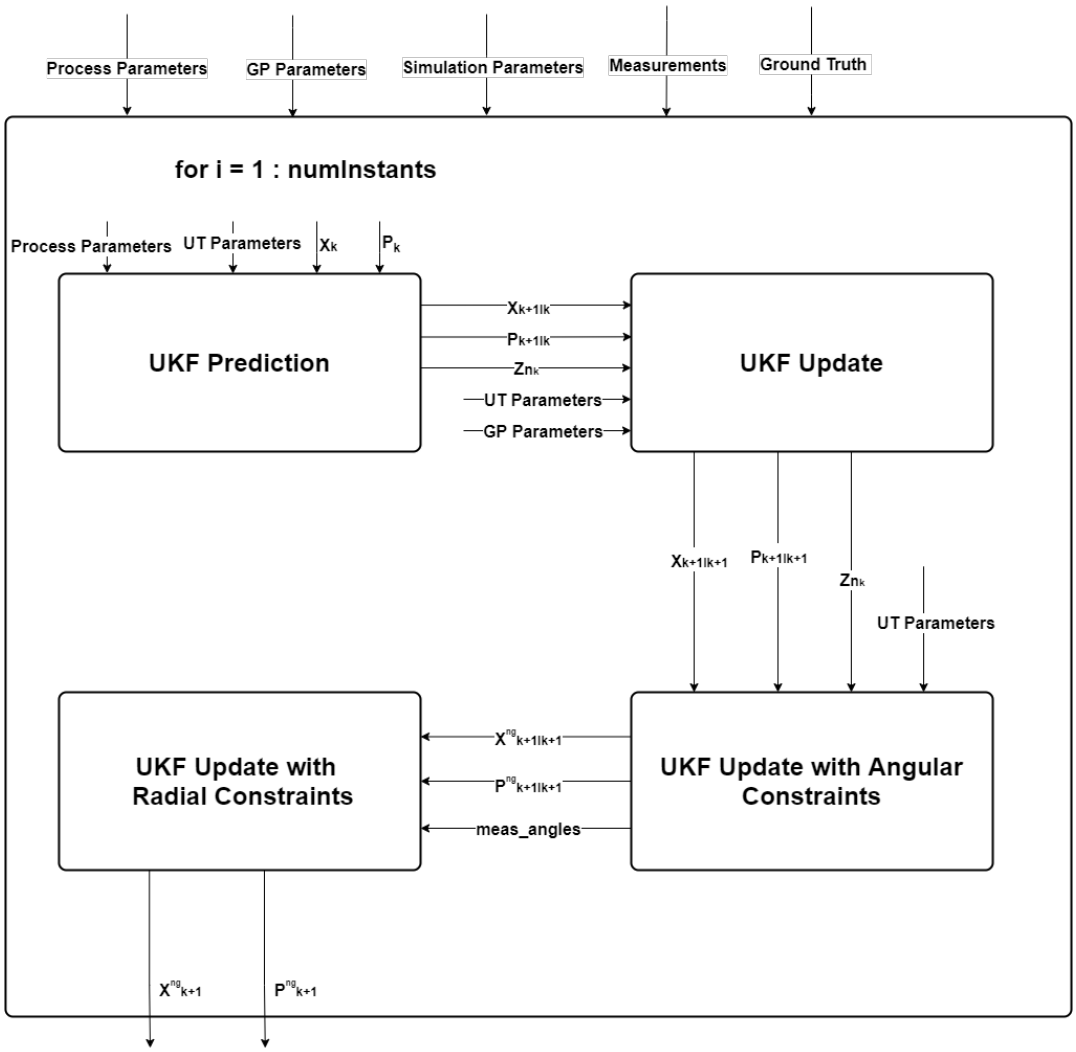

Figure 4.15: Overview of UKF Ng. ETTGP 3D Algorithm

CHAPTER 5

SIMULATION RESULTS AND DISCUSSIONS

In this chapter, simulations conducted for ETTGP 2D and ETTGP 3D algorithms are discussed. Simulated measurements are generated in MATLAB using a ETTGP scenario generation framework. This framework enabled us to conduct combinations of different simulations easily. By utilizing such a framework, we aim to compare different types of extended target tracking algorithms in a controlled environment. Figures and performance metrics are automatically generated and stored in .tex format, and simulations are easily repeatable. Details of this simulation environment will be explained in Sec. 5.1.1 and Sec. 5.2.1. Furthermore, necessary parameters for simulations are presented in this chapter such as GP parameters (prior variance, length scale etc.), object parameters (object type, dimensions), sensor parameters (sampling rate, standard deviation of measurement noise etc.), and lastly motion parameters (motion type, duration, velocity etc.). Moreover, we additionally conducted experiments using real data by using Kitti data set [32] for ETTGP 3D algorithms to demonstrate the effectiveness of the proposed algorithm. Finally, results of the simulations and discussions regarding results are given.

5.1 ETTGP 2D Simulation Results and Discussions

In this section, ETTGP 2D Scenario Generation Framework is discussed firstly. After that, parameters used in the ETTGP 2D simulations are given. Performance measures, IoU and RMSE, and scenario visualizations given later. Lastly, comments regarding the performance of different ETTGP 2D estimators such as EKF, UKF, Unscented Kalman Filter with Negative Information Fusion (UKF Ng.) EKS, UKS

and Unscented Kalman Smoother with Negative Information Fusion (UKS Ng.) are presented.

5.1.1 ETTGP 2D Framework

In this section, ETTGP 2D Scenario Generation Framework is introduced firstly. Framework mainly consists of three blocks as given in Fig. 5.1.

- **Initialization:** This block takes motion, object, GP and sensor parameters as inputs. These four inputs are defined as a class in MATLAB and different types of models can be incorporated to the framework easily. Using the input classes and some scenario specific parameters, initialization block performs necessary operations for simulation start-up.
- **Monte Carlo Runs:** This block takes initialization block's outputs and perform Monte Carlo runs for a number of different estimation algorithms. The block is designed to be flexible in a way that many algorithms can be comparatively ran. Moreover, online figures for observing target trajectory and estimators' output are plotted. RMSE and IoU metrics, which will be discussed later, are also estimated and saved by this block.
- **Performance Evaluator:** This block is used for post-processing the results we obtained from Monte Carlo runs block. Our aim is to store the results of the simulations, i.e., figures and tables, in an easily publishable format. For this purpose, different figures are plotted and saved in a Latex format by utilizing matlab2tikz package provided in [33]. Furthermore, tables for RMSE results of different algorithms are generated in a .tex format by utilizing latexTable function provided in [34].

There are 24 possible simulation configurations that are combinations of different target shapes, motion models and sensor models. Object, sensor and motion model types are given in Table 5.1. One can extend the framework by adding different types of motion models, object types and sensor models with seamless integration. Note that abbreviations in the parentheses will be used in the following figures' and

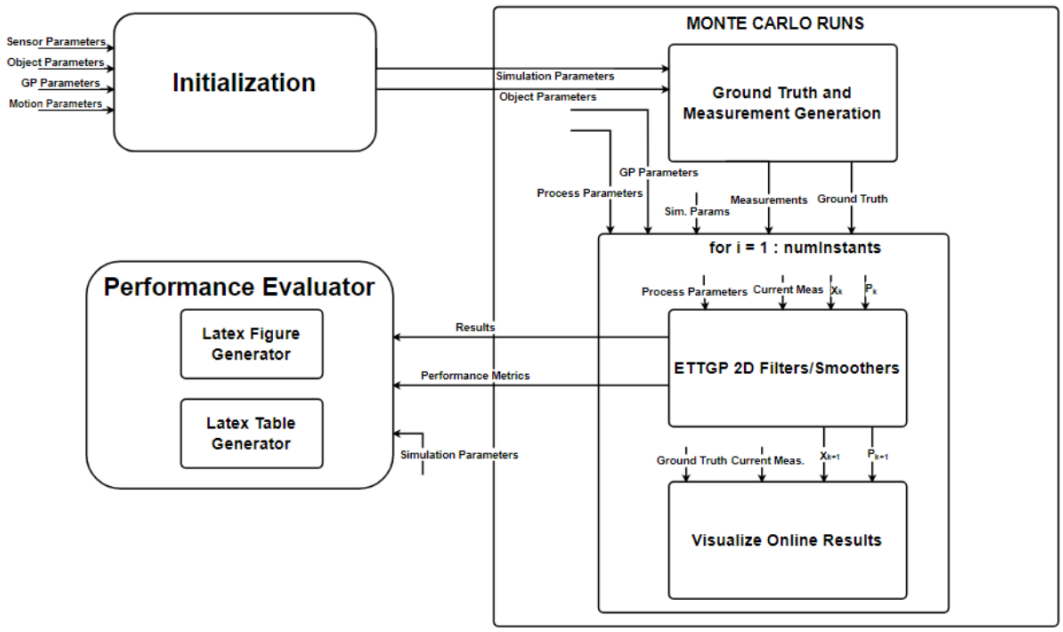

Figure 5.1: Schematic of ETTGP 2D framework

tables' captions. For instance, when scenario $O_2 - M_3 - S_1$ is referred, it means that rectangle moving in accordance with coordinated turn model is tracked by a sensor that generates uniform measurements along the extent of the object. In a similar manner, scenario $O_3 - M_2 - S_2$ refers to an isosceles triangle moving in accordance with constant velocity model which tracked by a LIDAR sensor that generates partial measurements along the extent of the object.

Table 5.1: Simulation configurations for ETTGP 2D algorithms

Object Type	Motion Model	Sensor Type
Ellipse (O_1)	Still (M_1)	Uniform (S_1)
Rectangle (O_2)	Constant Velocity (M_2)	LIDAR (S_2)
Isosceles Triangle (O_3)	Coordinated Turn (M_3)	
Plus Shaped (O_4)		

Parameter set in Table 5.2 are used in the ETTGP 2D simulations unless otherwise specified.

Table 5.2: Parameter set used in ETTGP 2D simulations

Abbreviation	Explanation	Value	Unit
r	GP Mean	0	m
σ_r	GP Mean Std. Dev.	0.5	m
σ_f	GP Prior Std. Dev.	2	m
l	GP Length Scale	$\pi/4$	-
γ	Forgetting Factor for Extent	10^{-4}	-
σ_z	Std. Dev. for Measurement Noise	0.02	m
T	Sampling Time	0.5	s
v	Velocity	0.4	m/s
σ_q	Std. Dev. of Process Noise for Position	0.03	m
σ_{q^ψ}	Std. Dev. of Process Noise for Orientation	0.003	rad
α	UT Parameter	$\frac{1}{\sqrt{n}}$	-
β	UT Parameter	2	-
κ	UT Parameter	1	-
n	State Dimension	56	-
N	Number of Basis Points for Extent Estimation	50	-
n_k	Number of Measurements per Instant	20	-
σ_w	Std. Dev. for Angular Constraint's Noise	0.00175	rad
σ_d	Std. Dev. for Radial Constraint's Noise	0.1	m

5.1.2 Performance Measures

Performances of the ETTGP 2D algorithms for extent estimation are calculated by the widely used Intersection over Union (IoU) measure

$$\text{IoU}(S_{gt}, \hat{S}) = \frac{\text{area}(S_{gt} \cap \hat{S})}{\text{area}(S_{gt} \cup \hat{S})}, \tag{5.1}$$

where S_{gt} denotes ground truth target shape and $\hat{S}$ denotes the shape estimated by the algorithm. IoU is calculated by dividing the area of the intersection of the shapes to

the union of the shapes. By this definition, IoU $\in [0, 1]$. IoU $= 1$ means the perfect estimation whereas IoU $= 0$ means estimates are entirely false. It is paramount to emphasize that IoU metric determines the quality of the kinematics estimates as well as extent. If extent of the target is erroneously estimated, IoU measure will be low. Even if we assume that the extent estimates are perfect, if kinematics of the target is erroneously estimated, IoU measure will again be low.

Performances of the ETTGP 2D algorithms for kinematics estimation are calculated by the widely used Root Mean Square Error (RMSE) measure

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(\bar{\mathbf{x}}_k - \bar{\mathbf{x}}_{k,gt})^2}, \tag{5.2}$$

where $\bar{\mathbf{x}}_k$ denotes kinematics estimates and $\bar{\mathbf{x}}_{k,gt}$ ground truth kinematics.

It is paramount to emphasize that GP modelling of the extent does not impose a unique center representation of the target. Thus, IoU results are more informative than RMSE results for representation of a target.

5.1.3 Computation Time

Compared ETTGP 2D algorithms utilized different inference techniques such as EKF, UKF, and UKF Ng. Estimates are recursively updated as new measurements obtained in each time step. Since all algorithms are recursive, computational complexity does not increase over time. Computational complexity only depends on the number of the measurements at a scan and the size of the state vector.

Simulations are run in MATLAB 2020b on a laptop with AMD Ryzen 5 PRO 4650U 2.10 GHz CPU with 16 GB of RAM. Computation time for a step of the algorithms are found as 2.02 ms for EKF ETTGP, 62.5 ms for UKF ETTGP and 85.5 ms for UKF Ng. ETTGP model on average. Note that algorithms are not computationally optimized with any method.

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.2193	0.1412	0.0219	0.9029
UKF	0.2091	0.1324	0.0218	0.9021
UKF Negative	1.1608	0.8817	0.6062	0.8327
EKS	0.2141	0.1399	0.0195	0.9092
UKS	0.2063	0.1301	0.0197	0.9084
UKS Negative	1.1584	0.8664	0.5927	0.8564

Table 5.3: Performances of ETTGP 2D algorithms for scenario $O_2 - M_3 - S_1$

5.1.4 Results and Discussions

In this section, results for ETTGP 2D algorithms are given. Firstly, detailed results for three scenarios, $O_2 - M_3 - S_1$, $O_2 - M_3 - S_2$, and $O_3 - M_3 - S_2$, are discussed. Then, performance measures for other scenarios such as $O_3 - M_3 - S_1$, $O_2 - M_2 - S_1$, $O_2 - M_2 - S_2$ etc. are presented. Unless otherwise specified, sensor is located at the origin, (0,0), for all scenarios.

In Fig. 5.2 results of a sample run from scenario $O_2 - M_3 - S_1$ are given whereas in Fig. 5.3, results of the orientation estimation averaged from 100 MC runs in scenario $O_2 - M_3 - S_1$ are given. Then, IoU estimates averaged from 100 MC runs in scenario $O_2 - M_3 - S_1$ are shown in Fig. 5.4. Lastly, performance measures are given in Table 5.3. In scenario $O_2 - M_3 - S_1$, all algorithms successively track the target. Orientation and IoU estimates of UKF Ng. are lower than EKF and UKF. Note that measurements are uniformly generated along the extent of the target in this scenario. In uniform sensor case, sensor to target geometry leads to wrong angular and radial constraints. Since, measurements are obtained throughout the extent, there cannot be negative information. Hence, results for uniform sensor case is not meaningful for UKF Ng. algorithm. IoU estimates are more erroneous at the start and end of the coordinated turn motion as expected. Note that since extent estimates are pretty accurate for all filters, IoU improvements from smoothers are marginal. Similarly, RMSE errors in smoothers are quite close to filters.

In Fig. 5.5, results of a sample run from $O_2 - M_3 - S_2$ are given. In Fig. 5.6, orienta-

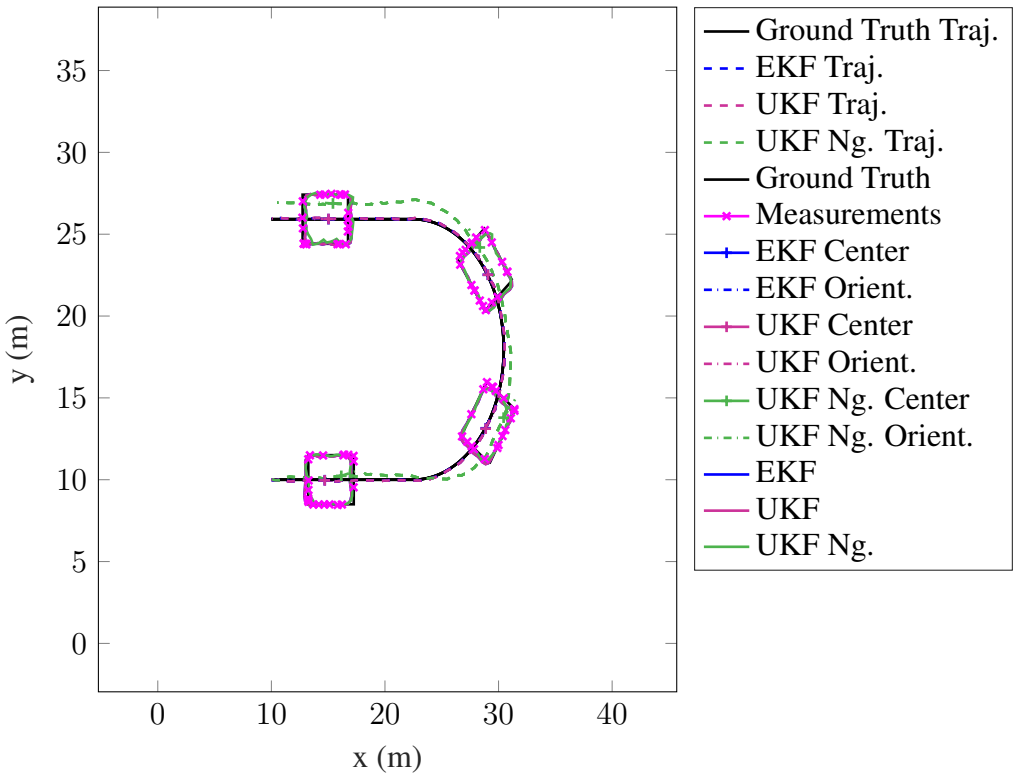

Figure 5.2: A sample run from $O_2 - M_3 - S_1$ Scenario

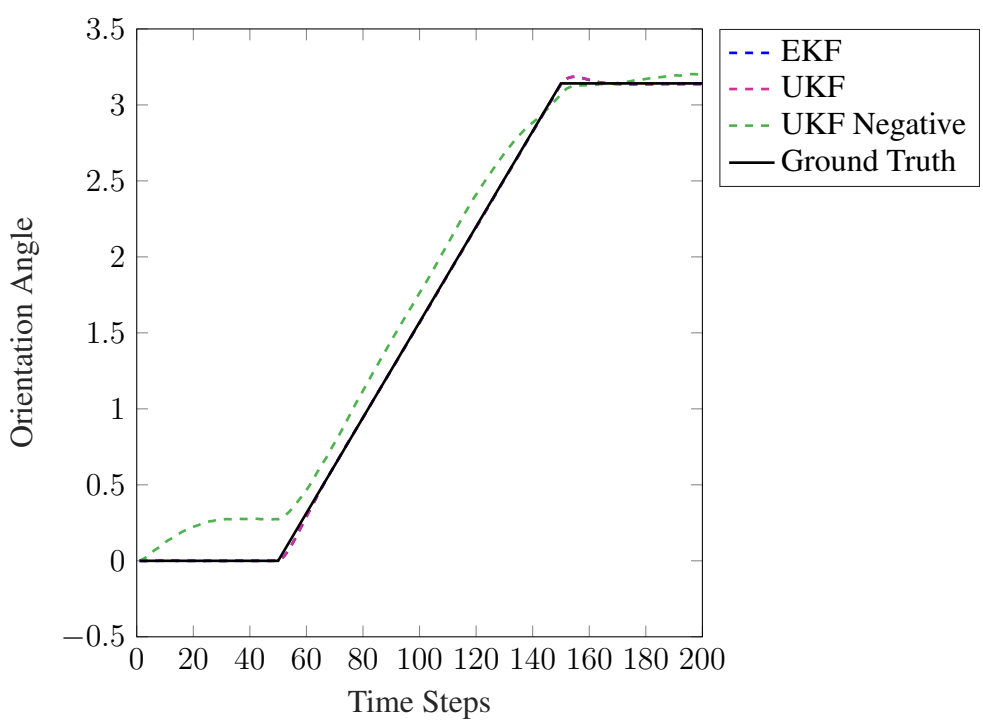

Figure 5.3: Orientation estimation in $O_2 - M_3 - S_1$ Scenario

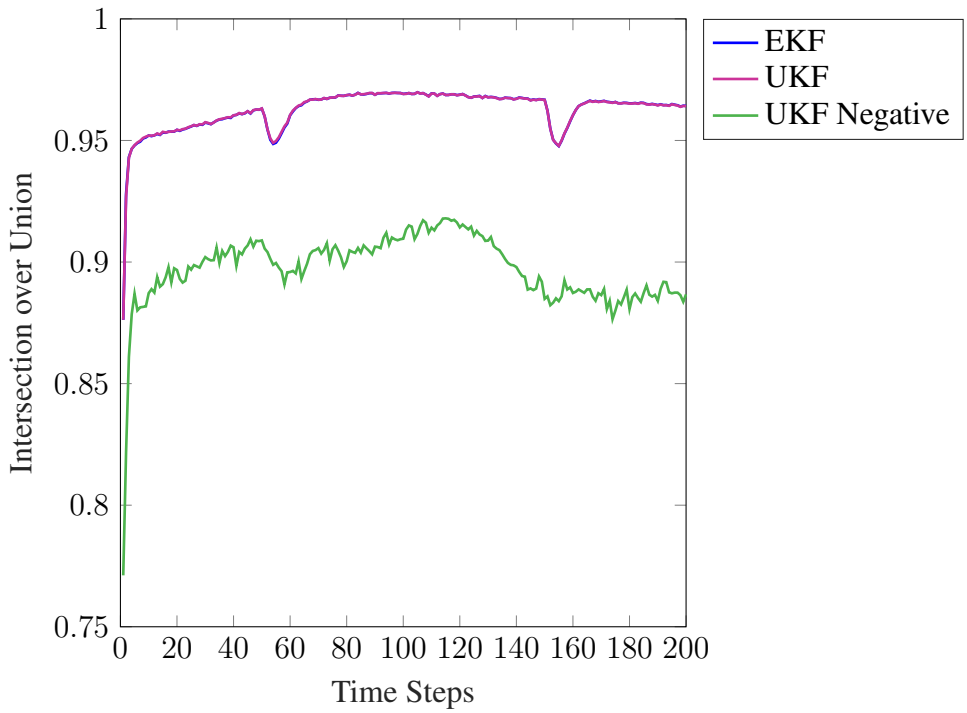

Figure 5.4: IoU results in $O_2 - M_3 - S_1$ Scenario

tion estimates averaged from 100 MC runs in scenario $O_2 - M_3 - S_2$ are given. IoU estimates averaged from 100 MC runs in scenario $O_2 - M_3 - S_2$ are given in Fig. 5.7. Lastly, performance measures are given in Table 5.4. In this scenario, measurements are obtained via LIDAR from observed parts of the target. At first, measurements are only obtained from left and bottom side of the target. Pseudo angular measurements in UKF Ng. algorithm limits the angular extent of the target. Hence, UKF Ng. algorithm outperforms EKF and UKF thanks to negative information fusion. Furthermore, orientation estimation of UKF Ng. is significantly better compared to other algorithms especially in coordinated turn part. For all algorithms, IoU estimates become better as scenario progresses since measurements from the unobserved parts of the target obtained. It is important to emphasize that both IoU and RMSE estimates of smoothers are considerably better than filters. Smoothers estimate the coordinated turn motion part of the scenario better than filters. This is expected since smoothers use all the available information in the scenario.

In Fig. 5.8, results of a sample run from $O_3 - M_3 - S_2$ are given. In Fig. 5.9, orientation estimates averaged from 100 MC runs in scenario $O_3 - M_3 - S_2$ are given.

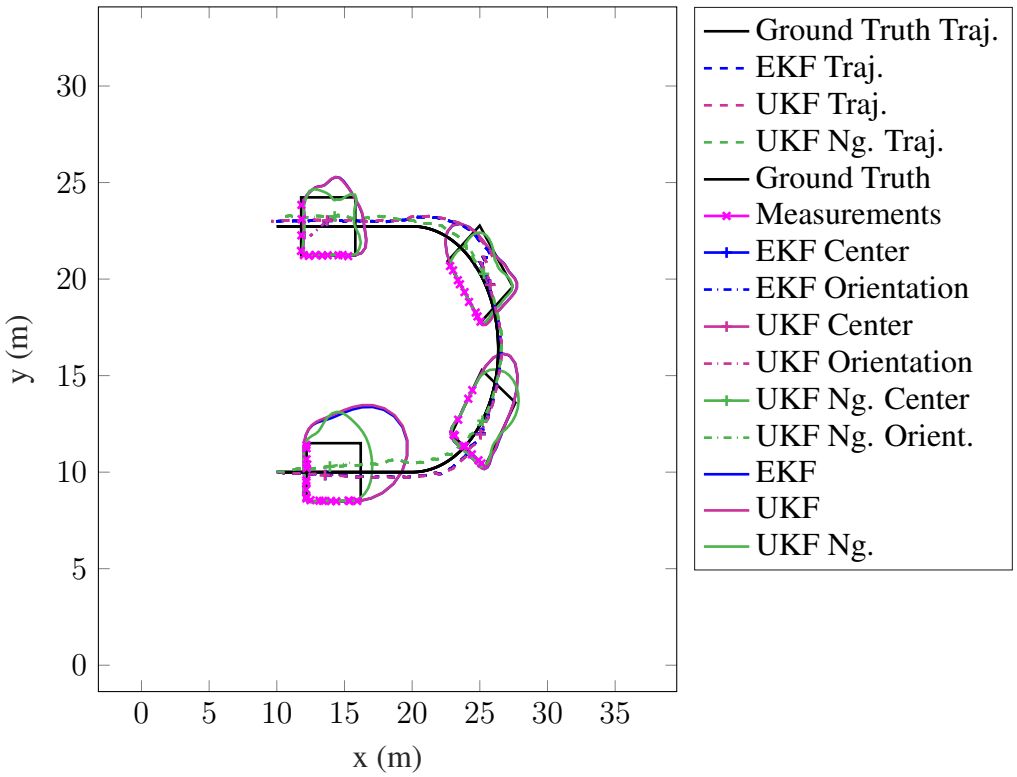

Figure 5.5: ETTGP 2D a sample run from scenario $O_2 - M_3 - S_2$

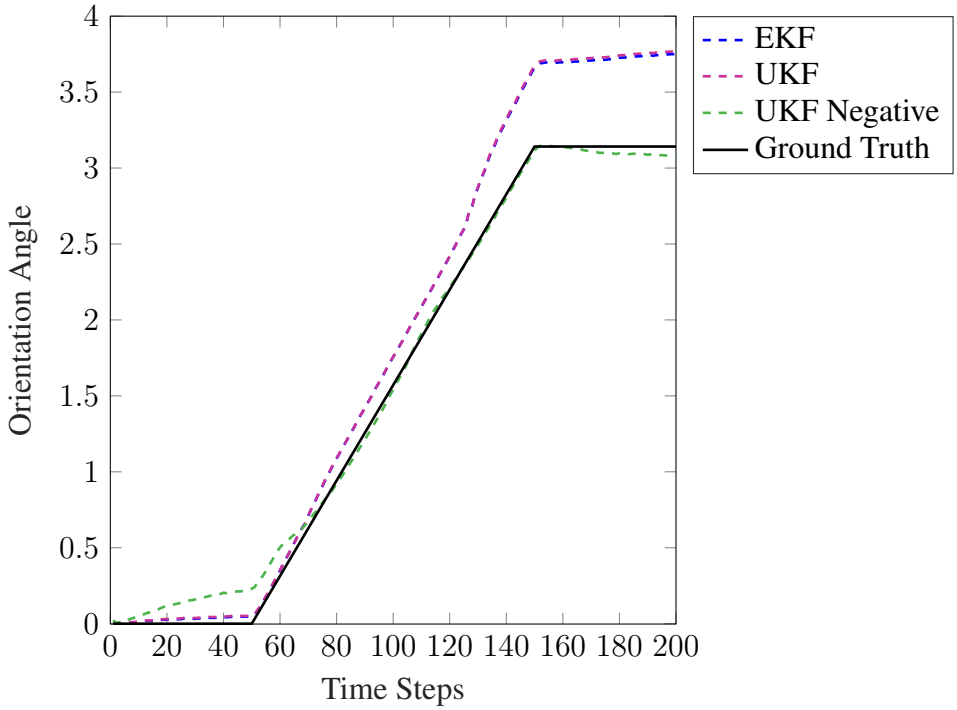

Figure 5.6: Orientation estimation in scenario $O_2 - M_3 - S_2$

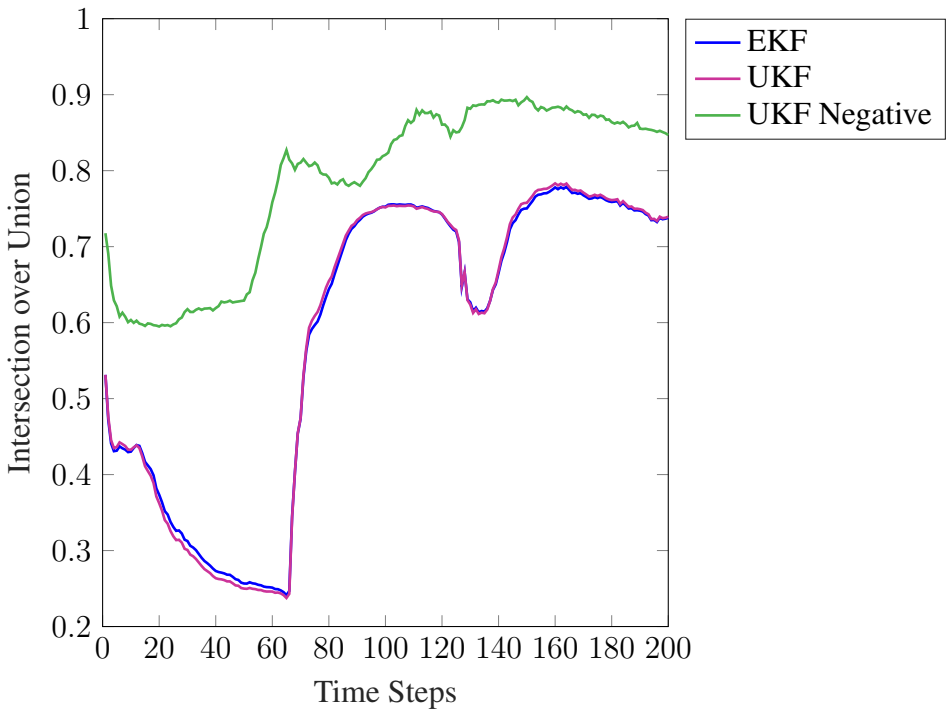

Figure 5.7: IoU estimations in scenario $O_2 - M_3 - S_2$

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.4691	0.4053	0.3439	0.5853
UKF	0.4943	0.4399	0.3521	0.5851
UKF Negative	0.2227	0.4013	0.1103	0.7838
EKS	0.4049	0.3515	0.3336	0.6958
UKS	0.4302	0.3824	0.3471	0.6964
UKS Negative	0.2156	0.3926	0.1088	0.8049

Table 5.4: Performances of ETTGP 2D algorithms for scenario $O_2 - M_3 - S_2$

IoU estimates averaged from 100 MC runs in scenario $O_3 - M_3 - S_2$ are given in Fig. 5.10. Lastly, performance measures are given in Table 5.10. In this scenario, measurements are obtained via LIDAR from observed parts of the target similar to the previous scenario. In the scenario, an isosceles triangle shaped target moves in accordance with coordinated turn model. At first, measurements are only obtained from right and bottom side of the target. Pseudo angular measurements in UKF Ng. algorithm limits the angular extent of the target. Hence, UKF Ng. algorithm outperforms EKF and UKF thanks to negative information fusion. Furthermore, orientation estimation of UKF Ng. is better compared to other algorithms. For all algorithms, IoU estimates become better as scenario progresses since measurements from the unobserved parts of the target obtained. It is important to emphasize that IoU estimates of smoothers are considerably better than filters. Smoothers estimate the coordinated turn motion part of the scenario better than filters. This is expected since smoothers use all the available information in the scenario.

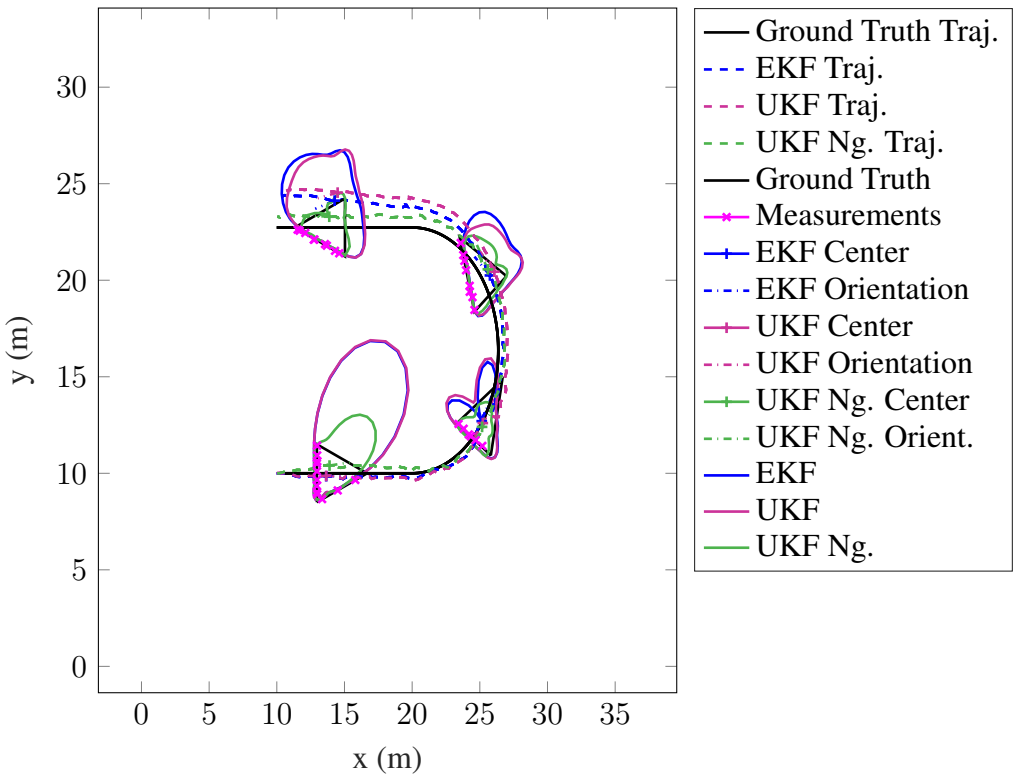

Figure 5.8: ETTGP 2D a sample run from scenario $O_3 - M_3 - S_2$

In Tables 5.5, 5.6, 5.7, 5.8, and 5.9 performance metrics for 100 MC runs are given.

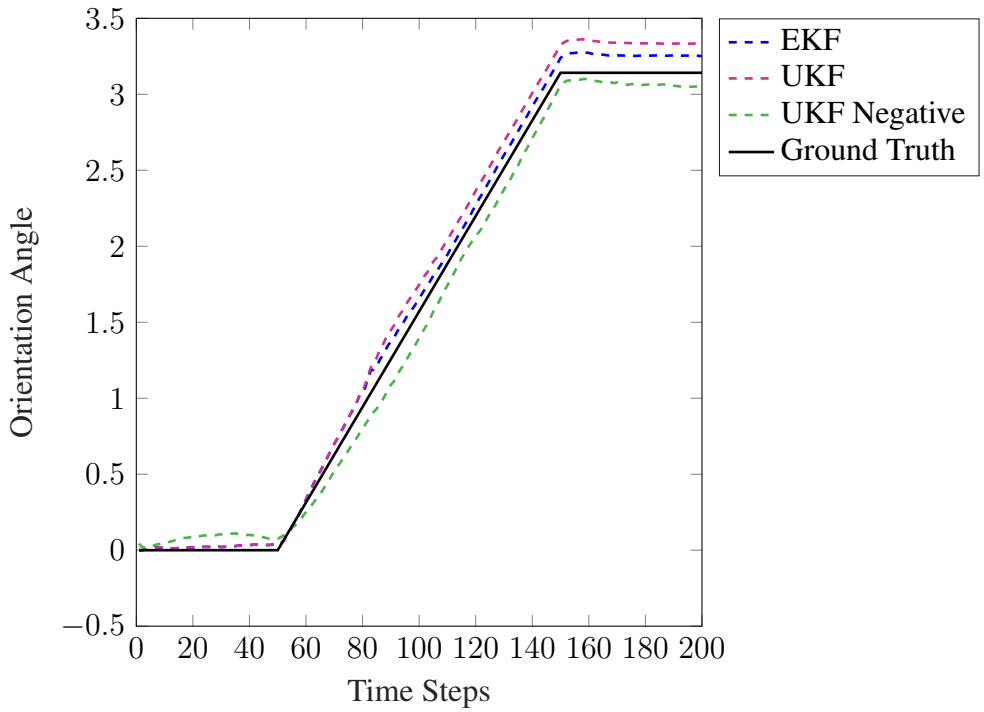

Figure 5.9: Orientation estimation in scenario $O_3 - M_3 - S_2$

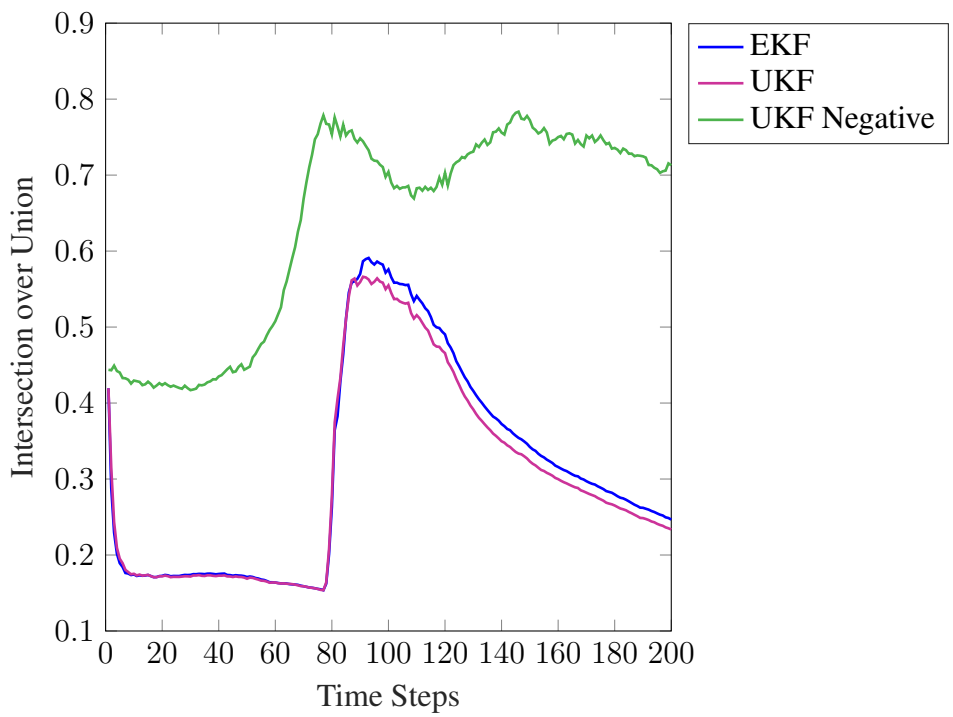

Figure 5.10: IoU estimations in scenario $O_3 - M_3 - S_2$

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.2592	0.0953	0.0175	0.9037
UKF	0.2394	0.0845	0.0164	0.9021
UKF Negative	1.4849	0.3652	0.3178	0.8145
EKS	0.2521	0.0878	0.0144	0.9086
UKS	0.2362	0.0793	0.0130	0.9072
UKS Negative	1.4273	0.3592	0.3139	0.8238

Table 5.5: Performances of ETTGP 2D algorithms for scenario $O_2 - M_2 - S_1$

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.6073	0.2571	0.0559	0.2820
UKF	0.6463	0.2895	0.0591	0.2761
UKF Negative	0.1202	0.5422	0.2772	0.6781
EKS	0.6052	0.2425	0.0544	0.2912
UKS	0.6357	0.2779	0.0578	0.2863
UKS Negative	0.1013	0.5273	0.2676	0.6941

Table 5.6: Performances of ETTGP 2D algorithms for scenario $O_2 - M_2 - S_2$

This results are similar to the discussions regarding the scenarios $O_2 - M_3 - S_1$, $O_2 - M_3 - S_2$, and $O_3 - M_3 - S_2$. UKF Ng. outperforms EKF and UKF in all LIDAR scenarios. It is paramount to emphasize that, when the target type is chosen as isosceles triangle, e.g. Tables 5.8, 5.10, UKF Ng. performs substantially better than EKF and UKF. Angular and radial constraints in UKF Ng. results in a better extent estimation since isosceles triangle does not possess two-line symmetry as rectangle does.

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.1033	0.0381	0.0272	0.9262
UKF	0.0966	0.0377	0.0271	0.9264
UKF Negative	0.5663	0.1407	0.1238	0.8651
EKS	0.0867	0.0297	0.0233	0.9399
UKS	0.0811	0.0293	0.0231	0.9401
UKS Negative	0.5657	0.1396	0.1186	0.8750

Table 5.7: Performances of ETTGP 2D algorithms for scenario $O_3 - M_2 - S_1$

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.3498	0.2321	0.1066	0.1545
UKF	0.3016	0.2661	0.0945	0.1588
UKF Negative	0.1712	0.1825	0.0617	0.4938
EKS	0.3247	0.1960	0.1014	0.1596
UKS	0.2877	0.2286	0.0921	0.1603
UKS Negative	0.1623	0.1620	0.0545	0.5067

Table 5.8: Performances of ETTGP 2D algorithms for scenario $O_3 - M_2 - S_2$

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.1021	0.0539	0.0290	0.9264
UKF	0.0958	0.0513	0.0288	0.9266
UKF Negative	0.3934	0.3323	0.1355	0.8808
EKS	0.0884	0.0481	0.0246	0.9400
UKS	0.0835	0.0469	0.0244	0.9401
UKS Negative	0.3804	0.3171	0.1306	0.8960

Table 5.9: Performances of ETTGP 2D algorithms for scenario $O_3 - M_3 - S_1$

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
EKF	0.4761	0.8789	0.1474	0.3097
UKF	0.5335	0.9637	0.1690	0.2991
UKF Negative	0.3003	0.4693	0.1158	0.6367
EKS	0.4656	0.8575	0.1459	0.4235
UKS	0.5271	0.9366	0.1635	0.4185
UKS Negative	0.2936	0.4625	0.1132	0.6791

Table 5.10: Performances of ETTGP 2D algorithms for scenario $O_3 - M_3 - S_2$

	$RMSE_x$	$RMSE_y$	$RMSE_\psi$	IoU
UKF Ng.	0.2053	0.3273	0.1088	0.7970
UKF Ng. Ang.	0.4387	0.3653	0.0908	0.7434
UKF Ng. Rad.	0.5934	0.9932	0.1123	0.4546

Table 5.11: Individual effects of angular and radial constraints on the performances for scenario $O_2 - M_3 - S_2$

Individual Effects of Angular and Radial Constraints: Individual effects of angular and radial constraints are demonstrated with a sample run in Fig. 5.11 for scenario $O_2 - M_3 - S_2$. In Fig. 5.12, orientation estimates averaged from 100 MC runs are given. IoU estimates averaged from 100 MC runs are given in Fig. 5.13. Lastly, performance measures are given in Table 5.11.

UKF Ng. abbreviation corresponds to the derived UKF Ng. ETTGP 2D algorithm. UKF Ng. Ang. consists of only angular constraints, radial constraints are left out. Similarly, UKF Ng. Rad. consists of only radial constraints. UKF Ng. Rad. underperforms compared to UKF Ng. and UKF Ng. Ang. Extent estimation of UKF Ng. Rad. is quite erroneous since angular constraints does not limit the angular extent of the target. UKF Ng. and UKF Ng. Ang. have similar angular extent estimations. However, UKF Ng. outperforms UKF Ng. Ang. in IoU estimation as can be seen from Table 5.11.

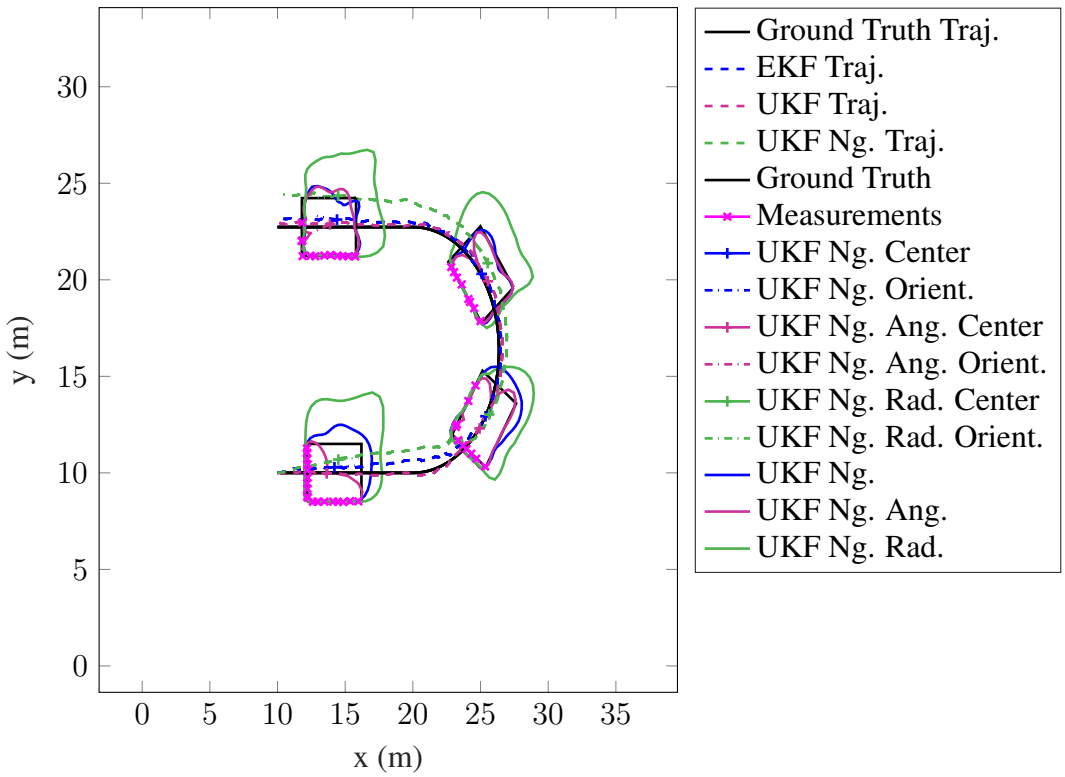

Figure 5.11: Individual effects of angular and radial constraints in scenario $O_2 - M_3 - S_2$

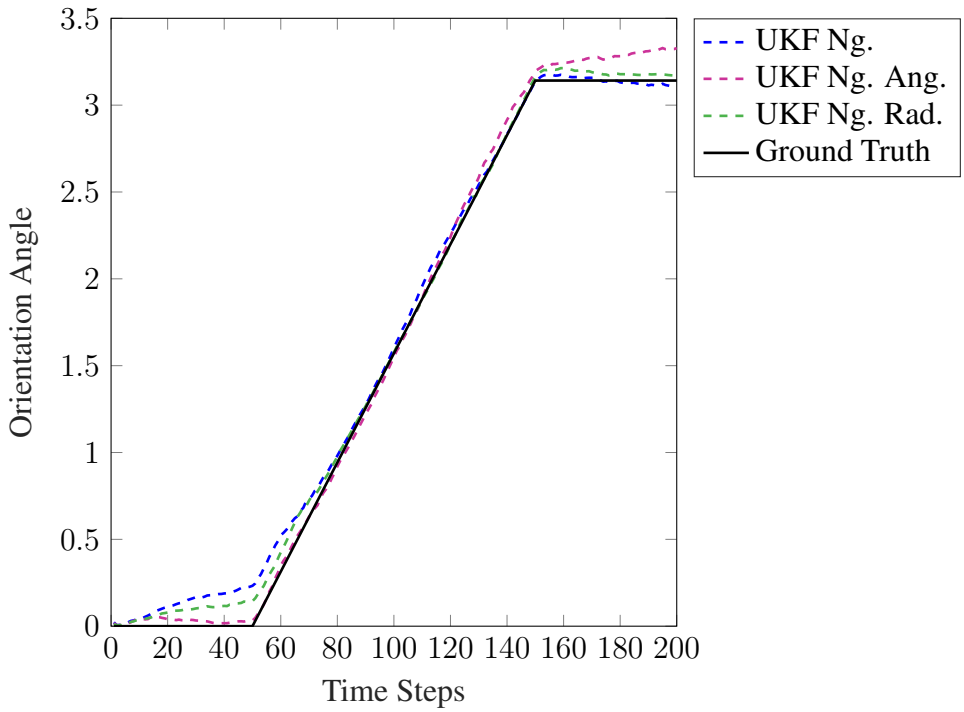

Figure 5.12: Orientation estimations in scenario $O_2 - M_3 - S_2$

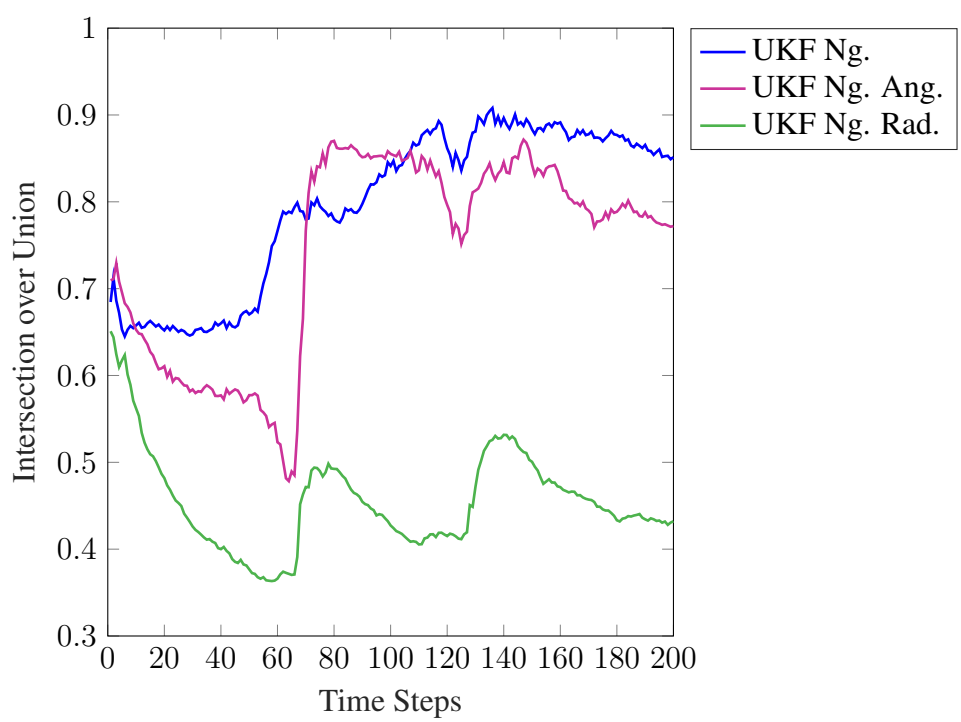

Figure 5.13: IoU estimations in scenario $O_2 - M_3 - S_2$

5.2 ETTGP 3D Simulation Results and Discussions

In this section, ETTGP 3D Scenario Generation Framework is discussed firstly. After that, parameters used in the ETTGP 3D simulations are given. Performance measures, IoU and RMSE, and scenario visualizations given later. Lastly, comments regarding the performance of different ETTGP 3D estimators such as EKF, UKF, and UKF Ng. are presented.

5.2.1 ETTGP 3D Framework

In this section, ETTGP 3D Scenario Generation Framework is presented. Framework for ETTGP 3D algorithm is quite similar to the ETTGP 2D algorithm. However, for the sake of completeness and indicating deviations from ETTGP 2D framework, framework is again discussed. ETTGP 3D Framework mainly consists of three blocks as given in Fig. 5.14.

- **Initialization:** This block takes motion, object, GP and sensor parameters as inputs. These four inputs are defined as a class in MATLAB and different types of models can be incorporated to the framework easily. Using the input classes and some scenario specific parameters, Initialization block performs necessary operations for simulation start-up.
- **Monte Carlo Runs:** This block takes Initialization block's outputs and perform Monte Carlo runs for a number of different estimation algorithms. The block is designed to be flexible in a way that many algorithms can be comparatively ran. Moreover, online figures for observing target trajectory and estimators' output are plotted. RMSE and IoU metrics, which will be discussed later, are also estimated and saved by this block.
- **Performance Evaluator:** This block is used for post-processing the results we obtained from Monte Carlo runs block. Our aim is to store the results of the simulations, i.e., figures and tables, in an easily publishable format. For this purpose, different figures are plotted and saved in a Latex format by utilizing matlab2tikz package provided in [33]. Furthermore, tables for RMSE results

of different algorithms are generated in a .tex format by utilizing latexTable function provided in [34].

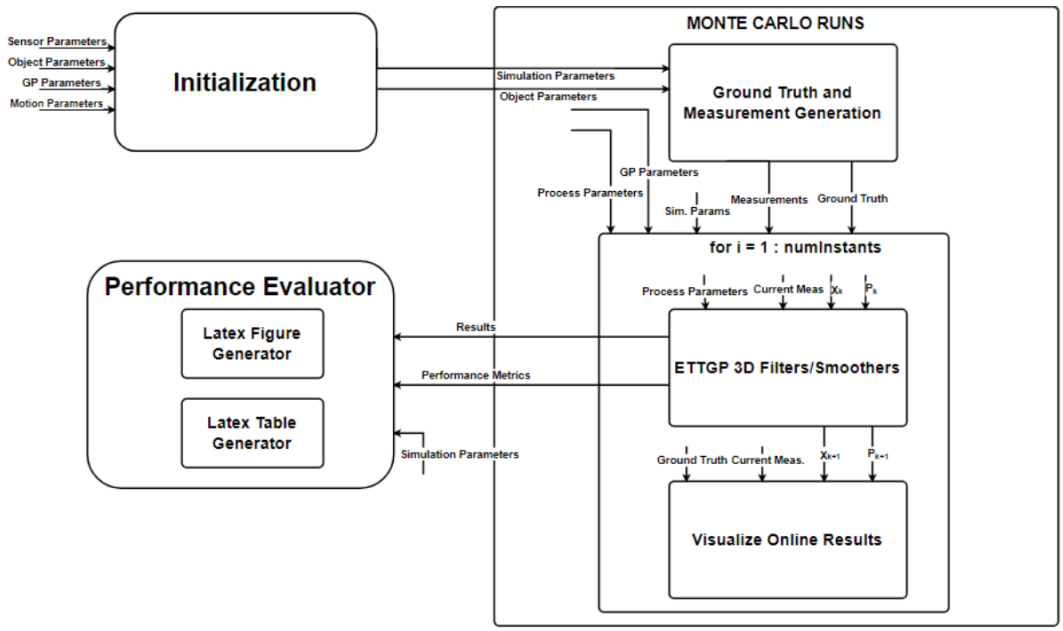

Figure 5.14: Schematic of ETTGP 3D Framework

There is a variety of simulation configurations that are combinations of different 3D target shapes, motion models and sensor models. Object, sensor and motion model types are given in Table 5.12. One can extend the framework by adding different types of motion models, object types and sensor models with seamless integration. Note that abbreviations in the parentheses will be used in the following figures' and tables' captions. For instance, when scenario $O_2 - M_3 - S_2$ is referred, it means that rectangular prism moving with coordinated turn is tracked by a LIDAR sensor that generates partial measurements over the surface of the object.

Table 5.12: Simulation configurations for ETTGP 3D algorithms

Object Type	Motion Model	Sensor Type
Ellipsoid	Still	Uniform
Rectangular Prism	Constant Velocity (CV)	LIDAR
Cone	Coordinated Turn (CT)	

Parameter set in Table 5.13 are used in ETTGP 3D simulations unless otherwise spec-

ified.

Table 5.13: Parameter set used in ETTGP 3D simulations

Abbreviation	Explanation	Value	Unit
r	GP Mean	0	m
σ_r	GP Mean Std. Dev.	0.2	m
σ_f	GP Prior Std. Dev.	1	m
l	GP Length Scale	$\pi/8$	-
σ_z	Std. Dev. for Measurement Noise	0.03	m
T	Sampling Time	0.1	s
v	Velocity	5	m/s
σ_c	Std. Dev. of Process Noise for Center	0.2	m
σ_q	Std. Dev. of Process Noise for Quaternions	0.01	rad
λ	Scaling factor for Extent Dynamics	0.99	-
α	UT Parameter	$\frac{1}{\sqrt{n}}$	-
β	UT Parameter	2	-
κ	UT Parameter	1	-
n	State Dimension	174	-
N	Number of Basis Points for Extent Estimation	162	-
n_k	Number of Measurements per Instant	20	-
σ_w	Std. Dev. for Angular Constraint's Noise	0.005	rad
σ_d	Std. Dev. for Radial Constraint's Noise	0.1	m

5.2.2 Performance Measures

Performances of the ETTGP 3D algorithms for extent estimation are calculated by the widely used Intersection over Union (IoU) measure

$$\text{IoU}(S_{gt}, \hat{S}) = \frac{\text{volume}(S_{gt} \cap \hat{S})}{\text{volume}(S_{gt} \cup \hat{S})}, \tag{5.3}$$

where S_{gt} denotes ground truth target shape and $\hat{S}$ denotes the shape estimated by

the algorithm. IoU is calculated by dividing the volume of intersection of two shapes to union of the shapes. By this definition, IoU $\in [0, 1]$. IoU $= 1$ means the perfect estimation whereas IoU $= 0$ means estimates are entirely false. It is paramount to emphasize that IoU metric determines the quality of the kinematics estimates as well as extent. If the shape of the target is erroneously estimated, IoU measure will be low. Even if we assume that the shape estimates are perfect, if kinematics of the target is erroneously estimated, IoU measure will again be low.

Performances of the ETTGP 3D algorithms for kinematics estimation are calculated by the widely used Root Mean Square Error (RMSE) measure

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(\bar{x}_k - \bar{x}_{k,gt})^2}, \tag{5.4}$$

where $\bar{x}_k$ denotes kinematics estimates and $\bar{x}_{k,gt}$ ground truth kinematics.

It is paramount to emphasize that GP modelling of the extent does not impose a unique center representation of the target. Thus, IoU results are more informative than RMSE results for representation of a target.

5.2.3 Computation Time

ETTGP 3D algorithms which utilized different inference techniques such as EKF, UKF, and UKF Ng. Estimates are recursively updated as new measurements obtained in each time step. Since all algorithms are recursive, computational complexity does not increase over time. Computational complexity only depends on the number of the measurements at a scan and the size of the state vector.

Simulations are run in MATLAB 2020b on a laptop with AMD Ryzen 5 PRO 4650U 2.10 GHz CPU with 16 GB of RAM. Computation time for a step of the algorithms is found as 30.4 ms for EKF ETTGP, 885.6 ms for UKF ETTGP and 1160 ms for UKF Ng. ETTGP model on average. Note that algorithms are not computationally optimized with any method.

5.2.4 Results and Discussions for Simulated Data

In this section, results of simulated data for ETTGP 3D algorithms are given. Detailed results for two scenarios, $O_2 - M_2 - S_2$ and $O_2 - M_3 - S_2$, are discussed. Sensor is located at the origin, (0,0,0), for all scenarios.

In Fig. 5.15 results of a sample run from scenario $O_2 - M_2 - S_2$ are given. IoU estimates averaged from 50 MC runs in scenario $O_2 - M_2 - S_2$ are shown in Fig. 5.20. Lastly, performance measures are given in Table 5.14. In Fig. 5.15, there are three snapshots from different instances. Upper part of the figure is a snapshot from the beginning of the tracking experiment. Middle part of the figure is approximately from the half duration of the experiment whereas bottom part is from ending of the experiment. UKF Ng. estimates the shape quite well compared to EKF and UKF as can be seen from the Fig. 5.15. It is important to emphasize that, uncertainties in EKF and UKF estimates grew as experiment progresses especially along z axis. Furthermore, angular extent of the object is represented quite well in UKF Ng. as opposed to EKF and UKF. This result is expected since there are no measurements obtained from top and bottom sides of the target. Pseudo angular measurements in UKF Ng. algorithm limits the angular extent of the target along the z axis. Moreover, IoU estimates in Table. 5.14, indicates the superior performance of UKF Ng. to EKF and UKF.

In Fig. 5.16 results of a sample run from scenario $O_2 - M_3 - S_2$ are given. In Fig. 5.17, 5.18, and 5.19, projections onto $X - Y$, $X - Z$, and $Y - Z$ planes are given respectively. IoU estimates averaged from 50 MC runs in scenario $O_2 - M_3 - S_2$ are shown in Fig. 5.21. Lastly, performance measures are given in Table 5.15. In Fig. 5.16, it can be seen that UKF Ng. estimates the shape of the target quite well compared to EKF and UKF in coordinated turn part. Notice that, estimated extents by EKF and UKF grew as the target makes a turn. Moreover, UKF Ng. adapted to the turn of object very well as opposed to EKF and UKF. Estimated target extent brim over the true target shape for EKF and UKF algorithms as can be seen from Fig. 5.17, 5.18, and 5.19. Furthermore, IoU estimates in Table. 5.15, highlights the superior performance of UKF Ng. to EKF and UKF.

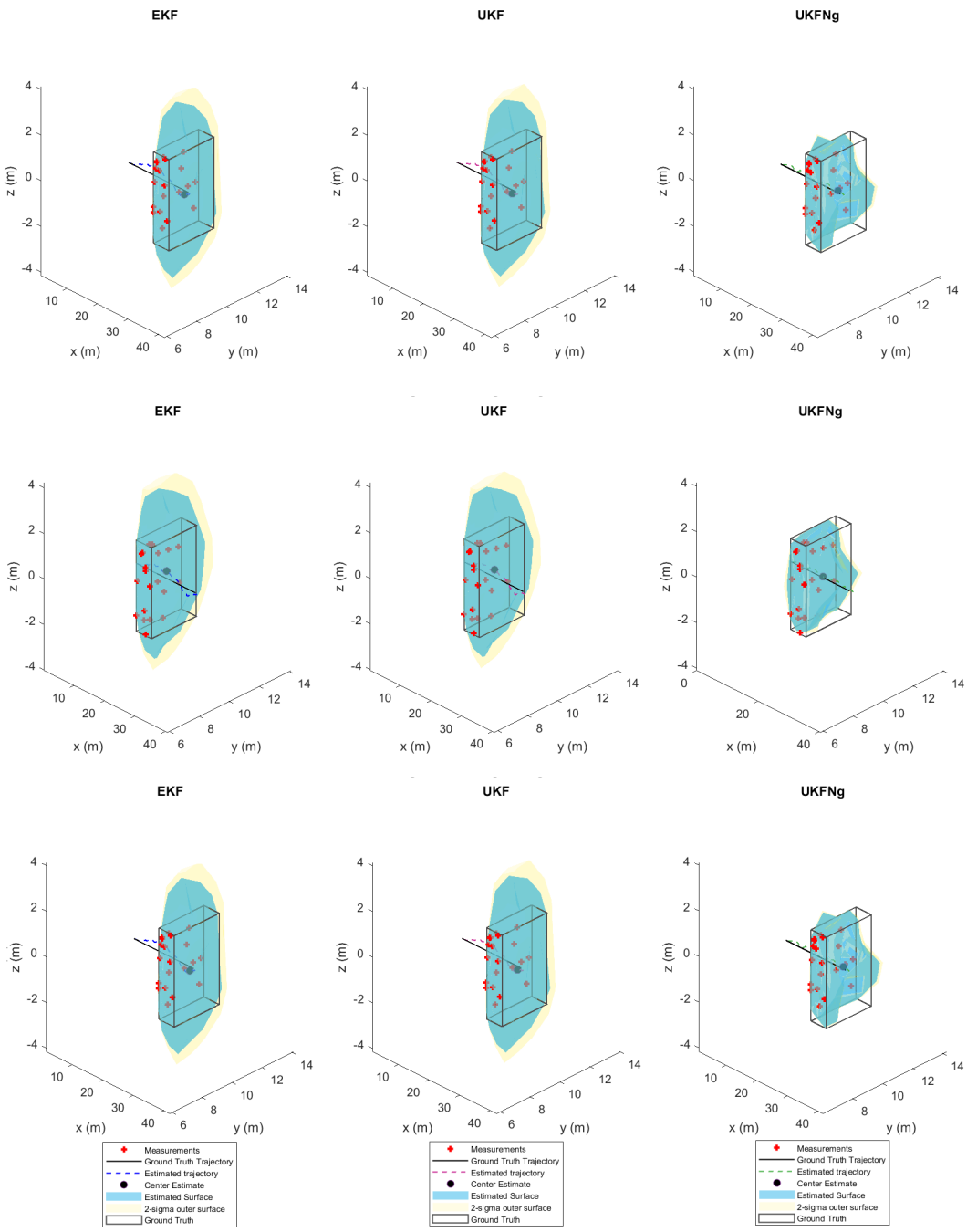

Figure 5.15: ETTGP 3D sample run from scenario $O_2 - M_2 - S_2$

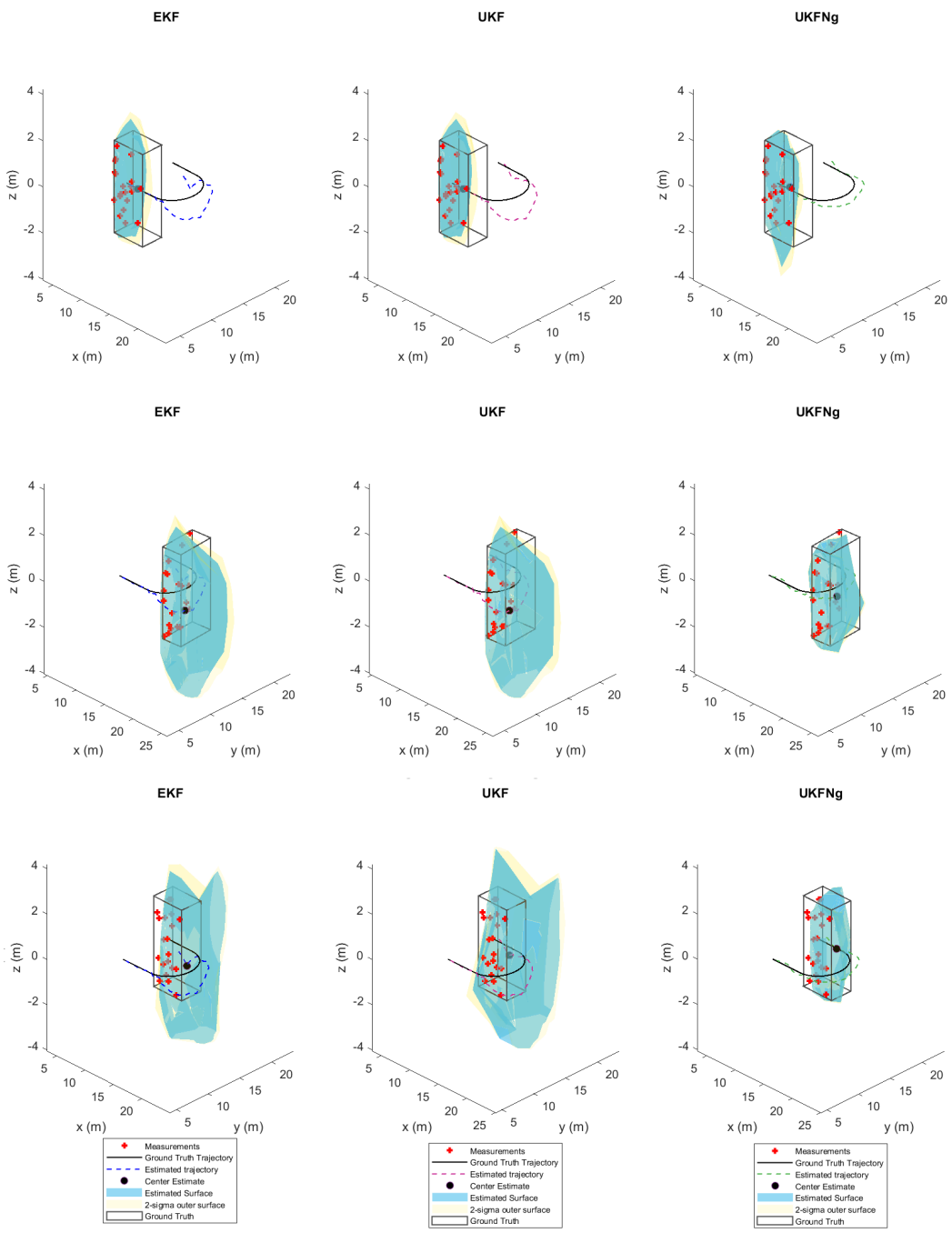

Figure 5.16: ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$

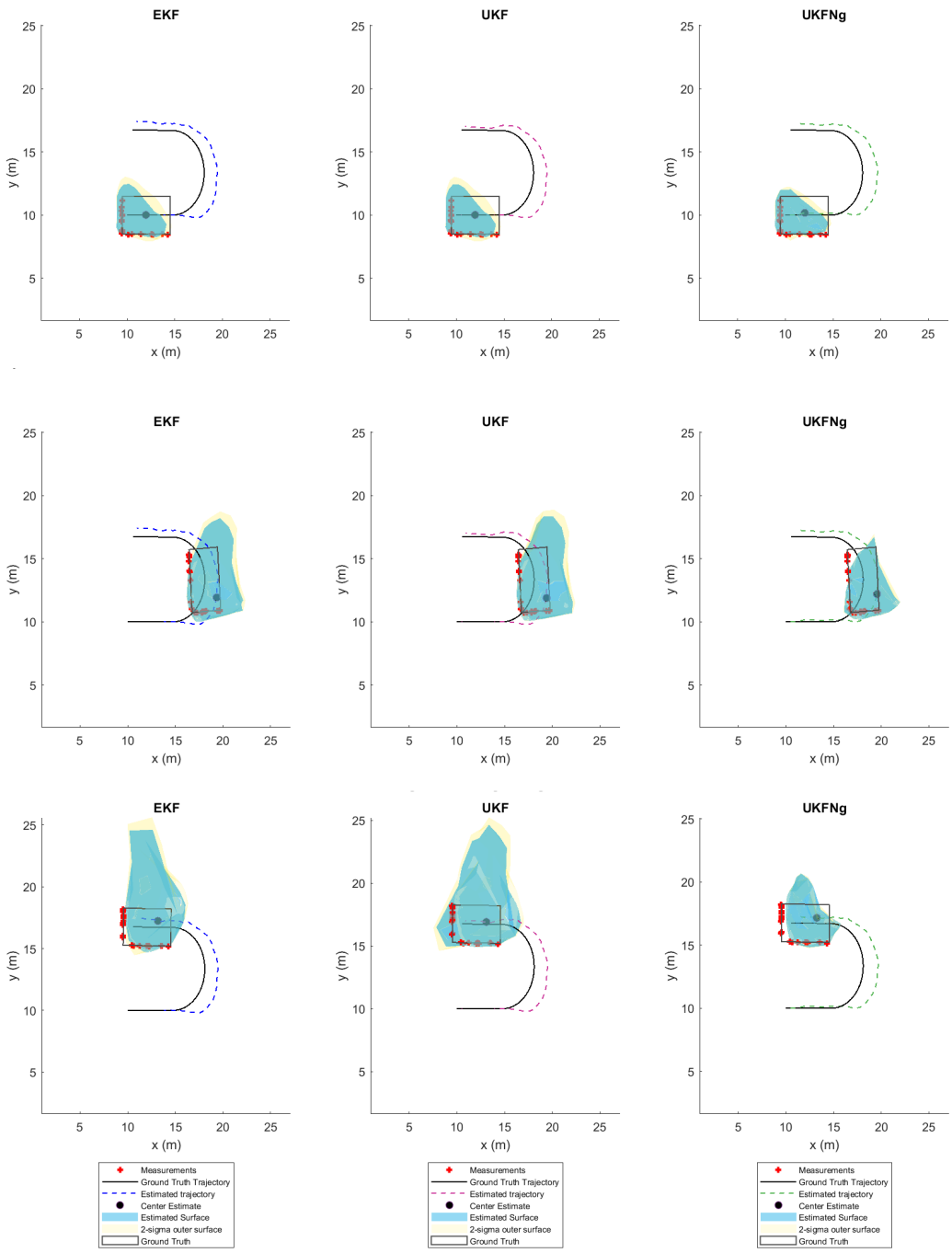

Figure 5.17: ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$ from (Top view)

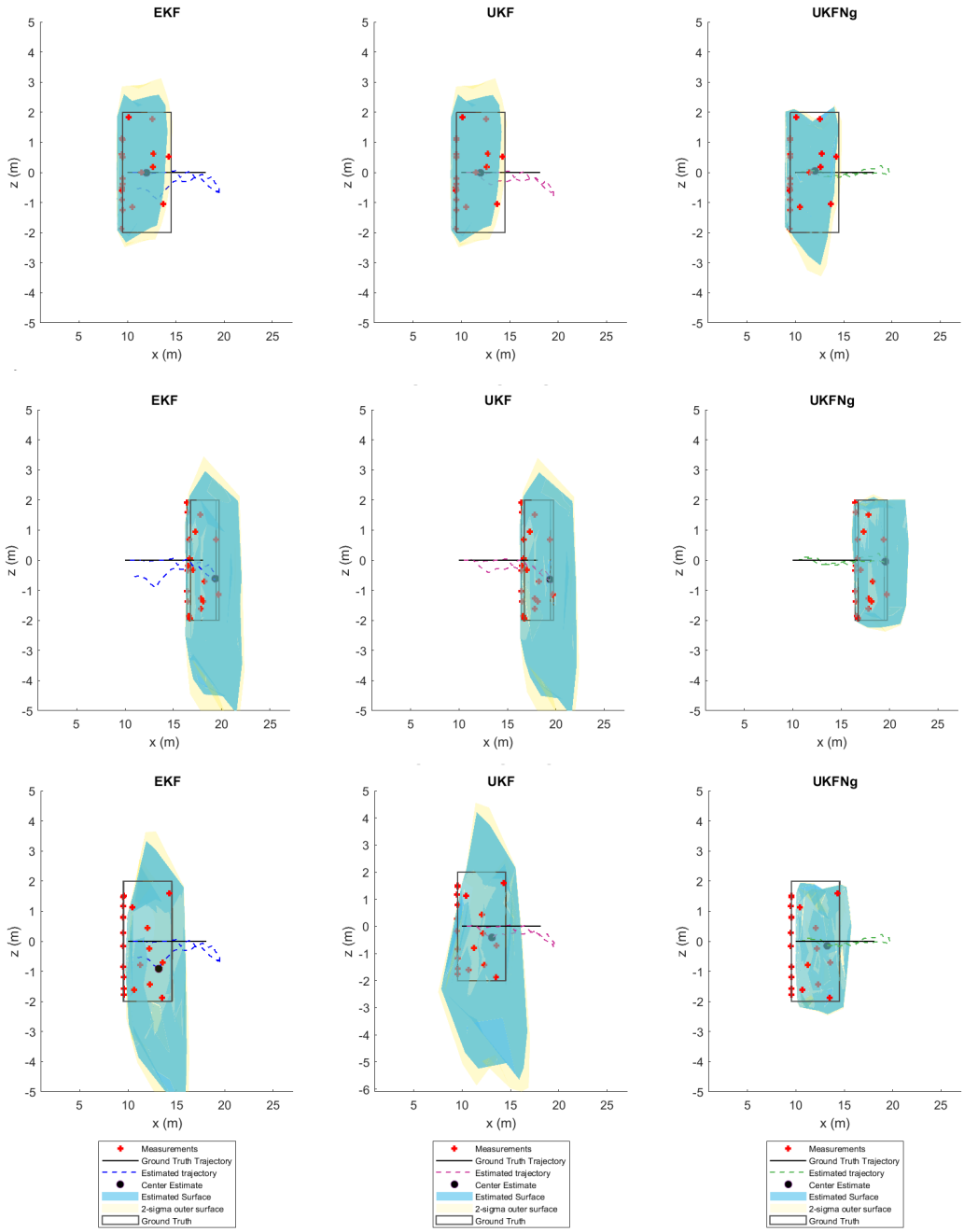

Figure 5.18: ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$ (Right view)

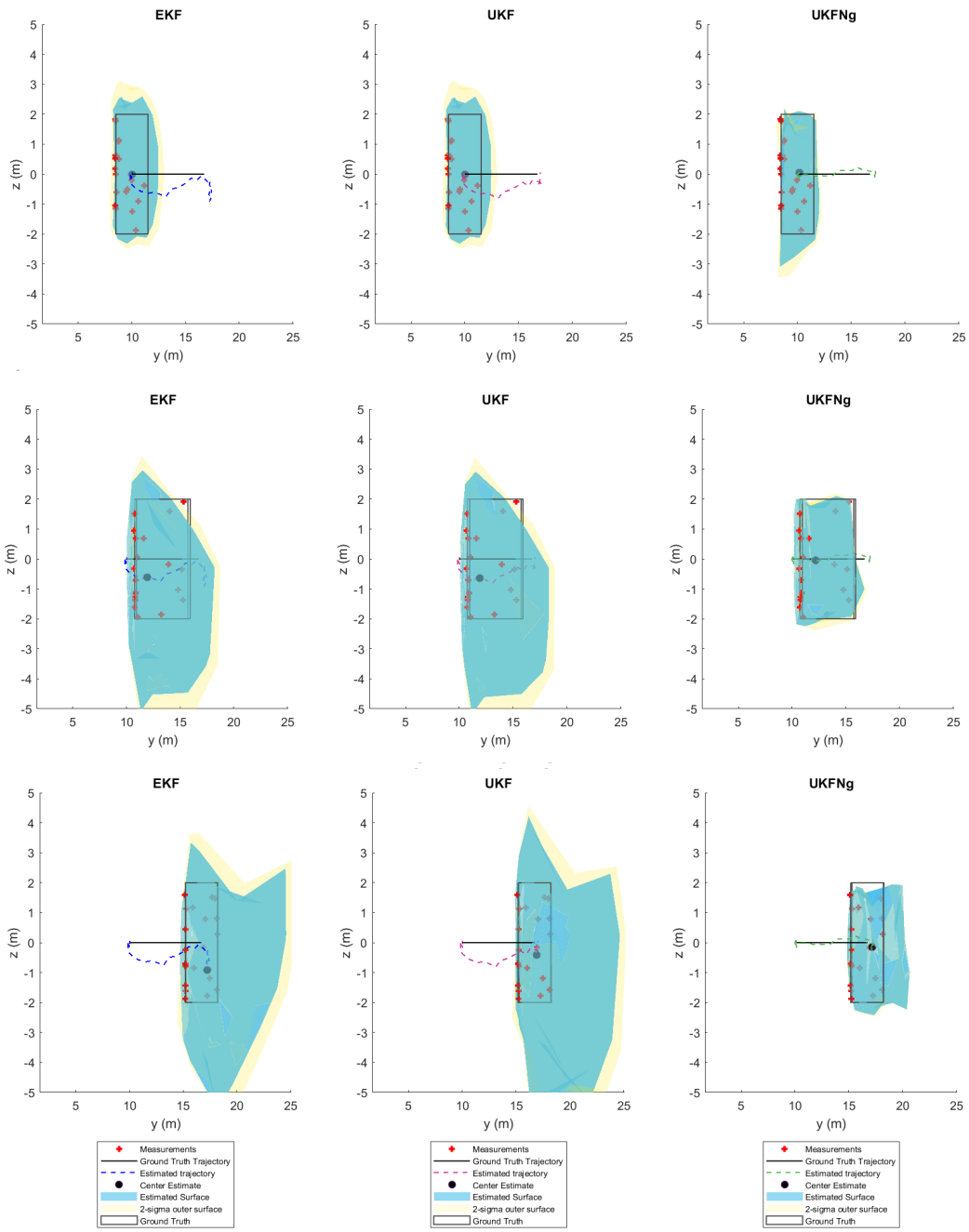

Figure 5.19: ETTGP 3D sample run from scenario $O_2 - M_3 - S_2$ (Front view)

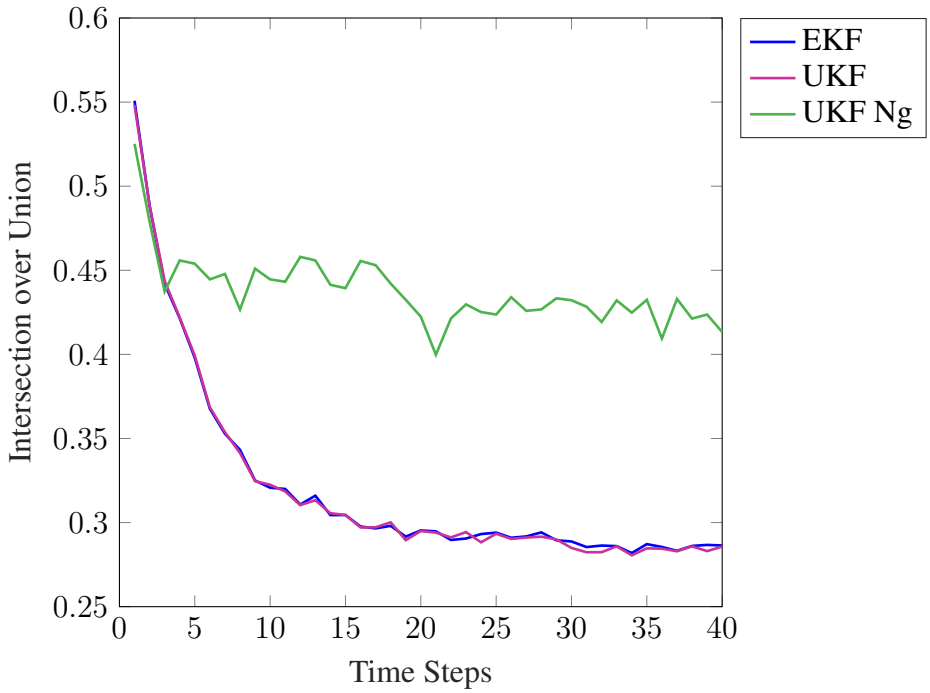

Figure 5.20: ETTGP 3D IoU estimations in scenario $O_2 - M_2 - S_2$

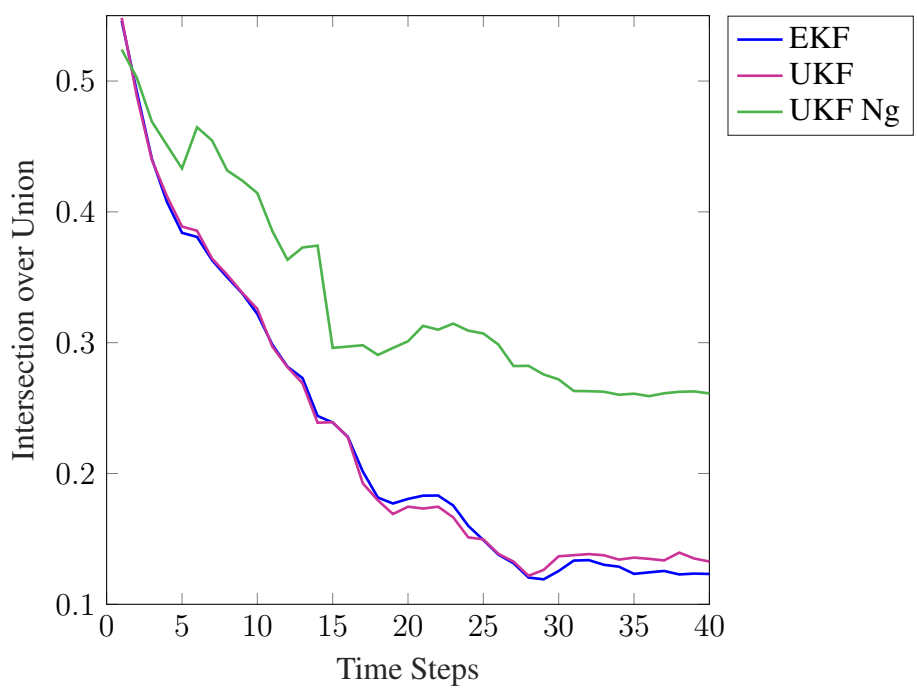

Figure 5.21: ETTGP 3D IoU estimations in scenario $O_2 - M_3 - S_2$

	$RMSE_x$	$RMSE_y$	$RMSE_z$	IoU
EKF	0.4793	0.0282	0.1737	0.3204
UKF	0.4788	0.0276	0.1710	0.3198
UKF Ng.	0.6674	0.0969	0.0864	0.4375

Table 5.14: Performances of ETTGP 3D algorithms for scenario $O_2 - M_2 - S_2$

	$RMSE_x$	$RMSE_y$	$RMSE_z$	IoU
EKF	0.9173	0.5326	0.4614	0.2271
UKF	0.8884	0.4615	0.4167	0.2285
UKF Ng.	0.9526	0.3921	0.1446	0.3356

Table 5.15: Performances of ETTGP 3D algorithms for scenario $O_2 - M_3 - S_2$

Individual Effects of Angular and Radial Constraints: Individual effects of angular and radial constraints are demonstrated with a sample run in Fig. 5.22 for scenario $O_2 - M_3 - S_2$. IoU estimates averaged from 50 MC runs are given in Fig. 5.23. Lastly, performance measures are given in Table 5.16.

UKF Ng. abbreviation corresponds to the derived UKF Ng. ETTGP 3D algorithm. UKF Ng. Ang. consists of only angular constraints, radial constraints are left out. Similarly, UKF Ng. Rad. consists of only radial constraints. UKF Ng. Rad. underperforms compared to UKF Ng. and UKF Ng. Ang. Extent estimation of UKF Ng. Rad. is quite erroneous since angular constraints does not limit the angular extent of the target. UKF Ng. and UKF Ng. Ang. have similar angular extent estimations. However, UKF Ng. outperforms UKF Ng. Ang. in IoU estimation as can be seen from Table 5.16.

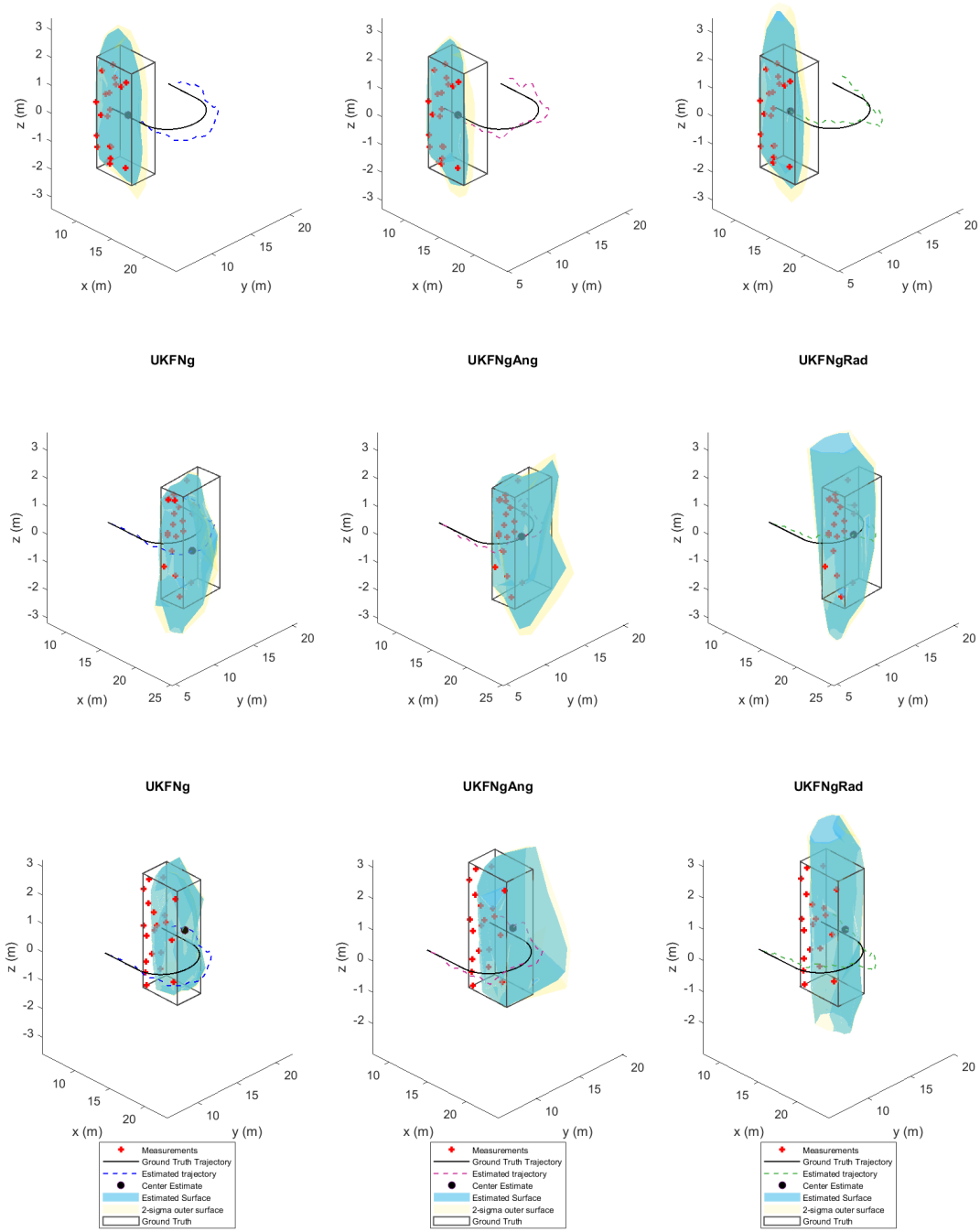

Figure 5.22: Individual effects of angular and radial constraints in scenario $O_2 - M_3 - S_2$

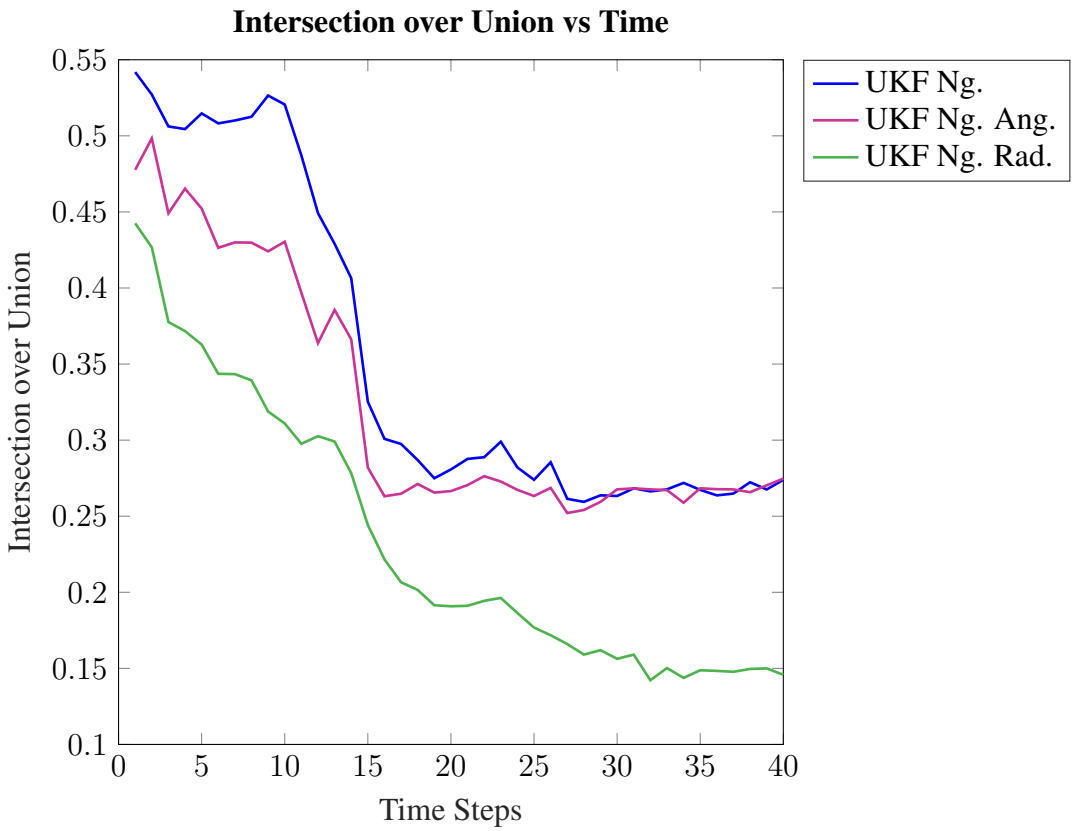

Figure 5.23: IoU estimations in scenario $O_2 - M_3 - S_2$

	$RMSE_x$	$RMSE_y$	$RMSE_z$	IoU
UKF Ng.	0.9232	0.4663	0.1317	0.3436
UKF Ng. Ang.	0.8812	0.4359	0.1929	0.3272
UKF Ng. Rad.	0.9741	0.3858	0.2497	0.2588

Table 5.16: Individual effects of angular and radial constraints on the performances for scenario $O_2 - M_3 - S_2$

5.2.5 Results and Discussions for Real Data

Experiments with real data are conducted with Kitti data set. The data set consists of several records of real-world traffic scenarios. Measurements are collected with different types of sensors that are mounted on a ego vehicle. In the experiments, measurements acquired by Velodyne HDL-64E lazer scanner are used. It is important to emphasize that raw point cloud data is not preprocessed, e.g. ground removal, association, in the experiments, rather labels provided within the data set are used. The set of parameters that are used in real data experiments is same with Table 5.12.

Three different scenarios are chosen to demonstrate the effectiveness of the proposed ETTGP 3D UKF Ng. algorithm. Scenario 1, 2, and 3 are visualized in Fig. 5.24, 5.25, and 5.26 respectively.

Figure 5.24: Visualization of Kitti data set scenario 1

In Fig. 5.24, yellow ellipse shows the car be tracked. The car consistently get away from the ego vehicle that is moving in the same direction. Note that car is accelerating during the experiment. Upper part of the Fig. 5.24 shows the beginning of the tracking experiment whereas lower part of the figure shows the ending of the tracking

experiment. Notice that measurements are only collected from the back and right side of the car throughout the experiment. Thus, uncertainties in the unobserved section of the car increase during the experiment.

Figure 5.25: Visualization of Kitti data set scenario 2

In Fig. 5.25, yellow rectangle shows the car be tracked. The car moves with a nearly constant speed. Ego vehicle is stationary at sidewalk. Similar to the Fig. 5.24 , upper part of the Fig. 5.25 shows the beginning of the tracking experiment whereas lower part of the figure shows the ending of the tracking experiment. It is important to emphasize that occlusions caused by pedestrians walking on the pavement and the column of the building imposes challenges.

In Fig. 5.26, yellow rectangle shows the car be tracked. The car makes a half turn with a nearly constant speed. Ego vehicle is stationary at red light. Similar to the Fig. 5.24 and Fig. 5.25, upper part of the Fig. 5.26 shows the beginning of the tracking experiment whereas lower part of the figure shows the ending of the tracking experiment. This scenario is specifically chosen to examine the performance of the proposed algorithm in a real life maneuvering scenario.

In Fig. 5.27, a sample run for Kitti scenario 1 is shown. In Fig. 5.27, there are three snapshots from different instances. Upper part of the figure is a snapshot from the beginning of the tracking experiment. Middle part of the figure is approximately

Figure 5.26: Visualization of Kitti data set scenario 3

from the half duration of the experiment whereas bottom part is from ending of the experiment. UKF Ng. estimates the shape quite well compared to EKF and UKF as can be seen from the Fig. 5.27. It is important to emphasize that, extent estimates of EKF and UKF grew erroneously as the experiment progresses. In this scenario, movement of the target is mainly in y direction. Measurements are mostly obtained from one side of the target. Extent along y direction is represented fairly well for all algorithms. However, estimated extents brim over the true target extent for EKF and UKF algorithms in x and z directions. Furthermore, angular extent of the object is represented quite well in UKF Ng. as opposed to EKF and UKF. Pseudo angular measurements in UKF Ng. algorithm limits the angular extent of the target along the x and z directions.

In Fig. 5.28, a sample run for Kitti scenario 2 is shown. In Fig. 5.28, it can be seen that UKF Ng. estimates the shape of the target quite well compared to EKF and UKF. Notice that, extent estimates of EKF and UKF grew erroneously as the experiment progresses especially in z direction.

In Fig. 5.29, a sample run for Kitti scenario 3 is shown. In Fig. 5.30, 5.31, and 5.32, projections onto $X-Y$, $X-Z$, and $Y-Z$ planes are given respectively. In Fig. 5.29, it can be seen that UKF Ng. estimates the shape of the target quite well compared to EKF and UKF. Notice that, extent estimates of EKF and UKF grew erroneously as the experiment progresses. Estimated target extent brim over the true target shape for

EKF and UKF algorithms as can be seen from Fig. 5.30, 5.31, and 5.32. Especially, UKF Ng. performs much better extent estimation along z direction compared to EKF and UKF algorithm as can be seen from Fig. 5.31 and 5.32.

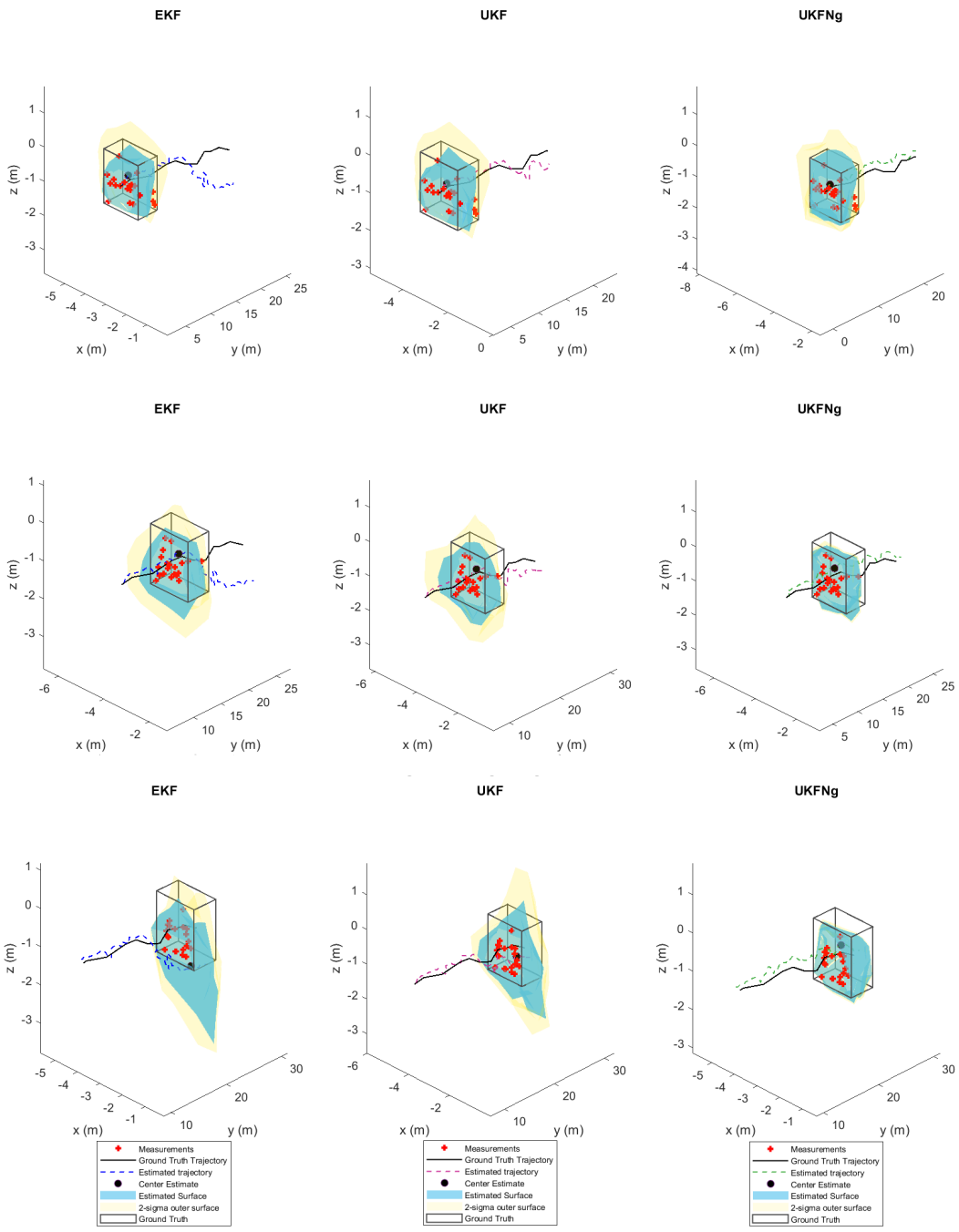

Figure 5.27: ETTGP 3D Kitti scenario 1 sample run

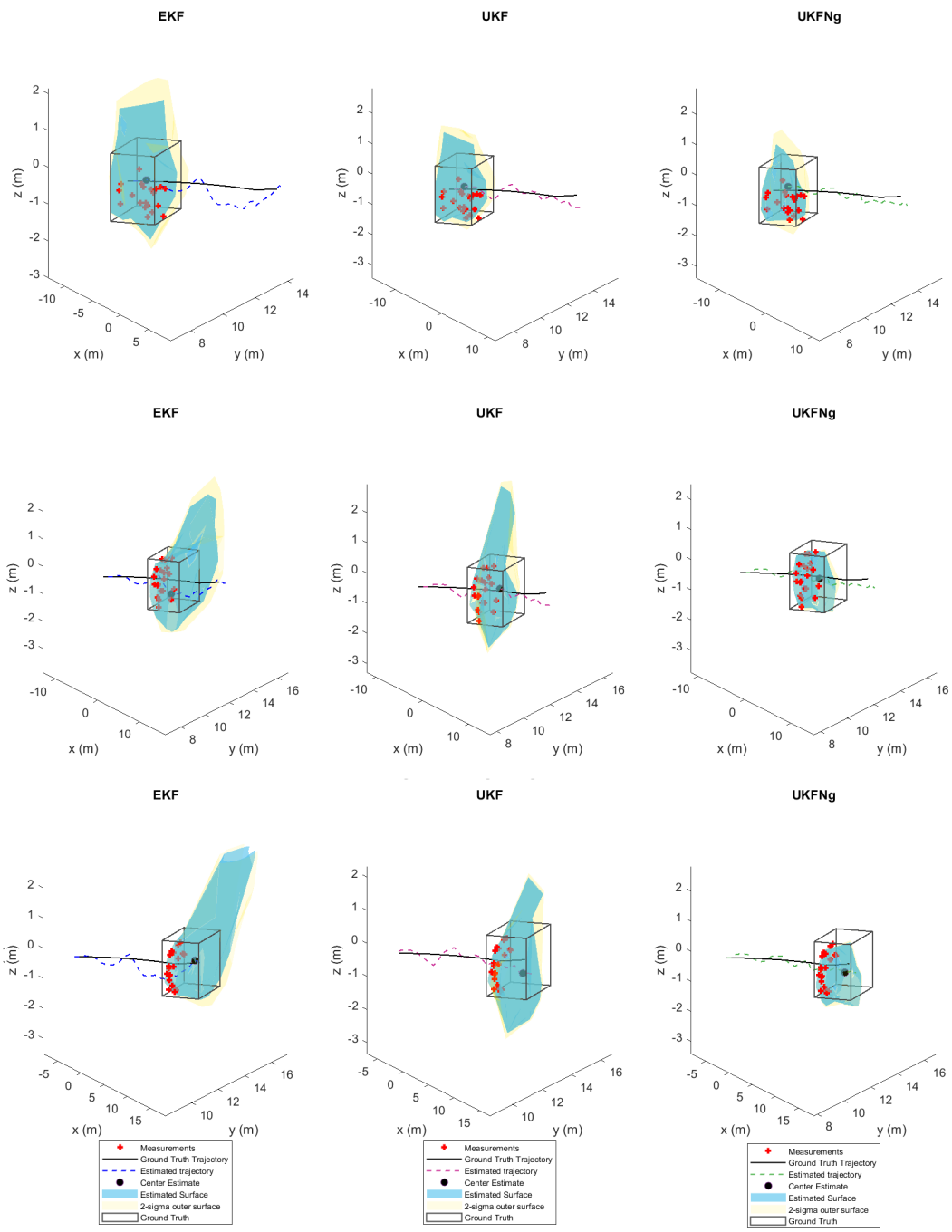

Figure 5.28: ETTGP 3D Kitti scenario 2 sample run

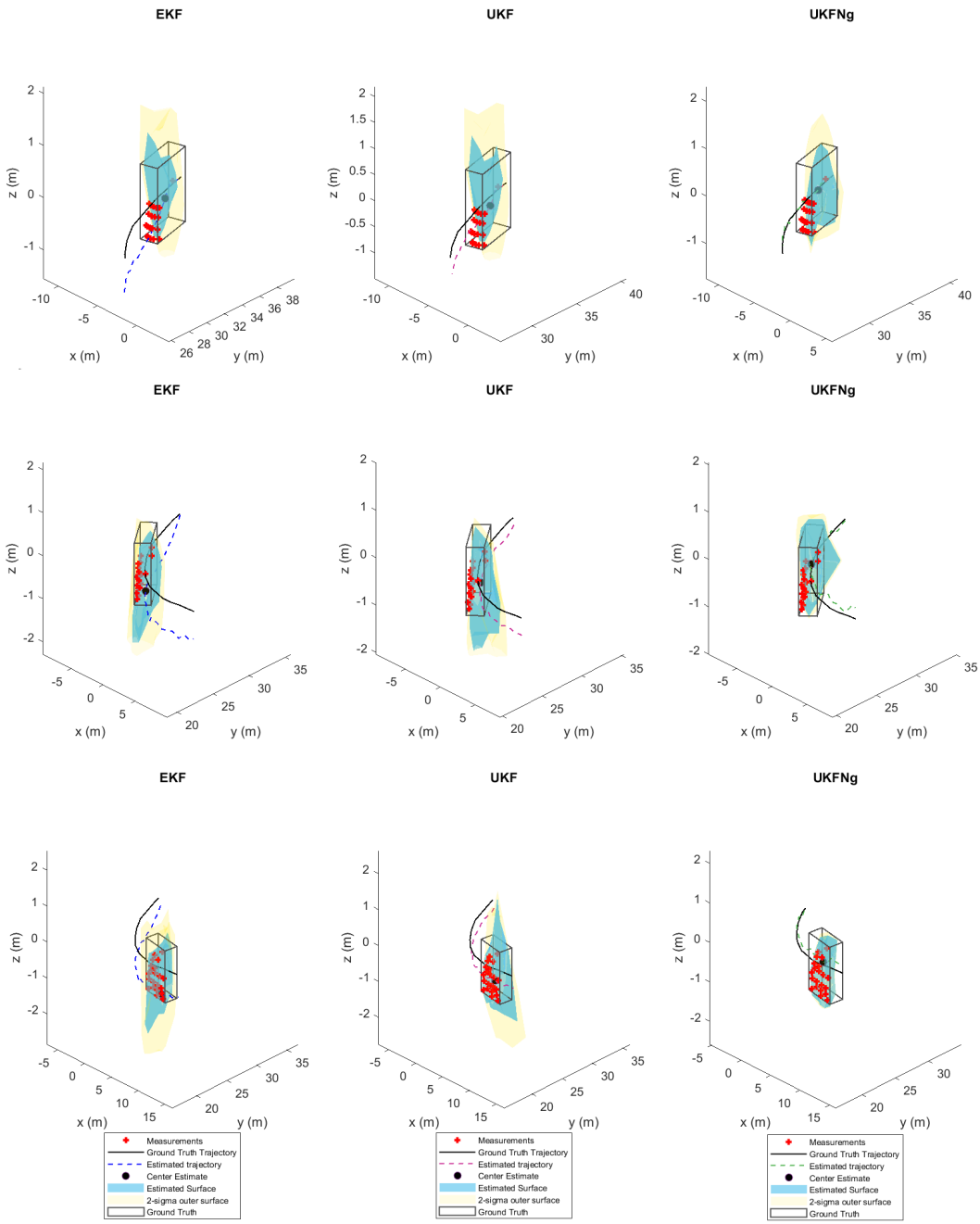

Figure 5.29: ETTGP 3D Kitti scenario 3 sample run

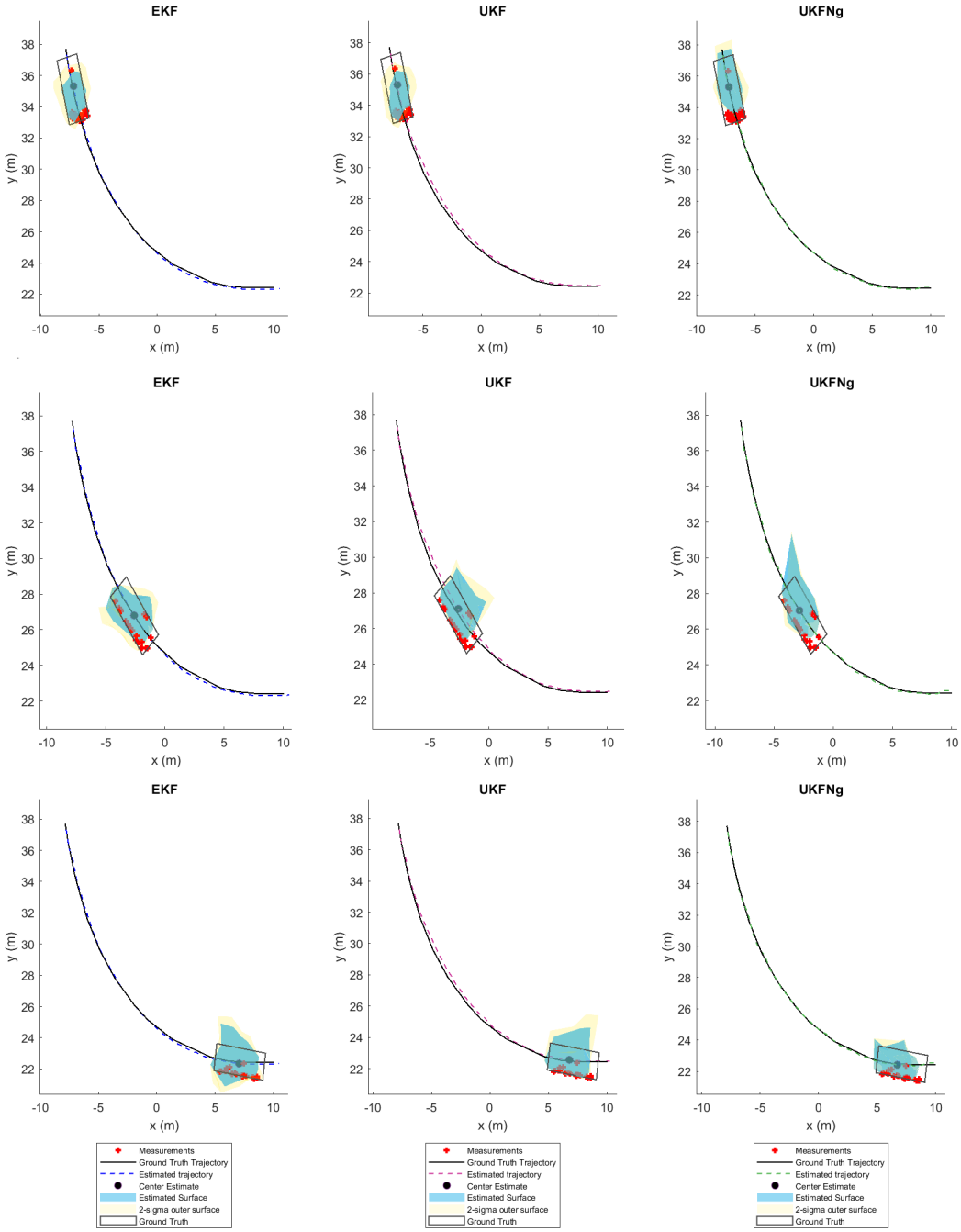

Figure 5.30: ETTGP 3D Kitti scenario 3 sample run (Top view)

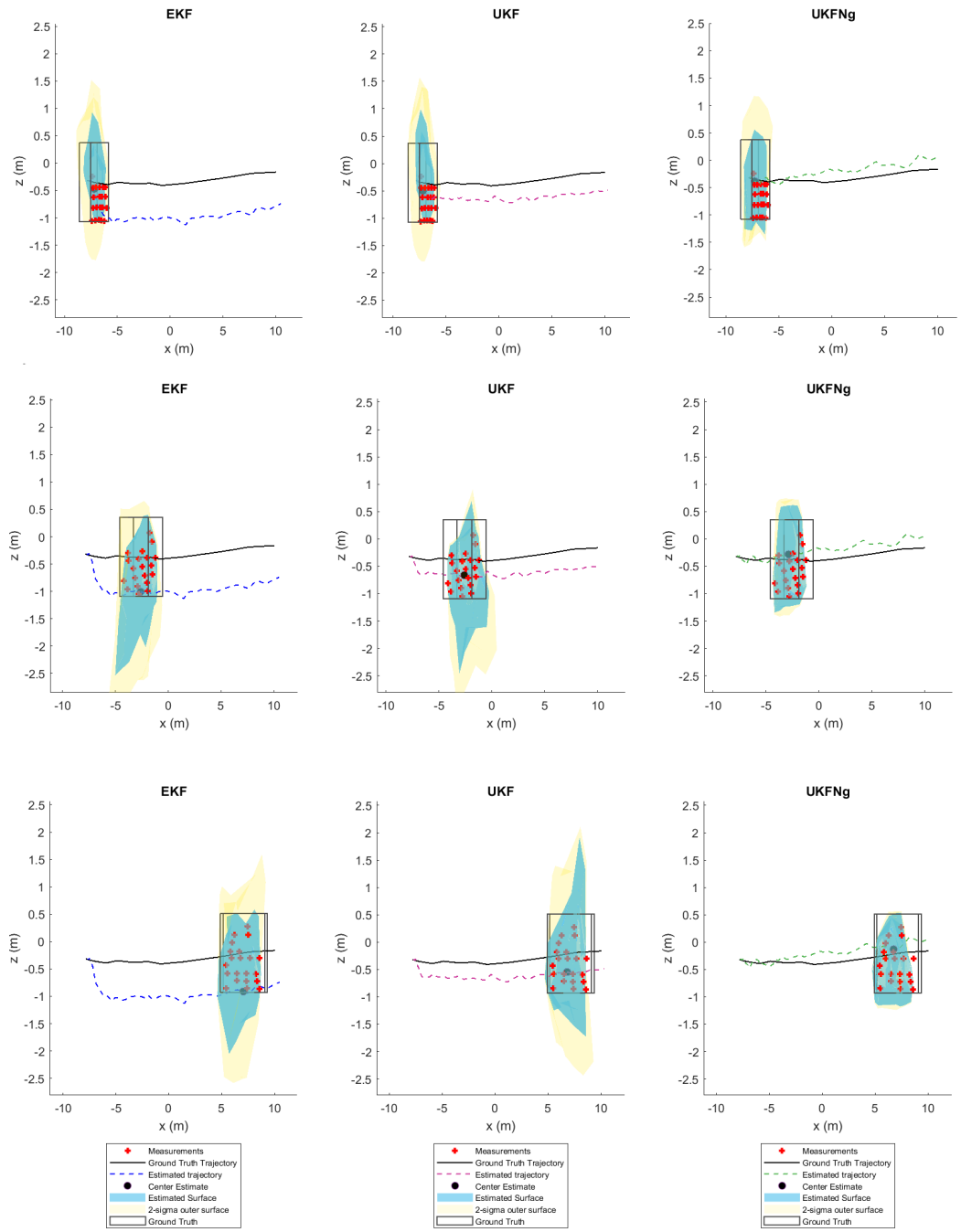

Figure 5.31: ETTGP 3D Kitti scenario 3 sample run (Right view)

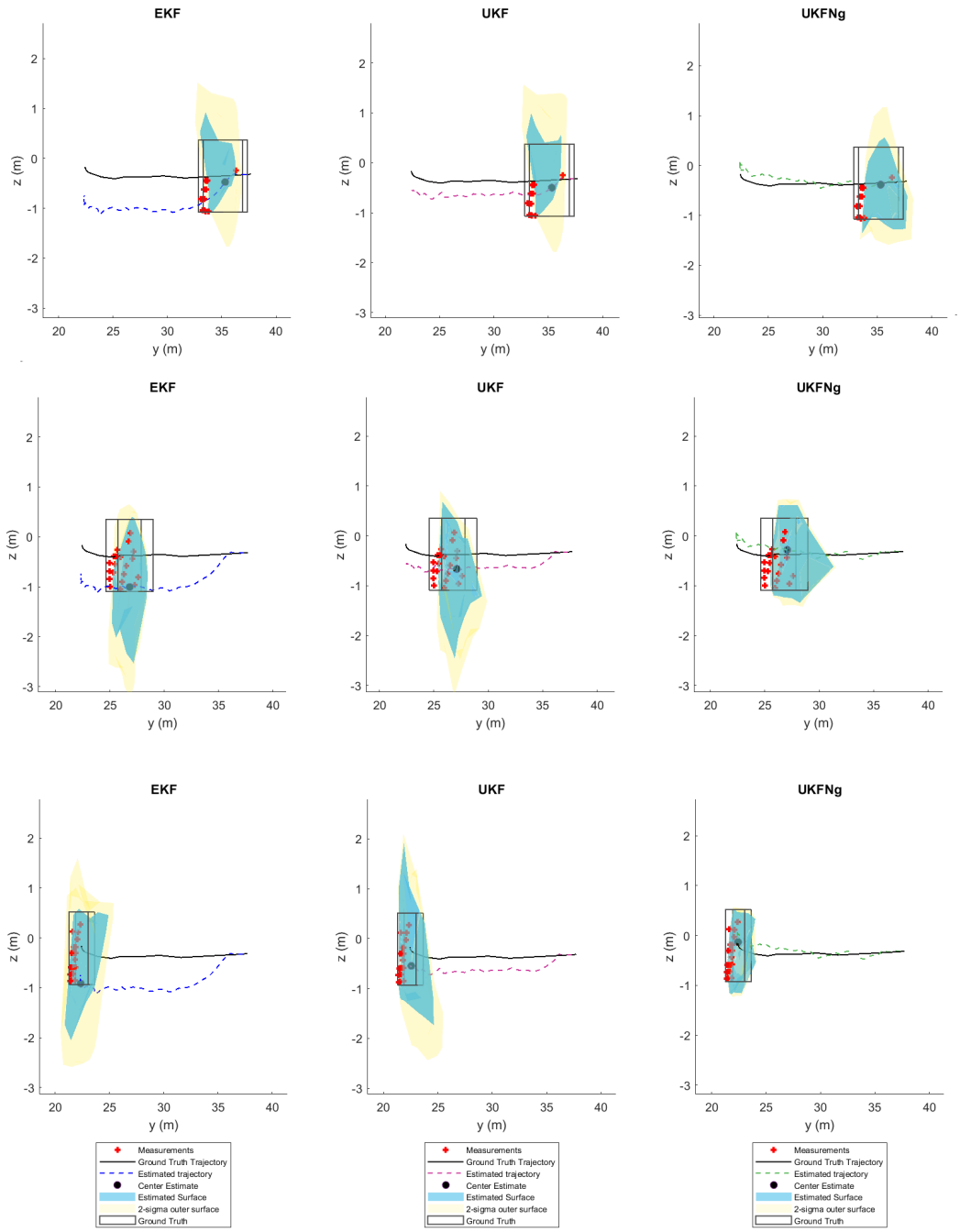

Figure 5.32: ETTGP 3D Kitti scenario 3 sample run (Front view)

CHAPTER 6

CONCLUSIONS

In this thesis, performances of different ETTGP 2D and 3D algorithms are evaluated. EKF, UKF, UKF Ng., EKS, UKS, and UKS Ng. are used for inference in ETTGP algorithms for a variety of scenarios. Furthermore, a method for ETTGP 3D which utilizes negative information fusion is proposed. This method incorporates measurements along with negative information to obtain better extent estimates. In order to utilize negative information, angular and radial constraints are formulated as pseudo measurements. These pseudo measurements are used in additional measurement update steps. Significant improvements in extent estimation are achieved with incorporated negative information.

In the thesis, extended target tracking problem is defined in detail firstly. Then, necessary literature for deriving ETTGP algorithms are given such as Gaussian processes, negative information fusion concept, and different inference techniques. ETTGP algorithms for 2D and 3D are derived step by step in the subsequent chapter. Extent, state space, and inference models for both algorithms are discussed in detail. In Ch. 5, ETTGP scenario generation framework is introduced. Results for a variety of simulations including both simulated and real data are given and elaborated.

Simulations are carried out on a modular environment called ETTGP scenario generation framework. The framework enabled us to conduct a variety of simulations easily. By utilizing such a framework, we compared different types of extended target tracking algorithms in a controlled environment. Figures and performance metrics were automatically generated and stored in .tex format.

A variety of simulations are conducted with different target shapes, motion models,

and sensor models to compare ETTGP 2D algorithms' performances. For uniform sensor scenarios, UKF Ng. performs worse compared to EKF and UKF. In uniform sensor case, sensor to target geometry leads to wrong angular and radial constraints. Since, measurements are obtained throughout the extent, there cannot be negative information. Hence, results for uniform sensor case is not meaningful for UKF Ng. algorithm. However, results are shown that negative information fusion model, UKF Ng., outperforms both EKF and UKF in LIDAR scenarios. Extent estimates are much more accurate for UKF Ng. algorithm compared to other algorithms especially in the sides that are not any measurements generated. In general, smoothers outperformed filters. This is expected since smoothers utilized all the available information in the scenario unlike filters.

Both simulated data and real data experiments are conducted to compare different ETTGP 3D algorithms' performances. The method proposed in the thesis, i.e., UKF Ng. ETTGP 3D algorithm, outperformed EKF and UKF ETTGP algorithms in both real and simulated data experiments. Same set of parameters are used for both simulated and real data experiments to demonstrate the robustness of the proposed algorithm. IoU estimates of UKF Ng. ETTGP were substantially better than existing algorithms.

Individual effects of angular and radial constraints are examined for both ETTGP 2D and ETTGP 3D algorithms. Radial constraints does not improve IoU estimation alone. However, angular constraints significantly improves the IoU estimations alone. Proposed UKF Ng. algorithm with both angular and radial constraints outperforms both UKF Ng. Ang. and UKF Ng. Rad. Nonetheless, IoU results of UKF Ng. and UKF Ng. Ang. are quite similar. Thus, one may omit radial constraints in order to save computational resources in trade off slight performance increase.

As a future work, more simulations for different types of sensors, targets and motion models can be done to demonstrate the effectiveness of the proposed model. Moreover, extension of the proposed measurement update to multi target tracking applications can be studied. With this study, multi sensor multi target scenarios can be investigated to research the performance of the new model. Another future work can be using the proposed measurement update in different ETTGP 3D algorithms.

REFERENCES

[1] K. Granström, C. Lundquist, and U. Orguner, "Tracking rectangular and elliptical extended targets using laser measurements," *Fusion 2011 - 14th International Conference on Information Fusion*, pp. 1–8, 2011.

[2] K. Granström and C. Lundquist, "On the use of multiple measurement models for extended target tracking," *Proceedings of the 16th International Conference on Information Fusion*, pp. 1534–1541, 2013.

[3] M. Baum, V. Klumpp, and U. D. Hanebeck, "A novel Bayesian method for fitting a circle to noisy points," in *13th Conference on Information Fusion, Fusion 2010*, IEEE Computer Society, 2010.

[4] W. Koch, "Bayesian approach to extended object and cluster tracking using random matrices," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 3, pp. 1042 – 1059, 2008.

[5] M. Baum and U. D. Hanebeck, "Random hypersurface models for extended object tracking," *IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2009*, pp. 178–183, 2009.

[6] N. Wahlstrom and E. Ozkan, "Extended Target Tracking Using Gaussian Processes," *IEEE Transactions on Signal Processing*, vol. 63, no. 16, pp. 4165–4178, 2015.

[7] K. Granström, M. Baum, and S. Reuter, "Extended Object Tracking: Introduction, Overview and Applications," Tech. Rep. 2, 2017.

[8] C. E. R. Williams and C. K., *Gaussian Processes for Machine Learning*. MA, USA: MIT Press, 2006.

[9] D. Zhang and G. Lu, "Study and evaluation of different Fourier methods for image retrieval," *Image and Vision Computing*, vol. 23, no. 1, pp. 33–49, 2005.

[10] M. F. Huber, "Recursive Gaussian process regression," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 3362–3366, 2013.

[11] M. F. Huber, "Recursive Gaussian process: On-line regression and learning," *Pattern Recognition Letters*, vol. 45, no. 1, pp. 85–91, 2014.

[12] W. Koch, "On 'negative' information in tracking and sensor data fusion: Discussion of selected examples," *Proceedings of the Seventh International Conference on Information Fusion, FUSION 2004*, vol. 1, pp. 91–98, 2004.

[13] W. Koch, "On exploiting 'negative' sensor evidence for target tracking and sensor data fusion," *Information Fusion*, vol. 8, no. 1 SPEC. ISS., pp. 28–39, 2007.

[14] J. Hoffmann, M. Spranger, D. Göhring, and M. Jüngel, "Making use of what you don't see: Negative information in Markov localization," *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pp. 2947–2952, 2005.

[15] A. Zea, F. Faion, J. Steinbring, and U. D. Hanebeck, "Exploiting negative measurements for tracking star-convex extended objects," *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, vol. 0, no. Mfi, pp. 622–628, 2016.

[16] S. Lee and J. McBride, "Extended Object Tracking via Positive and Negative Information Fusion," *IEEE Transactions on Signal Processing*, vol. 67, no. 7, pp. 1812–1823, 2019.

[17] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering, Transactions of the ASME*, vol. 82, no. 1, pp. 35–45, 1960.

[18] Y. Bar-Shalom, X.-R. Li, T. Kirubarajan, and A, *Estimation with Applications To Tracking and Navigation*. John Wiley & Sons, 2001.

[19] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA Journal*, vol. 3, no. 8, pp. 1445–1450, 1965.

[20] J. Hartikainen and S. Särkkä, "Optimal filtering with Kalman filters and smoothers – a Manual for Matlab toolbox EKF / UKF," *Journal of Interprofessional Care*, vol. 25, no. Imm, pp. 1–57, 2007.

[21] H. Cox, "On the estimation of state variables and parameters for noisy dynamic systems," *IEEE Transactions on Automatic Control*, vol. 9, no. 1, pp. 5–12, 1964.

[22] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "New approach for filtering nonlinear systems," *Proceedings of the American Control Conference*, vol. 3, no. June, pp. 1628–1632, 1995.

[23] J.K.Uhlmann, *Simultaneous map building and localization for realtime applications*. PhD thesis, Oxford, Oxford, U.K., 1994.

[24] S. Särkkä, "Unscented Rauch-Tung-Striebel smoother," *IEEE Transactions on Automatic Control*, vol. 53, pp. 845–849, apr 2008.

[25] M. Kumru and E. Özkan, "Three-Dimensional Extended Object Tracking and Shape Learning Using Gaussian Processes," pp. 1–13, 2019.

[26] E. Özkan, V. Šmídl, S. Saha, C. Lundquist, and F. Gustafsson, "Marginalized adaptive particle filtering for nonlinear models with unknown time-varying noise parameters," *Automatica*, vol. 49, no. 6, pp. 1566–1575, 2013.

[27] J. L. Crassidis, F. Landis Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[28] F. L. Markley, "Attitude error representations for Kalman filtering," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, 2003.

[29] U. Maeder and M. Morari, "Attitude estimation for vehicles with partial inertial measurement," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 4, pp. 1496–1504, 2011.

[30] F. L. Markley, "Attitude estimation or quaternion estimation?," *The Journal of the Astronautical Sciences*, vol. 52, no. 1-2, p. 221–238, 2004.

[31] R. Gill, M. W. Mueller, and R. D'Andrea, "Full-order solution to the attitude reset problem for kalman filtering of attitudes," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 7, p. 1232–1246, 2020.

[32] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.

[33] E. Geerardyn, "matlab2tikz package." `https://github.com/matlab2tikz/matlab2tikz`, 2008. Accessed: 2020-07-26.

[34] E. Duenisch, "latextable package." `https://github.com/eliduenisch/latexTable`, 2016. Accessed: 2020-07-19.