Research on Middle-term Vehicular Motion Prediction and its Applications Based on Dynamic Map

Lu TAO

ABSTRACT

Currently, with the development of sensor and communication technologies, autonomous driving becomes one of the hottest research areas. Vehicles can derive information from not only onboard sensors but also surrounding vehicles and roadside units. Sharing these traffic data among traffic participants can pave the way to achieve a safe and green traffic society. Dynamic Map (DM) is proposed in such a condition. As a city-level traffic data platform, DM manages road maps, static information, dynamic information and prediction information of whole city in real time. The prediction information, including short-term, middle-term and long-term prediction information, is generated by prediction algorithms. This dissertation is devoted to studying the middle-term vehicle motion prediction algorithm, whose prediction range covers not only onboard sensors' detection areas but also the neighboring areas that are out of sensor range. To achieve the goal, three novel studies are carried out in this dissertation.

Uncertainty is the knottiest problem in vehicle motion prediction. Kalman Filter (KF) is the most popular method in coping with uncertainty. Therefore, KFs and vehicular kinematic motion models are regarded as basic tools in this dissertation. Their properties should be investigated at first. However, in vehicle state estimation and motion prediction area, the comparative performance assessment, regarding accuracy and efficiency, of the most popular Unscented Kalman Filter (UKF) and Extended Kalman Filter (EKF) is rarely discussed. Hence, this dissertation is firstly devoted to empirically evaluating the performance of UKF and EKF incorporating different motion models and investigating the models properties and the affecting factors in motion prediction. Real world experiments have been carried out and the results show that EKF and UKF have roughly identical accuracy in state estimation; however, EKF is faster than UKF generally; the fastest filter is about 2.6 times faster than the slowest. The motion prediction experiments reveal that the velocity estimate and the used motion model affect

motion prediction; the more realistically the model reflects the vehicles driving status, the more reliable its predictions.

Then, this dissertation tries to quantify the uncertainty in vehicle motion prediction and applies it in an advanced collision detection application. The internet of things plays an indispensable role in the development of connected vehicles, which will pave the way for road safety applications. In recent years, the concept of a Cooperative Collision Warning System (CCWS) has been introduced and developed to enhance road safety, and it has been seen as a typical internet of vehicles application. In most CCWSs, it is vital to have a detection mechanism based on trajectory predictions where the uncertainties associated with vehicular state and motion are complex. However, most available approaches in this regard did not consider these uncertainties. This dissertation proposes a new Collision Risk Assessment (CRA) method where sigma trajectories that include multiple possible trajectories considering multiple aspects of vehicular motion are designed to cope with vehicular uncertainties. The method is implemented in a novel server-based architecture based on DM, and it is different from the commonly used vehicle-based controlled CCWSs. The CRA is provided as a service by a cloud server. The proposed method and architecture are validated and evaluated through real-world experiments. Experimental results show that our method outperforms a referenced method in terms of CRA and achieves better robustness in tolerating communication delays and dropouts. Latencies in CRA service were analyzed, and it was found that powerful computing resources provided by cloud servers can significantly decrease computational cost, which will indirectly compensate for communication costs in the future. Based on our high-performance CRA method, the proposed architecture can be regarded as a novel option for CCWS design.

Finally, this dissertation attempts to reduce the uncertainty in vehicle motion prediction through spatial kinematic trajectory data. Due to the uncertainties of vehicle dynamics and complexities of surroundings, middle-term motion prediction is never trivial work. As they combine effects of humans, vehicles and

environments, kinematic trajectory data reflect several aspects of vehicles' spatial behaviors. This dissertation proposes a novel method that leverages spatial database and kinematic trajectory data to achieve middle-term vehicular motion prediction. In this dissertation, a spatial database system is initially embedded in an EKF framework. The spatial kinematic trajectory data are managed through the database and directly used in motion prediction; namely, weighted means are derived from the spatially retrieved kinematic data and used to update EKF predictions. The proposed method is validated in the real world. The experiments indicate that different weighting methods make a slight accuracy difference. The proposed method is not data-and-computation-consumed; its performance is acceptable in the limited data conditions and its prediction accuracy is improved as the size of used data sets increases; the method can predict in real time. The efficiency of an UKF is compared with that of the EKF. The results show that the UKF can hardly meet real-time requirements.

This dissertation details the three studies, including methodologies and their experimental validations and evaluations. Through these studies, middle-term vehicular motion prediction is achieved and successfully applied in some advanced applications via DM platform.

ACKNOWLEDGEMENTS

Completing this dissertation in my second life has been the happiest time I have ever experienced. Life itself is an incredible gift that makes everything possible, and I dedicate this dissertation to my second life at its beginning, hoping to always cherish the happiness it brings me.

I owe an immeasurable debt of gratitude to my parents, who gave me birth twice and have been my unwavering supporters throughout my life. Words cannot express how deeply grateful I am to them for their love and sacrifice.

I would also like to extend my sincere thanks to my advisor, Dr. Hiroaki Takada, for allowing me to pursue a Ph.D. degree and for his continuous support and guidance. My heartfelt appreciation also goes to Dr. Yousuke Watanabe, who provided me with meticulous concerns and kind suggestions during my academic journey, and to Dr. Gang Zeng, who has been a great help to me in both my academic and personal life. I am grateful to the members of TAKADA Lab for the wonderful experience we had together.

I would like to express my sincere gratitude to the people who helped me through the toughest days of my therapy sessions: Dr. Wei Han, my attending doctor, whose kindness saved my life; Dr. Dunyao Zhu and Dr. Teng Fei, my previous advisors, who generously assisted me; my brothers and best friends, Dawei Gui, Yan Wang, Bowen Li, Haijun Li, Tianze Li, Yanjun Qiao, Ye Tian, Yuntao Wang, Qin Zhang, Jia Zheng, Hao Zhang, Lidong Yang, and Pan Zhang, who supported me every step of the way; and my dearest friend, Yuan Zhou, who always cares for me. I am grateful to those who selflessly helped me, and I regret not being able to mention everyone here due to space limitations.

Thanks to all the people who have been a shining sun in my world.

To life.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF ABBREVIATIONS

ACDT Advance Collision Detection Time

AD Autonomous Driving

ADAS Advanced Driver Assistance System

AED Average Euclidean Distance

AEE Average Euclidean Error

AI Artificial Intelligence

ASRA Adaptive Spatial Retrieve Algorithm

AW Average Weighting

CA Constant Acceleration

CCWS Cooperative Collision Warning System

CDF Cumulative Distribution Function

CITS Cooperative Intelligent Transportation System

CP Conflict Point

CRA Collision Risk Assessment

CTRA Constant Turn Rate and Acceleration

CTRV Constant Turn Rate and Velocity

CV Constant Velocity

CWS Collision Warning System

DCC Data-and-Computation-Consumed

DM Dynamic Map

DM2.0PF Dynamic Map 2.0 platform

DSRC Dedicated Short Range Communication

EKF Extended Kalman Filter

GNSS Global Navigation Satellite System

GPS Global Positioning System

HD High-Definition

IDW Inverse Distance Weighting

IMU Inertial Measurement Unit

IoV Internet of Vehicles

KF Kalman Filter

LiDAR Light Detection and Ranging

LoS Line-of-Sight

LS Least Squares

LSTM Long Short Term Memory

LTE Long Term Evolution

NLoS Non-Line-of-Sight

PDF Probability Density Function

RAEE Relative Average Euclidean Error

RLS Recursive Least Squares

RMSE Root Mean Square Error

ROS Robot Operating System

RRMSE Relative Root Mean Square Error

SCWS Standalone Collision Warning System

SRUKF Square-Root Unscented Kalman Filter

SV Subject Vehicle

TTC Time to Collision

UKF Unscented Kalman Filter

V2V Vehicle-to-Vehicle

VM Virtual Machine

VPN Virtual Private Network

WLS Weighted Least Squares

CHAPTER 1

INTRODUCTION

1.1 Background

Advanced Driver Assistance System (ADAS) and Autonomous Driving (AD) technologies have been research hotspots for several years. ADAS uses various sensor technologies to provide information, warnings and assistance to the driver to improve his/her ability to react to dangers on the road through a human-machine interface. Further, AD allows vehicles to drive safely without any human intervention based on underlying perception, planning, decision and control systems. There is a common and foundational purpose among the ADAS and AD studies: safe driving.

In order to achieve the goal of building a safe traffic society, various methodologies, technologies and applications are proposed and implemented by researchers and engineers, for decades. The vehicle's current and future states are the cornerstone for safe driving because they enable the vehicle to understand current and upcoming situations. To obtain reliable current and future vehicle state estimates/predictions, two technologies are involved at least: vehicle state estimation and motion prediction. And to build the safe traffic society, more and more advanced technologies and applications are under research and development, such as Internet of Vehicles (IoV) and Cooperative Collision Warning System (CCWS).

1.1.1 Vehicle State Estimation

In order to obtain current vehicle state, such as position, velocity and acceleration, multiple sensors are mounted on the vehicle, for example Global Positioning System (GPS), Inertial Measurement Unit (IMU) and Light Detection and Rang-

ing (LiDAR). Unfortunately, these sensors inherently suffer from noise, meaning that the raw sensor data cannot be used directly, in general. The method introduced by Kalman (1960) is the method most commonly used for addressing such noise. However, the standard Kalman Filter (KF) is designed for linear systems. When the system is nonlinear, KF is extended by linearising the nonlinear models by Gelb et al. (1974) and this is the so-called Extended Kalman Filter (EKF). In EKF, the nonlinearities are approximated analytically through first-order Taylor expansion. For decades, EKF has been the dominant state estimator for nonlinear problems. However, it has two flaws: (1) the linearisation may lead to poor performance and divergence when the nonlinearities are considerable, because the higher order terms of Taylor expansion are ignored; (2) the derivation of the Jacobian matrices may be nontrivial in some applications. In contrast with EKF, the Unscented Kalman Filter (UKF) (Julier et al., 1995) is founded on the intuition that it is easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function or transformation (Julier & Uhlmann, 1997). In UKF, a minimal set of deterministically sampled sigma points is utilized to capture completely the true mean and covariance of the Gaussian random variables and when propagated through the true nonlinearity, captures the posterior mean and covariance, accurate to the third order (Taylor expansion) for any nonlinearity; in contrast, EKF only achieves first-order accuracy (Wan & Van Der Merwe, 2000). There are ample studies that compare the performance of EKF and UKF; and the properties of some kinematic models are also investigated, such as Constant Velocity (CV) model, Constant Acceleration (CA) model, Constant Turn Rate and Velocity (CTRV) model and Constant Turn Rate and Acceleration (CTRA) model.

However, the available studies compared either EKF/UKF or motion models separately, focusing on the one-sided aspect of accuracy. In their conclusions, only an appropriate KF form or motion model was recommended. In practice, the best filter is a tradeoff between accuracy and efficiency.

In this dissertation, the mathematical principles and algorithms of KF, EKF, UKF and Square-Root Unscented Kalman Filter (SRUKF) are elaborated. The mathematical derivations of CV, CA, CTRV and CTRA models are also given. These mathematical expressions could help the reader make a deep understanding on vehicle state estimation and motion prediction. Following that, the properties, including accuracy and efficiency, of the KFs and the kinematic motion models are evaluated (Tao, Watanabe, Yamada, & Takada, 2021).

1.1.2 Uncertainty in Vehicle Motion Prediction and its Application in CCWS

Collision avoidance is a critical function for either ADAS or AD. Collision warning systems can shorten drivers' reaction time, reducing automobile accident rates (Chang et al., 2009). For Collision Warning System (CWS), there are two ways in deriving surrounding knowledge: (1) Standalone Collision Warning System (SCWS) uses onboard ranging sensors to detect nearby vehicles; (2) Cooperative CWS utilizes wireless communication to exchange information with neighboring vehicles. Standalone CWS entirely depend on ranging sensors, which are limited by Non-Line-of-Sight (NLoS) problems, sensors' properties, and environment visibility. Recent developments in communication and sensor technologies have precipitated the evolution of SCWS into CCWS, thus overcoming the above-stated limitations. According to Sengupta et al. (2007), a CCWS is a form of inter-vehicle safety cooperation through a data communication system.

Connected vehicles refer to the wireless connectivity-enabled vehicles that can communicate with their internal and external environments. These interactions enhance the situation awareness of vehicles to reduce uncertainties and provide people with a rich information travel environment. Connected vehicles are the building blocks of emerging IoV and will pave the way for road safety applications (N. Lu et al., 2014) and CCWS has been regarded as a typical IoV application (Zhuang et al., 2019). However, some common fundamental engineering

limitations hamper CCWS development (Tan & Huang, 2006): (1) any object, including vehicles, pedestrians, etc., that cannot communicate effectively creates a black hole in the system, jeopardizing the system safety; (2) the reliability of CCWSs mainly depends on the capability and performance of the communication system. To overcome these limitations, synergy among industry, academia, and governments is needed.

Intuitively, collision is a clash between vehicles' trajectories. Therefore, most CCWSs relied on trajectory prediction to assess crash hazards. To predict vehicle trajectory, knowledge about vehicles' current state and vehicular kinematics are needed. However, in reality, this knowledge cannot be fully realized due to the two major inherent uncertainties: state and motion uncertainties (Tao, Watanabe, Li, et al., 2021). Ignoring these uncertainties will reduce vehicle trajectory prediction's reliability, and thus reduces the efficiency of Collision Risk Assessment (CRA). In literature, some studies on CCWS did not consider these uncertainties. For example, in the studies of Miller & Huang (2002); X. Xu et al. (2018); Tu & Huang (2010), trajectory predictions were performed without considering the two uncertainties, reducing the reliability of their systems. Some CCWS only considered one kind of uncertainty, such as the study of Tan & Huang (2006), the state uncertainty estimation was performed using an open-loop KF. To date, only a few related studies have sufficiently considered the two uncertainties.

The vehicle-based control architecture is predominantly used in CCWS and this architecture's feasibility has been validated repeatedly. However, there are some limitations in the Vehicle-to-Vehicle (V2V) communication-based framework: (1) messages from vehicles are sent in a broadcast channel and all neighboring vehicles have to process received messages to avoid collisions. This imposes heavy workloads on vehicles, especially in a high-traffic environment; (2) Line-of-Sight (LoS) path of V2V communication is often blocked by buildings at road intersections (N. Lu et al., 2014), which is a traffic accident hotspot, and it has been reported that the NLoS problems may severely degrade V2V communication

performance (Lyu et al., 2019); 3) data contents and computing resources in local vehicles are limited. Recently, a CCWS based on so-called fog nodes, such as Wi-Fi access points and parked vehicles, has been proposed by X. Xu et al. (2018); whereas, computing resources and data contents in the fog nodes are limited. Unfortunately, a commonly used approach to mitigate the effects of these limitations, namely a server-based architecture, has rarely been discussed so far.

In order to fix the flaws of present studies, this dissertation aims to leverage the two uncertainties in predicting vehicles' trajectories and achieving a better CRA and attempts to provide the CRA service for connected vehicles using a cloud-server-based architecture (Tao, Watanabe, Li, et al., 2021).

1.1.3 Reducing the Uncertainty in Vehicle Motion Prediction

Plentiful methods have been proposed to explain vehicle motion evolution in different time range (Lefèvre et al., 2014) to obtain future vehicle motion states, such as position and velocity in the next 3 seconds. However, vehicle motion prediction is never trivial work due to uncertainties concerning vehicular dynamics and complexities of surroundings. In general, there are five main genres in vehicle motion prediction studies: (1) physical model-based methods that use explicit mathematical expressions to describe vehicle motion evolution, such as the methods proposed by Tan & Huang (2006) and Lytrivis et al. (2011); (2) trajectory-matching-based methods that map vehicles' trajectories into typical motion patterns to achieve motion prediction, such as the method proposed by Hermes et al. (2009); (3) machine-learning-based methods that learn prediction models from historical data, such as the methods proposed by Jeong et al. (2017); Altché & de La Fortelle (2017); Jiang et al. (2022); Lin et al. (2021); (4) map-aided methods that leverage map data, particularly geometries of High-Definition (HD) maps, to realize vehicle motion prediction, such as the method proposed by Petrich et al. (2013); (5) hybrid methods that make use of at least two of the above methods, such as the method proposed by Yalamanchi et al. (2020).

These methods try to cope with the uncertainties and complexities in vehicle motion prediction from different perspectives and each kind of method has pros and cons and some challenges remain. The physical model-based methods are straightforward and efficient; however, a vehicle is governed by not only physical laws but also a human being and traffic environments. Our previous work showed that a single physical model was not able to make a reliable long-term prediction (Tao, Watanabe, Yamada, & Takada, 2021). When the physical models are used to predict vehicle motion in some safety-related applications, the associated uncertainties should be considered meticulously (Tao, Watanabe, Li, et al., 2021). Trajectory-matching- and machine-learning-based methods receive reasonable predictions due to utilizing huge volumes of pre-prepared historical data. They therefore are computationally expensive and data-consumed; the prediction accuracy is heavily dependent on the richness of collected historical data. Map-aided methods take into account both functionality and efficiency. With mass productions of HD maps, there have been some typical map-aided methods proposed in recent years, such as the work of Petrich et al. (2013); Yalamanchi et al. (2020); Kawasaki & Tasaki (2018). However, these lack reasonable bases to fuse static map data that are defined by map makers with dynamic vehicle motion. A forced combination of them will make the prediction converge to static map data/attributes and the dynamics of the vehicle will surely be lost, for example, their trajectory prediction will converge to the lane center line and the predicted velocity will converge to the fixed velocity attributes in maps.

A particular trajectory can be regarded as an outcome of interactions between a particular vehicle and driver under particular environments. As records of dynamic vehicle motions, vehicles' spatial kinematic trajectory data, such as position, velocity, yaw, yaw rate and acceleration, in fact reflect vehicles' spatial behaviors in several aspects; for example, a position near the right side of a lane indicates that the vehicle would turn right; velocities and accelerations along different road segments were reflections of driving styles in different spaces. In vehicle

motion predictions, which full of uncertainties, this spatial information is crucial for corrections of mathematical models' predictions.

In order to overcome the disadvantages of the mentioned genres in motion prediction, this dissertation, in a novel contribution, leverages spatial kinematic trajectory data that are retrieved through spatial relations to predict vehicle motions in KF and spatial database frameworks (Tao et al., 2022).

1.2 Dynamic Map and This Study

Dynamic Map (DM) is a logical data set that enables sensor data to be overlaid onto a HD map through location reference method. DM is seen as the next-generation road map (Watanabe et al., 2020). Since 2016, DM has been studied and developed by Dynamic Map 2.0 consortium, which consists of several universities and companies in Japan (NCES, 2019).

From the standpoint of data, DM is a traffic information platform. There are four types information in DM: road maps, static information, dynamic information and predicted information as demonstrated in Figure 1.1. The road maps are essential data in DM; they play the role of sensor data interpreter, meaning that the road maps provide semantic explanations for the sensor data through semantic maps and spatial reference system. Static information includes the fixed attributes of the objects on the maps. Dynamic information are streaming sensor data that are generated by onboard and roadside sensors. Prediction data are produced by the prediction algorithm, including short-term, middle-term and long-term prediction data. This study is devoted to the algorithm development of middle-term prediction. The position that this study locates in DM study is indicated by the red arrow in Figure 1.1. To be noticed, in DM, there are two kinds of prediction algorithm, corresponding to different prediction data. (1) Dynamic-data-based prediction algorithm that uses dynamic sensor data to predict vehicle/pedestrians motion based on physical law and traffic rules; this kind of algorithm outputs short-term and middle-term prediction data, which have high demands on both

accuracy and efficiency. (2) Statistics-based prediction algorithm that uses the big data that are collected in a long term to predict, for example the traveling time, traffic jam. This kind of algorithm produces long-term prediction data that have lower accuracy and efficiency requests.

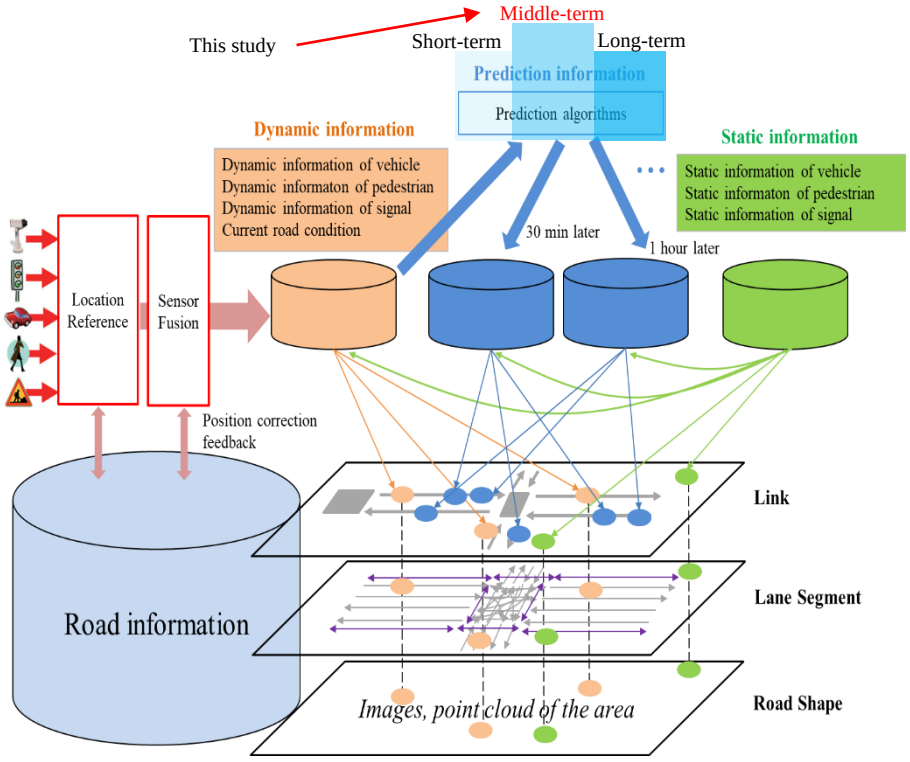


Figure 1.1: Dynamic map and this study.

From the standpoint of functionality, DM is a traffic application platform. It provides: (1) A distributed architecture for real-time processing for large volume of data. DM comprises embedded devices, edge servers and cloud servers with the architecture shown in Figure 1.2. Through the cloud/edge/embedded systems and their collaboration, DM is able to process a huge volume of traffic data, as well as meet real-time demands. (2) A common data model for heterogeneous traffic data; in DM, all types of data are treated as tables using a common relational model. (3) High precision road maps, including link-level, lane-level and physical-shape-level map data, for traffic data integration. (4) Common access method for heterogeneous traffic data.

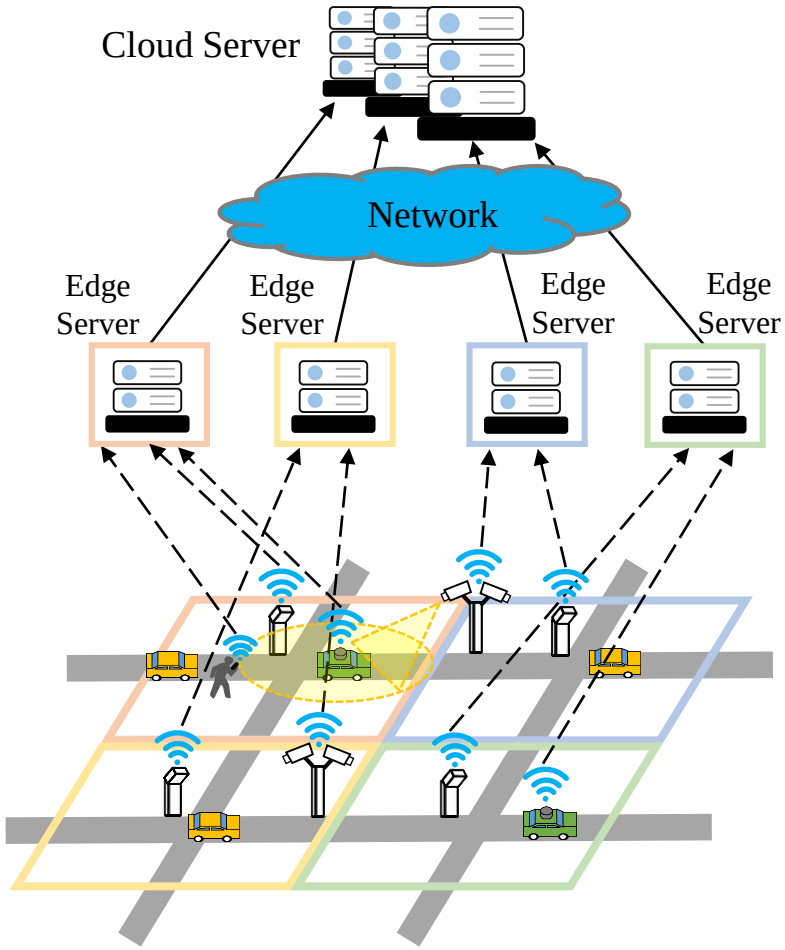


Figure 1.2: The geographically distributed architecture of DM.

1.3 Problem Definitions

As aforementioned, there are three kinds of predictions in DM: short-term prediction, middle-term prediction and long-term prediction. This dissertation is devoted to the middle-term vehicular motion prediction.

In available studies, vehicle motion predictions are roughly divided into two groups: short-term prediction (less than one second) and long-term prediction (more than one second) (Lefèvre et al., 2014). This classification cannot meet the actual requests in DM application development. Therefore, we classify vehicular motion prediction into the three categories in this study: short-term, middle-term and long-term prediction, as Figure 1.3 shows:

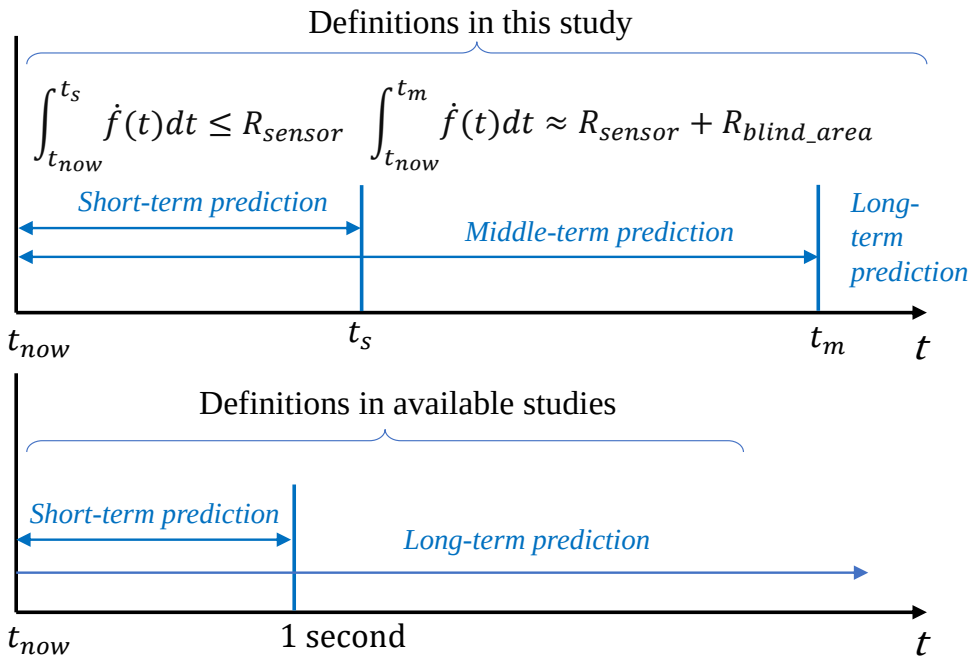


Figure 1.3: Problem definitions. The definition of short-, middle- and long-term predictions in this study. Different with the available studies (Lefèvre et al., 2014) that briefly define vehicle motion prediction as short- and long-term prediction (the bottom), there are three types of motion predictions in this study (the top).

More precisely, a short-term vehicle motion prediction is the process to predict vehicular future motions whose ranges are not more than on-board sensors' detection range:

$$\text{short-term prediction: } \int_{t_{now}}^{t_s} \dot{f}(t)dt \leq R_{sensor} \tag{1.3-1}$$

where t_{now} indicated current time and t_s is time range of short-term prediction; $f(t)$ denotes a vehicular motion model; R_{sensor} indicates the on-board sensers' detection range.

A middle-term vehicle motion prediction is the process to predict vehicular future motions whose ranges cover not only the range of on-board sensors, but also the blind area of the sensors (it is called unknown area in DM):

$$\text{middle-term prediction: } \int_{t_{now}}^{t_m} \dot{f}(t)dt \approx R_{sensor} + R_{blind\_area} \tag{1.3-2}$$

where t_m is time range of middle-term prediction; generally, $3s \leq t_m \lesssim 10s$. R_{blind_area} is the range of blind area. From this definition, it is obvious that the aim of middle-term prediction is to enhance the awareness ability of vehicles. To be pointed out, both the time range t_s and t_m are determined based on particular situations.

A long-term vehicle motion prediction is the process to predict vehicular motions whose ranges are much farther than middle-term prediction. If we regard the short- and middle-term predictions as local; then, the long-term prediction is global. For example, arriving time prediction between starting and destination points.

1.4 Contributions

Herein, the contributions of this dissertation, which is briefly mentioned in Section 1.1, are detailed.

As the most important theoretical basis of this dissertation, chapter 3 elaborates the mathematical derivations of involved vehicular kinematic models and Kalman filters. These can help the reader establish a deep understanding for the mathematical foundations of this dissertation.

In chapter 4, the properties of KFs and kinematic models are investigated through comparative experiments. The contributions are as following:

- In vehicle state estimation, the accuracy and efficiency of both EKF and UKF, incorporating CTRV and CTRA models, are evaluated. The analyses can guide the reader to make an appropriate filter/estimator considering their specific demands.

- In vehicle motion prediction, the affecting factors and the models' properties, towards accuracy performance, are investigated.

Chapter 5 is devoted to solving the knotty uncertainties from theoretical and technical ways. The contributions are as following:

- A novel CRA method that leverages vehicular state and motion uncertainties was proposed. For trajectory prediction, the method considers both the state and motion uncertainties, which involve multiple aspects, including position, velocity, acceleration, heading, and yaw rate. To make use of the uncertainties in vehicle motion prediction, so-called sigma trajectory was proposed.

- Server-based architecture was utilized and implemented based on DM. This is different from the conventional vehicle-based control architecture mostly adopted by CCWS. This dissertation builds a novel cloud-server-based architecture for CCWS where CRA is provided as a service.

- Both the proposed method and architecture were validated and evaluated in the real world. A total of 54 experiments have been carried out. Compared with the simulation-based studies, in the experiments, only the collision had not occurred in the physical world and all the others were real. The experiments, therefore, can reveal certain facts hidden in simulation-based experiments.

To reduce the uncertainties in motion prediction, historical trajectory data are generally used. In chapter 6, a novel vehicle motion prediction method that leverages spatial database and kinematic trajectory data was proposed. The contributions are as following:

- A novel vehicle motion prediction method based on spatial database and kinematic trajectory data is proposed. Different from existing historical-data-based methods that learn knowledge from huge volumes of data, the method retrieves relevant information based on spatial relations through a well-organized spatial database. In addition, the neglected personal factors in the present methods, such as driver and vehicle information, are taken into account.

- A spatial database system is initially embedded in a classical KF framework. This combination makes our system lightweight and the utilization of a spatial search makes our algorithm able to find the most spatially related data quickly.

- Both accuracy and efficiency of algorithms are discussed.

1.5 Dissertation Outline

The remainder of this dissertation is organized as follows.

The related work of this study is reviewed in chapter 2, including three involved research topics: (1) vehicle state estimation, (2) cooperative collision warning system and (3) long-term vehicle motion prediction.

Chapter 3 is the foundation of this dissertation. This chapter elaborates the most powerful technology for extracting reliable information from uncertainty, namely Kalman filter. The novelly-proposed methods in chapter 5 and chapter 6 are based on this technology. In this chapter, firstly, four widely used vehicle kinematic models are introduced: CV, CA, CTRV and CTRA. Then, the most important KF is derived from least squares method for linear systems. In order to deal with nonlinear systems, the derivation of EKF is also given. To be pointed out, these mathematical processes are presented to help with understanding the basic principles of this dissertation. Besides, the algorithms of UKF and SRUKF are also given in chapter 3.

In chapter 4, the properties of the state estimators that consist of the most popular CTRV/CTRA models and EKF/UKF in vehicle state estimation application are investigated. The affecting factors and the models' properties in vehicle motion prediction are also analyzed.

Chapter 5 focuses on coping with and making use of the uncertainties in middle-term vehicle motion prediction. There are two ways in dealing with the uncertainties: (1) in a theoretical way, so-called current-state-centered multidimensional

sampling method and sigma trajectory are proposed to cope with the uncertainties in motion prediction process. (2) in a technical way, DM is utilized to obtain the information that is hidden in onboard sensors' blind spots. Using the sigma trajectory, a new collision risk assessment method is proposed. The risk indicators like collision probability, Time to Collision (TTC) and Conflict Point (CP) are used to quantify collision risk. It is remarkable that the proposed method is implemented in a novel server-based architecture based on DM, which is totally different with predominant vehicle-based architecture. The collision risk assessment is provided as a service in IoV environment. In the experiments, both the proposed method and architecture are validated and evaluated in real world.

Chapter 6 is devoted to reducing the uncertainties in vehicle motion prediction based on spatial kinematic trajectory data. As they combine effects of humans, vehicles and environments, kinematic trajectory data reflect several aspects of vehicles' spatial behaviors. In this chapter, a novel method that leverages spatial database and kinematic trajectory data is proposed to achieve middle-term vehicular motion prediction in a lightweight way. In the proposed system, a spatial database system is initially embedded in an EKF framework. The spatial kinematic trajectory data are managed through the database and directly used in motion prediction; namely, weighted means are derived from the spatially retrieved kinematic data and used to update EKF prediction. The proposed method is validated in the real world to investigate its performance.

Finally, summary, conclusions and future work of this dissertation are presented in chapter 7.

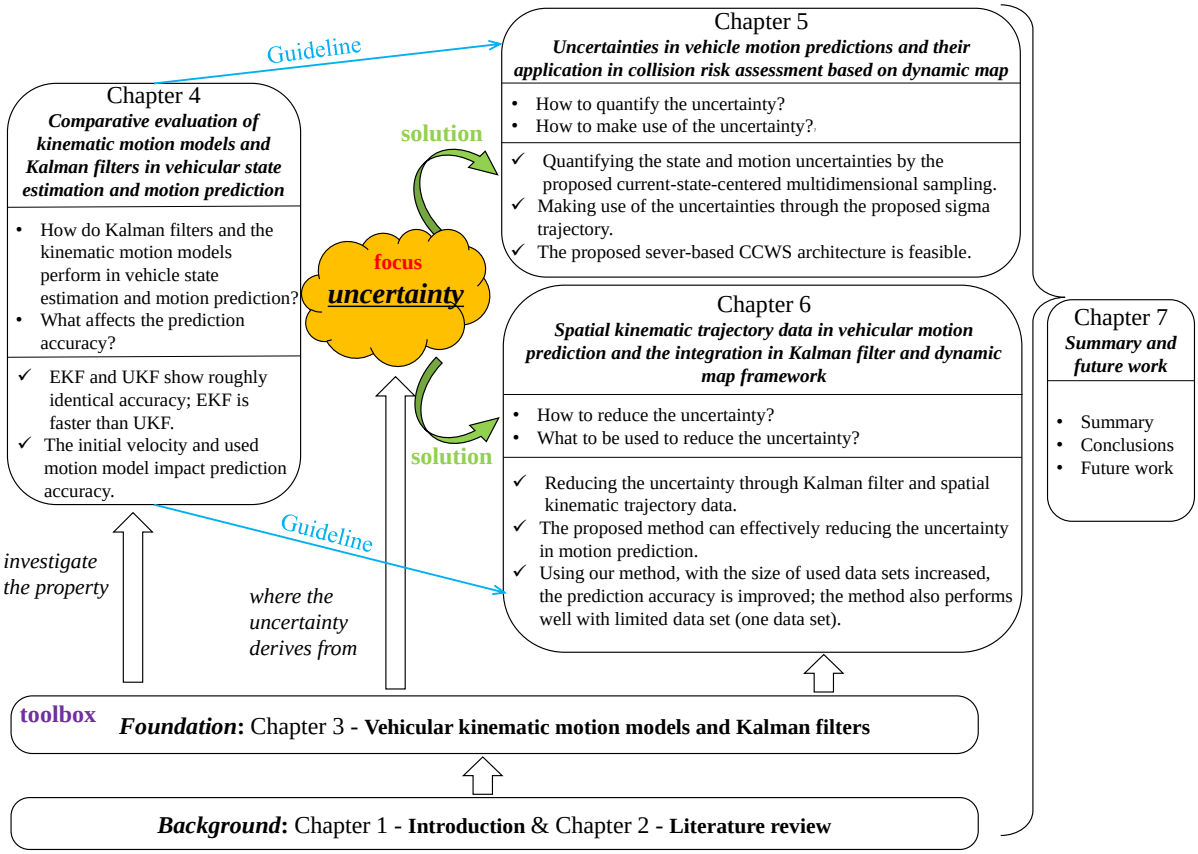


Figure 1.4: The framework of this dissertation.

Figure 1.4 demonstrates the framework of this dissertation. Briefly, chapter 1 and chapter 2 introduce the background of this study, where the related background and literature are reviewed and the problems of this study are defined. Chapter 3 is the essential foundation of this dissertation and it can be regarded as the toolbox of this study. Uncertainty, which can be traced back to chapter 3, is the focus in solving the middle-term vehicle motion prediction problems. To solve the problem, it is necessary to know the properties of the "tools" to be used. Therefore, in chapter 4, real world experiments are carried out to investigate the properties of KFs and kinematic motion models. The conclusions of chapter 4 are the guidelines of chapter 5 and chapter 6.

Chapter 5 and chapter 6 are devoted to solving the uncertainty in middle-term motion prediction. Specifically, in chapter 5, the uncertainty is quantified by a new sampling method and the novel sigma trajectory is proposed to make use of

the uncertainty; however, chapter 5 does not provide an effective theoretical way to reduce the uncertainty in prediction; the uncertainty remains. Fortunately, this question is novelly solved in chapter 6; the uncertainty, in middle-term prediction, is reduced as much as possible using spatial kinematic trajectory data based on EKF and spatial database.

Finally, this dissertation is summarized and concluded in chapter 7.

CHAPTER 2

LITERATURE REVIEW

This chapter reviews related literature, including the following research topics: vehicle state estimation, cooperative collision warning system and long-term vehicle motion prediction.

2.1 Vehicle State Estimation

The Kalman filters, including classical linear KF and nonlinear KFs, are the most popular state estimation technologies. In practice, most systems are nonlinear and in this section, three nonlinear KFs (EKF, UKF and SRUKF) are concerned. As we know, EKF approximates the nonlinear system around recent state estimates analytically through first-order Taylor expansion. This makes the information from high order terms lost. To fix the flaw, UKF that uses sigma points to capture completely true mean and covariance is proposed. However, UKF has to compute the square root of the state covariance matrix at each iteration through, like, Cholesky factorization. This requires that the covariance matrix keeps positive definiteness. However, in practice, the covariance matrix may lose positive definiteness due to some numerical problems, like precision lose in floating point computations; and that causes UKF to fail. To prevent the numerical instability of UKF, SRUKF that guarantees the state covariance positive semi-definite is proposed by Van Der Merwe & Wan (2001). In literature, EKF and UKF are widely used in state estimation and they properties are studied.

There are many studies that investigate the performance of EKF and UKF. Lefebvre et al. (2004) theoretically compared the performance in process and measurement update step of some KFs. In vehicle localization, the performances of EKF

and UKF are compared experimentally and they were found to behave similarly in terms of accuracy and consistency (Ndjeng et al., 2009). In vehicle navigation, the accuracy and computational time of EKF and UKF were evaluated by St-Pierre & Gingras (2004); it was found that UKF had just a slight improvement in accuracy whereas it cost much more computational time than EKF. Yang et al. (2017) investigated the performance of EKF and UKF with different feedback control models and showed the superiority of UKF. In vehicular state estimation, the performance of EKF and UKF is rarely discussed, because UKF is seen as a better choice and sufficient for most current applications where only several vehicles are involved.

Besides the KF forms, the process model also affects the performance of a state estimator. An elaborate process model significantly improves the filter's performance (Julier & Durrant-Whyte, 2003). In road safety related applications, kinematic models prevail (Lefèvre et al., 2014), such as CV, CA, CTRV and CTRA models. A comprehensive comparison and evaluation of these models was carried out by Schubert et al. (2008, 2011); they concluded that the sophisticated model did not demonstrate better performance in any case, and the appropriateness of model heavily depended on the application. In the literature, CTRV and CTRA models are popular choices in various applications, such as autonomous driving (Madhavan et al., 2006), collision avoidance (Polychronopoulos et al., 2007) and trajectory prediction (Lytrivis et al., 2011; Houenou et al., 2013).

A common defect of these studies is that they compare either EKF/UKF or motion models separately and just focus on the one-sided aspect of accuracy. With the development of communicating and networking technologies, Cooperative Intelligent Transportation System (CITS) has been regarded as the next important step towards the vision of accident-free driving (Weiß, 2011). The applications involved with CITS and vehicular networking are of real-time nature (Karagiannis et al., 2011). Therefore, the efficiency of a state estimator should be seriously treated. On other words, the best filter in actual application is a trade-off between

accuracy and efficiency. However, in present literature, especially in vehicle state estimation application, the accuracy and efficiency of state estimators are poorly investigated. This work is completed in chapter 4 of this dissertation.

2.2 Cooperative Collision Warning System

Situation awareness is vital for CCWS. There are two approaches to the situation awareness algorithms' design: non-predictive and predictive. Non-predictive approaches rely on a real-time vehicular state to assess potential collision risk. The easiest way to implement a non-predictive approach is to monitor the distance between vehicles and warn drivers when vehicles are close to each other, as demonstrated in the work of Sengupta et al. (2007); Zhao et al. (2019); J. Liu et al. (2013); Gómez et al. (2016). Hafner et al. (2013) proposed a formal control method to avoid collisions at road intersections, where the collision area was known a priori. The set of all initial conditions under which no control input can prevent a collision was called a capture set. The proposed method guaranteed vehicles never enter the capture set simultaneously. Non-predictive approaches demand strict real-time monitoring of CCWS; latency degrades CCWS performance. Predictive approaches inherently tolerate delays and dropouts. Technically, common straightforward paradigms can be found adopted in such situations (Tan & Huang, 2006; Miller & Huang, 2002; Tu & Huang, 2010; Joerer et al., 2013; Lytrivis et al., 2011): (1) a state estimator fuses sensor data to generate accurate state estimate of a Subject Vehicle (SV); (2) a communicator transmits SV's information to other vehicles; (3) a predictor generates vehicle trajectories; (4) a collision assessor identifies potential risk to assist safe driving. Therein, KF (Tu & Huang, 2010), EKF (Tan & Huang, 2006) and UKF (Lytrivis et al., 2011) were the most used state estimators. Vehicular motion models were mostly employed as trajectory predictors, for instance, kinematic models (Tan & Huang, 2006; Miller & Huang, 2002; Joerer et al., 2013; Lytrivis et al., 2011). Collision risk was assessed through indicators such as intersection points (Lytrivis et al., 2011), distance (Zhao et al.,

2019; Gómez et al., 2016), TTC (Tan & Huang, 2006; Miller & Huang, 2002; Tu & Huang, 2010) , and probability (Joerer et al., 2013). Among the above paradigms, trajectory prediction is the most challenging because state and motion uncertainties are difficult to solve. Collision detection was formulated into a minimum distance problem by Tu & Huang (2010). The algorithm was founded on the hypothesis that vehicles perfectly followed predefined trajectories. The vehicles' trajectory was predicted using a kinematic model under constant acceleration and yaw rate assumption; collision risk was assessed based on the assumption that predicted trajectories were fully trusted (Tan & Huang, 2006), i.e., they are fail-proof. Both collision assessor approaches adopted by Tu & Huang (2010) and Tan & Huang (2006) did not consider motion uncertainty; thus, the reliability of their systems decreased as prediction time increased.

To calculate collision probability, "all" possible driver behaviors were modeled mathematically by Joerer et al. (2013). Vehicles' trajectories were generated by a CA model with different acceleration values within a fixed interval. To the best of our knowledge, it is one of state-of-the-art CCWS solutions among the published papers. However, this approach considered uncertainties related to acceleration and ignored other possible behaviors associated with other aspects. In this dissertation, a trajectory set that incorporates multiple aspects of vehicles' state and motion uncertainties is designed for CRA, in chapter 5.

Besides the algorithms, CCWS have two basic hardware components: sensing and communication modules. The sensing module provides state measurements of a SV through sensor suites, such as the GPS and inertial sensors (Tan & Huang, 2006; Tu & Huang, 2010). The SV state information is then transmitted to neighboring vehicles through the communication module. There are two popular communication solutions: Dedicated Short Range Communication (DSRC) and cellular network technologies (Abboud et al., 2016). The primary motivation for deploying DSRC is to enable collision prevention applications (Kenney, 2011); hence, in literature, despite DSRC's low market penetration, it is the most

widely used communication framework in vehicle-based controlled CCWSs (Sengupta et al., 2007; Tu & Huang, 2010; Hafner et al., 2013; Zhao et al., 2019). In contrast, cellular network technology is an off-the-shelf solution for vehicle-to-everything communications, providing high capacity, high mobility support, wide coverage, high penetration, and pervasive infrastructures (Araniti et al., 2013; Seo et al., 2016). There are some common communication problems such as delays and dropouts. In this dissertation, we do not attempt to solve these problems directly but make our algorithms tolerant of them.

There are some related approaches focusing on another situation of collision avoidance. Standalone collision avoidance systems in vehicles are widely studied (Lee et al., 2017; Kim & Kum, 2017; Shangguan et al., 2019; Yoon et al., 2019) . These systems highly depend on ranges of in-vehicle sensors. On the other hand, CCWS can exchange information from vehicles via network. The paper of Gao et al. (2017) proposes collision avoidance in mixture of DSRC-equipped vehicles and non-DSRC-equipped vehicles. The paper of Segata et al. (2017) estimates collision between a vehicle and a cyclist. Their research targets are different from ours.

2.3 Long-term Vehicle Motion Prediction

In literature, long-term vehicular motion prediction is an important research area among AD and ADAS. However, the uncertainty is knotty in prediction, due to it propagates over time and dramatically increases if there is no additional information to be used to correct the prediction.

In a previous study of Tao, Watanabe, Yamada, & Takada (2021), CTRV and CTRA models were directly used to predict vehicle motion in an open loop KF framework without any external information correction; the results show that a sole mathematical model was not suitable for long-term vehicle motion prediction. Although a stand-alone physical vehicle motion model cannot predict reliably, switching between different motion models in different scenes using a Dempster-

Shafer reasoning system can produce an accomplished prediction (Lytrivis et al., 2011). However, to date, few studies have only used physical models to conduct vehicle motion prediction due to some inevitable constant hypotheses of motion models being unreasonable in a long time range. A compensation is to fully consider the uncertainties in predictions, such as the methods proposed by Tao, Watanabe, Li, et al. (2021); Hafner et al. (2013); Joerer et al. (2013).

Using historical data to reduce the uncertainty in vehicle motion prediction is a popular and dominant methodology. An early study can be dated back to 2009; the authors proposed a long-term motion method that combined trajectory classification and a particle filter framework. This was a so-called trajectory-matching-based method; they used a quaternion-based rotationally invariant longest common subsequence metric to measure trajectory similarity. Speed and timing profiles were introduced as presentations of surrounding environments by Shan et al. (2011), in which a particle filter was used to incorporate information of environments and motion models. However, the algorithm only predicted one-dimensional positions along routes. Artificial Intelligence (AI) is a powerful paradigm for traffic prediction (Y. Xu et al., 2021). The Gaussian process was used to learn parameters of vehicular trajectories (Tran & Firl, 2013); moreover, to take into account vehicles' interactions, a dynamic Bayesian network was utilized (Gindele et al., 2015). In the papers of Zhang et al. (2022); Y. Lu et al. (2022); Schmidt et al. (2022), graph neural networks were employed to model complex interactions between vehicles and roadside infrastructures. Deep neural networks were also adopted to predict ego-vehicle paths using environment observation in the paper of Baumann et al. (2018). Long Short Term Memory (LSTM) networks are widely used in sequence tasks, such as traffic and trajectory predictions. Altché & de La Fortelle (2017) adopted an LSTM network to predict vehicle trajectories on highways. Spatial and temporal attention mechanisms were introduced into LSTM networks by Jiang et al. (2022); Lin et al. (2021). A common drawback of the above methods is that they have to collect large amounts of historical data

for training in advance, and their computing cost is much higher; in this study, we call that Data-and-Computation-Consumed (DCC). In addition, these methods treat trajectory data from different drivers and vehicles as the same, which is unreasonable. For example, it is not applicable to use trajectory data of the old drivers to predict a young driver's vehicle motion; it is also improper to use a sports car's velocity profile to predict the velocity of a school bus. Personal characteristics, such as the driver's and his/her vehicle's information, should be taken into account.

The commercialization of HD maps provides a new solution to reduce the prediction uncertainty and its feasibility has been validated in some studies. Petrich et al. (2013) used an EKF to update the prediction made by a kinematic bicycle model using information acquired from HD maps, such as position, heading, and velocity. In the paper, squared Mahalanobis distance was used to assign traffic lane as an access to the map data. Similarly, Kawasaki & Tasaki (2018) utilized an EKF to incorporate a uniform acceleration motion model and velocity model that was built based on HD map and observed velocities. Yalamanchi et al. (2020) proposed an uncertainty-aware stitching method that combined short-term trajectories predicted by learned models with long-term actor goals derived from associated lanes. The map-aided methods force combinations of dynamic vehicle motions with static map data, which causes vehicle dynamics to be lost in prediction.

CHAPTER 3

VEHICULAR KINEMATIC MOTION MODELS AND KALMAN FILTERS

This chapter is devoted to elaborating the kinematic models and Kalman filters that are widely used in vehicle state estimation and motion prediction. The author does not claim any novel contributions in this chapter; the only propose to write this chapter is to help the reader to understand the concepts and contents in following chapters, especially chapter 5 and chapter 6.

3.1 Vehicular Kinematic Motion Models

In this section, four vehicular kinematic models: CV, CA, CTRV and CTRA models, are introduced.

3.1.1 Constant Velocity Model

State vector of CV model

In CV model, the state of a vehicle at time k is defined as:

$$\mathbf{x}_k = \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_k \tag{3.1-1}$$

where x, y are position coordinates and v_x, v_y are the velocity components in x and y direction.

State transition function of CV model

Constant velocity model assumes that the velocity of a vehicle keeps constant. It is easy to know that the state transition function, from time k to time k+1, is as following:

$$\begin{aligned} x_{k+1} &= x_k + v_x \Delta t \\ y_{k+1} &= y_k + v_y \Delta t \\ v_{x|k+1} &= v_{x|k} \\ v_{y|k+1} &= v_{y|k} \end{aligned} \tag{3.1-2}$$

where $\Delta t = t_{k+1} - t_k$. Equation 3.1-2 can be written in following matrix formulation:

$$\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k \tag{3.1-3}$$

Equation 3.1-3 is state transition function of CV model, in matrix form. The matrix $\mathbf{F}_k$ is called state transition matrix of CV model.

3.1.2 Constant Acceleration Model

State vector of CA model

In CA model, the vehicle state at time k is defined as:

$$\mathbf{x}_k = \begin{bmatrix} x & y & v_x & v_y & a_x & a_y \end{bmatrix}_k^T \tag{3.1-4}$$

where a_x, a_y are the acceleration components in x and y direction.

State transition function of CA model

Constant acceleration model assumes that a vehicle's acceleration is constant. Therefore, the state at time k+1 can be known easily from kinematics law:

$$\begin{aligned} x_{k+1} &= x_k + v_{x|k}\Delta t + \frac{1}{2}a_{x|k}\Delta t^2 \\ y_{k+1} &= y_k + v_{y|k}\Delta t + \frac{1}{2}a_{y|k}\Delta t^2 \\ v_{x|k+1} &= v_{x|k} + a_{x|k}\Delta t \\ v_{y|k+1} &= v_{y|k} + a_{y|k}\Delta t \\ a_{x|k+1} &= a_{x|k} \\ a_{y|k+1} &= a_{y|k} \end{aligned} \tag{3.1-5}$$

Similarly, Equation 3.1-5 can also be written in matrix formulation:

$$\mathbf{x}_{k+1} = \mathbf{F}_k\mathbf{x}_k = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k \tag{3.1-6}$$

Equation 3.1-6 is state transition function of CA model, in matrix form. The matrix $\mathbf{F}_k$ is called state transition matrix of CA model.

3.1.3 Constant Turn Rate and Velocity Model

State vector of CTRV model

In CTRV model, the vehicle state at time k is defined as

$$\mathbf{x}_k = \begin{bmatrix} x & y & v & \theta & \omega \end{bmatrix}_k^T \tag{3.1-7}$$

where θ denotes heading (yaw) angle and ω denotes the turn (yaw) rate.

State transition function of CTRV model

Constant turn rate and velocity model assumes that a vehicle keeps constant turn rate and velocity, which results in a circular trajectory, see Figure 3.1 (the trajectory is simulated with initial state $[0m, 100m, 0deg, 3.14m/s, -18deg/s]^T$ and duration 200 s). The state at time $k+1$ can be derived from the following equation:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \int_{t_k}^{t_{k+1}} \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{v}(t) \\ \dot{\theta}(t) \\ \dot{\omega}(t) \end{bmatrix} dt = \mathbf{x}_k + \begin{bmatrix} \int_{t_k}^{t_{k+1}} v(t)\cos(\theta(t))dt \\ \int_{t_k}^{t_{k+1}} v(t)\sin(\theta(t))dt \\ 0 \\ \omega_k \Delta t \\ 0 \end{bmatrix} \tag{3.1-8}$$

where

$$\begin{aligned} \int_{t_k}^{t_{k+1}} v(t)\cos(\theta(t))dt &= v_k \int_{t_k}^{t_{k+1}} \cos(\theta_k + \omega_k(t-t_k))dt \\ &= v_k \int_{t_k}^{t_{k+1}} \cos(\theta_k + \omega_k(t-t_k))d(\theta_k + \omega_k(t-t_k))\frac{1}{\omega_k} \\ &= \frac{v_k}{\omega_k}\sin\left(\theta_k + \omega_k(t-t_k)\right)\Big|_{t_k}^{t_{k+1}} \\ &= \frac{v_k}{\omega_k}\left[\sin(\theta_k + \omega_k\Delta t) - \sin(\theta_k)\right] \end{aligned} \tag{3.1-9}$$

and the same for:

$$\begin{aligned} \int_{t_k}^{t_{k+1}} v(t)\sin(\theta(t))dt &= v_k \int_{t_k}^{t_{k+1}} \sin(\theta_k + \omega_k(t-t_k))dt \\ &= v_k \int_{t_k}^{t_{k+1}} \sin(\theta_k + \omega_k(t-t_k))d(\theta_k + \omega_k(t-t_k))\frac{1}{\omega_k} \\ &= -\frac{v_k}{\omega_k}\cos\left(\theta_k + \omega_k(t-t_k)\right)\Big|_{t_k}^{t_{k+1}} \\ &= \frac{v_k}{\omega_k}\left[-\cos(\theta_k + \omega_k\Delta t) + \cos(\theta_k)\right] \end{aligned} \tag{3.1-10}$$

Substituting Equations 3.1-9 and 3.1-10 into Equation 3.1-8, the state transi-

tion functions of CTRV model can be summarized as:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k) = \begin{bmatrix} x_k + \frac{v_k}{\omega_k}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\ y_k + \frac{v_k}{\omega_k}\left[-\cos(\theta_k + \omega_k \Delta t) + \cos(\theta_k)\right] \\ v_k \\ \theta_k + \omega_k \Delta t \\ \omega_k \end{bmatrix} \tag{3.1-11}$$

Jacobian of CTRV model

As we can see that Equation 3.1-11 cannot be written in a matrix form, due to CTRV model is nonlinear and it thus cannot be presented in linear form. To linearize a nonlinear model, Jacobian is usually involved.

Giving a vector function:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \mathbf{f}_1(\mathbf{x}) & \mathbf{f}_2(\mathbf{x}) & \cdots & \mathbf{f}_m(\mathbf{x}) \end{bmatrix}^T \tag{3.1-12}$$

where $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}$. The Jacobian matrix of the vector function $\mathbf{f}(\mathbf{x})$ can be calculated based on the following equation:

$$\textbf{Jacobian} = \begin{bmatrix} \frac{\partial \mathbf{f}_1}{\partial x_1} & \frac{\partial \mathbf{f}_1}{\partial x_2} & \cdots & \frac{\partial \mathbf{f}_1}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \mathbf{f}_m}{\partial x_1} & \frac{\partial \mathbf{f}_m}{\partial x_2} & \cdots & \frac{\partial \mathbf{f}_m}{\partial x_n} \end{bmatrix} \tag{3.1-13}$$

To obtain the Jacobian of CTRV model, we firstly let

$$\begin{aligned} \mathbf{f}_x(\mathbf{x}_k) &= x_k + \frac{v_k}{\omega_k}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\ \mathbf{f}_y(\mathbf{x}_k) &= y_k + \frac{v_k}{\omega_k}\left[-\cos(\theta_k + \omega_k \Delta t) + \cos(\theta_k)\right] \\ \mathbf{f}_v(\mathbf{x}_k) &= v_k \\ \mathbf{f}_\theta(\mathbf{x}_k) &= \theta_k + \omega_k \Delta t \\ \mathbf{f}_\omega(\mathbf{x}_k) &= \omega_k \end{aligned} \tag{3.1-14}$$

Based on Equation 3.1-13, the Jacobian of CTRV model is:

$$\mathbf{J}_{CTRV} = \begin{bmatrix} 1 & 0 & \frac{\partial \mathbf{f}_x}{\partial v} & \frac{\partial \mathbf{f}_x}{\partial \theta} & \frac{\partial \mathbf{f}_x}{\partial \omega} \\ 0 & 1 & \frac{\partial \mathbf{f}_y}{\partial v} & \frac{\partial \mathbf{f}_y}{\partial \theta} & \frac{\partial \mathbf{f}_y}{\partial \omega} \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & \frac{\partial \mathbf{f}_\theta}{\partial \omega} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1-15}$$

where

$$\begin{aligned} \frac{\partial \mathbf{f}_x}{\partial v} &= \frac{1}{\omega_k}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\ \frac{\partial \mathbf{f}_x}{\partial \theta} &= \frac{v_k}{\omega_k}\left[\cos(\theta_k + \omega_k \Delta t) - \cos(\theta_k)\right] \\ \frac{\partial \mathbf{f}_x}{\partial \omega} &= -\frac{v_k}{\omega_k^2}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] + \frac{v_k}{\omega_k}\cos(\theta_k + \omega_k \Delta t)\Delta t \\ \frac{\partial \mathbf{f}_y}{\partial v} &= \frac{1}{\omega_k}\left[-\cos(\theta_k + \omega_k \Delta t) + \cos(\theta_k)\right] \\ \frac{\partial \mathbf{f}_y}{\partial \theta} &= \frac{v_k}{\omega_k}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\ \frac{\partial \mathbf{f}_y}{\partial \omega} &= \frac{v_k}{\omega_k^2}\left[\cos(\theta_k + \omega_k \Delta t) - \cos(\theta_k)\right] + \frac{v_k}{\omega_k}\sin(\theta_k + \omega_k \Delta t)\Delta t \\ \frac{\partial \mathbf{f}_\theta}{\partial \omega} &= \Delta t \end{aligned} \tag{3.1-16}$$

3.1.4 Constant Turn Rate and Acceleration Model

State vector of CTRA model

In CTRA model, the vehicle state at time k is defined as

$$\mathbf{x}_k = \begin{bmatrix} x & y & v & \theta & a & \omega \end{bmatrix}_k^T \tag{3.1-17}$$

where a denotes the acceleration.

State transition function of CTRA model

Constant turn rate and acceleration model assumes that the turn rate and acceleration of a vehicle are constant. The trajectory of CTRA model follows a

clothoid, see Figure 3.1 (the trajectory is simulated with initial state $[0m, 100m, 0deg, 3.14m/s, 0.11m/s^2, 1.8deg/s]^T$ and duration 200 s). The state at next time step k+1 can be calculated by:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \int_{t_k}^{t_{k+1}} \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{v}(t) \\ \dot{\theta}(t) \\ \dot{a}(t) \\ \dot{\omega}(t) \end{bmatrix} dt = \mathbf{x}_k + \begin{bmatrix} \int_{t_k}^{t_{k+1}} v(t)\cos(\theta(t))dt \\ \int_{t_k}^{t_{k+1}} v(t)\sin(\theta(t))dt \\ \int_{t_k}^{t_{k+1}} a(t)dt \\ \int_{t_k}^{t_{k+1}} \omega(t)dt \\ 0 \\ 0 \end{bmatrix} \tag{3.1-18}$$

where

$$\begin{aligned} &\int_{t_k}^{t_{k+1}} v(t)\cos(\theta(t))dt \\ &= \int_{t_k}^{t_{k+1}} \left[v_k + a_k(t-t_k)\right]\cos\left[\theta_k + \omega_k(t-t_k)\right]dt \\ &= \int_{t_k}^{t_{k+1}} v_k\cos\left[\theta_k + \omega_k(t-t_k)\right]dt + \int_{t_k}^{t_{k+1}} a_k(t-t_k)\cos\left[\theta_k + \omega_k(t-t_k)\right]dt \end{aligned} \tag{3.1-19}$$

The integration of the first term in Equation 3.1-19 can be obtained by followings:

$$\begin{aligned} &\int_{t_k}^{t_{k+1}} v_k\cos\left[\theta_k + \omega_k(t-t_k)\right]dt \\ &= v_k\int_{t_k}^{t_{k+1}} \cos\left[\theta_k + \omega_k(t-t_k)\right]d\left[\theta_k + \omega_k(t-t_k)\right]\frac{1}{\omega_k} \\ &= \frac{v_k}{\omega_k}\sin\left[\theta_k + \omega_k(t-t_k)\right]\Big|_{t_k}^{t_{k+1}} \\ &= \frac{v_k}{\omega_k}\left[\sin(\theta_k + \omega_k\Delta t) - \sin(\theta_k)\right] \end{aligned} \tag{3.1-20}$$

For the second term's integration in Equation 3.1-19, let $x = t - t_k$, then we have $dx = dt, \quad 0 \leqslant x \leqslant \Delta t$; and we arrive:

$$\begin{aligned}
&\int_{t_k}^{t_{k+1}} a_k(t-t_k)\cos\left[\theta_k + \omega_k(t-t_k)\right] dt \\
&= a_k \int_0^{\Delta t} x\cos(\theta_k + \omega_k x)dx \\
&= a_k \int_0^{\Delta t} x d\left[\frac{1}{\omega_k}\sin(\theta_k + \omega_k x)\right] \\
&= a_k \left[\frac{x}{\omega_k}\sin(\theta_k + \omega_k x)\Big|_0^{\Delta t} - \int_0^{\Delta t} \frac{1}{\omega_k}\sin(\theta_k + \omega_k x)dx\right] \\
&= \frac{a_k \Delta t}{\omega_k}\sin(\theta_k + \omega_k \Delta t) - \int_0^{\Delta t} \frac{a_k}{\omega_k}\sin(\theta_k + \omega_k x)d(\theta_k + \omega_k x)\frac{1}{\omega_k} \\
&= \frac{a_k \Delta t}{\omega_k}\sin(\theta_k + \omega_k \Delta t) + \frac{a_k}{\omega_k^2}\cos(\theta_k + \omega_k x)\Big|_0^{\Delta t} \\
&= \frac{a_k \Delta t}{\omega_k}\sin(\theta_k + \omega_k \Delta t) + \frac{a_k}{\omega_k^2}\cos(\theta_k + \omega_k \Delta t) - \frac{a_k}{\omega_k^2}\cos(\theta_k)
\end{aligned} \tag{3.1-21}$$

Substituting Equation 3.1-20 and Equation 3.1-21 into Equation 3.1-19, we obtain

$$\begin{aligned}
\Delta x &= \int_{t_k}^{t_{k+1}} v(t)\cos(\theta(t))dt \\
&= \frac{(v_k + a_k \Delta t)\sin(\theta_k + \omega_k \Delta t) - v_k \sin(\theta_k)}{\omega_k} + \frac{a_k\left[\cos(\theta_k + \omega_k \Delta t) - \cos(\theta_k)\right]}{\omega_k^2}
\end{aligned} \tag{3.1-22}$$

And the same to

$$\begin{aligned}
&\int_{t_k}^{t_{k+1}} v(t)\sin(\theta(t))dt \\
&= \int_{t_k}^{t_{k+1}} \left[v_k + a_k(t-t_k)\right]\sin\left[\theta_k + \omega_k(t-t_k)\right] dt \\
&= \int_{t_k}^{t_{k+1}} v_k \sin\left[\theta_k + \omega_k(t-t_k)\right] dt + \int_{t_k}^{t_{k+1}} a_k(t-t_k)\sin\left[\theta_k + \omega_k(t-t_k)\right] dt
\end{aligned} \tag{3.1-23}$$

The integration of the first term in Equation 3.1-23 can be obtained by followings.

$$\begin{aligned}
&\int_{t_k}^{t_{k+1}} v_k \sin\left[\theta_k + \omega_k(t-t_k)\right] dt \\
&= v_k \int_{t_k}^{t_{k+1}} \sin\left[\theta_k + \omega_k(t-t_k)\right] d\left[\theta_k + \omega_k(t-t_k)\right] \frac{1}{\omega_k} \\
&= -\frac{v_k}{\omega_k} \cos\left[\theta_k + \omega_k(t-t_k)\right] \Big|_{t_k}^{t_{k+1}} \\
&= -\frac{v_k}{\omega_k} \left[\cos(\theta_k + \omega_k \Delta t) - \cos(\theta_k)\right]
\end{aligned} \tag{3.1-24}$$

Similarly, for the second term's integration in Equation 3.1-23, let $x = t - t_k$, then we have $dx = dt, \quad 0 \leqslant x \leqslant \Delta t$; and we arrive:

$$\begin{aligned}
&\int_{t_k}^{t_{k+1}} a_k(t-t_k) \sin\left[\theta_k + \omega_k(t-t_k)\right] dt \\
&= a_k \int_0^{\Delta t} x \sin(\theta_k + \omega_k x) dx \\
&= a_k \int_0^{\Delta t} x d\left[\frac{-1}{\omega_k} \cos(\theta_k + \omega_k x)\right] \\
&= a_k \left[\frac{-x}{\omega_k} \cos(\theta_k + \omega_k x)\Big|_0^{\Delta t} - \int_0^{\Delta t} \frac{-1}{\omega_k} \cos(\theta_k + \omega_k x) dx\right] \\
&= -\frac{a_k \Delta t}{\omega_k} \cos(\theta_k + \omega_k \Delta t) + \int_0^{\Delta t} \frac{a_k}{\omega_k} \cos(\theta_k + \omega_k x) d(\theta_k + \omega_k x) \frac{1}{\omega_k} \\
&= -\frac{a_k \Delta t}{\omega_k} \cos(\theta_k + \omega_k \Delta t) + \frac{a_k}{\omega_k^2} \sin(\theta_k + \omega_k x)\Big|_0^{\Delta t} \\
&= -\frac{a_k \Delta t}{\omega_k} \cos(\theta_k + \omega_k \Delta t) + \frac{a_k}{\omega_k^2} \sin(\theta_k + \omega_k \Delta t) - \frac{a_k}{\omega_k^2} \sin(\theta_k)
\end{aligned} \tag{3.1-25}$$

Substituting Equation 3.1-24 and Equation 3.1-25 into Equation 3.1-23, we obtain

$$\begin{aligned}
\Delta y &= \int_{t_k}^{t_{k+1}} v(t) \sin(\theta(t)) dt \\
&= -\frac{(v_k + a_k \Delta t) \cos(\theta_k + \omega_k \Delta t) - v_k \cos(\theta_k)}{\omega_k} + \frac{a_k \left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right]}{\omega_k^2}
\end{aligned} \tag{3.1-26}$$

Finally, it is easy to know that

$$\Delta v = \int_{t_k}^{t_{k+1}} a(t) dt = a_k \Delta t \tag{3.1-27}$$

and

$$\Delta\theta = \int_{t_k}^{t_{k+1}} \omega(t)dt = \omega_k \Delta t \tag{3.1-28}$$

Substituting Equations 3.1-22, 3.1-26, 3.1-27, and 3.1-28 into Equation 3.1-18, the state transition functions of CTRA model can be obtained:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k) = \begin{bmatrix} x_k + \Delta x \\ y_k + \Delta y \\ v_k + \Delta v \\ \theta_k + \Delta\theta \\ a_k \\ \omega_k \end{bmatrix} \tag{3.1-29}$$

Jacobian of CTRA model

The following Jacobian is used to linearize CTRA mode. Firstly, let

$$\begin{aligned}
\mathbf{f}_x(\mathbf{x}) &= x_k + \frac{(v_k + a_k\Delta t)\sin(\theta_k + \omega_k\Delta t) - v_k\sin(\theta_k)}{\omega_k} + \frac{a_k\left[\cos(\theta_k + \omega_k\Delta t) - \cos(\theta_k)\right]}{\omega_k^2} \\
\mathbf{f}_y(\mathbf{x}) &= y_k - \frac{(v_k + a_k\Delta t)\cos(\theta_k + \omega_k\Delta t) - v_k\cos(\theta_k)}{\omega_k} + \frac{a_k\left[\sin(\theta_k + \omega_k\Delta t) - \sin(\theta_k)\right]}{\omega_k^2} \\
\mathbf{f}_v(\mathbf{x}) &= v_k + a_k\Delta t \\
\mathbf{f}_\theta(\mathbf{x}) &= \theta_k + \omega_k\Delta t \\
\mathbf{f}_a(\mathbf{x}) &= a_k \\
\mathbf{f}_\omega(\mathbf{x}) &= \omega_k
\end{aligned} \tag{3.1-30}$$

Then, based on Equation 3.1-13, the Jacobian of CTRA is as follows

$$\mathbf{J}_{CTRA} = \begin{bmatrix} 1 & 0 & \frac{\partial \mathbf{f}_x}{\partial v} & \frac{\partial \mathbf{f}_x}{\partial \theta} & \frac{\partial \mathbf{f}_x}{\partial a} & \frac{\partial \mathbf{f}_x}{\partial \omega} \\ 0 & 1 & \frac{\partial \mathbf{f}_y}{\partial v} & \frac{\partial \mathbf{f}_y}{\partial \theta} & \frac{\partial \mathbf{f}_y}{\partial a} & \frac{\partial \mathbf{f}_y}{\partial \omega} \\ 0 & 0 & 1 & 0 & \frac{\partial \mathbf{f}_v}{\partial a} & 0 \\ 0 & 0 & 0 & 1 & 0 & \frac{\partial \mathbf{f}_\theta}{\partial \omega} \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1-31}$$

where

$$\begin{aligned}
\frac{\partial \mathbf{f}_x}{\partial v} =& \frac{1}{\omega_k}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\
\frac{\partial \mathbf{f}_x}{\partial \theta} =& \frac{1}{\omega_k}\left[(v_k + a_k \Delta t)\cos(\theta_k + \omega_k \Delta t) - v_k \cos(\theta_k)\right] - \frac{a_k}{\omega_k^2}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\
\frac{\partial \mathbf{f}_x}{\partial a} =& \frac{1}{\omega_k}\left[\sin(\theta_k + \omega_k \Delta t)\Delta t\right] + \frac{1}{\omega_k^2}\left[\cos(\theta_k + \omega_k \Delta t) - \cos(\theta_k)\right] \\
\frac{\partial \mathbf{f}_x}{\partial \omega} =& \frac{1}{\omega_k}\left[(v_k + a_k \Delta t)\cos(\theta_k + \omega_k \Delta t)\Delta t\right] - \frac{1}{\omega_k^2}\left[(v_k + 2a_k \Delta t)\sin(\theta_k + \omega_k \Delta t) - v_k \sin(\theta_k)\right] \\
& - \frac{2a_k}{\omega_k^3}\left[\cos(\theta_k + \omega_k \Delta t) - \cos(\theta_k)\right] \\
\frac{\partial \mathbf{f}_y}{\partial v} =& \frac{1}{\omega_k}\left[-\cos(\theta_k + \omega_k \Delta t) + \cos(\theta_k)\right] \\
\frac{\partial \mathbf{f}_y}{\partial \theta} =& \frac{1}{\omega_k}\left[(v_k + a_k \Delta t)\sin(\theta_k + \omega_k \Delta t) - v_k \sin(\theta_k)\right] + \frac{a_k}{\omega_k^2}\left[\cos(\theta_k + \omega_k \Delta t) - \cos(\theta_k)\right] \\
\frac{\partial \mathbf{f}_y}{\partial a} =& \frac{-1}{\omega_k}\left[\cos(\theta_k + \omega_k \Delta t)\Delta t\right] + \frac{1}{\omega_k^2}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\
\frac{\partial \mathbf{f}_y}{\partial \omega} =& \frac{1}{\omega_k}\left[(v_k + a_k \Delta t)\sin(\theta_k + \omega_k \Delta t)\Delta t\right] + \frac{1}{\omega_k^2}\left[(v_k + 2a_k \Delta t)\cos(\theta_k + \omega_k \Delta t) - v_k \cos(\theta_k)\right] \\
& - \frac{2a_k}{\omega_k^3}\left[\sin(\theta_k + \omega_k \Delta t) - \sin(\theta_k)\right] \\
\frac{\partial \mathbf{f}_v}{\partial a} =& \Delta t \\
\frac{\partial \mathbf{f}_\theta}{\partial \omega} =& \Delta t
\end{aligned} \tag{3.1-32}$$

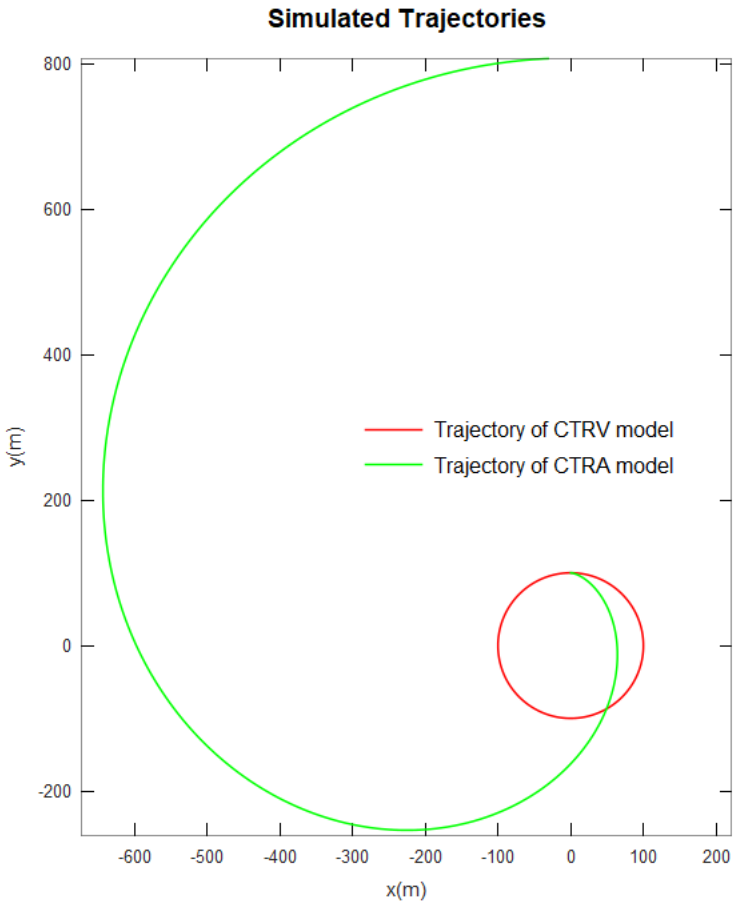


Figure 3.1: Simulated trajectories of CTRV and CTRA models. The trajectory of CTRV model is simulated with initial state $[0m, 100m, 0deg, 3.14m/s, -18deg/s]^T$ and duration 200 s; the trajectory of CTRA model is simulated with initial state $[0m, 100m, 0deg, 3.14m/s, -18deg/s]^T$ and duration 200 s.

3.2 Kalman Filters

In this section, the classical linear KF, and its nonlinear derivatives EKF, UKF, SRUKF are elaborated. The basic mathematical principles and algorithms of KF and EKF are detailed. For UKF and SRUKF, just their algorithms are introduced.

3.2.1 Involved Sensors

Kalman filter is widely used in sensor fusion. At the beginning of this section, the sensors involved in this dissertation are introduced. The sensors are mounted on the vehicle in Figure 3.2.

Figure 3.2: Involved sensors in this dissertation: the LiDAR: Velodyne HDL-64ES3; the GPS: Trimble NetR9; the IMU: Xsens MTi-300.

In this dissertation, both the LiDAR (Velodyne HDL-64ES3) and GPS (Trimble NetR9) can provide raw measurements of position (x, y), velocity $(v_{\mathbf{y}}^x, v_{\mathbf{y}}^y)$ and heading (θ) . The IMU (Xsens MTi-300) provides raw measurements of turn rate (ω) and acceleration $(a_{\mathbf{y}}^x, a_{\mathbf{y}}^y)$. The bold subscript $\mathbf{y}$ indicates that it is an element of measurement vector $\mathbf{y}$. The superscript x and y respectively indicate that it is a component in x and y direction.

These measurements are output through ROS (2022) messages from Autoware (2022) platform. The raw measurements are fused with the information that is derived from kinematic motion models by KFs to obtain a more accurate vehicle state estimate.

All the vehicle states are observed directly, except the velocity and acceleration. The measurement models are given in the following equations.

$$\begin{aligned} x_{\mathbf{y}} &= x_{\mathbf{x}} \quad \theta_{\mathbf{y}} = \theta_{\mathbf{x}} \quad v_{\mathbf{y}}^x = v_{\mathbf{x}} \cos(\theta_{\mathbf{x}}) \quad a_{\mathbf{y}}^x = a_{\mathbf{x}} \cos(\theta_{\mathbf{x}}) \\ y_{\mathbf{y}} &= y_{\mathbf{x}} \quad \omega_{\mathbf{y}} = \omega_{\mathbf{x}} \quad v_{\mathbf{y}}^y = v_{\mathbf{x}} \sin(\theta_{\mathbf{x}}) \quad a_{\mathbf{y}}^y = a_{\mathbf{x}} \sin(\theta_{\mathbf{x}}) \end{aligned} \tag{3.2-33}$$

where the bold subscript $\mathbf{x}$ indicate that it is an element of system state vector.

3.2.2 Linear Kalman Filter

We derive linear KF from Least Squares (LS) method.

Least Squares Method

Suppose $\mathbf{y}$ is a k-element noisy measurement vector of $\mathbf{x}$, which is a constant but unknown n-element vector. To find out the best estimate $\hat{\mathbf{x}}$ of $\mathbf{x}$, we assume that each element of $\mathbf{y}$ is a linear combination of the elements of $\mathbf{x}$, adding some measurement noise.

$$\begin{aligned} y_1 &= H_{11}x_1 + H_{12}x_2 + \cdots + H_{1n}x_n + v_1 \\ y_2 &= H_{21}x_1 + H_{22}x_2 + \cdots + H_{2n}x_n + v_2 \\ &\vdots \\ y_k &= H_{k1}x_1 + H_{k2}x_2 + \cdots + H_{kn}x_n + v_k \end{aligned} \tag{3.2-34}$$

where v_k is the measurement noise term. These equations can be written in matrix form:

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v} \tag{3.2-35}$$

The measurement residual $\mathbf{e}_y$ is defined as the difference between the noisy measurements and the vector $\mathbf{H}\hat{\mathbf{x}}$:

$$\mathbf{e}_y = \mathbf{y} - \mathbf{H}\hat{\mathbf{x}} \tag{3.2-36}$$

As Gauss wrote in 1857: the most probable value of vector $\mathbf{x}$ is the vector $\hat{\mathbf{x}}$ that minimizes the sum of squares between the observed values $\mathbf{y}$ and the vector

$\mathbf{H}\hat{\mathbf{x}}$. Therefore, a cost function J is defined as:

$$\begin{aligned} J =& e_{y1}^2 + e_{y2}^2 + \cdots + e_{yk}^2 \\ =& \mathbf{e}_y^T \mathbf{e}_y \\ =& (\mathbf{y} - \mathbf{H}\hat{\mathbf{x}})^T (\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}) \\ =& \mathbf{y}^T\mathbf{y} - \hat{\mathbf{x}}^T\mathbf{H}^T\mathbf{y} - \mathbf{y}^T\mathbf{H}\hat{\mathbf{x}} + \hat{\mathbf{x}}^T\mathbf{H}^T\mathbf{H}\hat{\mathbf{x}} \end{aligned} \tag{3.2-37}$$

To minimize J with respect to (w.r.t) $\hat{\mathbf{x}}$, it's partial derivative should be equal to zero:

$$\frac{\partial J}{\partial \hat{\mathbf{x}}} = 0 \tag{3.2-38}$$

The following partial derivative rules are used in computing the partial derivative.

$$\begin{array}{ll} \frac{\partial x^T A}{\partial x} = \frac{\partial A^T x}{\partial x} = A & \frac{\partial x^T A x}{\partial x} = Ax + A^T x = 2Ax \text{(if A is symmetric)} \\ \frac{\partial x^T x}{\partial x} = 2x & \frac{\partial A^T x x^T B}{\partial x} = AB^T x + BA^T x \end{array}$$

From Equations 3.2-38 and 3.2-37, we obtain

$$-2\mathbf{y}^T\mathbf{H} + 2\hat{\mathbf{x}}^T\mathbf{H}^T\mathbf{H} = 0 \tag{3.2-39}$$

Solving Equation 3.2-39 for $\hat{\mathbf{x}}$, we obtain:

$$\hat{\mathbf{x}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y} \tag{3.2-40}$$

Equation 3.2-40 is the LS solution for Equation 3.2-34.

Weighted Least Squares Method

In previous section, all the measurements are treated equally for amount of confidence. Now, we suppose that we have more confidence in some measurements than others. To solve this problem, Weighted Least Squares (WLS) method is introduced.

Firstly, we write Equation 3.2-34 into detailed matrix form:

$$\begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} H_{11} & \cdots & H_{1n} \\ \vdots & \ddots & \vdots \\ H_{k1} & \cdots & H_{kn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} v_1 \\ \vdots \\ v_k \end{bmatrix} \tag{3.2-41}$$

In Equation 3.2-41, we assume that the variance of the measurement noise v_i is distinct and each measurement noise is zero-mean and independent. Then the measurement covariance matrix is:

$$\mathbf{R} = \mathbb{E}(\mathbf{v}\mathbf{v}^T) = \begin{bmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_k^2 \end{bmatrix}, \text{ where } \sigma_i^2 = \mathbb{E}(v_i^2),\ i = 1, \cdots, k \tag{3.2-42}$$

Then, let us minimize weighted sum of squares. Firstly, the weighted sum can be written as:

$$J = \frac{e_{y1}^2}{\sigma_1^2} + \frac{e_{y2}^2}{\sigma_2^2} + \cdots + \frac{e_{yk}^2}{\sigma_k^2} \tag{3.2-43}$$

Writing J in matrix formulation:

$$\begin{aligned} J =& \mathbf{e}_y^T \mathbf{R}^{-1} \mathbf{e}_y \\ =& (\mathbf{y} - \mathbf{H}\hat{\mathbf{x}})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}) \\ =& \mathbf{y}^T \mathbf{R}^{-1} \mathbf{y} - \hat{\mathbf{x}}^T \mathbf{H}^T \mathbf{R}^{-1} \mathbf{y} - \mathbf{y}^T \mathbf{R}^{-1} \mathbf{H}\hat{\mathbf{x}} + \hat{\mathbf{x}}^T \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H}\hat{\mathbf{x}} \end{aligned} \tag{3.2-44}$$

In order to obtain the best estimate $\hat{\mathbf{x}}$, we take the partial derivative of J w.r.t $\hat{\mathbf{x}}$ and set it equal to zero:

$$\frac{\partial J}{\partial \hat{\mathbf{x}}} = -\mathbf{y}^T \mathbf{R}^{-1} \mathbf{H} + \hat{\mathbf{x}}^T \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} = -\mathbf{H}^T \mathbf{R}^{-1} \mathbf{y} + \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H}\hat{\mathbf{x}} = 0 \tag{3.2-45}$$

Solving Equation 3.2-45, the WLS solution of Equation 3.2-34 is obtained:

$$\hat{\mathbf{x}} = (\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{R}^{-1} \mathbf{y} \tag{3.2-46}$$

Recursive Least Squares Method

Equations 3.2-40 and 3.2-46 provide two ways to get the optimal estimate of a constant. In the equations, $\mathbf{H}$ is a $k \times n$ matrix and when we obtain measurements sequentially and want to update the estimate $\hat{\mathbf{x}}$ with the new measurements, we have to augment $\mathbf{H}$ and recompute $\hat{\mathbf{x}}$. To avoid that, Recursive Least Squares (RLS) is introduced to compute the weighted least squares estimate of a constant without solving Equation 3.2-46 repeatedly.

Suppose we have an optimal estimate $\hat{\mathbf{x}}_{k-1}$ after k-1 measurements and a new measurement $\mathbf{y}_k$ is obtained. To be more general, hereafter, the measurement is now regarded as an m-element vector. In this condition, we assume that a linear recursive estimator can be written in following form:

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x} + \mathbf{v}_k \tag{3.2-47}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k-1} + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k-1}) \tag{3.2-48}$$

In Equation 3.2-48, we assume that the new estimate $\hat{\mathbf{x}}_k$ is a linear combination of previous estimate $\hat{\mathbf{x}}_{k-1}$ and the correction term $(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k-1})$, $\mathbf{K}_k$ is a $n \times m$ gain matrix to be determined. $\mathbf{H}_k$ is an $m \times n$ matrix.

At time step k, the estimation error's expectation of the linear recursive estimator is:

$$\begin{aligned}
\mathbb{E}(\mathbf{e}_{x|k}) =& \mathbb{E}(\mathbf{x} - \hat{\mathbf{x}}_k) \\
=& \mathbb{E}\left[\mathbf{x} - \hat{\mathbf{x}}_{k-1} - \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k-1})\right] \\
=& \mathbb{E}\left[\mathbf{e}_{x|k-1} - \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k-1})\right] \\
=& \mathbb{E}\left[\mathbf{e}_{x|k-1} - \mathbf{K}_k(\mathbf{H}_k\mathbf{x} + \mathbf{v}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k-1})\right] \\
=& \mathbb{E}\left[\mathbf{e}_{x|k-1} - \mathbf{K}_k\mathbf{H}_k(\mathbf{x} - \hat{\mathbf{x}}_{k-1}) - \mathbf{K}_k\mathbf{v}_k\right] \\
=& (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbb{E}(\mathbf{e}_{x|k-1}) - \mathbf{K}_k\mathbb{E}(\mathbf{v}_k)
\end{aligned} \tag{3.2-49}$$

In Equation 3.2-49, if $\mathbb{E}(\mathbf{e}_{x|k-1}) = 0$ and $\mathbb{E}(\mathbf{v}_k) = 0$, then $\mathbb{E}(\mathbf{e}_{x|k}) = 0$. Namely, if

the measurement noise $\mathbf{v}_k$ is zero-mean for all k and the initial estimate of $\mathbf{x}$ is equal to the expected value of $\mathbf{x}$, like $\hat{\mathbf{x}}_0 = \mathbb{E}(\mathbf{x})$, then the expected value of $\hat{\mathbf{x}}_k$ will be equal to $\mathbf{x}_k$ for all k. With this property, Equation 3.2-48 is called *unbiased*.

Then, let us determine the optimal value for $\mathbf{K}_k$. The sum of the variances of estimation errors at time k is:

$$\begin{aligned} J_k &= \mathbb{E}\left[(x_1 - \hat{x}_1)^2\right] + \mathbb{E}\left[(x_2 - \hat{x}_2)^2\right] + \cdots + \mathbb{E}\left[(x_n - \hat{x}_n)^2\right] \\ &= \mathbb{E}(e_{x1|k}^2 + e_{x2|k}^2 + \cdots + e_{xn|k}^2) \\ &= \mathbb{E}(\mathbf{e}_{x|k}^T \mathbf{e}_{x|k}) \\ &= \mathbb{E}\left[\mathrm{Tr}(\mathbf{e}_{x|k}\mathbf{e}_{x|k}^T)\right] \\ &= \mathrm{Tr}\left[\mathbb{E}(\mathbf{e}_{x|k}\mathbf{e}_{x|k}^T)\right] \\ &= \mathrm{Tr}(\mathbf{P}_k) \end{aligned} \tag{3.2-50}$$

where $\mathrm{Tr}(\cdot)$ is trace operator; $\mathbf{P}_k$ is the estimation-error covariance. $\mathbf{P}_k$ can be recursively calculated through:

$$\begin{aligned} \mathbf{P}_k =& \mathbb{E}(\mathbf{e}_{x|k}\mathbf{e}_{x|k}^T) \\ =& \mathbb{E}\left\{\left[(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{e}_{x|k-1} - \mathbf{K}_k\mathbf{v}_k\right]\left[(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{e}_{x|k-1} - \mathbf{K}_k\mathbf{v}_k\right]^T\right\} \\ =& (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbb{E}(\mathbf{e}_{x|k-1}\mathbf{e}_{x|k-1}^T)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T - \mathbf{K}_k\mathbb{E}(\mathbf{v}_k\mathbf{e}_{x|k-1}^T)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T \\ & - (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbb{E}(\mathbf{e}_{x|k-1}\mathbf{v}_k^T)\mathbf{K}_k^T + \mathbf{K}_k\mathbb{E}(\mathbf{v}_k\mathbf{v}_k^T)\mathbf{K}_k^T \end{aligned} \tag{3.2-51}$$

Since estimation error $\mathbf{e}_{x|k-1}$ is independent of the zero mean measurement noise $\mathbf{v}_k$,therefore:

$$\mathbb{E}(\mathbf{e}_{x|k-1}\mathbf{v}_k^T) = \mathbb{E}(\mathbf{v}_k\mathbf{e}_{x|k-1}^T) = 0 \tag{3.2-52}$$

Considering Equation 3.2-52, Equation 3.2-51 becomes:

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k-1}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \tag{3.2-53}$$

where $\mathbf{R}_k$ is the covariance of $\mathbf{v}_k$.

When we try to find the best $\mathbf{K}_k$, the following partial derivative rules are useful:

$$\frac{\partial \operatorname{Tr}(ABA^T)}{\partial A} = 2AB \text{ (if B is symmetric)}; \qquad \frac{\partial \operatorname{Tr}(AB)}{\partial A} = B^T$$

The optimal $\mathbf{K}_k$ makes the cost function J_k minimum and we arrive:

$$\frac{\partial J_k}{\partial \mathbf{K}_k} = \frac{\partial \operatorname{Tr}(\mathbf{P}_k)}{\partial \mathbf{K}_k} = 2(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k-1}(-\mathbf{H}_k^T) + 2\mathbf{K}_k\mathbf{R}_k = 0 \tag{3.2-54}$$

$$\Rightarrow \mathbf{K}_k = \mathbf{P}_{k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \tag{3.2-55}$$

Recursive least squares algorithm We summarize RLS algorithm here.

1. Initialization:

$$\hat{\mathbf{x}}_0 = \mathbb{E}(\mathbf{x}) \tag{3.2-56}$$

$$\mathbf{P}_0 = \mathbb{E}\left[(\mathbf{x} - \hat{\mathbf{x}}_0)(\mathbf{x} - \hat{\mathbf{x}}_0)^T\right] \tag{3.2-57}$$

2. For $k = 1, 2, \cdots$

2.1 Get the measurement $\mathbf{y}_k$, which is given by:

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x} + \mathbf{v}_k \tag{3.2-58}$$

in which $\mathbf{v}_k$ is a zero-mean Gaussian white noise with covariance $\mathbf{R}_k$.

2.2 Update the estimate of $\mathbf{x}$ and the estimation error covariance $\mathbf{P}$ through:

$$\mathbf{K}_k = \mathbf{P}_{k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \tag{3.2-59}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k-1} + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k-1}) \tag{3.2-60}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k-1}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \tag{3.2-61}$$

Alternate forms of RLS Some alternate forms of RLS are provided here. To be pointed out, these alternate forms are mathematically identical. The details of

these forms, please refer to Simon (2006, chap. 3)

$$\begin{aligned}
\mathbf{K}_k =& \mathbf{P}_{k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1} = \mathbf{P}_k\mathbf{H}_k^T\mathbf{R}_k^{-1} && (3.2\text{-}62)\\
\hat{\mathbf{x}}_k =& \hat{\mathbf{x}}_{k-1} + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k-1}) && (3.2\text{-}63)\\
\mathbf{P}_k =& (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k-1}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T = (\mathbf{P}_{k-1}^{-1} + \mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k)^{-1}\\
=& (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k-1} && (3.2\text{-}64)
\end{aligned}$$

Propagation of State and Covariance

In vehicle state estimation and motion prediction applications, we have the following linear discrete-time system.

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \omega_{k-1} \tag{3.2-65}$$

Equation 3.2-65 is called state propagation equation and $\mathbf{F}_{k-1}$ is called state propagation/transition matrix, where ω_{k-1} is Gaussian zero-mean white noise with covariance $\mathbf{Q}_k$. Taking the expected value of both side of Equation 3.2-65, we get:

$$\bar{\mathbf{x}}_k = \mathbf{F}_{k-1}\bar{\mathbf{x}}_{k-1} \tag{3.2-66}$$

Then, the covariance of $\mathbf{x}_k$ can be calculated:

$$\begin{aligned}
\mathbf{P}_k =& \mathbb{E}[(\mathbf{x}_k - \bar{\mathbf{x}}_k)(\mathbf{x}_k - \bar{\mathbf{x}}_k)^T]\\
=& \mathbb{E}\left[(\mathbf{F}_{k-1}\mathbf{x}_{k-1} + \omega_{k-1} - \bar{\mathbf{x}}_k)(\mathbf{F}_{k-1}\mathbf{x}_{k-1} + \omega_{k-1} - \bar{\mathbf{x}}_k)^T\right]\\
=& \mathbb{E}\left[(\mathbf{F}_{k-1}\mathbf{x}_{k-1} + \omega_{k-1} - \mathbf{F}_{k-1}\bar{\mathbf{x}}_{k-1})(\mathbf{F}_{k-1}\mathbf{x}_{k-1} + \omega_{k-1} - \mathbf{F}_{k-1}\bar{\mathbf{x}}_{k-1})^T\right]\\
=& \mathbb{E}[\mathbf{F}_{k-1}(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1})(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1})^T\mathbf{F}_{k-1}^T + \mathbf{F}_{k-1}(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1})\omega_{k-1}^T +\\
& \omega_{k-1}\mathbf{F}_{k-1}(\mathbf{x}_{k-1} - \bar{\mathbf{x}}_{k-1}) + \omega_{k-1}\omega_{k-1}^T]\\
=& \mathbf{F}_{k-1}\mathbf{P}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}
\end{aligned} \tag{3.2-67}$$

Please note that the estimation error $(\mathbf{x}_{k-1}-\bar{\mathbf{x}}_{k-1})$ is uncorrelated with the process noise ω_{k-1}. Equation 3.2-67 is called covariance propagation equation.

The Kalman Filter

In this section, we derive the classical linear KF based on the equations of previous sections. Suppose we have a linear discrete-time system:

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \omega_{k-1} \tag{3.2-68}$$

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v_k} \tag{3.2-69}$$

The process and measurement noise ω_k and $\mathbf{v}_k$ are white, zero-mean, uncorrelated and have known covariance $\mathbf{Q}_k$ and $\mathbf{R}_k$:

$$\begin{aligned}
&\omega_k \sim (0, \mathbf{Q}_k) && \mathbb{E}(\omega_k\omega_j^T) = \mathbf{Q}_k\delta_{k-j} \\
&\mathbf{v}_k \sim (0, \mathbf{R}_k) && \mathbb{E}(\mathbf{v}_k\mathbf{v}_j^T) = \mathbf{R}_k\delta_{k-j} \\
&\delta_{k-j} = \begin{cases} 1, & k-j=0 \\ 0, & k-j\neq 0 \end{cases} \\
&\mathbb{E}(\mathbf{v}_k\omega_j^T) = 0
\end{aligned}$$

The first symbol that is used in this section is defined as:

$$\textit{a posteriori estimate: } \hat{\mathbf{x}}_k^+ = \mathbb{E}(\mathbf{x}_k|\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_k) \tag{3.2-70}$$

The superscript "+" denotes that the estimate is a posteriori. The way to get the a posteriori estimate is to compute the expected value of $\mathbf{x}_k$ conditioned on all the measurements up to and including time k. If we have all of the measurements before but not including time k, we can form an a priori estimate by computing the expected value of $\mathbf{x}_k$ conditioned on all the measurement. The second symbol is then defined as:

$$\textit{a priori estimate: } \hat{\mathbf{x}}_k^- = \mathbb{E}(\mathbf{x}_k|\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{k-1}) \tag{3.2-71}$$

The superscript "-" denotes that the estimate is a priori.

Giving a initial estimate of $\mathbf{x}_0$ and and its covariance before any measurements are available:

$$\hat{\mathbf{x}}_0^+ = \mathbb{E}(\mathbf{x}_0) \tag{3.2-72}$$

$$\mathbf{P}_0^+ = \mathbb{E}\left[(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)^T\right] \tag{3.2-73}$$

and based on the state and covariance propagation functions, Equations 3.2-65 and 3.2-67, we can get a priori estimate of $\mathbf{x}_1$ and a priori covariance of $\mathbf{P}_1$ at time $k = 1$:

$$\hat{\mathbf{x}}_1^- = \mathbf{F}_0\hat{\mathbf{x}}_0$$

$$\mathbf{P}_1^- = \mathbf{F}_0\mathbf{P}_0^+\mathbf{F}_0^T + \mathbf{Q}_0$$

We can write the above equations in more general formulation as following:

$$\hat{\mathbf{x}}_k^- = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1} \tag{3.2-74}$$

$$\mathbf{P}_k^- = \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1} \tag{3.2-75}$$

Equations 3.2-74 and 3.2-75 are called time update functions for state and covariance.

Now, we derive measurement update equations. To be noticed, the only difference between $\hat{\mathbf{x}}_k^-$ and $\hat{\mathbf{x}}_k^+$ is that $\hat{\mathbf{x}}_k^+$ takes into account $\mathbf{y}_k$, see Equations 3.2-70 and 3.2-71. In other words, in order to obtain the a posteriori estimate $\hat{\mathbf{x}}_k^+$, we just need to take the new measurement $\mathbf{y}_k$ into account to update the a priori estimate $\hat{\mathbf{x}}_k^-$. Such a problem has been already solved by previous RLS algorithm, in which $\hat{\mathbf{x}}_{k-1}$ and $\mathbf{P}_{k-1}$ are respectively the estimate of $\mathbf{x}$ and its covariance *before* the new measurement $\mathbf{y}_k$ is processed; and $\hat{\mathbf{x}}_k$ and $\mathbf{P}_k$ are respectively the estimate of $\mathbf{x}$ and its covariance *after* the new measurement $\mathbf{y}_k$ is processed. The terms $\hat{\mathbf{x}}_{k-1}$ and $\mathbf{P}_{k-1}$, of previous section, correspond to $\hat{\mathbf{x}}_k^-$ and $\mathbf{P}_k^-$ of this section, respectively;

and the terms $\hat{\mathbf{x}}_k$ and $\mathbf{P}_k$, of previous section, correspond to $\hat{\mathbf{x}}_k^+$ and $\mathbf{P}_k^+$ of this section, respectively.

We replace the terms in Equations 3.2-62, 3.2-63 and 3.2-64 with corresponding terms of this section, then we obtain:

$$\begin{aligned}
\mathbf{K}_k =& \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k)^{-1} = \mathbf{P}_k^+ \mathbf{H}_k^T \mathbf{R}_k^{-1} && \text{(3.2-76)} \\
\hat{\mathbf{x}}_k^+ =& \hat{\mathbf{x}}_k^- + \mathbf{K}_k (\mathbf{y}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^-) && \text{(3.2-77)} \\
\mathbf{P}_k^+ =& (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^- (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^T = [(\mathbf{P}_k^-)^{-1} + \mathbf{H}_k^T \mathbf{R}_k^{-1} \mathbf{H}_k]^{-1} \\
=& (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^- && \text{(3.2-78)}
\end{aligned}$$

These are called measurement update equations. $\mathbf{K}_k$ is called Kalman gain matrix.

The discrete-time KF algorithm We summarize KF algorithm as follows.

1. The dynamic system:

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{F}_{k-1} \mathbf{x}_{k-1} + \omega_{k-1} && \text{(3.2-79)} \\
\mathbf{y}_k &= \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k && \text{(3.2-80)} \\
\mathbb{E}(\omega_k \omega_j^T) &= \mathbf{Q}_k \delta_{k-j} \\
\mathbb{E}(\mathbf{v}_k \mathbf{v}_j^T) &= \mathbf{R}_k \delta_{k-j} \\
\mathbb{E}(\mathbf{v}_k \omega_j^T) &= 0
\end{aligned}$$

2. Initialization of KF:

$$\begin{aligned}
\hat{\mathbf{x}}_0^+ &= \mathbb{E}(\mathbf{x}_0) && \text{(3.2-81)} \\
\mathbf{P}_0^+ &= \mathbb{E}\left[(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)^T \right] && \text{(3.2-82)}
\end{aligned}$$

3. KF iteration for $k = 1, 2, \cdots$

3.1 Prediction (a priori estimation/time update)

$$\hat{\mathbf{x}}_k^- = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1}^+ \tag{3.2-83}$$

$$\mathbf{P}_k^- = \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1} \tag{3.2-84}$$

3.2 Correction (a posteriori estimation/measurement update)

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)^{-1} = \mathbf{P}_k^+\mathbf{H}_k^T\mathbf{R}_k^{-1} \tag{3.2-85}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-) \tag{3.2-86}$$

$$\begin{aligned}\mathbf{P}_k^+ =& (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T = [(\mathbf{P}_k^-)^{-1} + \mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k]^{-1} \\ =& (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-\end{aligned} \tag{3.2-87}$$

3.2.3 Nonlinear Kalman Filter

All the above discussion is about linear system. Unfortunately, linear systems do not really exist. All systems are ultimately nonlinear. Although nonlinear filtering is a difficult and complex subject, some nonlinear estimation methods have become widespread, such as EKF, UKF and SRUKF. In this section, the derivation process and algorithm of EKF are given. For both UKF and SRUKF, just their algorithms are introduced.

The Extended Kalman Filter

The idea of EKF is linearizing the nonlinear system around the closest state estimate and making estimation based on the linearized systems.

In vehicle state estimation, suppose we have the following nonlinear system

model:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} \tag{3.2-88}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \tag{3.2-89}$$

$$\mathbb{E}(\omega_k \omega_j^T) = \mathbf{Q}_k \delta_{k-j}$$

$$\mathbb{E}(\mathbf{v}_k \mathbf{v}_j^T) = \mathbf{R}_k \delta_{k-j}$$

$$\mathbb{E}(\mathbf{v}_k \omega_j^T) = 0$$

and we have an initial estimate and its covariance:

$$\hat{\mathbf{x}}_0^+ = \mathbb{E}(\mathbf{x}_0) \tag{3.2-90}$$

$$\mathbf{P}_0^+ = \mathbb{E}\left[(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)^T\right] \tag{3.2-91}$$

Assume we have an a posteriori estimate $\hat{\mathbf{x}}_{k-1}^+$ and its covariance $\mathbf{P}_{k-1}^+$ at time $k-1$, then we can obtain the a priori estimate at time k:

$$\begin{aligned} \hat{\mathbf{x}}_k^- =& \mathbb{E}(\mathbf{x}_k | \mathbf{y}_1, \cdots, \mathbf{y}_{k-1}) \\ =& \mathbb{E}\left[\mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} | \mathbf{y}_1, \cdots, \mathbf{y}_{k-1}\right] \\ =& \mathbb{E}\left[\mathbf{f}(\mathbf{x}_{k-1}) | \mathbf{y}_1, \cdots, \mathbf{y}_{k-1}\right] \end{aligned} \tag{3.2-92}$$

We perform a Taylor series expansion on the nonlinear function $\mathbf{f}(\mathbf{x}_{k-1})$ in Equation 3.2-92 around $\hat{\mathbf{x}}_{k-1}^+$:

$$\mathbf{f}(\mathbf{x}_{k-1}) = \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{J}_{\mathbf{f}}(\hat{\mathbf{x}}_{k-1}^+)(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}^+) + \text{high order terms} \tag{3.2-93}$$

where

$$\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_n(\mathbf{x})]^T$$
$$\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$$
$$\mathbf{J_f} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

$\mathbf{J_f}$ is Jacobian of $\mathbf{f}(\cdot)$. Dropping the high order terms in Equation 3.2-93, we obtain:

$$\begin{aligned} \mathbf{f}(\mathbf{x}_{k-1}) \approx & \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}^+) \\ = & \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ \end{aligned} \tag{3.2-94}$$

Substituting Equation 3.2-94 into Equation 3.2-92, we arrive:

$$\begin{aligned} \hat{\mathbf{x}}_k^- = & \mathbb{E}\left[\mathbf{f}(\mathbf{x}_{k-1})|\mathbf{y}_1, \cdots, \mathbf{y}_{k-1}\right] \\ \approx & \mathbb{E}\left[\mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+|\mathbf{y}_1, \cdots, \mathbf{y}_{k-1}\right] \\ = & \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbb{E}(\mathbf{e}_{k-1}^+|\mathbf{y}_1, \cdots, \mathbf{y}_{k-1}) \end{aligned}$$

Since the estimation error $\mathbf{e}_k$ is independent from measurement, we obtain:

$$\hat{\mathbf{x}}_k^- \approx \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) \tag{3.2-95}$$

Recalling Equation 3.2-94, the a priori estimation error at time k is:

$$\begin{aligned} \mathbf{e}_k^- = & \mathbf{x}_k - \hat{\mathbf{x}}_k^- \\ \approx & \mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} - \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) \\ = & \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ + \omega_{k-1} \end{aligned} \tag{3.2-96}$$

and the corresponding covariance can be calculated:

$$\begin{aligned}\mathbf{P}_k^- =&\mathbb{E}[\mathbf{e}_k^-(\mathbf{e}_k^-)^T]\\ =&\mathbb{E}\left\{\left[\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ + \omega_{k-1}\right]\left[\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ + \omega_{k-1}\right]^T\right\}\\ =&\mathbb{E}[\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+(\mathbf{e}_{k-1}^+)^T\mathbf{J_f}^T(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+\omega_{k-1}^T\\ &+ \omega_{k-1}(\mathbf{e}_{k-1}^+)^T\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+) + \omega_{k-1}\omega_{k-1}^T]\end{aligned} \tag{3.2-97}$$

Since $\mathbf{e}_k$ and ω_k are independent, thus $\mathbb{E}(\mathbf{e}_k\omega_k^T) = 0$, the above equation becomes:

$$\begin{aligned}\mathbf{P}_k^- =&\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbb{E}\left[\mathbf{e}_{k-1}^+(\mathbf{e}_{k-1}^+)^T\right]\mathbf{J_f}^T(\hat{\mathbf{x}}_{k-1}^+) + \mathbb{E}(\omega_{k-1}\omega_{k-1}^T)\\ =&\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{P}_{k-1}^+\mathbf{J_f}^T(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{Q}_{k-1}\end{aligned} \tag{3.2-98}$$

Equations 3.2-95 and 3.2-98 are called time update functions of EKF.

Now, we derive measurement update functions for EKF. Firstly, we suppose the a posteriori estimate is in the following formulation:

$$\hat{\mathbf{x}}_k^+ = A + \mathbf{K}_k\mathbf{y}_k \tag{3.2-99}$$

For unbiased estimation, we have:

$$\begin{aligned}&\mathbb{E}(\mathbf{x}_k - \hat{\mathbf{x}}_k^+|\mathbf{y}_k) = 0\\ \Rightarrow&\mathbb{E}\left[(\hat{\mathbf{x}}_k^- + \mathbf{e}_k^-) - (A + \mathbf{K}_k\mathbf{y}_k)|\mathbf{y}_k\right] = 0\\ \Rightarrow&\mathbb{E}\left[\hat{\mathbf{x}}_k^- + \mathbf{e}_k^- - A - \mathbf{K}_k(\mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k)|\mathbf{y}_k\right]\\ \Rightarrow&\mathbb{E}\left[\hat{\mathbf{x}}_k^- + \mathbf{e}_k^- - A - \mathbf{K}_k\mathbf{h}(\mathbf{x}_k) - \mathbf{K}_k\mathbf{v}_k|\mathbf{y}_k\right] = 0\\ \Rightarrow&\hat{\mathbf{x}}_k^- + \mathbb{E}(\mathbf{e}_k^-|\mathbf{y}_k) - A - \mathbf{K}_k\mathbb{E}(\mathbf{h}(\mathbf{x}_k)|\mathbf{y}_k) - \mathbf{K}_k\mathbb{E}(\mathbf{v}_k|\mathbf{y}_k) = 0\\ \Rightarrow&\hat{\mathbf{x}}_k^- - A - \mathbf{K}_k\mathbb{E}(\mathbf{h}(\mathbf{x}_k)|\mathbf{y}_k) = 0\\ \Rightarrow&A = \hat{\mathbf{x}}_k^- - \mathbf{K}_k\mathbb{E}(\mathbf{h}(\mathbf{x}_k)|\mathbf{y}_k)\end{aligned} \tag{3.2-100}$$

Substituting Equation 3.2-100 into Equation 3.2-99, we obtain:

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left[\mathbf{y}_k - \mathbb{E}(\mathbf{h}(\mathbf{x}_k)|\mathbf{y}_k)\right] \tag{3.2-101}$$

Now, we expand the function $\mathbf{h}(\mathbf{x}_k)$ in Equation 3.2-101 around $\hat{\mathbf{x}}_k^-$ and drop the high order terms:

$$\begin{aligned} \mathbf{h}(\mathbf{x}_k) &= \mathbf{h}(\hat{\mathbf{x}}_k^-) + \mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-) + \text{high order terms} \\ &\approx \mathbf{h}(\hat{\mathbf{x}}_k^-) + \mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-)(\mathbf{x}_k - \hat{\mathbf{x}}_k^-) \\ &= \mathbf{h}(\hat{\mathbf{x}}_k^-) + \mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-)\mathbf{e}_k^- \end{aligned} \tag{3.2-102}$$

in which

$$\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \cdots, h_m(\mathbf{x})]^T$$

$$\mathbf{J}_\mathbf{h} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial h_m}{\partial x_1} & \frac{\partial h_m}{\partial x_2} & \cdots & \frac{\partial h_m}{\partial x_n} \end{bmatrix}$$

$\mathbf{J}_\mathbf{h}$ is Jacobian of $\mathbf{h}(\cdot)$. Substituting Equation 3.2-102 into Equation 3.2-101, we get:

$$\begin{aligned} \hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left\{\mathbf{y}_k - \mathbb{E}\left[\mathbf{h}(\hat{\mathbf{x}}_k^-) + \mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-)\mathbf{e}_k^-|\mathbf{y}_k\right]\right\} \\ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\mathbf{y}_k - \mathbf{K}_k\mathbf{h}(\hat{\mathbf{x}}_k^-) + \mathbf{K}_k\mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-)\mathbb{E}(\mathbf{e}_k^-|\mathbf{y}_k) \\ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-)) \end{aligned} \tag{3.2-103}$$

The a posteriori estimation error is:

$$
\begin{aligned}
\mathbf{e}_k^+ &= \mathbf{x}_k - \hat{\mathbf{x}}_k^+ \\
&= \mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} - \hat{\mathbf{x}}_k^- - \mathbf{K}_k(\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-)) \\
&\approx \mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} - \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) - \mathbf{K}_k(\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-)) \\
&= \mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} - \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) - \mathbf{K}_k\left[\mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-)\right] \\
&= \mathbf{f}(\mathbf{x}_{k-1}) - \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) - \mathbf{K}_k\left[\mathbf{h}(\mathbf{x}_k) - \mathbf{h}(\hat{\mathbf{x}}_k^-)\right] + \omega_{k-1} - \mathbf{K}_k\mathbf{v}_k \\
&\approx \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\mathbf{e}_k^- - \mathbf{K}_k\mathbf{v}_k + \omega_{k-1} \\
&\approx \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)(\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ + \omega_{k-1}) - \mathbf{K}_k\mathbf{v}_k + \omega_{k-1} \\
&= \left[\mathbf{I} - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\right]\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ + \left[\mathbf{I} - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\right]\omega_{k-1} - \mathbf{K}_k\mathbf{v}_k
\end{aligned}
\tag{3.2-104}
$$

Substituting Equation 3.2-104 in following equations, the a posteriori estimation covariance can be computed:

$$
\begin{aligned}
\mathbf{P}_k^+ &= \mathbb{E}(\mathbf{e}_k^+\mathbf{e}_k^{+T}) \\
&= \mathbb{E}\left\{\left[\left[\mathbf{I} - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\right]\mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{e}_{k-1}^+ + \left[\mathbf{I} - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\right]\omega_{k-1} - \mathbf{K}_k\mathbf{v_k}\right][\cdots]^T\right\} \\
&\quad \cdots \\
&= \left[\mathbf{I} - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\right]\mathbf{P}_k^-\left[\mathbf{I} - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\right]^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T
\end{aligned}
\tag{3.2-105}
$$

As aforementioned, the optimal $\mathbf{K}_k$ makes the trace of covariance minimum. Therefore, we perform the partial derivative to $\mathbf{P}_k^+$ w.r.t $\mathbf{K}_k$ and set it equal to zero, then we obtain:

$$
\begin{aligned}
&\frac{\partial \operatorname{Tr}(\mathbf{P}_k^+)}{\partial \mathbf{K}_k} = 0 \\
\Rightarrow &-2\left[\mathbf{I} - \mathbf{K}_k\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\right]\mathbf{P}_k^-{\mathbf{J_h}}^T(\hat{\mathbf{x}}_k^-) + 2\mathbf{K}_k\mathbf{R}_k = 0 \\
\Rightarrow &\mathbf{K}_k = \mathbf{P}_k^-{\mathbf{J_h}}^T(\hat{\mathbf{x}}_k^-)\left[\mathbf{J_h}(\hat{\mathbf{x}}_k^-)\mathbf{P}_k^-{\mathbf{J_h}}^T(\hat{\mathbf{x}}_k^-) + \mathbf{R}_k\right]^{-1}
\end{aligned}
\tag{3.2-106}
$$

Substituting Equation 3.2-106 back in Equation 3.2-105, we get:

$$\mathbf{P}_k^+ = \left[\mathbf{I} - \mathbf{K}_k \mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-)\right] \mathbf{P}_k^- \tag{3.2-107}$$

The discrete-time EKF algorithm We summarize the EKF algorithm for discrete-time system here.

1. The dynamic system:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} \tag{3.2-108}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \tag{3.2-109}$$

$$\mathbb{E}(\omega_k \omega_j^T) = \mathbf{Q}_k \delta_{k-j}$$

$$\mathbb{E}(\mathbf{v}_k \mathbf{v}_j^T) = \mathbf{R}_k \delta_{k-j}$$

$$\mathbb{E}(\mathbf{v}_k \omega_j^T) = 0$$

2. Initialization:

$$\hat{\mathbf{x}}_0^+ = \mathbb{E}(\mathbf{x}_0) \tag{3.2-110}$$

$$\mathbf{P}_0^+ = \mathbb{E}\left[(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)^T\right] \tag{3.2-111}$$

3. EKF iteration for $k = 1, 2, \cdots$

3.1 Prediction (a priori estimation/time update)

$$\hat{\mathbf{x}}_k^- \approx \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) \tag{3.2-112}$$

$$\mathbf{P}_k^- = \mathbf{J}_\mathbf{f}(\hat{\mathbf{x}}_{k-1}^+) \mathbf{P}_{k-1}^+ {\mathbf{J}_\mathbf{f}}^T(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{Q}_{k-1} \tag{3.2-113}$$

3.2 Correction (a posteriori estimation/measurement update)

$$\mathbf{K}_k = \mathbf{P}_k^- {\mathbf{J}_\mathbf{h}}^T(\hat{\mathbf{x}}_k^-) \left[\mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-) \mathbf{P}_k^- {\mathbf{J}_\mathbf{h}}^T(\hat{\mathbf{x}}_k^-) + \mathbf{R}_k\right]^{-1} \tag{3.2-114}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-)) \tag{3.2-115}$$

$$\mathbf{P}_k^+ = \left[\mathbf{I} - \mathbf{K}_k \mathbf{J}_\mathbf{h}(\hat{\mathbf{x}}_k^-)\right] \mathbf{P}_k^- \tag{3.2-116}$$

The Unscented Kalman Filter

The extended Kalman filter has two flaws: (1) the linearization may lead to poor performance and divergence when the nonlinearities are considerable, because the higher order terms of Taylor expansion are ignored, see Equations 3.2-94 and 3.2-102; (2) the derivation of the Jacobian matrices may be nontrivial in some applications. To overcome these flaws, the UKF is proposed. Unlike EKF, UKF copes with the nonlinearities by deterministically sampling around the optimal estimate $\hat{\mathbf{x}}_k^+$. UKF can capture the posterior mean and covariance, accurate to the third order (Taylor expansion) for any nonlinearity; in contrast, EKF only achieves first-order accuracy. In this section, we introduce the UKF algorithm.

The unscented Kalman filter begins with a deterministic sampling after the same initialization process with the EKF. Firstly, $2n+1$ (n is the dimension of state vector) sigma points $\mathcal{X}_{k|i}$ (with the weights $\mathcal{W}_i$) are selected through:

$$\begin{aligned}
\mathcal{X}_{k|0} &= \hat{\mathbf{x}}_k^+ \\
\mathcal{X}_{k|i} &= \hat{\mathbf{x}}_k^+ + \left(\sqrt{(n+\lambda)\mathbf{P}_k}\right)_i, \quad i = 1, \cdots, n \\
\mathcal{X}_{k|i} &= \hat{\mathbf{x}}_k^+ - \left(\sqrt{(n+\lambda)\mathbf{P}_k}\right)_{i-n}, \quad i = n+1, \cdots, 2n \\
\mathcal{W}_0^{(m)} &= \frac{\lambda}{n+\lambda} \\
\mathcal{W}_0^{(c)} &= \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta) \\
\mathcal{W}_i^{(m)} &= \mathcal{W}_i^{(c)} = \frac{1}{2(n+\lambda)}, \quad i = 1, \cdots, 2n
\end{aligned} \tag{3.2-117}$$

where $\left(\sqrt{(n+\lambda)\mathbf{P}_k}\right)_i$ is the i-th column of the matrix square root. $\lambda = \alpha^2(n+\kappa) - n$ is the scaling parameter; α determines the spread of the sigma points around $\hat{\mathbf{x}}_k^+$ and is usually set to a positive value not more than 1; κ is the second scaling parameter usually set to 0; β is used to incorporate prior knowledge of the distribution of $\mathbf{x}$; for Gaussian distribution, $\beta = 2$ is optimal (Wan & Van Der Merwe,

2000). Then, each sigma point is propagated through the state transition function:

$$\mathcal{X}_{k+1|i} = \mathbf{f}(\mathcal{X}_{k|i}) \tag{3.2-118}$$

The state and covariance prediction is derived from the weighted mean and covariance of the transformed sigma points:

$$\hat{\mathbf{x}}_{k+1}^{-} = \sum_{i=0}^{2n} \mathcal{W}_i^{(m)} \mathcal{X}_{k+1|i} \tag{3.2-119}$$

$$\mathbf{P}_{k+1}^{-} = \sum_{i=0}^{2n} \mathcal{W}_i^{(c)} (\mathcal{X}_{k+1|i} - \hat{\mathbf{x}}_{k+1}^{-})(\mathcal{X}_{k+1|i} - \hat{\mathbf{x}}_{k+1}^{-})^T + \mathbf{Q}_k \tag{3.2-120}$$

The above equations are called time update equations of UKF.

In the correction step, the observations are predicted through:

$$\mathcal{Y}_{k+1|i} = \mathbf{h}(\mathcal{X}_{k+1|i}), \quad i = 0, \cdots, 2n \tag{3.2-121}$$

$$\hat{\mathbf{y}}_{k+1}^{-} = \sum_{i=0}^{2n} \mathcal{W}_i^{(m)} \mathcal{Y}_{k+1|i} \tag{3.2-122}$$

Then the predicted state and covariance are corrected via the following measurement update functions:

$$\hat{\mathbf{x}}_{k+1}^{+} = \hat{\mathbf{x}}_{k+1}^{-} + \mathbf{K}_{k+1}(\mathbf{y}_{k+1} - \hat{\mathbf{y}}_{k+1}^{-}) \tag{3.2-123}$$

$$\mathbf{P}_{k+1} = \mathbf{P}_{k+1}^{-} - \mathbf{K}_{k+1} \mathbf{P}_{yy} \mathbf{K}_{k+1}^{T} \tag{3.2-124}$$

where the Kalman gain matrix $\mathbf{K}_{k+1}$ is:

$$\mathbf{K}_{k+1} = \mathbf{P}_{xy} \mathbf{P}_{yy}^{-1} \tag{3.2-125}$$

where

$$\mathbf{P}_{yy} = \sum_{i=0}^{2n} \mathcal{W}_i^{(c)} (\mathcal{Y}_{k+1|i} - \hat{\mathbf{y}}_{k+1}^-)(\mathcal{Y}_{k+1|i} - \hat{\mathbf{y}}_{k+1}^-)^T + \mathbf{R}_{k+1} \tag{3.2-126}$$

$$\mathbf{P}_{xy} = \sum_{i=0}^{2n} \mathcal{W}_i^{(c)} (\mathcal{X}_{k+1|i} - \hat{\mathbf{x}}_{k+1}^-)(\mathcal{Y}_{k+1|i} - \hat{\mathbf{y}}_{k+1}^-)^T \tag{3.2-127}$$

Both UKF and EKF are based on the standard KF framework; they just differ in addressing the nonlinearities: UKF approximates the state probability distribution, while EKF approximates the nonlinear state space models. Theoretically, UKF outperforms EKF and they have equal computational complexity of $\mathcal{O}(n^3)$. However, UKF needs to transform each sigma point through the process and observation functions at each prediction and update process, which may cost more time for high dimensional problems.

Square Root Unscented Kalman Filter

The unscented Kalman filter has to compute the square root of the state covariance matrix at each iteration. This requires that the covariance matrix keeps positive definiteness. However, in practice, the covariance matrix may lose positive definiteness due to some numerical problems, like precision lose in floating point computations; and that makes UKF to fail. To prevent the numerical instability of UKF, SRUKF that guarantees the state covariance positive semi-definite is proposed by Van Der Merwe & Wan (2001). In this section, the SRUKF is introduced.

Square-Root UKF is initialized by calculating the matrix square-root of the state covariance through Cholesky factorization (denoted by $\mathbf{chol}\{\cdot\}$).

$$\hat{\mathbf{x}}_0^+ = \mathbb{E}\left[\mathbf{x}_0\right] \quad \mathbf{S}_0 = \mathbf{chol}\left\{\mathbb{E}\left[\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+\right)\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+\right)^T\right]\right\} \tag{3.2-128}$$

Then, the sigma points are computed by:

$$\mathcal{X}_{k-1} = \begin{bmatrix} \hat{\mathbf{x}}_{k-1}^{+} & \hat{\mathbf{x}}_{k-1}^{+} - \eta \mathbf{S}_k & \hat{\mathbf{x}}_{k-1}^{+} + \eta \mathbf{S}_k \end{bmatrix} \tag{3.2-129}$$

where $\eta = \sqrt{n + \lambda}$, n is the dimension of state, and λ is the scaling parameter. The time update equations are as follows:

$$\begin{aligned} &\mathcal{X}_{k|k-1} = \mathbf{F}(\mathcal{X}_{k-1}) \quad \hat{\mathbf{x}}_k^{-} = \sum_{i=0}^{2n} W_i^{(m)} \mathcal{X}_{i,k|k-1} \\ &\mathbf{S}_k^{-} = \mathbf{qr}\left\{ \begin{bmatrix} \sqrt{W_1^{(c)}} (\mathcal{X}_{1:2n,k|k-1} - \hat{\mathbf{x}}_k^{-}) & \sqrt{\mathbf{Q}} \end{bmatrix} \right\} \\ &\mathbf{S}_k^{-} = \mathbf{cholupdate}\left\{ \mathbf{S}_k^{-}, \mathcal{X}_{0,k} - \hat{\mathbf{x}}_k^{-}, W_0^{(c)} \right\} \\ &\mathcal{Y}_{k|k-1} = \mathbf{H}(\mathcal{X}_{k|k-1}) \quad \hat{\mathbf{y}}_k^{-} = \sum_{i=0}^{2n} W_i^{(m)} \mathcal{Y}_{i,k|k-1} \end{aligned} \tag{3.2-130}$$

where $\mathbf{qr}\{*\}$ denotes QR decomposition function that returns the transpose of the Cholesky factor; $\mathbf{cholupdate}\{*\}$ denotes the function of rank-1 update to the Cholesky factorization. The measurement update equations are:

$$\begin{aligned} &\mathbf{S}_{\tilde{\mathbf{y}}_k} = \mathbf{qr}\left\{ \begin{bmatrix} \sqrt{W_1^{(c)}} (\mathcal{Y}_{1:2n,k} - \hat{\mathbf{y}}_k^{-}) & \sqrt{\mathbf{R}}) \end{bmatrix} \right\} \\ &\mathbf{S}_{\tilde{\mathbf{y}}_k} = \mathbf{cholupdate}\left\{ \mathbf{S}_{\tilde{\mathbf{y}}_k}, \mathcal{Y}_{0,k} - \hat{\mathbf{y}}_k^{-}, W_0^{(c)} \right\} \\ &\mathbf{P}_{\mathbf{x}_k \mathbf{y}_k} = \sum_{i=0}^{2n} W_i^{(c)} [\mathcal{X}_{i,k|k-1} - \hat{\mathbf{x}}_k^{-}][\mathcal{Y}_{i,k|k-1} - \hat{\mathbf{y}}_k^{-}]^T \\ &\mathbf{G}_k = (\mathbf{P}_{\mathbf{x}_k \mathbf{y}_k} / \mathbf{S}_{\tilde{\mathbf{y}}_k}^T) / \mathbf{S}_{\tilde{\mathbf{y}}_k} \\ &\hat{\mathbf{x}}_k^{+} = \hat{\mathbf{x}}_k^{-} + \mathbf{G}_k (\mathbf{y}_k - \hat{\mathbf{y}}_k^{-}) \\ &\mathbf{S}_k = \mathbf{cholupdate}\left\{ \mathbf{S}_k^{-}, \mathbf{G}_k \mathbf{S}_{\tilde{\mathbf{y}}_k}, -1 \right\} \end{aligned} \tag{3.2-131}$$

CHAPTER 4

COMPARATIVE EVALUATION OF KINEMATIC MOTION MODELS AND KALMAN FILTERS IN VEHICLE STATE ESTIMATION AND MOTION PREDICTION

In this chapter, real world experiments are carried out to investigate the properties of two widely used nonlinear Kalman filters, EKF and UKF, and two widely used kinematic models, CTRV and CTRA, in vehicle state estimation and motion prediction applications.

4.1 Experimental System

The experimental system of this chapter is illustrated in Figure 4.1. In this system, LiDAR, GPS and IMU are used. Due to the sensor data inherently containing greater or lesser uncertainties, it is not suggested that the raw sensor measurements should be used directly in application. Especially in motion prediction, the uncertainties will be propagated and accumulated. In practice, it is necessary initially to employ a state estimator to reduce the uncertainties considering real-time sensor data and vehicular motion model. An accurate vehicular state estimate helps both current and future situation awareness. To gain awareness of the upcoming situations, motion prediction is usually performed in ADAS and AD applications. In this experiment, a future path is defined as a moving trajectory between the current time (instant k) and few seconds later. In each instant, a future path is iteratively predicted by the kinematic model. In this system, vehicular motion models, specifically the CTRV and CTRA models, play the roles of

KF's process model and path predictor. Certainly, more sophisticated models do exist, e.g., that of Julier & Durrant-Whyte (2003). The reasons these two models are chosen are: (1) they are sufficient for vehicle state estimation and motion prediction applications; (2) more elaborate models may contain unobservable parameters; (3) they are widely used in intelligent vehicles. The obvious advantages of utilizing the models to predict path are feasibility and simplicity. Given an initial state, a path can be generated iteratively in real time. Such predicted path has been used in some vital applications, such as collision warning (Tan & Huang, 2006) and emergency electronic brake lights (Lytrivis et al., 2011). The estimated state and predicted path are compared with the baseline to assess their relative difference.

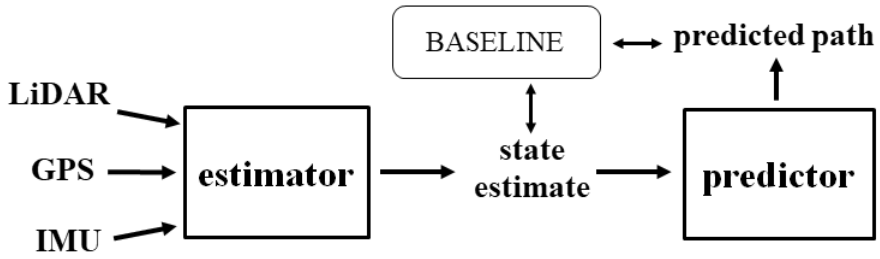


Figure 4.1: Overview of experimental system.

4.2 Experimental Design

The design of the experiments is described in Figure 4.2. The experiments were implemented in C++. GPS, IMU and LiDAR are installed on the *Autoware* (Kato et al., 2015) platform to provide vehicle state measurements, including position, heading, velocity, acceleration and angular rate. The measurements are fused by the four estimators considering the kinematic models to generate vehicle state estimates. In the experiments, all the combinations of the most popular KF derivatives (EKF and UKF) and motion models (CTRV and CTRA) are investigated. In both theory and practice, UKF outperforms EKF (Wan & Van Der Merwe, 2000; Julier, 2002; Yang et al., 2017); and the CTRA model

outperforms other kinematic models (Schubert et al., 2008). Therefore, in this study, the estimate (E0) of the UKF-CTRA filter is regarded as the baseline with which the other filters' estimates (E1, E2 and E3) are compared to find out the relative difference.

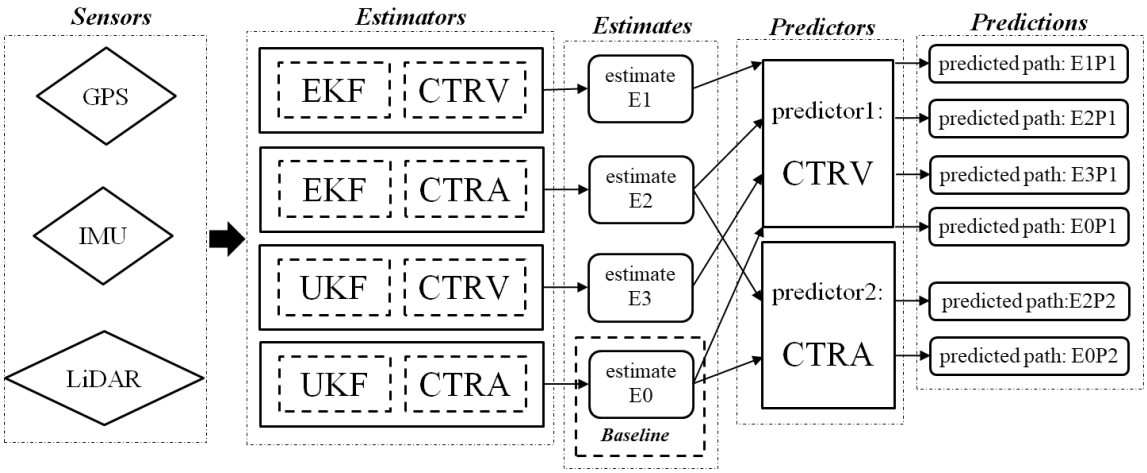


Figure 4.2: Experimental design.

The predictors, CTRV and CTRA, generate 3 s-ahead paths using the estimated states and the paths' reliabilities are checked through comparison with the baseline. The notation of the path indicates the used state estimate and predictor. For example, E0P2 is generated by predictor2 based on estimate E0. The relationship of measurement, estimate and prediction is illustrated by Figure 4.3, where, for the sake of brevity, only some of them are drawn. The blue point represents the raw position measurement that inevitably contains noise. The KF is used to reduce the noise. As the figure illustrates, the position estimates (the triangles) are closer to the real trajectory. In practice, the real trajectory cannot be completely known but approximated, since the uncertainties cannot be removed completely. Based on different state estimates, different paths are predicted by the motion models solely and their reliabilities decrease over time. The efficiency of the estimator is also investigated through the mean computational time of each iteration.

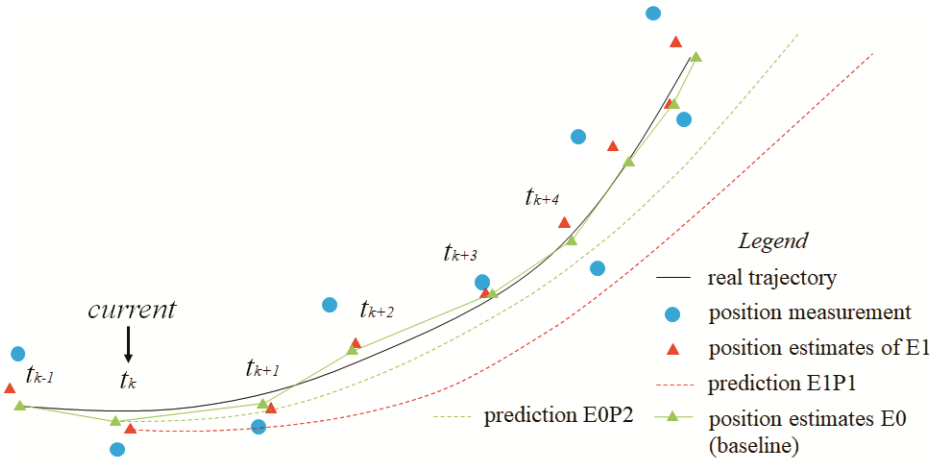


Figure 4.3: Relationship of measurement, estimate and prediction.

4.3 Experimental Configurations

4.3.1 Configurations of Sensors

The experiments were carried out via the Toyota PHV vehicle of Nagoya University, shown in Figure 4.4. Several kinds of sensors are mounted on the vehicle. In the experiments, LiDAR (Velodyne HDL-64ES3), IMU (Xsens MTi-300) and Global Navigation Satellite System (GNSS) receiver (Trimble NetR9) were used. Only fixed GPS data were received in the test fields.

Figure 4.4: Test vehicle.

In these experiments, *Autoware* just generates raw vehicle state measurements within a unified coordinate system. The LiDAR measurements were generated at 10 Hz, the GPS at 20 Hz (in two drives, GPS worked at 10 Hz) and the IMU 100 Hz. In urban areas, the measurements of LiDAR and IMU were fused by the estimators; on the highway, GPS and IMU were used instead. In KF, the prediction process is performed at 100 Hz and the update process is performed as soon as a new measurement is available. The observation functions are presented in Equation 4.3-1, where the bold subscript $\mathbf{x}$ and $\mathbf{y}$ respectively indicate the system state and observation vector. As the equation expresses, all the states are observed directly, except the velocity and acceleration.

$$\begin{aligned} x_{\mathbf{y}} &= x_{\mathbf{x}} \quad \theta_{\mathbf{y}} = \theta_{\mathbf{x}} \quad v^x_{\mathbf{y}} = v_{\mathbf{x}} \cos(\theta_{\mathbf{x}}) \quad a^x_{\mathbf{y}} = a_{\mathbf{x}} \cos(\theta_{\mathbf{x}}) \\ y_{\mathbf{y}} &= y_{\mathbf{x}} \quad \omega_{\mathbf{y}} = \omega_{\mathbf{x}} \quad v^y_{\mathbf{y}} = v_{\mathbf{x}} \sin(\theta_{\mathbf{x}}) \quad a^y_{\mathbf{y}} = a_{\mathbf{x}} \sin(\theta_{\mathbf{x}}) \end{aligned} \tag{4.3-1}$$

4.3.2 Configurations of EKF and UKF

For EKF and UKF, it is necessary to select the appropriate process and observation noise covariance matrix $\mathbf{Q}$ and $\mathbf{R}$ and to start the algorithm with initial $\hat{\mathbf{x}}_0^+$ and $\mathbf{P}_0^+$. In this study, $\mathbf{Q}$ and $\mathbf{R}$ are considered to be constant. The setting of these parameters follows the approach proposed by Schneider & Georgakis (2013), and some of them are tuned empirically. Since $\mathbf{Q}$ and $\mathbf{R}$ are system and sensor dependent, they should be determined based on the user's system and sensors. The parameters used in the experiments are listed in Tables 4.1 and 4.2. In the experiments, CTRV and CTRA use identical process noise parameters. The first aligned observation is assigned to $\hat{\mathbf{x}}_0^+$, and $\mathbf{P}_0^+$ is set by:

$$\mathbf{P}_0 = \sigma^2 \mathbf{I} \tag{4.3-2}$$

In this experiment $\sigma = 10$. In addition, the parameters α, β and κ of UKF are set to 0.1, 2 and 0, respectively.

Table 4.1: Standard deviation of process noise.

Parameter	Value	Description
w_x, w_y	$0.10m$	position process noise
w_θ	$3.16 \times 10^{-4} rad$	heading process noise
w_v	$3.16 \times 10^{-3} m/s$	velocity process noise
w_a	$3.16 \times 10^{-3} m/s^2$	acceleration process noise
w_ω	$3.16 \times 10^{-4} rad/s$	turn rate process noise

Table 4.2: Standard deviation of observation noise.

Parameter	Value	Description
v_{x_lidar},v_{y_lidar}	$0.50m$	position observation noise of LiDAR
v_{x_gps},v_{y_gps}	$3.00m$	position observation noise of GPS
v_{θ_lidar}	$7.07 \times 10^{-3} rad$	heading observation noise of LiDAR
v_{θ_gps}	$4.47 \times 10^{-2} rad$	heading observation noise of GPS
v_{vx_lidar},v_{vy_lidar}	$7.07 \times 10^{-2} m/s$	velocity observation noise of LiDAR
v_{vx_gps},v_{vy_gps}	$0.22m/s$	velocity observation noise of GPS
v_{ax},v_{ay}	$0.80m/s^2$	acceleration observation noise
v_ω	$0.04rad/s$	turn rate observation noise

4.4 Experimental Data

Four typical routes in urban and highway scenarios were selected for the experiments presented in this chapter. The left of Figure 4.5 shows the routes on Nagoya highway (HW) and the right of Figure 4.5 shows the routes in the Nagoya University campus (NU). In both the HW and NU experiments, drives were repeated three and four times, respectively. The statistics of all these drives are summarized in Tables 4.3 and 4.4. The total length of the driving routes was more than 105 km and the total driving time was nearly 2 h. In these drives, most of the commonest driving manoeuvres were performed: car following, overtaking and lane change on Nagoya highway; braking, stop, accelerating and turning on the campus of Nagoya University.

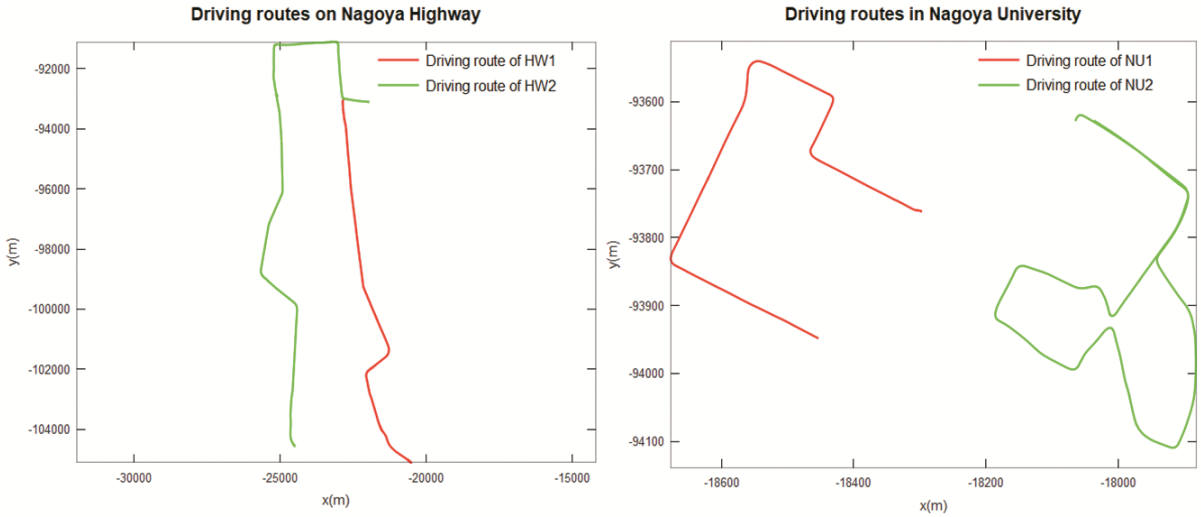


Figure 4.5: Driving routes of experiments.

Table 4.3: Statistics of each drive at Nagoya University.

Drive	Length (m)	Duration	Mean(v) (m/s)	Std(v) (m/s)	Mean(a) (m/s^2)	Std(a) (m/s^2)
NU1-1	981.74	2min44sec	6.1264	2.2486	-0.053812	0.69399
NU1-2	974.05	2min50sec	5.7021	1.7541	-0.066964	0.49773
NU1-3	992.68	2min53sec	5.7723	2.0001	0.034385	0.62067
NU1-4	966.11	3min01sec	5.341	1.8286	-0.074118	0.48275
NU2-1	1733.3	5min19sec	5.4971	2.0158	0.058261	0.47464
NU2-2	1727.5	4min30sec	6.4593	2.4218	0.034729	0.58905
NU2-3	1698.7	5min26sec	5.2544	2.2391	0.046688	0.51144
NU2-4	1746.3	5min01sec	5.8214	2.6551	0.046192	0.62631
TOTAL	10820.38	31min44sec	-	-	-	-

Table 4.4: Statistics of each drive on Nagoya highway.

drive	Length (m)	Duration	Mean(v) (m/s)	Std(v) (m/s)	Mean(a) (m/s2)	Std(a) (m/s2)
HW1-1	13138.77	10min42sec	20.455	3.2057	-0.016844	0.22684
HW1-2	13124.09	12min57ses	16.841	2.5291	-0.07341	0.22356
HW1-3	13122.98	12min24sec	17.67	2.2747	0.0094033	0.181
HW2-1	19010.99	16min44sec	18.942	2.2481	0.028385	0.2221
HW2-2	17794.26	15min35sec	19.025	2.1233	0.059228	0.22881
HW2-3	19253.90	17min42sec	18.121	2.428	0.033483	0.257
TOTAL	95444.98	1h26min04sec	-	-	-	-

4.5 Comparative Evaluation of the Filters' Performance in Vehicular State Estimation

4.5.1 Relative Difference

The most popular indicator for estimation accuracy is root-mean-square error. In this experiment, the relative difference evaluation was provided, instead of the absolute error evaluation. In the NU and HW experiments, the state estimates E1, E2 and E3 are respectively compared with the corresponding baseline E0 to assess their relative difference. (Both the baselines of NU and HW were generated by UKF-CTRA, integrating LiDAR-IMU and GPS-IMU respectively. The configurations are presented in section 4.3.2) The Relative Root Mean Square Error (RRMSE) is formulized as

$$RRMSE(x) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - x_i^{baseline})^2} \tag{4.5-3}$$

The mean RRMSEs of all the state estimates of each route are presented in Tables 4.5∼4.8. From the tables, it can be found that the four filters have not made a significant difference to state estimation (not positioning). UKF and the EKF have roughly identical performance in contrast to the simulation-based experiments where UKF shows a distinct improvement of accuracy (Tsogas et al., 2005; Yang et al., 2017).

Table 4.5: The mean RRMSE of HW1.

Model	CTRV		CTRA	
KF	EKF	UKF	EKF	UKF
Estimate	E1	E3	E2	E0
$x(m)$	0.02094	0.01884	0.03841	0
$y(m)$	0.03227	0.03002	0.01225	0
$\theta(deg)$	0.00289	0.00108	0.00523	0
$v(m/s)$	0.11288	0.11288	0.00044	0
$a(m/s^2)$	-	-	0.00041	0
$\omega(deg/s)$	0.00177	0.00010	0.00278	0

Table 4.6: The mean RRMSE of HW2.

Model	CTRV		CTRA	
KF	EKF	UKF	EKF	UKF
Estimate	E1	E3	E2	E0
$x(m)$	0.05695	0.05487	0.03683	0
$y(m)$	0.05904	0.05671	0.05054	0
$\theta(deg)$	0.00369	0.00148	0.00532	0
$v(m/s)$	0.15800	0.15799	0.00157	0
$a(m/s^2)$	-	-	0.00082	0
$\omega(deg/s)$	0.00291	0.00017	0.00339	0

Table 4.7: The mean RRMSE of NU1.

Model	CTRV		CTRA	
KF	EKF	UKF	EKF	UKF
Estimate	E1	E3	E2	E0
$x(m)$	0.02355	0.01850	0.01011	0
$y(m)$	0.02975	0.02224	0.01493	0
$\theta(deg)$	0.00883	0.00027	0.00877	0
$v(m/s)$	0.34869	0.34869	0.00045	0
$a(m/s^2)$	-	-	0.00192	0
$\omega(deg/s)$	0.00749	0.00011	0.00751	0

Table 4.8: The mean RRMSE of NU2.

Model	CTRV		CTRA	
KF	EKF	UKF	EKF	UKF
Estimate	E1	E3	E2	E0
$x(m)$	0.02471	0.01833	0.01794	0
$y(m)$	0.02887	0.02144	0.01475	0
$\theta(deg)$	0.01280	0.00042	0.01281	0
$v(m/s)$	0.37153	0.37152	0.00053	0
$a(m/s^2)$	-	-	0.00234	0
$\omega(deg/s)$	0.00529	0.00018	0.00549	0

A trend can be found in position estimation that E2 has the smallest relative difference, then E3 and E1 has the largest relative difference, with respect to E0. This matches the conclusions of the existing literature that UKF outperforms EKF (Yang et al., 2017) and CTRA based filters make slightly better position estimates than CTRV based filters (Schubert et al., 2008, 2011). The heading and turn rate estimates of the four filters appear to be exactly the same, which is due to the CTRV and CTRA models having the same constant turn rate hypotheses.

Compared with other state estimates, the velocity estimates of CTRA and CTRV based filters produced relatively obvious differences, in both highway and urban scenarios. The reason is the different velocity assumptions of the two models. The CTRA model takes acceleration into account and the velocity is driven by acceleration; however, the CTRV model assumes the velocity remains constant. In velocity alone, it can be observed that the relative difference of the velocity estimate in urban driving is larger than that of highway driving, which is caused by different driving styles on highway and urban routes. On the highway, drivers prefer to drive the car at an even speed most of the time; whereas, in urban areas, many accelerating/decelerating manoeuvres have to be performed, especially at road intersections. That is why acceleration in urban driving shows higher standard deviations than highway driving (Tables 4.3 and 4.4). The changes in acceleration result in the greater relative difference in velocity estimation. In low dynamic situations (Tables 4.5 and 4.6), the performance of the two models is approximately the same.

The overall results indicate that the motion model has more effect on the filter's performance than the KF form; with the same process model, the accuracy performance of EKF and UKF is almost identical. In high dynamic situations and where the speed estimate is critical, the CTRA model based filter is recommended. Otherwise, the performance of the CTRV model based filter is also acceptable.

The negligible estimation differences among the four filters are explained by the non-severe nonlinearities of the CTRV and CTRA models, which mean that the linearization in EKF does not cause much information loss. Besides, the low dynamics of the vehicle and the high sampling rate of the sensor make the vehicular motion quasi-linear. UKF thus does not have obvious superiority.

4.5.2 Time Efficiency

Limited by the space of this study, in this section, the results of HW1-1 and HW2-1 drives, where GPS worked at 10Hz, are not presented. In the experiments,

when using LiDAR and IMU, a complete iteration consists of 10 predictions, 10 IMU updates and one LiDAR update; for GPS and IMU, a complete iteration consists of five predictions, five IMU updates and one GPS update. This is done by using an Robot Operating System (ROS) time synchroniser (for more information, please see http://wiki.ros.org/message_filters). The experiments were run on a normal personal computer (Inter Core i7-4790 CPU 3.6 GHz, RAM 8G, Windows 10). Each drive's experiment was repeated 10 times. The running time of each estimator was recorded and the average time cost (microsecond) per iteration was calculated and shown in Tables 4.9∼4.12.

In Tables 4.9∼4.12, each row records the time cost of EKF and UKF, using the same model; and each column is the time cost of the filter using identical KF form and different models. The UKF/EKF column records the time cost rate of UKF and EKF with the same model, and the CTRA/CTRV row records the time cost rate of the filters using identical KF form and different models. In particular, the bottom-right cell is the time cost rate of the slowest (UKF-CTRA) and the fastest filters (EKF-CTRV). A clear conclusion can be drawn that, when utilizing the same model, EKF is faster than UKF; and when utilizing the same KF form, the filter using CTRV is faster than the filter using CTRA. For a higher dimension model, EKF is obviously faster than UKF since UKF has to calculate $2n + 1$ sigma points by taking a matrix square root of the state covariance matrix at each iteration and propagating these sigma points at the prediction and update steps.

Table 4.9: Average time cost per iteration of HW1 (GPS@20 Hz).

HW1	EKF	UKF	UKF/EKF
CTRV	9.56097	13.11299	1.37151
CTRA	12.86629	24.83430	1.93018
CTRA/CTRV	1.34571	1.89387	2.59747

Table 4.10: Average time cost per iteration of HW2 (GPS@20 Hz).

HW1	EKF	UKF	UKF/EKF
CTRV	9.66745	12.99528	1.34423
CTRA	13.21840	25.25782	1.91081
CTRA/CTRV	1.36731	1.94362	2.61267

Table 4.11: Average time cost per iteration of NU1 (LiDAR@10 Hz).

HW1	EKF	UKF	UKF/EKF
CTRV	16.14401	21.36358	1.32331
CTRA	22.95781	42.59849	1.85551
CTRA/CTRV	1.42206	1.99398	2.63866

Table 4.12: Average time cost per iteration of NU2 (LiDAR@10 Hz).

HW1	EKF	UKF	UKF/EKF
CTRV	16.84575	22.54865	1.33854
CTRA	23.85105	44.92495	1.88356
CTRA/CTRV	1.41585	1.99236	2.66684

In Tables 4.9 and 4.10, the average time cost of each filter is shorter than that in Tables 4.11 and 4.12. This is caused by the higher update frequency of GPS (20 Hz) compared with LiDAR (10 Hz). Despite that, the UKF/EKF and CTRA/CTRV rates are approximately equal. It is remarkable that the EKF-CTRV is about 2.6 times faster than the UKF-CTRA while their accuracies are almost equal (see Tables 4.55∼4.8). EKF-CTRV may not make a significant improvement in efficiency in most current applications, where just the states of several vehicles need to be estimated. When the amount of vehicles increases substantially, however, it becomes another situation, such as SENSORIS (2019) where data is exchanged between the vehicle sensors and a dedicated cloud. As Figure 4.6 illustrates, the data from vehicles and roadside sensors are gathered into the cloud, where vehicle state estimation is essential for mobility related applications. Such a system has urgent need of real-time processing. In this situation, the EKF-CTRV filter will show its distinguished performance. For example, in the same amount of time, the EKF-CTRV can process information from nearly 2,600 vehicles; however, the UKF-CTRA can only process information from 1,000 vehicles. The efficiency of the system can be greatly improved, with negligible accuracy loss.

In summary, for a specific application, an ideal filter is composed of the most appropriate motion model and KF form, taking both accuracy and efficiency into

consideration. In vehicular state estimation, EKF and UKF have not made distinct differences in accuracy; however, in the efficiency aspect, EKF is faster than UKF. The state estimation performance of the CTRV and CTRA model based filters is roughly identical.

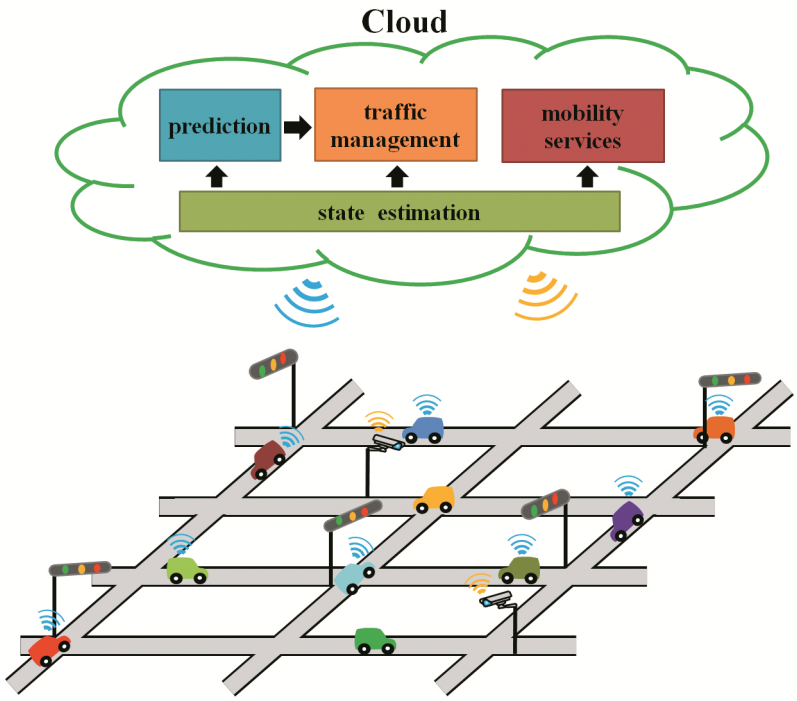


Figure 4.6: Dedicated cloud for vehicle and road side sensor.

4.6 Comparison of the Models' Performance in Motion Prediction

To investigate the models' properties and the affecting factors in motion prediction, the predicted path is defined:

$$\begin{aligned} &path = \{\mathbf{p}_{t_0}, \mathbf{p}_{t_1}, \cdots, \mathbf{p}_{t_n}\}, \\ &\text{where } \mathbf{p}_{t_i} = [x, y]^T = \begin{bmatrix} 1 & 0 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \end{bmatrix} \mathbf{f}(\mathbf{x}_{t_{i-1}}), t_i \in \{0, 0.1, \cdots, 3.0\}. \end{aligned} \tag{4.6-4}$$

where the path consists of sequential position points predicted by the kinematic model $\mathbf{f}(\cdot)$ (CTRV or CTRA). Given an initial state $\mathbf{x}_{t_0}$, a path can be generated

iteratively via Equation 4.6-4.

The Relative Average Euclidean Error (RAEE) at instant t_i of a certain $PATH$ defined in Equation 4.6-5 is used to measure the path's reliability evolution, where $path_j$ denotes the j-th predicted path; N is the number of predicted paths and $PATH \in \{E1P1, E3P1, E2P1, E0P1, E2P2, E0P2\}$.

$$RAEE(t_i, PATH) = \frac{1}{N}\sum_{j=1}^{N}\left\|\mathbf{p}_{t_i}^{path_j} - \mathbf{p}_{t_i}^{baseline}\right\|_2, \mathbf{p}_{t_i}^{path_j} \in path_j, path_j \in PATH \tag{4.6-5}$$

Since the baseline estimated based on GPS measurements still retained residuals of several meters, for reasons of caution, the motion prediction experiment was not conducted on the highway drives. In contrast, LiDAR provided sub-metre positioning results, at least (R. Liu et al., 2020). In this section, just the experiments at Nagoya University are discussed.

The evolution of RAEE over the time of each drive on routes NU1 and NU2 is presented in Figures 4.7 and 4.8, respectively. In the figures, the RAEE evolution of a certain $PATH$ is represented by a marked line with different color. The red and green lines indicate CTRV and CTRA based prediction, respectively. The triangular and circular marks, respectively, indicate that the EKF or UKF estimate is used in the prediction. The color of the mark indicates the process model used in state estimation; red is CTRV and green is CTRA.

In Figures 4.7 and 4.8, there are three outstanding characteristics: (1) the red lines with red marks have the largest RAEE, which means the paths predicted by CTRV using the state estimates of CTRV-based filters have the largest relative difference; (2) the triangular and circular marks are superposed, which means the KF form (EKF or UKF) barely affects the path prediction; (3) all the RAEEs increase exponentially over time. It can be inferred that (1) is caused by the different initial velocity estimates because the red lines with green marks have the lowest RAEE and the only obvious differences between them are the initial velocities, see Tables 4.7 and 4.8. The reason for (2) is the negligible estimate

difference between identical-model based EKF and UKF, see Tables 4.7 and 4.8. The cause of (3) is that the prediction, in fact, is an open-loop KF without any updates, thus the prediction becomes divergent over time.

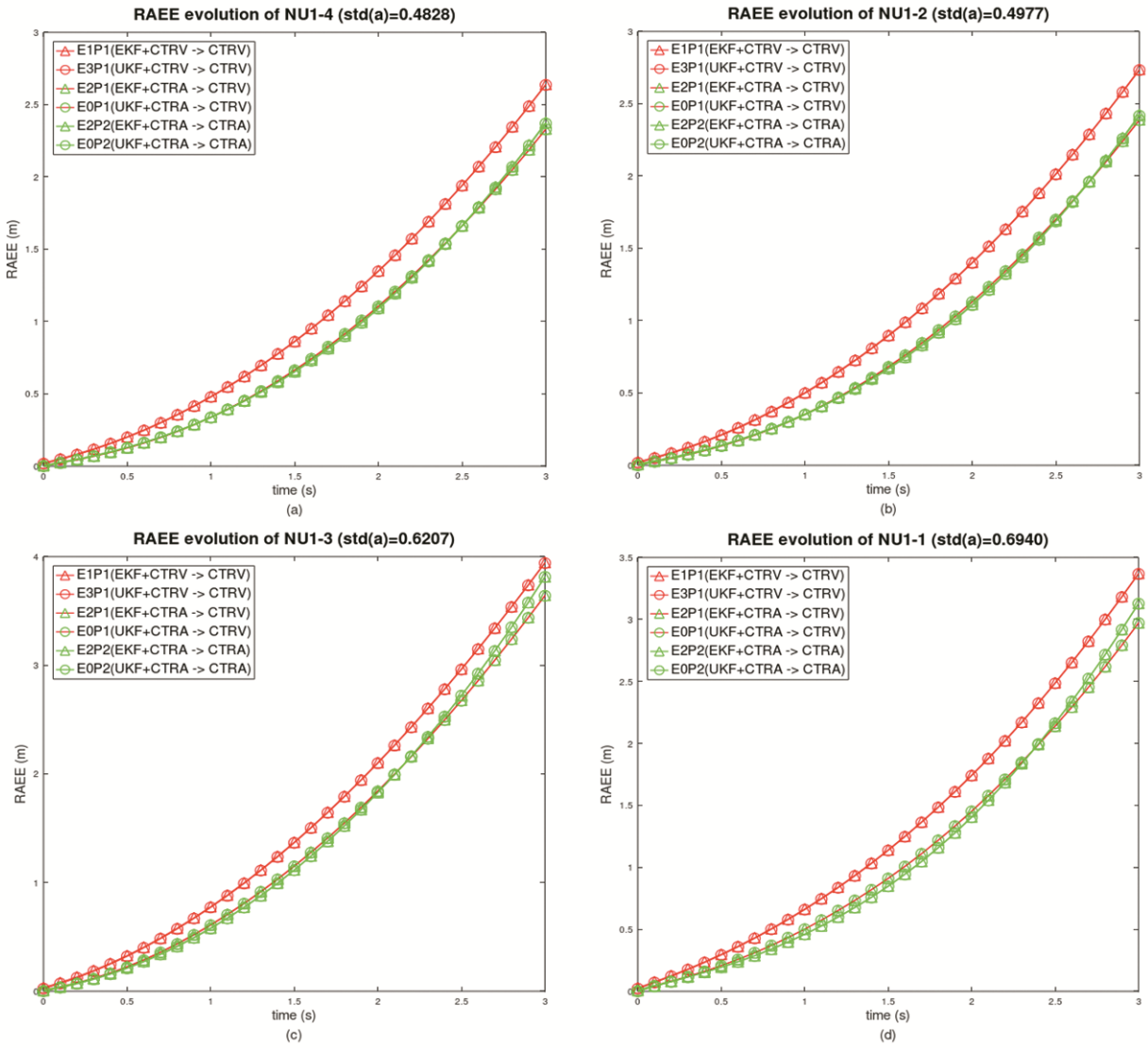


Figure 4.7: RAEE evolution over time of each drive on route NU1.

It can be observed in NU1-4 and NU1-2 drives [Figures 4.7(a) and 4.7(b)] that the CTRV and CTRA models have almost identical performance. This is because the acceleration was steady in the drives [low standard deviation of acceleration (std-a)] which results in low standard deviation of velocity (std-v), see Table 4.3. In such condition, the vehicle was likely to move uniformly; therefore, the two models produced almost identical predictions. However, with low std-a, in NU2-1 and NU2-3 drives [Figures 4.8(a) and 4.8(b)], CTRV predicted better than CTRA.

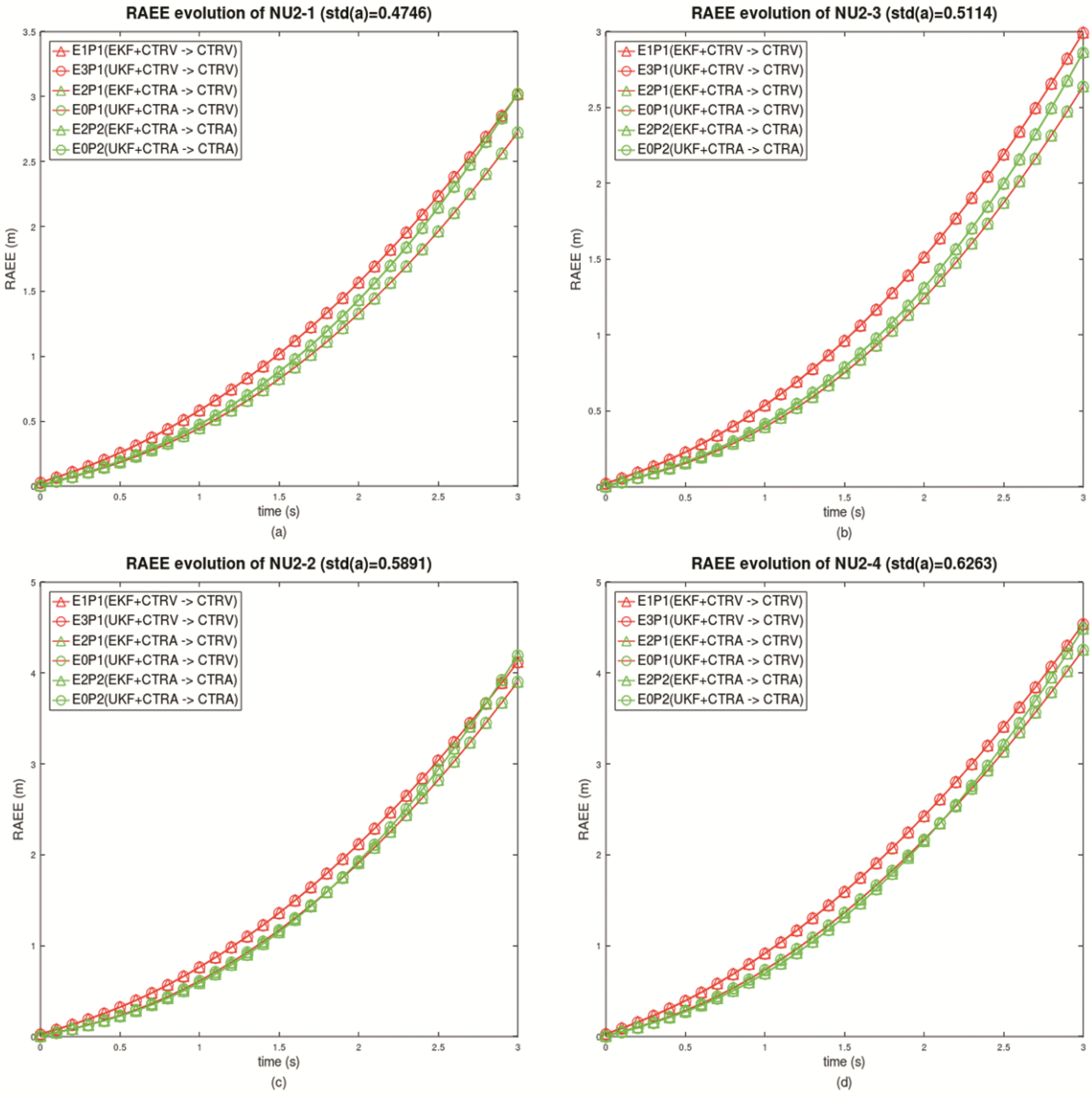


Figure 4.8: RAEE evolution over time of each drive on route NU2.

The reasons are associated with the road conditions. The NU2 route is the most curved and sloped, so it is harder for the driver to keep a constant acceleration than a constant velocity on NU2. Therefore, the constant velocity hypothesis of CTRV is more realistic than the constant acceleration hypothesis of CTRA. It can be summarized that in low std-a situations (low dynamics) CTRV model predicts more reliably. With std-a increasing, RAEE of the CTRA prediction decreased with respect to that of CTRV at the beginning (Figures 4.7 and 4.8, (c) and (d)]; however, RAEE of the CTRA prediction became larger than that of CTRV about 2 s later. This reveals the fact that continuous long-term accelerating is not permitted on the Nagoya University campus. It can be inferred that the CTRA model predicts more reliably when the vehicle has high dynamics [Figures 4.7(d) and 4.8(d)].

From the experiments, some conclusions can be drawn. Firstly, the reliability of predicted path decreases over time; a standalone model should not be used for long-term motion prediction. Secondly, velocity estimate affects path prediction. Thirdly, the model that matches with the vehicle's driving status predicts better; the prediction model should be chosen dynamically considering the vehicle's behavior, as done by Lytrivis et al. (2011).

4.7 Future Work

The experiments showed that a standalone motion model could not make a reliable prediction over a long time. In order to make a reliable prediction, some additional information should be taken into account in future study, such as the uncertainty in motion prediction, map and historical driving data.

4.8 Conclusions

In this chapter, a comparative study was carried out to compare the accuracy and efficiency performance of UKF and EKF with different motion models, in

vehicular state estimation. The models' properties and the factors affecting motion prediction have also been investigated. The experimental results indicated that UKF and EKF showed roughly the same accuracy; the only obvious difference occurs in velocity estimation which is caused by different velocity hypotheses of the CTRA and CTRV models. However, they differed significantly in efficiency. With an identical process model, EKF works faster than UKF; with identical KF form, the filter using CTRV is faster than that using CTRA. The fastest filter, EKF-CTRV, is about 2.6 times faster than the slowest, UKF-CTRA. For the application that needs to process mass data with strict real-time demands, EKF-CTRV might be an ideal choice. The velocity estimate and the motion model used affect the reliability in motion prediction. A realistic model that reflects the real driving status generates a reliable path.

CHAPTER 5

UNCERTAINTIES IN VEHICLE MOTION PREDICTION AND THEIR APPLICATION IN COLLISION RISK ASSESSMENT BASED ON DYNAMIC MAP

This chapter mainly focuses on coping with the uncertainties in middle-term vehicle motion prediction. In this chapter, we firstly define the associated uncertainties in vehicle motion prediction and then introduce a novel deterministic sampling method to quantify the uncertainties; following that, so-called sigma trajectory is proposed to make use of the uncertainties in a novel collision risk assessment application based on DM. Finally, the proposed methods and application are validated and evaluated in real world experiments.

5.1 State and Motion Uncertainties

In order to predict vehicle motion, knowledge about vehicles' current state and vehicular kinematics are needed. However, in reality, this knowledge cannot be fully realized due to the following two major inherent uncertainties.

1) State uncertainty: vehicles' real-time state cannot be fully known because state measurements are provided by sensors, which inevitably suffer from noise interference and no technology can eliminate all noises. The state uncertainty is derived from the measurement functions:

   For linear system: $\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k$ (Equation 3.2-80)

   For nonlinear system: $\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k$ (Equation 3.2-89)

from which, it can be clearly observed that the measurement inherently contains random noise.

2) Motion uncertainty: no model can perfectly describe vehicles' motion and predict future motions. On the one hand, a perfect vehicular motion model can never be built. On the other hand, the uncertainty propagates and accumulates over time. They can be explained through the following system models and covariance propagation equations:

$$\text{For linear system:} \begin{cases} \mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \omega_{k-1} \text{ (Equation 3.2-79)} \\ \mathbf{P}_k^- = \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1} \text{ (Equation 3.2-84)} \end{cases}$$

$$\text{For nonlinear system:} \begin{cases} \mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \omega_{k-1} \text{ (Equation 3.2-108)} \\ \mathbf{P}_k^- = \mathbf{J_f}(\hat{\mathbf{x}}_{k-1}^+)\mathbf{P}_{k-1}^+\mathbf{J_f}^T(\hat{\mathbf{x}}_{k-1}^+) + \mathbf{Q}_{k-1} \text{ (Equation 3.2-113)} \end{cases}$$

5.2 A Comparison of Different Methods in Dealing with Uncertainty in CCWS

In this chapter, we quantify and leverage the state and motion uncertainties to assess vehicle collision risk and build a novel CCWS. A comparison between our system and some typical CCWSs is presented in Table 5.1.

Table 5.1: Comparison of our system and some CCWSs: (1) Tan & Huang (2006); (2) Sengupta et al. (2007); (3) Tu & Huang (2010); (4) Hafner et al. (2013); (5) Joerer et al. (2013); (6) Gómez et al. (2016); (7) X. Xu et al. (2018); (8) Zhao et al. (2019).

Method	Architecture	Considered Uncertainty	Predictive/ Non-predictive	Risk Indicator	Experiment
(1)	vehicle-based	state uncertainty	predictive	TTC	real, simulated
(2)	vehicle-based	none	non-predictive	distance	real
(3)	vehicle-based	none	predictive	TTC	simulated
(4)	vehicle-based	state uncertainty	non-predictive	bad set	real
(5)	vehicle-based	state and motion uncertainties of acceleration	predictive	probability	simulated
(6)	vehicle-based	none	non-predictive	distance	simulated
(7)	fog-node-based	none	predictive	distance	real
(8)	vehicle-based	state uncertainty of position	non-predictive	distance	real
Ours	server-based	state and motion uncertainties of multiple aspects	predictive	probability, TTC, conflict points	real

In the table, the most distinct differences are: (1) the architecture of our system is server-based; different from the present systems that are implemented in local vehicle, our system is implemented in a cloud/edge server based on DM. (2) our method consider the state and motion uncertainties of multiple aspects in vehicle motion, whereas the present methods neglect part or whole of the uncertainties. (3) there are more risk indicators in our system, reflecting spatial, temporal and probabilistic collision risk; and our experiments are conducted in real world. In following section, the detailed framework and architecture of our system are introduced.

5.3 Framework and Architecture of Proposed System

5.3.1 DynamicMap2.0

Our system is assumed to be based on Dynamic Map 2.0 platform (DM2.0PF), which is proposed as a logical city-level dataset that allows the overlaying of sensor data onto a HD map and is seen as the next-generation road map (Watanabe et al., 2020). Since 2016, DM2.0PF has been studied and developed by Dynamic Map 2.0 consortium, which consists of several universities and companies in Japan (NCES, 2019). DM2.0PF comprises embedded devices, edge servers and cloud servers, as shown in Figure 5.1.

Through the cloud/edge/embedded systems and collaboration between these systems, DM2.0PF can process a large volume of traffic data, as well as meet real-time demands. More details of DM2.0PF, please refer to section 1.2.

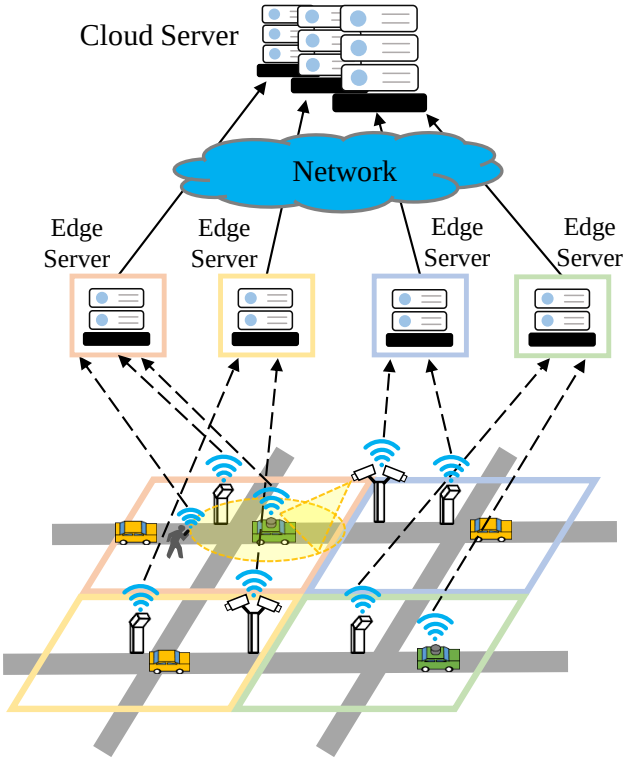


Figure 5.1: The cloud/edge/embedded systems of DM and their collaboration.

5.3.2 Target environment

We aim to provide CRA service for connected vehicles to enhance road safety in the IoT era. Our target operating environment is, therefore, hypothesized as follows:

(1) DM2.0PF is deployed in vehicles, edge servers and cloud servers. Massive traffic data are managed by DM2.0PF, and the platform can appropriately allocate tasks to avoid problems such as overloads. The communications between vehicles, edge servers and cloud servers are also handled by DM2.0PF; devices embedded with DM2.0PF will follow DM2.0PF schedule to avoid problems such as channel congestion and communication/computation contention.

(2) All vehicles have been equipped with communication devices, and the connections between them are well-maintained.

(3) Any vehicle can only be involved in one collision at a certain moment.

5.3.3 Architecture of the Proposed System

The proposed server-based architecture is shown in Figure 5.2. Our CRA is provided as a service through a dedicated server. There are two obvious advantages: (1) the entire system benefits from powerful computing resources on the server; (2) all connected vehicles benefit from various data contents that are available on the server.

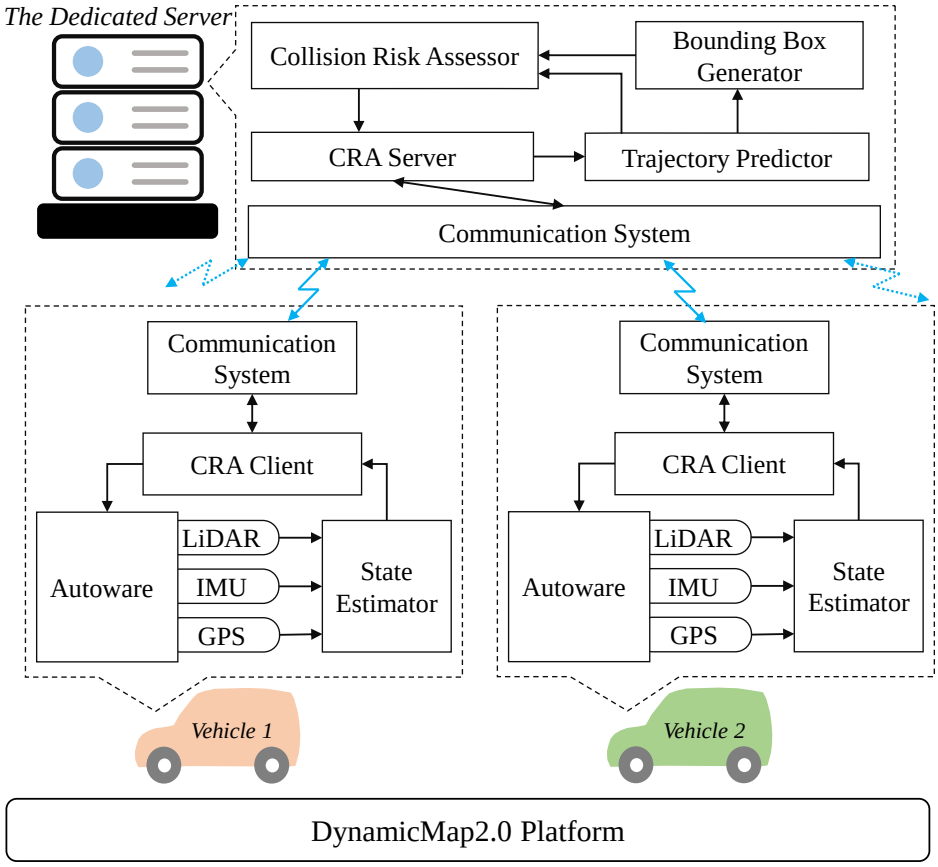


Figure 5.2: The proposed server-based architecture for CCWS

Autoware and Sensors

Multiple sensors are installed on *Autoware* for environment perception. In this system, LiDAR, IMU, and GPS are utilized. For most CCWSs, positioning accuracy should be within 1 m, and a further 0.5 m can produce significantly better performance, especially for blind spot warning (Shladover & Tan, 2006).

Hence, the LiDAR and IMU are to measure SV's state; GPS provides absolute time reference for data synchronization. Refer to (Kato et al., 2015, 2018) for more information on the sensors and *Autoware*.

Vehicular State Estimator

In our system, SRUKF (Van Der Merwe & Wan, 2001) is employed to provide vehicle state estimate and corresponding covariance at 10 Hz. These data are transmitted to the CRA server for trajectory prediction.

Communication System

In our experiments, vehicles connect to the dedicated server through Wi-Fi and Long Term Evolution (LTE) networks. In our system, an Azure cloud server is employed. A Virtual Private Network (VPN) is created in the cloud server for secure connections, using VNS3. The communication system is built based on the ROS multi-master system proposed by Juan & Cotarelo (2015). Similar configurations can be found in the paper of Hussein et al. (2017).

Collision Risk Assessment Client and Server

The CRA client and server are designed based on ROS service (AnisKoubaa, 2019) and respectively run in vehicles and the dedicated server. The CRA client prepares a request message and sends it to the server. The CRA server requests for trajectory predictions and CRA, and sends results to vehicles. In our experiments, the CRA client requests for CRA service as soon as a vehicle state estimation is completed. In the future, the timing of the CRA service call should be determined through negotiation between vehicles and the CRA server to avoid problems, such as channel congestion.

Trajectory Predictor and Bounding Box Generator

It is difficult to predict the true trajectory of a vehicle, even for a few seconds. Conversely, it is easier to predict a possible trajectory set that contains possible true trajectories. Computing the spatiotemporal relationships among such trajectory sets is computationally expensive. Thus, a bounding box generator is employed to build a bounding box for each trajectory set to speed up computation. Collision risk assessment is performed only when the trajectory sets' bounding boxes are intersecting.

Collision Risk Assessor

The collision risk assessor that runs on the dedicated server identifies potential crashes based on the predicted trajectories. The details are explained in the next section.

5.4 Coping with the Uncertainties

5.4.1 SRUKF State Estimator

A square-root UKF, which has been introduced in section 3.2.3 in chapter 3, is adopted to reduce the state uncertainty firstly. There are two reasons to choose the SRUKF here: (1) SRUKF can prevent numerical instability in UKF (Van Der Merwe & Wan, 2001). (2) the square root of state estimation covariance can be directly used in sigma trajectory generation, avoiding the calculation of that for each prediction process.

In the SRUKF, CTRA model (see section 3.1.4 in chapter 3) is selected to describe the vehicle motion. The state and measurement vectors of our system are as follows.

$$\mathbf{x} = \begin{bmatrix} x & y & \theta & v & a & \omega \end{bmatrix}^T \tag{5.4-1}$$

$$\mathbf{y} = \begin{bmatrix} x & y & \theta & v_x & v_y & a_x & a_y & \omega \end{bmatrix}^T \tag{5.4-2}$$

where x and y are position coordinates; θ is heading; v is velocity; a is acceleration; ω is yaw rate; v_x, v_y, a_x, and a_y are respectively horizontal and vertical velocity and acceleration.

5.4.2 Data Management on the Dedicated Server

As soon as a state estimation is completed, the request message in Equation 5.4-3 is prepared and sent to the CRA server by the CRA client.

$$req_k = \{ID_{SV}, t_k, \hat{\mathbf{x}}_k, \mathbf{S}_k\} \tag{5.4-3}$$

where ID_{SV} denotes SV's identification; t_k is the current time; $\hat{\mathbf{x}}_k$ (the same as $\hat{\mathbf{x}}_k^+$) is the a posteriori real-time vehicle state estimate produced by SRUKF algorithm; $\mathbf{S}_k$ is the square root of the state estimation covariance, see Equation 3.2-131.

As illustrated in Figure 5.3, suppose vehicle A sends a request, the CRA server then identifies the vehicle to be in an active vehicle database. If the vehicle is new, it will be inserted into the database; otherwise, the vehicle's old record will be updated.

Next, vehicles that can collide with vehicle A are approximately queried through DM2.0PF using certain rules, such as, are the lanes the vehicles are on potentially conflicting. The $\hat{\mathbf{x}}$ and $\mathbf{S}$ of all potential collision vehicles, in Equations 5.4-4 and 5.4-5, are then provided to the sigma trajectory predictor.

$$\mathbf{SC}^A = \begin{bmatrix} \hat{\mathbf{x}}^A & \mathbf{S}^A \end{bmatrix} \tag{5.4-4}$$

$$SC^{pcv} = \left\{\mathbf{SC}^B, \mathbf{SC}^C, \cdots, \mathbf{SC}^X\right\} \tag{5.4-5}$$

where $\mathbf{SC}$ is the state and square-root of state covariance. SC^{pcv} is the $\mathbf{SC}$ set of

the potential collision vehicles.

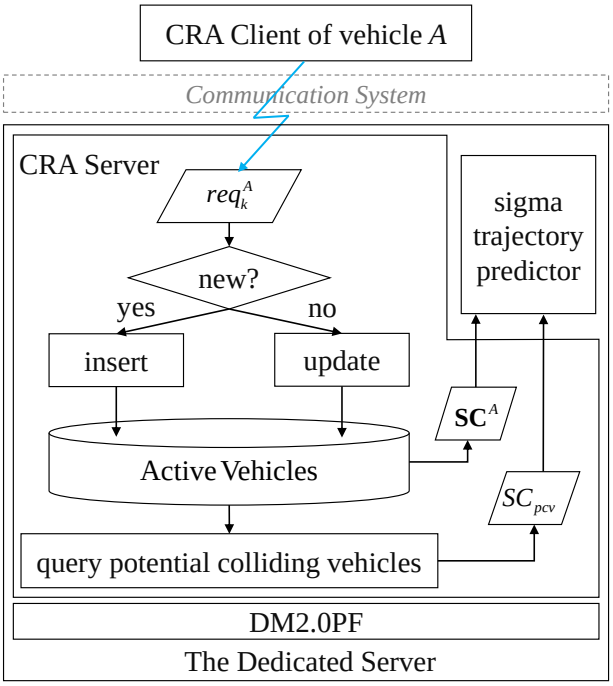


Figure 5.3: Data management on the dedicated server based on DM2.0PF

5.4.3 Quantification of Uncertainties: Current-state-centered Multi-dimensional Sampling

A sampling method, such as Monte Carlo sampling, is an effective method to address uncertainty. As mentioned above, in the work of Joerer et al. (2013), non-acceleration uncertainties were omitted. To tackle this problem, a current-state-centered multidimensional sampling method is proposed. Figure 5.4 illustrates the difference between different approaches to address uncertainties.

A vehicle cannot move randomly; any behavior of a vehicle in imminent future is related to current behaviors of the vehicle, as the vehicle's velocity at the next second cannot change stochastically, but the vehicle can increase or decrease its velocity. Intuitively, motion uncertainty can be approximately inferred from a vehicle's current state and corresponding uncertainty. A plausible way is to magnify the uncertainty around the current state, thus incorporating the state and motion uncertainties. As illustrated in Figure 5.4 (the right), the state uncertainty distributes around vehicles' current state (the blue sample), and the motion

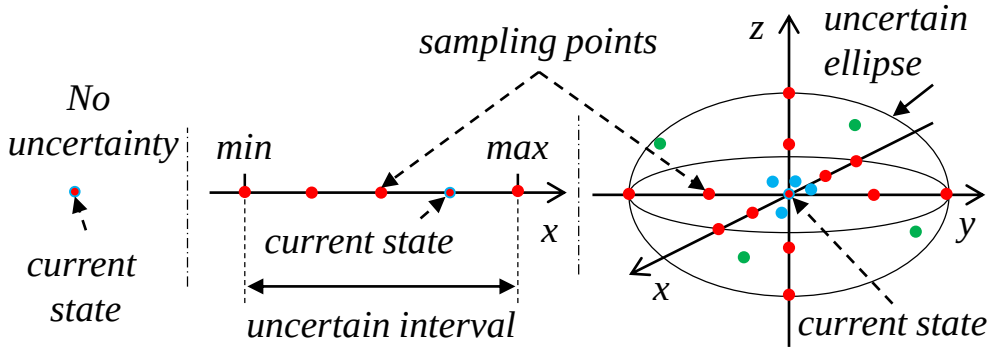


Figure 5.4: Comparison of different methods for addressing uncertainties. The left: no uncertainty was considered, as demonstrated by Tu & Huang (2010); Miller & Huang (2002); X. Xu et al. (2018). The center: only one-sided uncertainty was considered; different behaviors associated with a certain aspect, like acceleration, were considered, as demonstrated by Joerer et al. (2013). The right: the proposed current-state-centered multidimensional sampling method in which different behaviors associated with different aspects are considered.

uncertainty may distribute anywhere within the ellipse (the green sample). The uncertain ellipse includes the state and motion uncertainties, and its characteristics can be approximately captured through deterministic sampling using the red samples. For a vehicle whose $\mathbf{SC} = \begin{bmatrix} \hat{\mathbf{x}} & \mathbf{S} \end{bmatrix}$, the uncertainty is magnified by

$$\mathbf{U} = \mathbf{S} \times \mathbf{M} \tag{5.4-6}$$

where $\mathbf{M}$ is the diagonal magnifying matrix, and $\mathbf{U}$ is the magnified uncertainty matrix. The matrix $\mathbf{M}$ controls the shape of the uncertain ellipse. The larger the values in $\mathbf{M}$, the more uncertainties are incorporated. It is better to determine $\mathbf{M}$ based on vehicles' properties; however, for simplicity, the values of diagonal elements in $\mathbf{M}$ are set to 100 in our experiments. The samples are drawn from the ellipse deterministically using Equation 5.4-7, and they are called magnified sigma points. To be noticed, the associated uncertainties in vehicle motion prediction is also quantified by this equation.

$$\mathbf{MSP} = [\hat{\mathbf{x}} \pm psp(\mathbf{U})_{vsp}] \tag{5.4-7}$$

where the parameters psp and vsp control sampling diversities. $0 \leq psp \leq 1$ is the

possibility sampling parameters; to model more possible vehicular motions, more *psp* should be chosen; however, this will incur more computing time during collision detection. In this study, $psp \in \{0, 0.5, 1\}$. $vsp \in \{0, 1, 2, \cdots, 6\}$ is the variety sampling parameters. $(\mathbf{U})_{vsp}$ is the vsp-th column of matrix $\mathbf{U}$, and we defined $(\mathbf{U})_0 = \mathbf{0}$. To consider more uncertainty aspects, more vsp should be considered. Either $vsp \in \{0\}$ or $psp \in \{0\}$ means that no uncertainty is considered, and this is the left case in Figure 5.4. If $vsp \in \{5\}$, the behaviors associated with acceleration are modeled, and this is similar to the center case in Figure 5.4. In this study, $vsp \in \{3, 4, 5, 6\}$, different behaviors associated with velocity, acceleration, heading, and yaw rate are modeled. Any system with poor positioning accuracy should consider additional position uncertainty (let $vsp \in \{1, 2, \cdots, 6\}$).

5.5 Making Use of the Uncertainties in Collision Risk Assessment

5.5.1 Sigma Trajectory Generation

A crash indicates two vehicles are in the same location at the same time. This means, for a crash to happen, conflicts must have occurred in spatial and temporal domains. To facilitate collision detection, a vehicle's trajectory is defined as a sequence of tuples that compose of spatial position and time, as follows.

$$tra^j = \{(x, y, t)_k^j\}, k = 0, 1, \cdots, N_p; j = 1, 2, \cdots, N_t \tag{5.5-8}$$

where tra^j denotes the j-th trajectory, and $(x, y, t)_k^j$ denotes the k-th trajectory point of the j-th trajectory; N_p is the number of predicted trajectory points, and N_t is the number of predicted trajectories. In this study, $N_t = 17$.

As reported by Lytrivis et al. (2011); Houenou et al. (2013); Schubert et al. (2011), the CTRA model that is introduced in section 3.1.4 of chapter 3 can predict vehicle trajectory with ideal accuracy. Therefore, using the CTRA model and **MSP** matrix, all sigma trajectories are generated by Algorithm 1, where

$(\mathbf{MSP})_i$ is the i-th column of $\mathbf{MSP}$. In this study, the prediction time is 5 s with interval $T = 0.1$ s and $N_p = 50$.

Algorithm 1 Sigma trajectory generation algorithm

$i \leftarrow 1$
while $i \leq N_t$ **do**
  $\mathbf{x}_0 = (\mathbf{MSP})_i$
  $(x, y, t)_0^i = \left[\left[\begin{bmatrix} 1 & 0 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \end{bmatrix} \mathbf{x}_0\right]^T, 0\right]$
  $k \leftarrow 0$
  while $k \leq N_p$ **do**
    $\mathbf{x}_{k+1} = \mathbf{F}(\mathbf{x}_k)$
    $[x, y]_{k+1}^T = \begin{bmatrix} 1 & 0 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \end{bmatrix} \mathbf{x}_{k+1}$
    $(x, y, t)_{k+1}^i = ([x, y]_{k+1}, (k+1)T)$
    $k \leftarrow k + 1$
  end while
  $i \leftarrow i + 1$
end while

5.5.2 Bounding Box Generation

To accelerate collision detection, the bounding box of the sigma trajectory set is generated and stored. For instance, for vehicle A, these data are formulated as

$$vehicle_A = \left\{TRA^A, BB^A\right\}, \text{where } TRA^A = \left\{tra^j\right\} \text{ and } j = 1, 2, \cdots, N_t \tag{5.5-9}$$

where TRA^A and BB^A respectively denote vehicle A's sigma trajectory set and corresponding bounding box.

5.5.3 Collision Risk Assessment Methods

Suppose vehicle A requires CRA service, and vehicle B's bounding box, BB^B, is intersecting with vehicle A's bounding box, BB^A.

Conflict of Trajectories

From an engineering viewpoint, two trajectories are conflicting if the following statement is true.

$$\exists(x,y,t)^i \in tra^i, \exists(x,y,t)^j \in tra^j(\|(x,y)^i - (x,y)^j\|_2 \leq th_s \wedge \|t^i - t^j\| \leq th_t) \tag{5.5-10}$$

where th_s and th_t denote the threshold of spatial and temporal distances, respectively. The function in Equation 5.5-11 determines whether two trajectories collide.

$$\text{conlict}(tra^i, tra^j) = \begin{cases} 1, & \text{Equation 5.5-10 is true} \\ 0, & \text{otherwise.} \end{cases} \tag{5.5-11}$$

Collision Risk Indicators

The probability, TTC, and CPs are used to describe the probabilistic and temporal and spatial criticality of a collision.

For vehicles A and B, their collision probability is the accumulation of conflict probabilities of trajectory pairs that belong to TRA^A and TRA^B, as expressed in Equation 5.5-12.

$$\begin{aligned} p = \sum_{i=1}^{N_t^A} \sum_{j=1}^{N_t^B} p_t(tra^i) p_t(tra^j) \text{conflict}(tra^i, tra^j), \\ \text{where } tra^i \in TRA^A, \text{ and } tra^j \in TRA^B \end{aligned} \tag{5.5-12}$$

in which p is the collision probability; $p_t(tra^i)$ is the probability that tra^i is true. The trajectories of vehicles A and B are assumed as independent, which is plausible unless a driver realizes the potential hazard and changes the vehicle's status.

At instant t_k, the TTC and CPs are calculated by

$$TTC_k = \min(t_j^i), CP_k = \cup(x_j^i, y_j^i) \text{ where } (x,y,t)_j^i \in tra^i, \text{ and } tra^i \in TRA \tag{5.5-13}$$

where $(x, y, t)^i_j$ is the j-th conflict point of TRA; (x^i_j, y^i_j) is the conflict position. The response message in Equation 5.5-14 is sent to vehicle A to assist in collision avoidance.

$$res_k = \{t_k, p_k, TTC_k, CP_k\} \tag{5.5-14}$$

Probability Density Function of Sigma Trajectory

Hereto, the only problem is how to determine the value of $p_t(tra^i)$. The proposed Probability Density Function (PDF) of $p_t(tra^i)$ is shown in Figure 5.5. The PDF is based on the hypotheses that for safety and comfort, a vehicle always moves steadily and smoothly, which means the vehicle will not suddenly accelerate/decelerate or swerve; moreover, current state estimate is most likely to be true, and thus the trajectory generated by it is most likely to be a true trajectory.

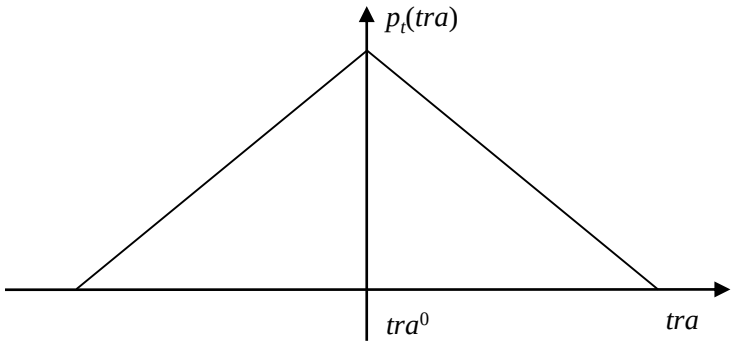


Figure 5.5: Symmetric triangular probability distribution of sigma trajectory.

5.6 Collision Risk Assessment Application: Experiments and Discussions

5.6.1 Experiment Setup

Both the methods proposed by us and Joerer et al. (2013) were implemented and evaluated using C++.

Experimental Vehicle and Sensors

The experiments were conducted using the Toyota PRIUS PHV where multiple sensors and *Autoware* were mounted, as shown in Figure 5.6c. In our experiments, the vehicle was driven by a human driver to ensure safety. The LiDAR (Velodyne HDL-64ES3), IMU (Xsens MTi-300) and GPS (Trimble NetR9) were utilized.

Experimental Field

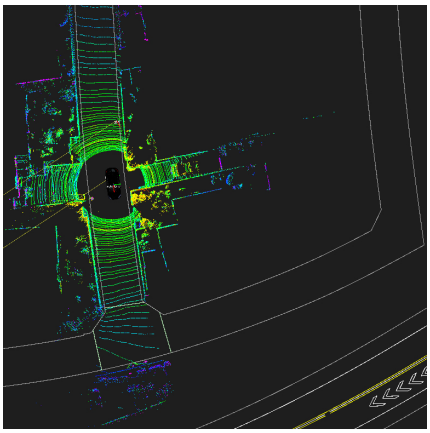
The experiments were performed at a public road (Figure 5.6d) in Kasugai city, Aichi prefecture, Japan. The experimental field is an uncontrolled blind intersection where dense buildings severely impair sensors' perception. As shown in Figures 5.6a and 5.6b, although the vehicle approached the intersection, the laser lights of the LiDAR were blocked, and this made vehicles to be hidden in the blind spot of each other. In such NLoS scenario, CCWSs are critically used to ensure safe driving.

Outdoor Data collecting

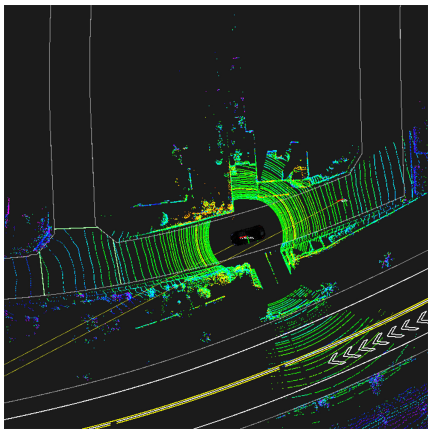
The outdoor experiments are demonstrated in Figure 5.6. We drove the vehicle on major and minor roadways and recorded all the messages outputted by *Autoware* into rosbags (JochenSprickerhof, 2015). The drive on the major and minor roadways was repeated three times, respectively, and a total of six rosbags were collected. These drives' kinematic statistics when approaching the intersection are presented in Table 5.2, and the trajectories are plotted in Figure 5.6d. Based on these six drives, nine crashes could occur.

Table 5.2: The kinematic statistics of the six drives

drive	mean v(m/s)	std v(m/s)	mean a(m/s^2)	std a(m/s^2)
major1	6.07010	0.86737	0.18034	0.40410
major2	7.07252	0.92781	0.30096	0.47941
major3	6.99085	0.91295	0.30722	0.39768
minor1	5.48219	1.70387	-0.15965	0.48760
minor2	5.57898	1.68495	-0.21417	0.44611
minor3	5.44822	2.04298	-0.14685	0.55309



(a)



(b)

(c)

(d)

Figure 5.6: Outdoor experiments. (a) The vehicle approaches the intersection from the minor roadway. (b) The vehicle approaches the intersection from the major roadway. (c) Experimental vehicle. (d) The six drives' trajectories

Indoor Experiments

[1] As illustrated in Figure 5.7, the rosbags were replayed in two computers to duplicate real-world driving situations when two vehicles were approaching the intersection. Meanwhile, the state estimator, CRA client, and *Autoware* were run in the two computers; our method's CRA program and that of referenced method were run on a Microsoft Azure Virtual Machine (VM). The vehicle and the dedicated server were connected through Wi-Fi (802.11ac)/LTE through the VNS3 VPN. For delay-critical applications, edge computing should be utilized (Zhuang et al., 2019; Shi et al., 2016). Regarding our system, deploying the dedicated server on the edge side of DM2.0PF is preferable; however, we leave this for future work.

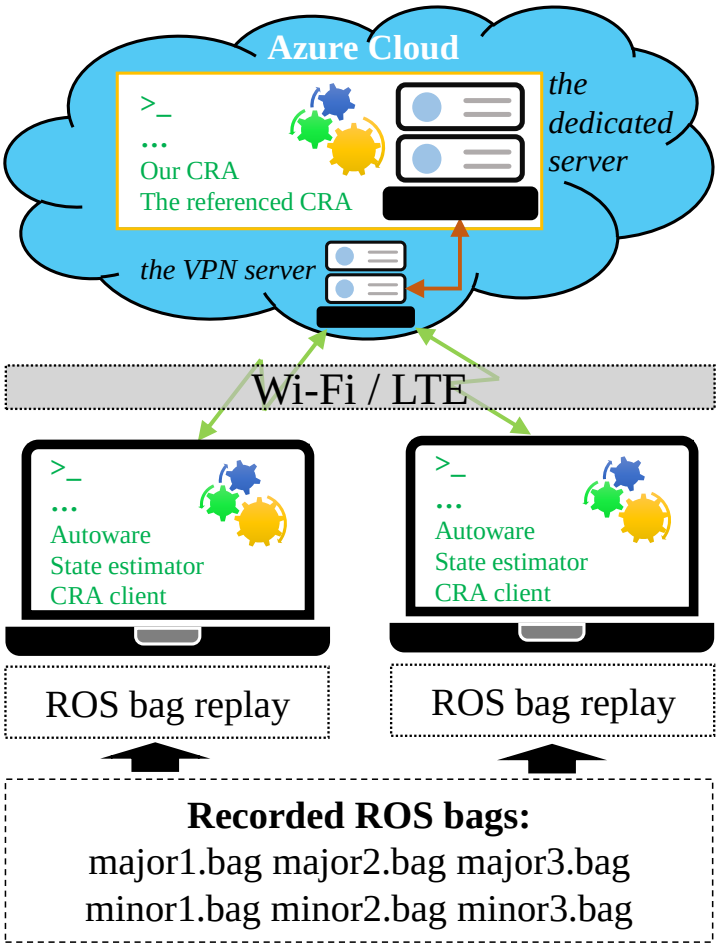


Figure 5.7: Indoor experiments.

[1] A demo of our system is available at https://youtu.be/YbeMhmsWTE4.

Collision Indicators

The distance between two vehicles was used to indicate whether a crash occurred. Our experimental scenario was a typical angle crash, therefore the following threshold was used.

$$th_s = 0.5(l_v + w_v) + e_p \tag{5.6-15}$$

where l_v and w_v denote the length and width of a vehicle. e_p is the positioning error. In our experiments, th_s = 3.3 m; the instant the distance between two vehicles is closest to th_s is regarded as a crash moment.

5.6.2 Experimental Results

Our experimental results are compared with that of the referenced method to evaluate the performance of our method. Each crash test was repeated three times, using Wi-Fi and LTE connection, respectively. In total, 54 tests were carried out. A crash test is notated with its major and minor roadway drives and test number.

Collision Probability Estimation

Probability is an important indicator for estimating potential collision hazard. An accurate and early collision probability estimate helps to adequately warn drivers/vehicles and ensure there is sufficient time for collision avoidance. As reported by Joerer et al. (2013), a probability value of 50 % could indicate no-crash and crash groups. In this study, we assume that when the collision probability is greater than 50 %, a collision is detected, and a warning message is given. The metric Advance Collision Detection Time (ACDT) defined in Equation 5.6-16 is used to evaluate the methods' performance in collision probability estimation. On the other hand, ACDT reflects the methods' survivability against latency and dropouts; the greater the ACDT, the more robust the methods.

$$ACDT = t_c - t_d \tag{5.6-16}$$

where t_c and t_d respectively denote the time a crash occurs and is detected. As shown in Figure 5.8, the major3-minor1-2 crash test is used to illustrate the collision probability's evolution.

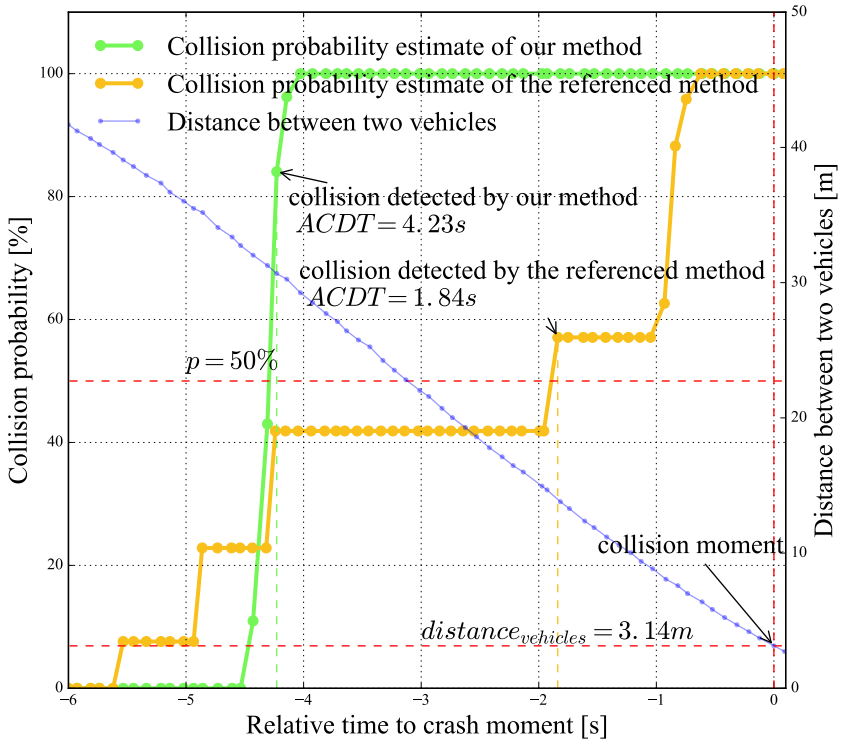
From the figures, we can see that the collision probability curves of our method rapidly increased and reached 50%. Our method could give a warning message over 2 s ahead than the referenced method. As shown in Figures 5.8a and 5.8b, the shapes of the collision probability curves of Wi-Fi and LTE based experiments are approximately identical; the only distinct difference is that the Wi-Fi-based test receives more data than the LTE-based test. This indicates delays and dropouts hardly impair collision probability estimation. The same observation can be seen in TTC and CP estimation experiments, and this has been reported by Tan & Huang (2006) and Joerer et al. (2013).

The superiority of our method is mainly due to the current-state-centered sampling, which is more realistic and practical than sampling at a fixed interval. Concretely, some samples that are drawn between the max and min acceleration are not appropriate, for example, hard braking and rapid acceleration are dangerous and barely used in reality. The negative effects could be observed in Figure 5.8 where the probability curves of the referenced method reacted earlier, which was caused by the samples of large accelerations resulting in long trajectories.

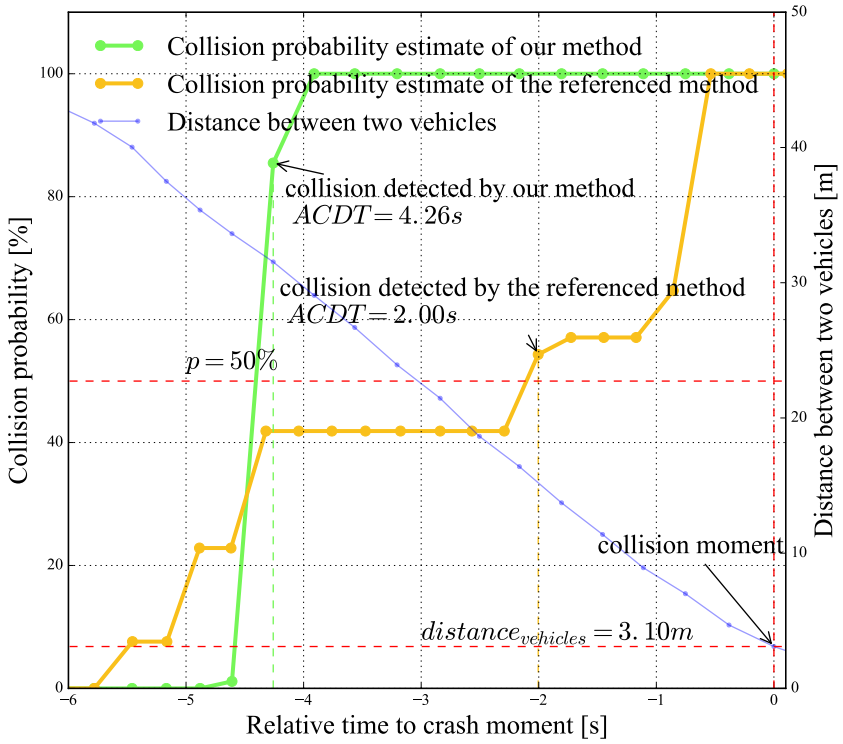
The mean ACDTs of all crash tests are plotted in a spider chart, as shown in Figure 5.9; the center of the spider web indicates crash moments. Specifically, using Wi-Fi/LTE, the mean ACDTs of our method and the referenced method are 4.4 s/4.3 s and 1.7 s/1.7 s, respectively. Our method yields greater ACDT, and this means more time reservation is provided to vehicles for collision avoidance, and our method is more robust to latency and dropouts than the referenced method.

Time-to-collision Estimation

TTC is another important indicator that reflects the criticality of a crash. The Root Mean Square Error (RMSE) defined in Equation 5.6-17 is adopted to



(a)



(b)

Figure 5.8: Collision probability estimates' evaluation in major3-minor1-2 test. (a) Wi-Fi-based experiment. (b) LTE-based experiment.

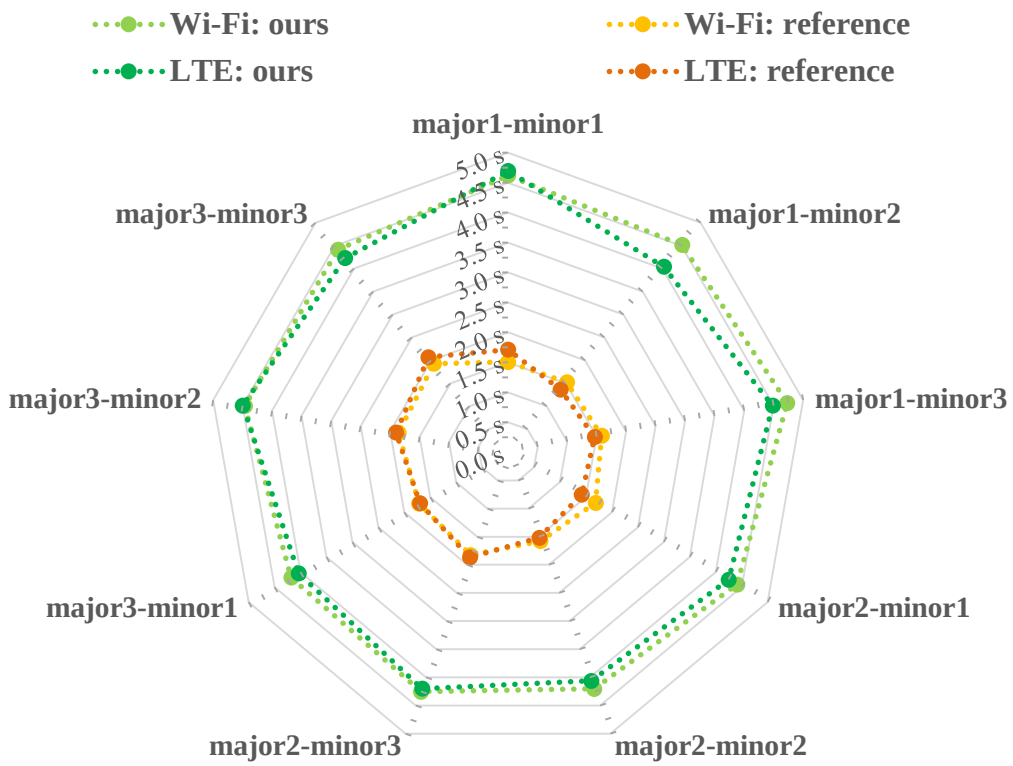


Figure 5.9: Mean ACDT of each crash test using Wi-Fi and LTE connections.

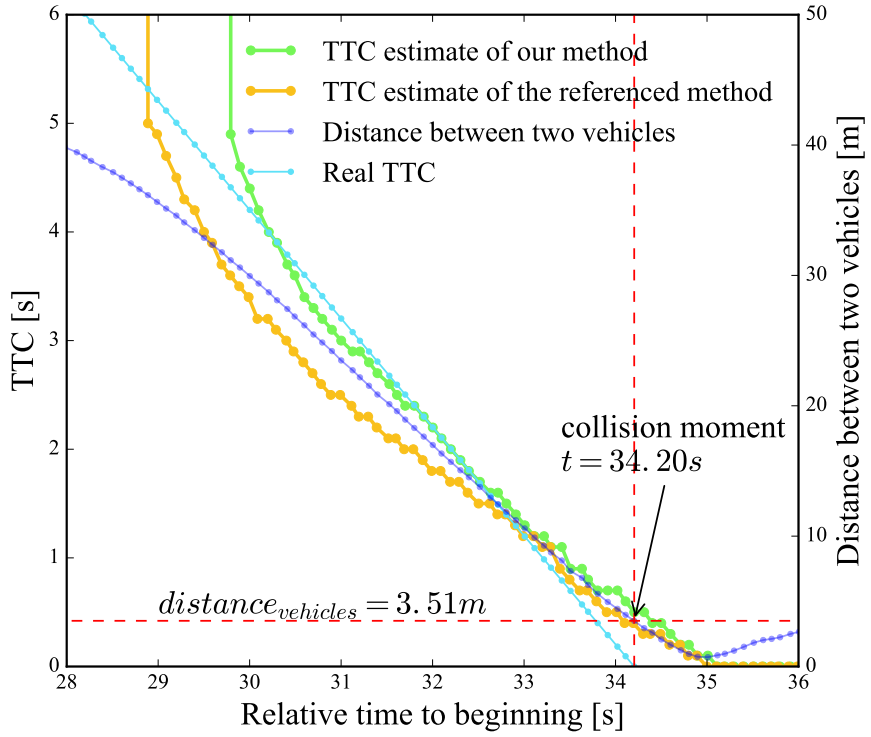
evaluate the methods' TTC estimation performance.

$$RMSE = \sqrt{\frac{\sum_{k=1}^{N}(TTC_k - TTC_{true})^2}{N}}, TTC_{true} = t_c - t_k \tag{5.6-17}$$

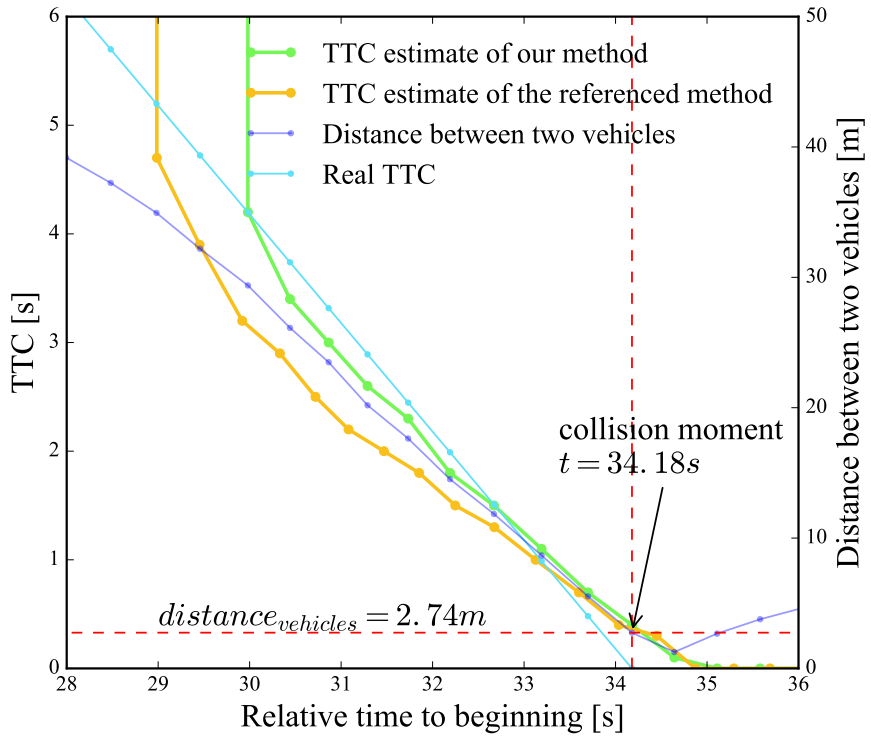
where TTC_{true} is true TTC; N is the estimate number. The major2-minor3-1 crash test in Figure 5.10 demonstrates the TTC estimates' evolution.

From the figures, it is obvious that the TTC estimated by our method is a closer approximation to the true TTC, which is due to: (1) the CTRA model produces more accurate predictions than the CA model; (2) our method considers velocity, which greatly impacts trajectory prediction.

We can observe that at a collision moment, the TTCs do not equal to zero. This is because the vehicles in this study are modeled as mass points. When a crash happened, the two mass points were not overlapped, and this induced non-zero TTCs. To overcome this problem, a more precise vehicle model should be used, such as a rectangle.



(a)



(b)

Figure 5.10: Evolution of TTC estimates in major2-minor3-1 test. (a) Wi-Fi-based experiment. (b) LTE-based experiment.

The mean TTC RMSEs of all the tests are summarized in Figure 5.11 from which it can be concluded that our method generates more accurate TTC estimates than the referenced method.

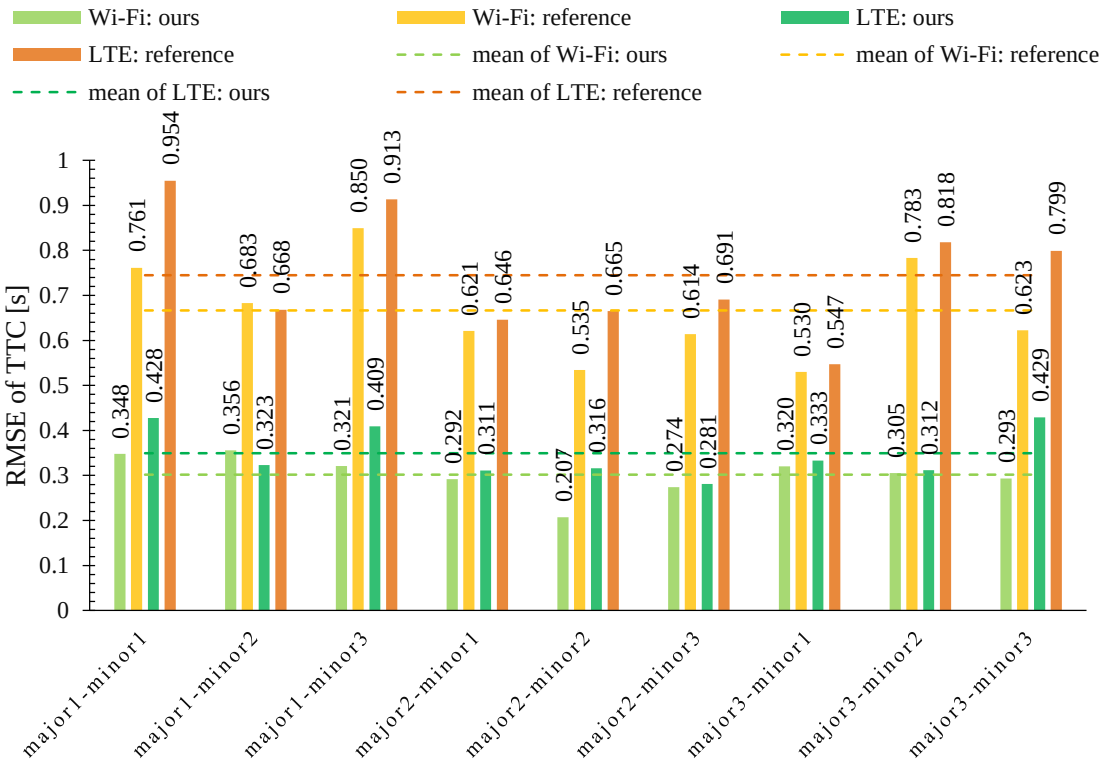


Figure 5.11: Mean TTC RMSE of each crash test using Wi-Fi and LTE connections.

Conflict Point Estimation

Conflict points are important for collision avoidance. The Average Euclidean Distance (AED) defined in Equation 5.6-18 is used to assess the methods' performance for CP estimation.

$$AED = \frac{1}{N}\sum_{i=1}^{N} \|(x,y)_i^k - (x,y)_{crash}\|_2, \text{where } (x,y)_i^k \in CP_k \tag{5.6-18}$$

where $(x,y)_i^k$ is the i-th estimated conflict point at t_k and $(x,y)_{crash}$ is the crash position represented by the midpoint between the two vehicles' positions at a crash moment; N is the count of estimated CP at t_k. The evolution of CP estimation is illustrated in Figure 5.12. It can be seen that our method, first, converges to the

real crash position ($AED \leq 3.3$ m) with smaller AED.

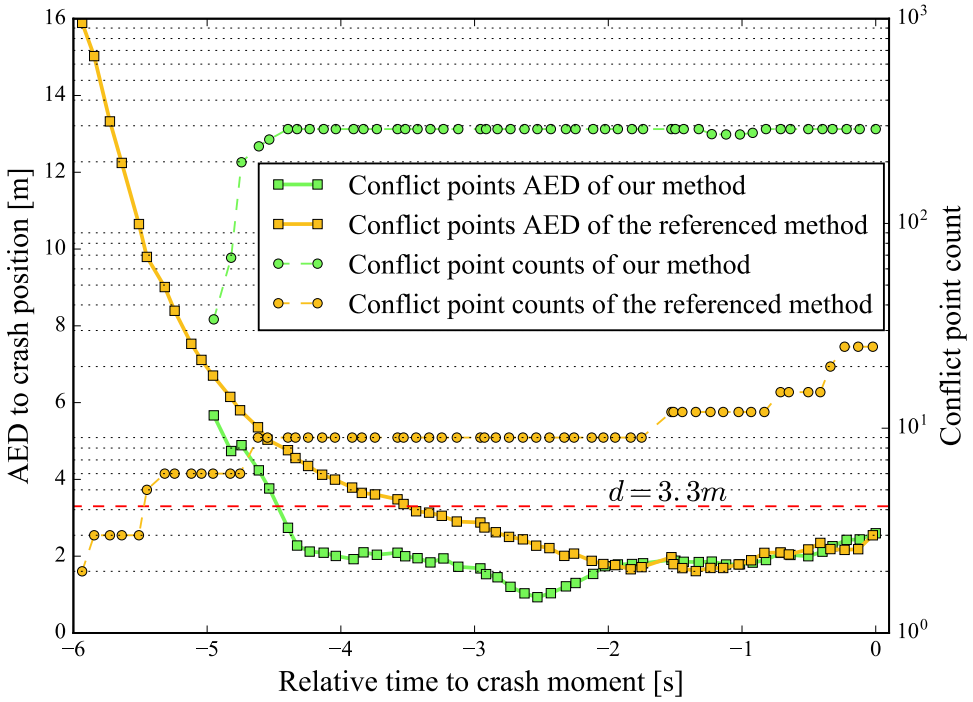
It can be inferred that the CTRA model and the current-state-centered sampling ensure an accurate and fast CP estimation. The results of CP estimations of all crash tests are shown in Equation 5.13. At average, our method can detect 275 potential CPs with 2.4-m AED; however, the referenced method can only detect 11 potential CPs with a larger 3.0-m AED.

Latency Assessments

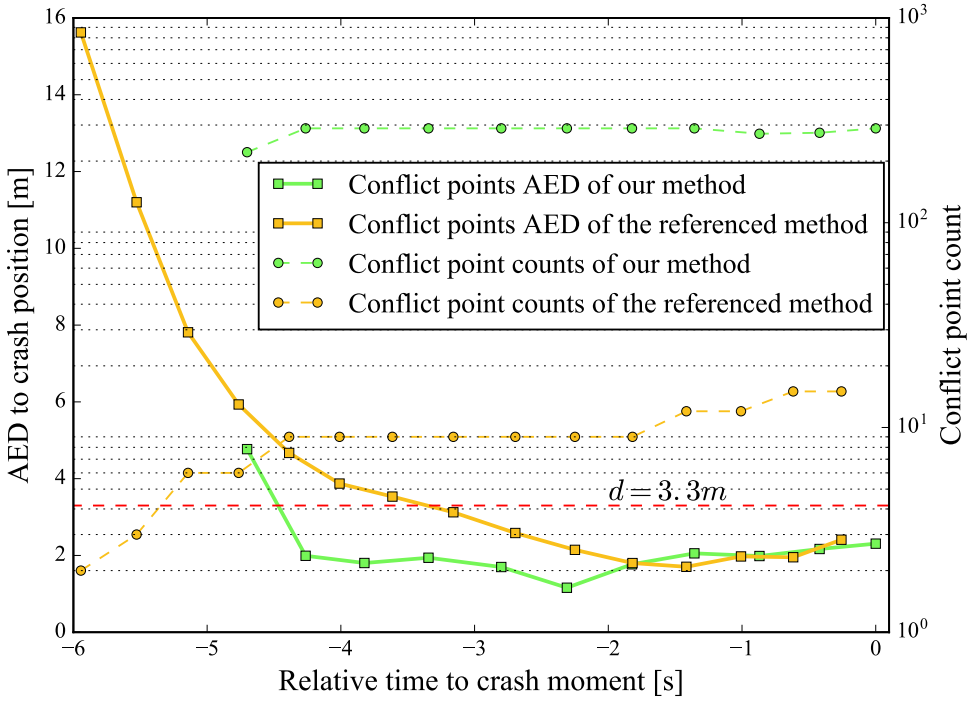
In safe driving applications, latency is the most critical criterion. In our experiments, latencies are derived from communication and computation. It is a nontrivial task to clearly distinguish between these two kinds of latencies because high communication latency always masks low computation latency. In the CRA service, only when the trajectory sets' bounding boxes are intersected, shall the collision detection be performed; otherwise, the CRA server does nothing and returns a null response message. The latency of the null response message is relatively regarded as communication latency. The overall communication latencies of the 54 crash tests are summarized in Figure 5.14. We can see that the communication latency of the Wi-Fi connection is much lower than that of the LTE connection, which fluctuates within a large interval. At average, Wi-Fi and LTE connections respectively introduced 23 ms and 285 ms latency in our experiments. The outliers in Wi-Fi-based experiments might be caused by the VPN server on the cloud server.

Next, we consider communication latency's effect on collision avoidance. As above experiments indicated, our method and the referenced method respectively yield 4.4 s and 1.7 s ACDTs; and according to Van Der Horst & Hogema (1993), a minimum time of 1.5 s is critical for collision avoidance. In this regard, the referenced method is vulnerable to latency using Wi-Fi, and becomes invalid using LTE; our method has a greater tendency to tolerate latency, up to 2.9 s.

Following the emergence of the 5G era, communication latency will be substan-



(a)



(b)

Figure 5.12: Evolution of CP AED and count in major1-minor3-3 test. (a) Wi-Fi-based experiment. (b) LTE-based experiment.

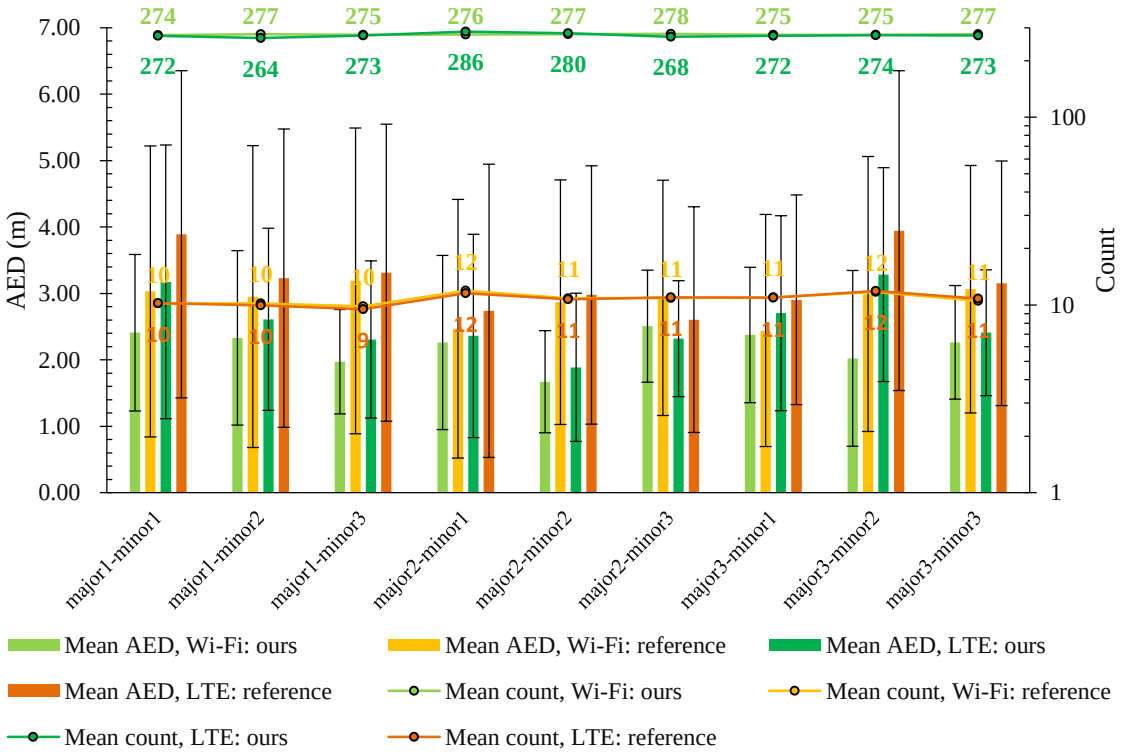


Figure 5.13: Mean AED and count of each crash test using Wi-Fi and LTE connections.

tially eliminated and computational cost may become a challenge due to the variety of data involved. To investigate the effect of computing resources on latency, six major1-minor1 crash tests under stable and low-latency Wi-Fi connection were conducted using different VMs, whose parameters are presented in Table 5.3.

Table 5.3: Parameters of Virtual Machines used

VM size	Family	vCPUs	RAM(GiB)	Data disks	Max IOPS	Temporary storage(GiB)
B2s	General purpose	2	4	4	1280	8
F8s	Compute optimized	8	16	32	25600	32

The experimental results are presented in Figure 5.15, where we can see that non-null response latency (marked with red + and ×) significantly increased, especially for our method, as shown in Figure 5.15a; the non-null response latency increase of the referenced method was relatively mild (see Figure 5.15b). This difference was because our method identified a crash between two vehicles from 17×17 cases; however, the referenced method was involved in 5×5 cases.

Figures 5.15c and 5.15d show that the powerful F8s VM decreased the latency

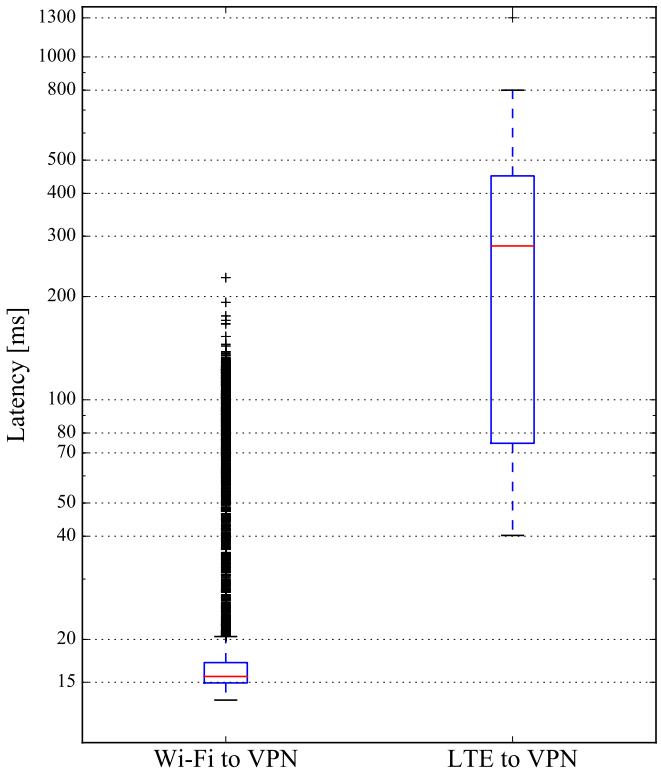


Figure 5.14: Communication latency using Wi-Fi and LTE connections.

at approximately 11.5 ms for our method, which can compensate for approximately 74 % communication cost (averagely, 15.5 ms). Using different VMs rarely impacted the referenced method's latency (see Figures 5.15e and 5.15f) because the B2s VM was sufficient for the method. From the above experiments, it can be inferred that in the future, for example, in the 5G era, for more complex algorithms involving large data contents, such as real-time sensor data, HD maps and historical driving data, exploiting powerful computing resources on cloud or edge servers will be the appropriate choice than putting all workloads on a local vehicle because the computation cost that is avoided may compensate for communication cost.

5.7 Future Work

Our method being a cloud/edge server-based solution, the increasing number of connected vehicles and unstable network connection (e.g., wireless interference,

poor signal) may lead to communication/computing contention and transmission failure, weakening our system performance. To our best knowledge, there are three possible ways to overcome these challenges in our future work:

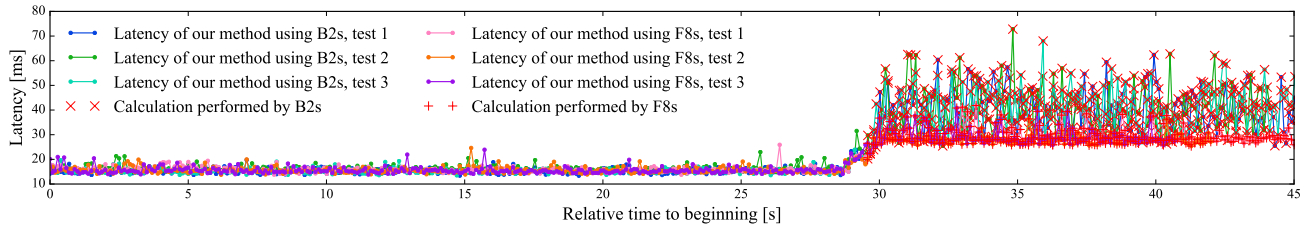
- Collaborating with SCWSs; if CCWSs become unworkable, SCWSs should be relied on.
- Deploying more dedicated servers distributed at various geographical locations; where a certain server will handle requests from a certain area.
- Assigning priorities to different requests; for example, a request sent from a blind intersection by an aged driver is of the highest priority.

To go forward in either way, as stated above, the proposed system must be initially validated and evaluated. In this regard, our study should be seen as a preliminary to implement the proposed system.

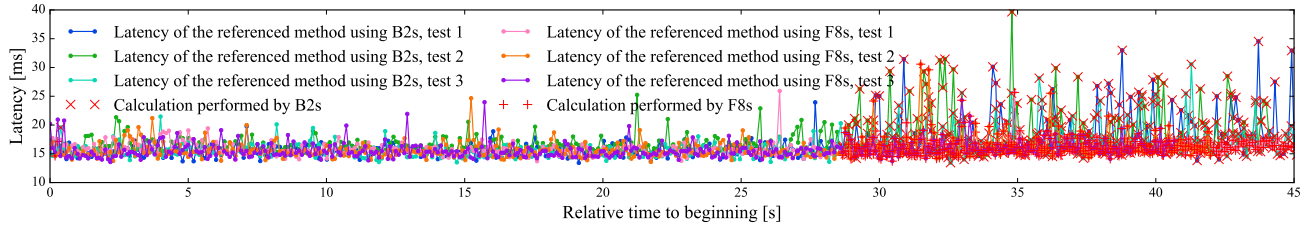
5.8 Conclusions

In this chapter, a new collision risk assessment method was proposed. The method leveraged vehicular state and motion uncertainties from several aspects to predict multiple possible vehicles' trajectories to assess collision risk. Furthermore, a novel server-based architecture for CCWSs was proposed. The method and architecture were validated and evaluated through many real-world experiments. Following the experimental results, our method outperformed the referenced method in terms of collision probability, TTC, and CP estimations. Our method detected a crash at approximately 4.4 s in advance. In contrast, the referenced method detected a crash (under the same experimental conditions as our method) at approximately 1.7 s in advance. As a result, our method can provide sufficient reservation time for collision avoidance, and this makes our system robust against latency and dropouts. For the communication modes, Wi-Fi has much lower latency than LTE. It was validated that the performance of the proposed architecture is improved when it leveraged powerful computing resources

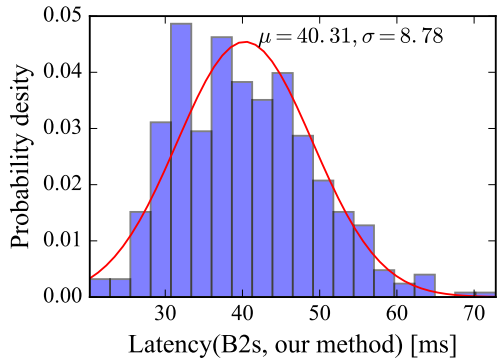
on a cloud server, which decreased computation cost that may compensate for communication costs in the future. The proposed architecture can be seen as a new and feasible option for CCWS design.



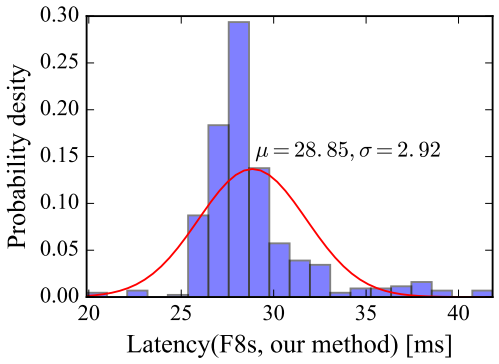
(a)



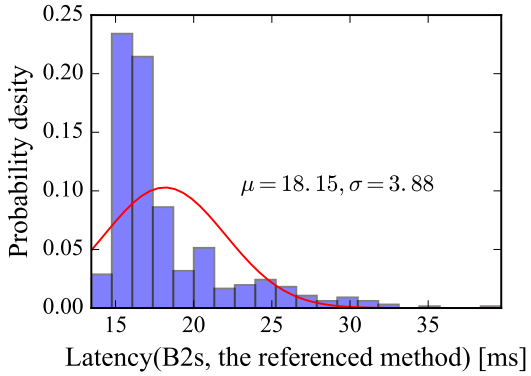
(b)



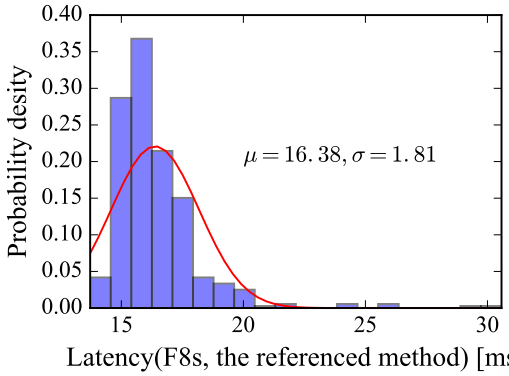
(c)



(d)



(e)



(f)

Figure 5.15: Computing resources' impact on latency. (a) Latency of our method using different VMs. (b) Latency of the referenced method using different VMs. (c) Non-null responses' latency distribution of our method using B2s. (d) Non-null responses' latency distribution of our method using F8s. (e) Non-null responses' latency distribution of the referenced method using B2s. (f) Non-null responses' latency distribution of the referenced method using F8s.

CHAPTER 6

SPATIAL KINEMATIC TRAJECTORY DATA IN VEHICULAR MOTION PREDICTION AND THEIR INTEGRATION IN KALMAN FILTER AND SPATIAL DATABASE FRAMEWORK

This chapter introduces a novel method that leverages Kalman filter and spatial kinematic trajectory data to reduce the uncertainties in middle-term motion prediction. In this chapter, the spatial kinematic trajectory is defined at first. Then the proposed system is introduced. Thirdly, the core of this chapter, the EKF and spatial database framework for kinematic trajectory data integration, is detailed. Then experiments are conducted to validate and evaluate our method. At last, the conclusions and future work are summarized.

6.1 Kinematic Trajectory and Spatial Database

6.1.1 Spatial Kinematic Trajectory

A spatial kinematic trajectory dataset $KT = \{\mathbf{p}_i\}_{i=0}^{M}$ is defined as a sequence of kinematic trajectory points $\mathbf{p}_i = \begin{bmatrix} x & y & \theta & v & a & \omega \end{bmatrix}$, in which noises have been reduced as much as possible through filtering or smoothing technologies that have been introduced in chapter 3. M is the point number. Besides the kinematic data, which contain spatial information of the point, other kinds of attributes are attached/linked to the kinematic trajectory points:

- semantic attributes, such as corresponding driver and vehicle information.

- topological attributes, such as the road a point located in; previous/next point.

6.1.2 Spatial Kinematic Trajectory Database

Up to date, there are no historical data available in DM. In this study, it is suggested to add historical data into DM as a new type of information. The spatial kinematic trajectory data are stored in a PostGIS (2022) database, which the DM is developed upon. In this study, three tables: kinematic trajectory point (ktp) table, kinematic trajectory (kt) table and road table of HD maps are mainly used, as Figure 6.1 shows.

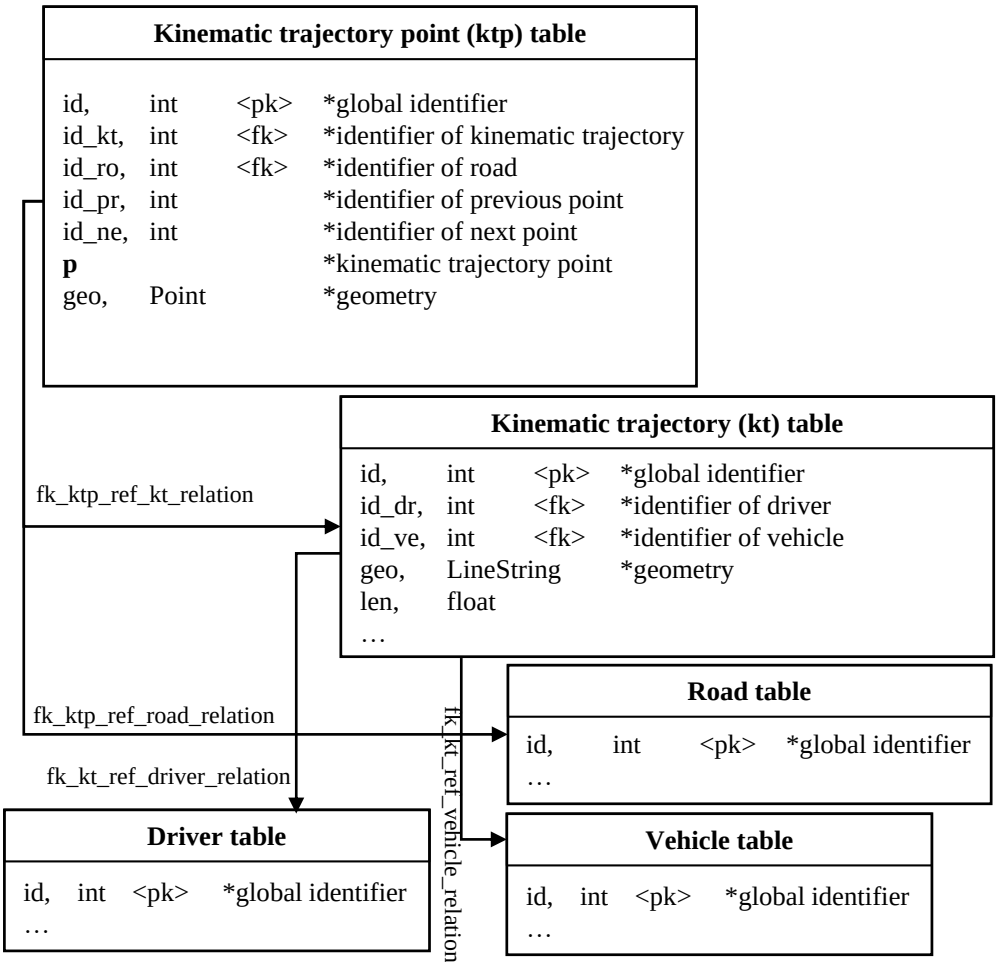


Figure 6.1: The physical data model of spatial kinematic trajectory in DM. * denotes explanations of the columns.

The ktp table stores key kinematic information of these points. The topological information of kinematic points is also maintained in this table. For example,

the ID of the road a kinematic point is located in can be easily known from the ktp table. The id_ro attribute is used to screen out unrelated points in a spatial search when a vehicle is driving on the road. This would speed up our queries. In addition, generalized search tree indexes are built on the ktp table to further accelerate the queries. We use id_kt to link the ktp table to the kt table, in which the statistic of the trajectories and semantic information of kinematic points are stored. The driver and vehicle table maintains the personal/private information of registered drivers and vehicles.

In this study, we assume that a global route is planned in advance. Therefore all the roads that a vehicle will pass through can be known. In a query, only data linked to the roads are scanned.

6.2 A Comparison of Different Vehicle Motion Prediction Methods

In order to highlight the merits of our method, a brief comparison of different vehicle motion prediction is conducted at first. In principle, the method proposed in this chapter is based on historical data. The difference is that our method makes use of spatial relationship. The comparison is presented in Table 6.1.

We can find from the table that: (1) our method does not belong to the present five genres, which is introduced in section 1.1.3 of chapter 1; novelly, it is implemented in a KF framework incorporating a spatial database; different from previous methods that need to collect a huge volume of training data and learn prediction models based on the data, which is DCC, our method directly retrieves information based on spatial relationships in the prediction process. The training process is not needed. (2) compared with the map-aided methods that lose vehicle dynamics in predictions, our method leverages the driver's spatial kinematic trajectory data to predict vehicle motions. Our predictions will converge to the driver's personal driving styles in different spaces and the dynamics will be maintained. (3) furthermore, our prediction is personalized; information concerning

the driver and the vehicle is considered.

Table 6.1: A comparison between our method and some typical methods selected from the five genres. The methods with learning/training processes are regarded as DCC.(1): Lytrivis et al. (2011); (2): Hermes et al. (2009); (3): Jiang et al. (2022); (4): Kawasaki & Tasaki (2018); (5): Yalamanchi et al. (2020). HDT: Historical Data for Training.

Study	Genre	Input	Output	Methodology	Scenario	DCC	Personalized
(1)	physical model based	vehicle state	position	kinematic models, Dempster-Shafer reasoning system	campus	No	No
(2)	trajectory matching based	odometry data, HDT	position, velocity, yaw and yaw angle	particle filter, trained trajectory classifier	intersection	Yes	No
(3)	machine learning based	the first 3s historical trajectories, HDT	position	LSTM	highway	Yes	No
(4)	map-aided	HD maps, vehicle state	position, velocity	EKF, cubic polynomial fitting	intersection	No	No
(5)	hybrid	HDT, HD maps	position	Uncertainty-aware Stitching	intersection	Yes	No
ours	spatial historical data based	vehicle state, spatial kinematic data	position, velocity	EKF, spatial search	campus	No	Yes

6.3 System Overview

The inspiration for proposed method is Tobler's first law of geography that: *everything is related to everything else, but near things are more related to each other* (Tobler, 1970). In our context, we suppose that in most cases vehicle behaviors are spatially correlated; for example, on a certain lane segment, vehicles always exhibit similar velocities, and when they approach an intersection or a bend, they have to slow down. Based on these intuitions, we further suppose that, in most cases, as vehicle behaviors' representations, vehicular states are spatially correlated. This makes middle-term motion prediction possible where and when spatial kinematic trajectory data are available. Our system is illustrated in Figure 6.2.

The core components of our system are listed below:

- A UKF state estimator. In real-world studies, prior to motion prediction, a real-time vehicle state estimator is necessary to reduce sensor noises; in our system, an UKF that cooperates with a CTRA model is adopted. The UKF fuses information from the CTRA model and onboard sensors to make a

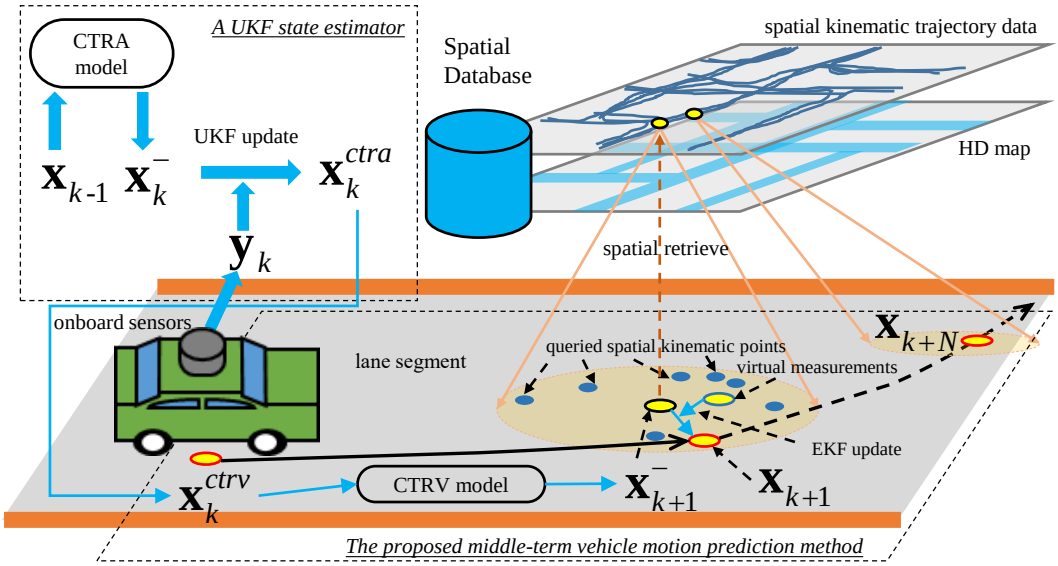


Figure 6.2: The system overview. The idea of this paper is inspired by the first law of geography that everything is related to everything else, but near things are more related to each other.

reliable real-time vehicle state estimate at 10 Hz.

- A spatial database for kinematic trajectory data management. The spatial database that maintains kinematic trajectory data and HD maps is a crucial component. The kinematic trajectory data, which contain spatial information, are stored in the spatial database to leverage a quick spatial query to realize real-time middle-term vehicle motion prediction. The kinematic data are linked to the HD maps to facilitate the spatial query.

- The lightweight middle-term vehicle motion prediction algorithm. The utilization of the spatial database and EKF makes our method lightweight. The quick spatial search functions of the database provide the most spatially related information to our algorithm and thus we do not need to learn knowledge from huge amounts of data. The efficient EKF ensures real-time data processing.

As illustrated in the figure, a current vehicle state estimate made by the UKF is sent to the middle-term motion prediction algorithm as an initial state. Then the state evolves into the next instant according to a CTRV model. At the predicted

position, the surrounding kinematic trajectory points are queried from the spatial database and the virtual measurements are calculated. Finally, the predictions are corrected by the EKF using the virtual measurements. This process is iterated 50 times to achieve 5 s middle-term prediction.

6.4 Methodology

6.4.1 Vehicle State Estimation

In order to predict vehicle motion, it is necessary to derive a vehicle's state at the current time, such as accurate position and velocity estimates. In our system, a UKF that is introduced section 3.2.3 in chapter 3 is adopted in real-time vehicle state estimation.

For the sake of brevity, we give only two key functions in state estimation in this section; namely the process and observation functions. The CTRA model that is introduced in section 3.1.4 in chapter 3 is selected as the process model in our system:

$$\mathbf{x}_k = \mathbf{F}_{ctra}(\mathbf{x}_{k-1}) = \begin{bmatrix} x_{k-1} + \frac{(v_{k-1}+a_{k-1}T)\sin(\theta_{k-1}+\omega_{k-1}T) - v_{k-1}\sin(\theta_{k-1})}{\omega_{k-1}} + \frac{a_{k-1}[\cos(\theta_{k-1}+\omega_{k-1}T) - \cos(\theta_{k-1})]}{\omega_{k-1}^2} \\ y_{k-1} - \frac{(v_{k-1}+a_kT)\cos(\theta_{k-1}+\omega_{k-1}T) - v_{k-1}\cos(\theta_{k-1})}{\omega_{k-1}} + \frac{a_{k-1}[\sin(\theta_{k-1}+\omega_{k-1}T) - \sin(\theta_{k-1})]}{\omega_{k-1}^2} \\ \theta_{k-1} + \omega_{k-1}T \\ v_{k-1} + a_{k-1}T \\ a_{k-1} \\ \omega_{k-1} \end{bmatrix} \tag{6.4-1}$$

In this model, $\mathbf{x}_k = [x, y, \theta, v, a, \omega]^T$ is the vehicle state at instant k. (x, y) denotes position coordinates. v and a are velocity and acceleration; θ and ω are heading and yaw rate. T is the time interval between instant k-1 and k.

The observation functions are given in Equation 6.4-2. It is noteworthy that

these functions are sensor- and system-dependent.

$$\begin{gathered} x_{\mathbf{y}} = x_{\mathbf{x}} \qquad y_{\mathbf{y}} = y_{\mathbf{x}} \qquad \theta_{\mathbf{y}} = \theta_{\mathbf{x}} \qquad \omega_{\mathbf{y}} = \omega_{\mathbf{x}} \\ v_{\mathbf{y}}^{x} = v_{\mathbf{x}} \cos\theta_{\mathbf{x}} \quad v_{\mathbf{y}}^{y} = v_{\mathbf{x}} \sin\theta_{\mathbf{x}} \quad a_{\mathbf{y}}^{x} = a_{\mathbf{x}} \cos\theta_{\mathbf{x}} \quad a_{\mathbf{y}}^{y} = a_{\mathbf{x}} \sin\theta_{\mathbf{x}} \end{gathered} \tag{6.4-2}$$

where the bold subscripts $\mathbf{x}$ and $\mathbf{y}$, respectively, denote the system state and observation vector; the superscripts x and y indicate the components in the x and y directions. Details of the configurations of the UKF algorithm are available in section 4.3.2 in chapter 3. The vehicle state estimate $\hat{\mathbf{x}}_k^{ctra}$ output by the UKF is used in vehicle motion prediction.

6.4.2 Adaptive Spatial Retrieve Algorithm

Kinematic trajectory points are not uniformly distributed in space; thus, it is unsuitable to use a fixed distance threshold in spatial searches. A recursive spatial retrieve algorithm is proposed. The algorithm adaptively sets the search distance to ensure that at least two associated kinematic trajectory points can be found. Its pseudocodes are shown in Figure 6.3.

Based on Tobler's first law of geography, Adaptive Spatial Retrieve Algorithm (ASRA) tries to find the closest associated kinematic points. Namely, around a specified position (ps.x, ps.y), the associated kinematic trajectory points must comply with the following rules:

- Spatial rules: the points must be within a certain distance 0.5 m * k, where k <5, and the heading difference must be less than $\pi/2$; otherwise, the points are kicked out; if $k \geq 5$ and the point number is less than 2, the search fails.
- Topological rules: the points must be located in the road that the vehicle is driving on; otherwise, the points are kicked out.
- Semantic rules: the points must be produced by the same vehicle that is driven by the same person; otherwise, the points are kicked out.

Adaptive Spatial Retrieve Algorithm (ASRA)

op: output spatial kinematic points
ps: a kinematic point around which a search occurs
rids: id set of the roads to be passed through
vid: vehicle id
did: driver id
k: search times, initial k=1

```
ASRA(ps, did, vid, rids, k):

    distance = 0.5m * k

    op = SELECT p FROM kpt t1, kt t2 WHERE t1.id_kt = t2.id      AND
    ST_DWithin(t1.geo, 'POINT(ps.x, ps.y)', distance)            AND
    t1.id_ro IN (rids)                                           AND
    t2.id_dr = did                                               AND
    t2.id_ve = vid                                               AND
    |ps.θ – t1.p. θ| < PI/2;

    if (size(op) < 2)
        k = k + 1
        ASRA(ps, did, vid, rids, k)

    return op
```

Figure 6.3: The proposed adaptive spatial retrieve algorithm.

The topological rules screen out plenty of unconcerned points to speed up the search. The spatial rules select all surrounding points that have a close heading angle within a certain distance. The semantic rules ensure only private data are selected. On the one hand, this protects the privacy of drivers; on the other hand, it is the key of personalized predictions. The searched kinematic points *op* are used to calculate virtual measurements in the following prediction algorithm. The details can be found in the section 6.4.3.

6.4.3 EKF Framework for Kinematic Trajectory Data Integration

As reported in chapter 4, the accuracy performances of EKF and UKF are almost identical; however, EKF is faster. Motion predictions are computation-consumed; the efficiency of an algorithm shall be seriously treated. Thus, an EKF (see section 3.2.3) that cooperates with the CTRV (see section 3.1.3) model is used in vehicle motion prediction.

In the CTRV model, an estimated vehicle state at instant k is defined as

$\hat{\mathbf{x}}_k = \begin{bmatrix} x & y & \theta & v & \omega \end{bmatrix}^T$. In our system, the current vehicle state estimate $\hat{\mathbf{x}}_k^{ctra}$, which is output by the UKF, is assigned to $\hat{\mathbf{x}}_k^{ctrv}$ as the initial state of our EKF predictor through:

$$\hat{\mathbf{x}}_k^{ctrv} = \left[\begin{array}{c|cc} \mathbf{I} & \multicolumn{2}{c}{\mathbf{0}} \\ \hline & 0 & 0 \\ \mathbf{0} & & \\ & 0 & 1 \end{array}\right]_{5\times 6} \times \hat{\mathbf{x}}_k^{ctra} \tag{6.4-3}$$

Hereafter, we denote $\hat{\mathbf{x}}_k^{ctrv}$ as $\hat{\mathbf{x}}_k$ for brevity.

Predicting

Firstly, the initial state $\hat{\mathbf{x}}_k$ (the same as $\hat{\mathbf{x}}_k^+$) and the corresponding covariance $\mathbf{P}_k$ (the same as $\mathbf{P}_k^+$) have evolved into the next instant's state $\hat{\mathbf{x}}_{k+1}^-$ and covariance $\mathbf{P}_{k+1}^-$ through Equations 6.4-4∼6.4-6.

$$\hat{\mathbf{x}}_{k+1}^- = \mathbf{F}_{ctrv}(\hat{\mathbf{x}}_k) \tag{6.4-4}$$

$$\mathbf{F}_{ctrv}(\hat{\mathbf{x}}_k) = \begin{bmatrix} x_k + \frac{v_k}{\omega_k}\sin(\theta_k + \omega_k T) - \frac{v_k}{\omega_k}\sin(\theta_k) \\ y_k - \frac{v_k}{\omega_k}\cos(\theta_k + \omega_k T) + \frac{v_k}{\omega_k}\cos(\theta_k) \\ \theta_k + \omega_k T \\ v_k \\ \omega_k \end{bmatrix} \tag{6.4-5}$$

$$\mathbf{P}_{k+1}^- = \mathbf{J}_F \mathbf{P}_k \mathbf{J}_F^T + \mathbf{Q}_k \tag{6.4-6}$$

where, $\mathbf{w}_k$ is the process noise. $\mathbf{J}_F$ and $\mathbf{Q}_k$ denote the Jacobian matrix of function $\mathbf{F}_{ctrv}$ and the covariance matrix of process noise, respectively. For more details on the Jacobian and covariance matrices, see chapter 3.

Spatial Search and Virtual Measurement Calculation

Secondly, we try to find the associated kinematic points around the predicted position and use these points to calculate a virtual measurement.

The ASRA is triggered around $\hat{\mathbf{x}}_{k+1}^-$ and the associated kinematic trajectory

points *op* are retrieved by ASRA. Then *op* is used to construct the following matrix:

$$\mathbf{op} = \begin{bmatrix} x_1 & y_1 & \theta_1 & v_1 & \omega_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & \theta_n & v_n & \omega_n \end{bmatrix}^T \tag{6.4-7}$$

where n is the element number of *op* and we use $(\mathbf{op})_i$ to denote the i-th column of **op**. Each column corresponds to an associated kinematic point. The following three weighting functions are proposed to calculated weights for each kinematic trajectory point.

$$w_i^j = w_i(d_j) = \begin{cases} 1 - \frac{d_j}{\sum_j^n d_j}, & i = 1 \\ 1, & i = 2 \\ e^{-kd_j}, & i = 3 \end{cases} \tag{6.4-8}$$

where d_j is the Euclidean distance between $\hat{\mathbf{x}}_{k+1}^-$ and $(\mathbf{op})_j$, defined in Equation 6.4-9.

$$d_j = ED(\hat{\mathbf{x}}_{k+1}^-, (\mathbf{op})_j) = \sqrt{(x_{\mathbf{x}} - x_{\mathbf{op}})^2 + (y_{\mathbf{x}} - y_{\mathbf{op}})^2} \tag{6.4-9}$$

The three weighting functions are illustrated in Figure 6.4; we can see that w_2 is an Average Weighting (AW) method; both w_1 and w_3 are Inverse Distance Weighting (IDW) methods, while w_1 is linear and yet w_3 is nonlinear. The weights are then normalized through Equation 6.4-10.

$$w_i^j = \frac{w_i^j}{\sum_{j=1}^n w_i^j}, \text{ where } i \in \{1, 2, 3\} \tag{6.4-10}$$

Using the calculated weights w_i^j, the weighted mean of the queried associated kinematic trajectory points is regarded as a virtual measurement $\mathbf{z}_{k+1}^{virtual}$, as ex-

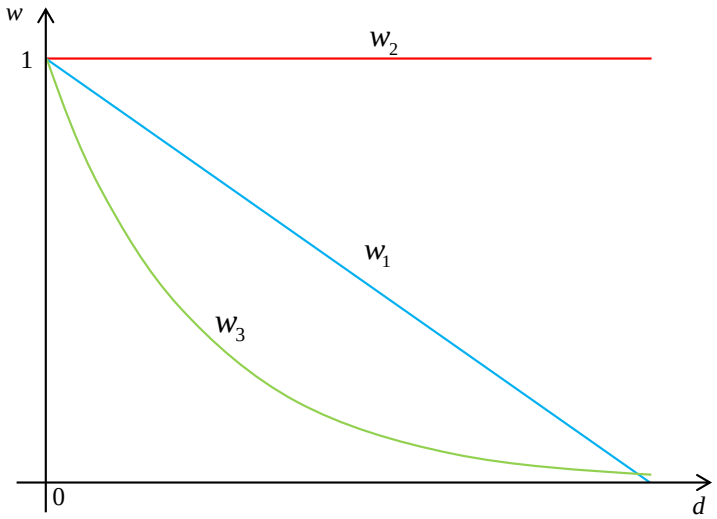


Figure 6.4: The proposed weighting functions. w_1 is a linear IDW method; w_2 is a AW method; w_3 is a nonlinear IDW method.

pressed in Equation 6.4-11.

$$\mathbf{z}_{k+1}^{virtual} = \mathbf{op} \times \begin{bmatrix} w_i^1 & \cdots & w_i^j & \cdots & w_i^n \end{bmatrix}^T \text{ ,where } i \in \{1,2,3\} \text{ and } j \in \{1,2,\cdots,n\} \tag{6.4-11}$$

Obviously, in our framework, the measurement function is as follows, where $\mathbf{e}_{k+1}$ is the measurement noise.

$$\mathbf{z}_{k+1} = \mathbf{x}_{k+1} + \mathbf{e}_{k+1} \tag{6.4-12}$$

Updating

Finally, in this section, the virtual measurement is used to update the prediction. First of all, the near-optimal Kalman gain $\mathbf{G}_{k+1}$ is calculated by:

$$\mathbf{G}_{k+1} = \mathbf{P}_{k+1}^{-}\mathbf{J}_H^T(\mathbf{J}_H\mathbf{P}_{k+1}^{-}\mathbf{J}_H^T + \mathbf{R}_{k+1})^{-1} \tag{6.4-13}$$

where $\mathbf{J}_H = \mathbf{I}$ and $\mathbf{R}_{k+1}$ is the covariance matrix of $\mathbf{e}_{k+1}$. Then, the state prediction is corrected by the virtual measurement through:

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_{k+1}^{-} + \mathbf{G}_{k+1}(\mathbf{z}_{k+1}^{virtual} - \hat{\mathbf{x}}_{k+1}^{-}) \tag{6.4-14}$$

and the a posteriori estimate covariance matrix is given by:

$$\mathbf{P}_{k+1} = (\mathbf{I} - \mathbf{G}_{k+1}\mathbf{J}_H)\mathbf{P}^-_{k+1} \tag{6.4-15}$$

The above process, from Equation 6.4-4 to Equation 6.4-15, is iterated 50 times to predict vehicle motions 5 s into the future.

6.5 Experimental Validation and Evaluation

6.5.1 Experimental Configurations

Real-world experiments were conducted on the campus of Nagoya University using the Toyota PRIUS PHV shown in Figure 5.6c. The LiDAR (Velodyne HDL-64ES3) and the IMU (Xsens MTi-300) mounted on the vehicle were used in our experiments. The sensors were connected to the *Autoware* platform (Kato et al., 2015, 2018; Autoware, 2022) and our system subscribed to the ROS (2022) messages published by the sensor nodes to estimate vehicle state and predict vehicle motion.

It is noteworthy that our experiments were conducted in a public space where the roads were curved and sloped and pedestrians, bikes and other vehicles coexisted. This made our driving behaviors complex; for example, we had to stop our car when pedestrians crossed the road and depart the planed lane when a vehicle parked on the road side. In our experiments, three drives' kinematic data, 12,112 points in total, were stored in the database and the other one was replayed to duplicate the real driving, as we had done in the experiments of chapter 5.

Our system was implemented based on C++ and ROS. Details of the configurations of experimental vehicle, sensors and KFs can be found in chapter 4.

6.5.2 Accuracy Performance Evaluations

Two factors that might impact our algorithm's accuracy performance—the used weighting function and data set size—are investigated in this section.

Used Metrics

The accuracy performance of our method is evaluated quantitatively, using the Average Euclidean Error (AEE) and max error metrics. AEE is used to analyze the overall prediction performance of our algorithm and is defined as:

$$AEE(t_i) = \frac{1}{N}\sum_{j=1}^{N}\sqrt{(x_{t_i}^j - x_{t_i}^r)^2 + (y_{t_i}^j - y_{t_i}^r)^2} \tag{6.5-16}$$

where $(x_{t_i}^j, y_{t_i}^j)$ is the predicted position at time t_i along the j_{th} predicted trajectory. $(x_{t_i}^r, y_{t_i}^r)$ is the corresponding real position. N is the total number of predicted trajectories. The velocity prediction errors are also calculated with the AEE method; however, it is one-dimensional:

$$velocity\ error(t_i) = \frac{1}{N}\sum_{j=1}^{N}|v_{t_i}^j - v_{t_i}^r| \tag{6.5-17}$$

Similarly, $v_{t_i}^j$ and $v_{t_i}^r$ are predicted and real velocity at time t_i along the j_{th} predicted trajectory, respectively.

Max error is the max prediction error along a predicted trajectory. The max error reflects the worst performance in a single trajectory prediction. Therefore, it can reveal some factors covered by the mean values that are derived from the overall predictions. More than 10,000 trajectories were predicted in each of following experiments and their prediction errors were discussed in detail.

Using Different Weighting Functions

In order to investigate the impact of different weighting functions, the prediction accuracy performance of three predictors that used different weight functions, w_1, w_2 and w_3, were compared. In this experiment, all the collected kinematic trajectory data in our database were used. The experimental results are shown in Figures 6.5 and 6.6.

An obvious trend can be found—the three weighting functions have not made

a distinct difference in accuracy aspect, for both position and velocity predictions. However, the IDW methods (w_1 and w_3) are slightly better than the AW method (w_2) in either position or velocity predictions. This is because our search radius is small (initial radius is 0.5 m and max search radius is not more than 2 m); the spatial differences among the searched kinematic trajectory points are therefore slight. Thus, different weighting functions cannot lead to obvious differences from a statistical standpoint. Compared with the state-of-the-art method proposed by Kawasaki & Tasaki (2018), whose position and velocity prediction errors are more than 4 m and 1.5 m/s at 4 s respectively, the performance of our method is acceptable.

For a further and meticulous investigation of prediction errors, their max prediction errors are analyzed. The max errors' CDFs when the three different weighting functions are used are drawn in the upper-left corner in Figure 6.5. We divide the max prediction errors into four groups: outstanding (max error $\leq$ 2 m), good (2 m $<$ max error $\leq$ 4 m), not bad (4 m $<$ max error $\leq$ 7 m) and bad (max error $>$ 7 m). The four max error groups' spatial distributions can be found in Figure 6.7.

From the CDFs in Figure 6.5, we can find that good predictions, including outstanding predictions, make up more than 60% using either weighting function, as the black arrow indicates. This means the good prediction rate of our method is over 60% in our complex experimental space. It is also noticeable, where the red arrow points, that w_3 (the red) yields a higher rate (34%, approximately) of outstanding predictions. Therefore, to obtain more outstanding predictions, w_3 is recommended.

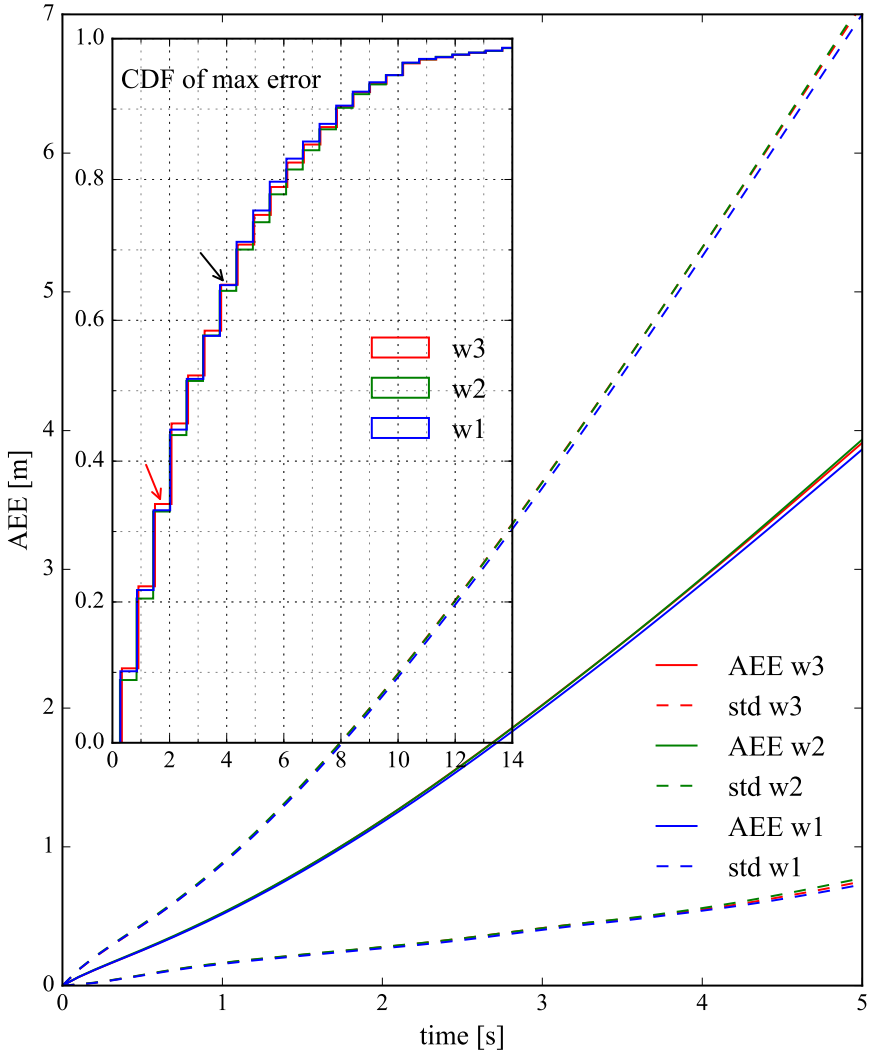


Figure 6.5: Statistics of position prediction errors. The solid and dotted lines are the AEEs and standard deviations (std) of position prediction errors, respectively. The red corresponds to the nonlinear IDW method w_3; the green corresponds to the AW method w_2; the blue corresponds to the linear IDW method w_1. The IDW methods (red and blue) are slightly better than the AW method (green). The CDFs of max prediction errors when w_3, w_2, w_1 are used are plotted in the upper-left box. The red arrow indicates that w_3 yields more outstanding predictions. The black arrow indicates a good prediction rate of either weighting function in our algorithm of more than 60%.

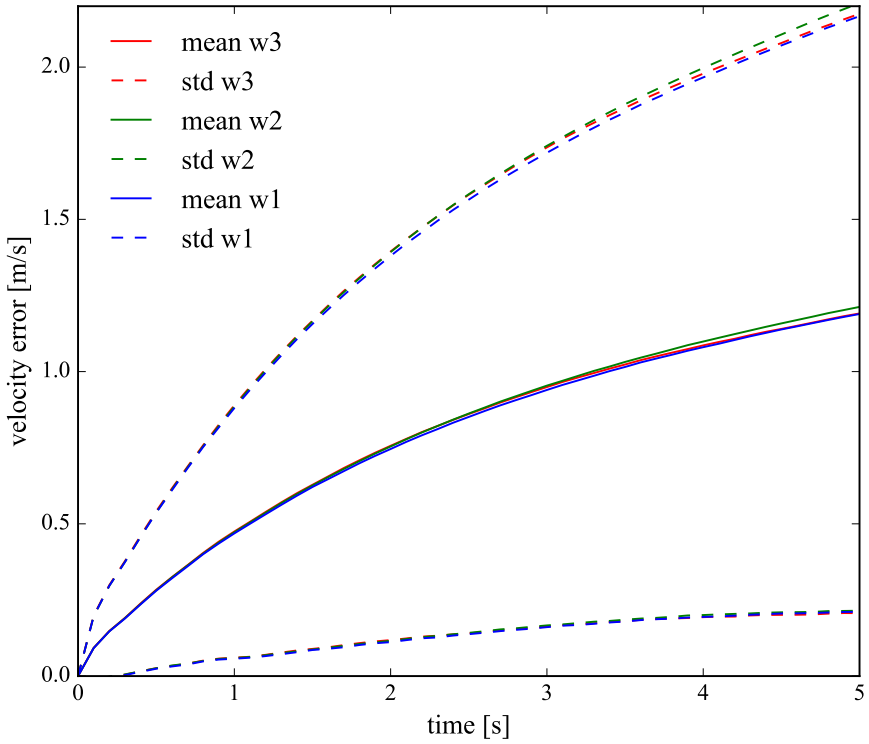


Figure 6.6: Statistics of velocity prediction errors. The solid and dotted lines are the means and standard deviations (std) of velocity prediction errors, respectively. The red corresponds to the nonlinear IDW method w_3; the green corresponds to the AW method w_2; the blue corresponds to the linear IDW method w_1. The IDW methods (red and blue) are slightly better than the AW method (green).

Figure 6.7: Spatial distribution of max prediction errors along the driving route.

Using Different Data Sets

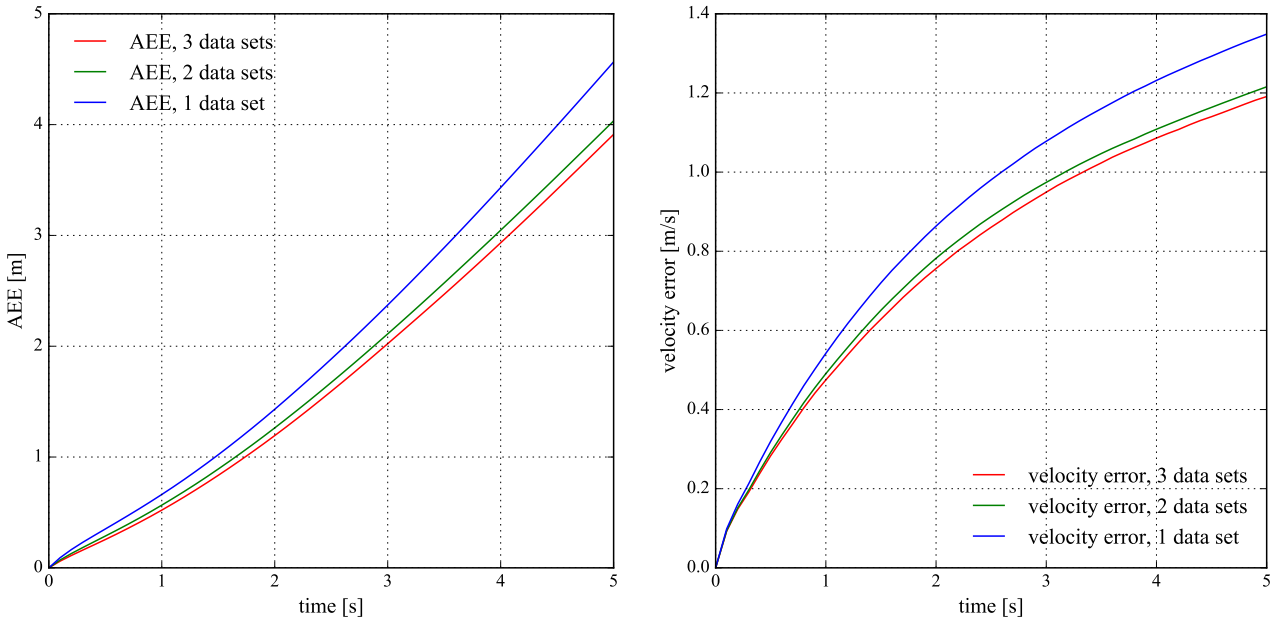
As mentioned before, the learning-based methods consume plenty of historical data; the model's performance is determined by the size of training data set. The following experiments are designed to answer two questions: (1) How does our method perform when the historical data that can be used are limited? For example, in the condition where just one trajectory is available. (2) How does our method perform when the size of historical data increases? The first question indicates the worst performance of our method in bad conditions. The second question assesses the performance potential of our method in good conditions.

We had collected three kinematic trajectory data sets $\{KT_1, KT_2, KT_3\}$ in our database. Each trajectory data set corresponded to a drive on the route in Figure 6.7 in our campus. Some information about the trajectories are listed in Table 6.2. Using w_3, our method was tested on different data sets, including one data set $\{\{KT_1\}, \{KT_2\}, \{KT_3\}\}$ that involves three experiments, two data

sets $\{\{KT_1, KT_2\}, \{KT_1, KT_3\}\{KT_2, KT_3\}\}$ that involve three experiments and three data sets $\{\{KT_1, KT_2, KT_3\}\}$ that involve one experiment. Their average prediction errors are summarized in Figure 6.8.

Table 6.2: The kinematic statistics of the three drives.

Trajectory	**Mean v (m/s)**	**Std v (m/s)**	**Mean a (m/s^2)**	**Std $a(m/s^2)$**	**Driver**	**Vehicle**	**Point Number**
KT_1	5.82139	2.65461	0.04619	0.62621	Yamata	PHV001	3895
KT_2	5.25437	2.23874	0.04669	0.51137	Yamata	PHV001	4142
KT_3	5.49708	2.01545	0.05826	0.47457	Yamata	PHV001	4075



(**a**) (**b**)

Figure 6.8: Prediction errors using different data sets. An obvious trend can be found—with the size of used data sets increased, the prediction accuracy of both position and velocity is improved. (**a**) The position prediction error. The red, green and blue curve, respectively, correspond to the position-prediction errors using 3, 2 and 1 data sets. (**b**) The velocity prediction errors. The red, green and blue curve, respectively, correspond to the velocity-prediction errors using 3, 2 and 1 data sets.

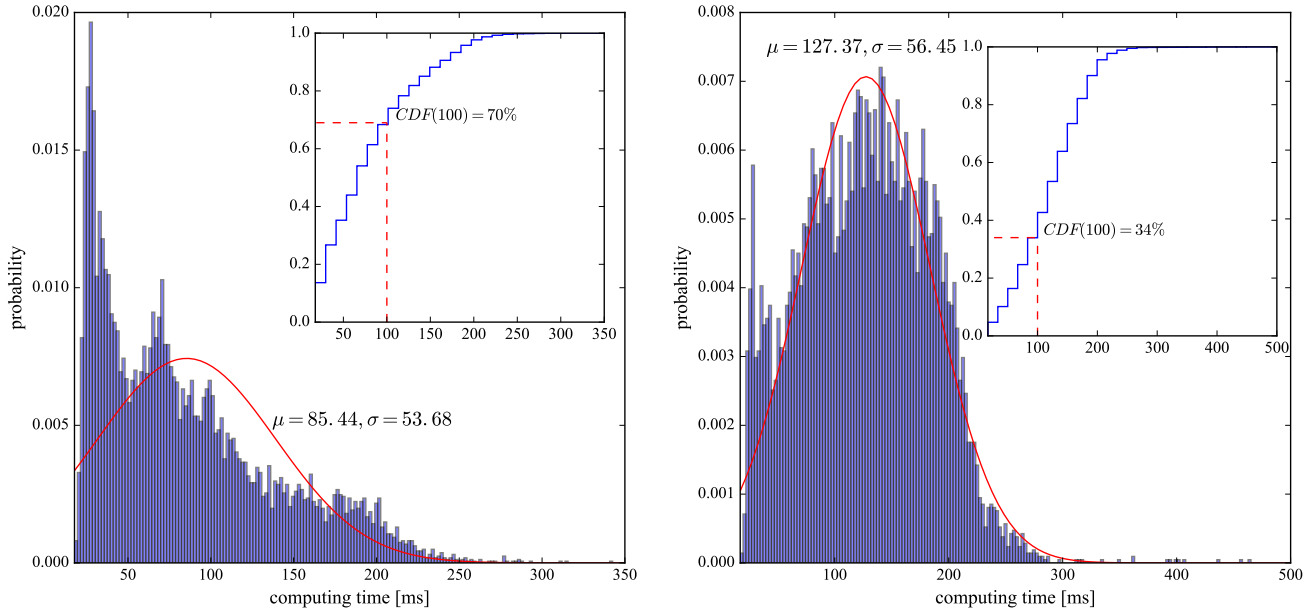
For the second question, Figure 6.8 clearly shows that with the size of the used data sets increased, the prediction accuracy of both position and velocity is improved. This figure reveals the promising application of our method in the future when collected trajectory data substantially increase. It can be inferred that the accuracy performance can be further improved if more data sets were

utilized. After all, achieving that significant accuracy performance, only three data sets were used at most.

For the first question, in Figure 6.8, an important point that should be noticed is that in the worst cases where just one data set was used, the prediction performance of our method (the blue curves) was acceptable, compared with the reported accuracy by Kawasaki & Tasaki (2018). This also proves that our method is not data-consumed. The more data sets utilized, the better our approach performs.

6.5.3 Efficiency Performance Evaluations

In most of the presented studies, efficiencies of algorithms were rarely discussed due to most learning-based methods not being lightweight. In our previous work, it was forecasted that with the amount of computations increased, EKF that has almost the same accuracy as UKF may remarkably outperform UKF in efficiency (Tao, Watanabe, Yamada, & Takada, 2021). This experiment was designed to investigate the efficiency of our method; on the other hand, we want to validate our previous forecast. In the below experiments, both EKF and UKF predictors were implemented; they cooperated with the same CTRV model and w_3 weighting function and were tested on the same three data sets. Each experiment was repeated three times and the computing-time statistics are presented in Figure 6.9.



(**a**) (**b**)

Figure 6.9: Computing-time comparisons. The histograms and corresponding fitted normal distribution curves of computing time using EKF and UKF are given. μ is mean value and σ is standard deviation. In the upper-right boxes, the CDFs of computing time are given. The value of CDF(100) is the rate that the prediction is completed in real time. (**a**) The EKF predictor. (**b**) The UKF predictor.

As the figures show, the EKF predictor (mean computing time: 85 ms) is obviously faster than the UKF predictor (mean computing time: 127 ms). As our vehicle state estimator works at 10 Hz, from the figures, we can roughly figure out that 70% EKF predictions are accomplished in time (CDF (100 ms) = 70%), while only 34% UKF predictions are accomplished in time (CDF (100 ms) = 34%).

In practice, the UKF predictor would lose many predictions in estimation-prediction cycles, and for some vital ADAS applications, such as collision detection, prediction data absence is a critical defect. UKF is thus not recommended in our system.

6.6 Future Work

There are several tasks remaining for future study.

- The experimental data in this chapter are limited. The proposed method

and system shall be further validated and evaluated using more datasets in different environments.

- As the above experiments indicates, the prediction accuracy is improved with the size of used data sets increased; therefore, the extreme accuracy of our method shall be investigated by increasing our data sets substantially.

- The data sets' interoperability between different vehicles and drivers shall also be examined.

- The spatial distributions of the four max error groups are shown in Figure 6.7. Unfortunately, we have not found a clear spatial distribution pattern so far. In order to improve the performance of our method, the spatial distribution patterns and the issues that affect the performance of our method shall be studied.

- With the size of collected data dramatically increased in the future, a new computing framework is demanded, as we did in chapter 5.

6.7 Conclusions

In this chapter, a novel lightweight middle-term motion prediction method was proposed. Kinematic trajectory data were the result of interactions between humans, vehicles and environments. The kinematic trajectory data were directly used in our middle-term motion prediction method and they were managed by a spatial database. A new KF framework that cooperated with the spatial database system was proposed to achieve middle-term motion prediction in real time. Our method was validated in the real world. The proposed IDW methods showed slight advantage in accuracy, compared with the AW method. The size of the used data sets impacts the accuracy performance of our method. The experiments showed that as the used data sets increased, the prediction accuracy was improved, and our method was not data-consumed. Concerning the efficiency aspect, our method

could meet real-time prediction requirements; the EKF predictor runs much faster than the UKF predictor, which hardly predicted in real time and thus was not recommended.

CHAPTER 7

SUMMARY AND FUTURE WORK

7.1 Summary and Conclusions

Middle-term vehicle motion prediction is important in DM research. This dissertation focused on the uncertainty in middle-term vehicle motion prediction and carried out three studies to cope with the uncertainty to achieve middle-term motion prediction and developed an advanced CCWS application based on DM.

Chapter 3 established a foundation for this dissertation. The most basic mathematical principles of this dissertation were elaborated in this chapter, including the derivations of four classical vehicular kinematic motion models (CV, CA, CTRV, CTRA), the derivations of the most popular technologies in state estimation (KF, EKF) and the algorithms of UKF and SRUKF.

Following that, chapter 4 conducted a comparative study to compare the accuracy and efficiency performance of UKF and EKF with different motion models, in vehicular state estimation. The models' properties and the factors affecting motion prediction have also been investigated. The experimental results indicated that UKF and EKF showed roughly the same accuracy; the only obvious difference occurs in velocity estimation which is caused by different velocity hypotheses of the CTRA and CTRV models. However, they differed significantly in efficiency. With an identical process model, EKF works faster than UKF; with identical KF form, the filter using CTRV is faster than that using CTRA. The fastest filter, EKF-CTRV, is about 2.6 times faster than the slowest, UKF-CTRA. For the application that needs to process mass data with strict real-time demands, EKF-CTRV might be an ideal choice. The velocity estimate and the motion model used affect the reliability in motion prediction. A realistic model that reflects the

real driving status generates a reliable path. A standalone motion model is not recommended for performing middle-term prediction.

Chapter 5 focused on the uncertainties in middle-term vehicle motion prediction. In this chapter, the uncertainties were quantified and applied in collision avoidance. Specifically, a new collision risk assessment method was proposed in this chapter. The method leveraged vehicular state and motion uncertainties from several aspects to predict multiple possible vehicles' trajectories to assess collision risk. Furthermore, a novel server-based architecture for CCWSs was proposed. The method and architecture were validated and evaluated through real-world experiments. Following the experimental results, our method outperformed the referenced method in terms of collision probability, TTC, and CP estimations. Our method detected a crash at approximately 4.4 s in advance. In contrast, the referenced method detected a crash (under the same experimental conditions as our method) at approximately 1.7 s in advance. As a result, our method can provide sufficient reservation time for collision avoidance, and this makes our system robust against latency and dropouts. For the communication modes, Wi-Fi has much lower latency than LTE. It was validated that the performance of the proposed architecture is improved when it leveraged powerful computing resources on a cloud server, which decreased computation cost that may compensate for communication costs in the future. The proposed architecture can be seen as a new and feasible option for CCWS design.

Chapter 6 tried to reduce the uncertainties in middle-term vehicle motion prediction. In this chapter, the spatial kinematic trajectory data was novelly integrated in EKF and spatial database frameworks. To be specific, a novel lightweight middle-term motion prediction method was proposed. Kinematic trajectory data were the result of interactions between humans, vehicles and environments. The kinematic trajectory data were directly used in our middle-term motion prediction method and they were managed by a spatial database. A new KF framework that cooperated with the spatial database system was proposed to achieve middle-term

motion prediction in real time. Our method was validated in the real world. The proposed IDW methods showed slight advantage in accuracy, compared with the AW method. The size of the used data sets impacts the accuracy performance of our method. The experiments showed that as the used data sets increased, the prediction accuracy was improved, and our method was not data-consumed. Concerning the efficiency aspect, our method could meet real-time prediction requirements; the EKF predictor runs much faster than the UKF predictor, which hardly predicted in real time and thus was not recommended.

In summary, this dissertation focused on the uncertainty in middle-term vehicle motion prediction and tries to solve it based on DM. Three studies were carried out. Chapter 4 investigated the properties of involved tools. It was found that EKF and UKF had roughly identical accuracy performance; however, EKF was much faster than UKF. The EKF-CTRV was suggested for the application that needed to process mass data with strict real-time demands and it was adopted and validated in spatial-trajectory-based middle-term motion prediction algorithm in chapter 6. Chapter 5 contributed to quantifying the uncertainties in middle-term motion prediction. A deterministic current-state-centered multidimensional sampling method was proposed to handle the so-called state and motion uncertainties. The uncertainties were novelly used in vehicle collision risk assessment through the proposed sigma trajectory. In addition, a new server-based CCWS architecture was built based on DM in this chapter. Both the proposed method and architecture were validated and evaluated in real world to show our superiority. Chapter 6 was devoted to reducing the uncertainties through spatial kinematic trajectory data based on a basic geographical law, novelly. An EKF and spatial database framework was constructed to integrate the kinematic trajectory data in middle-term motion prediction to reduce the uncertainties. The method achieved reliable accuracy with real-time ability.

7.2 Future Work

There are some interesting subjects that are deserved to be studied in the future.

7.2.1 AI-based Prediction Framework

There are many AI-based vehicle motion prediction approaches in literature. However, the research of AI based middle-term motion prediction under DM framework is lacked. In the future, AI-based middle-term motion prediction shall be studied and a new computation framework shall be constructed. Different with current methods that are based on a standalone computer in local, the new prediction framework shall fully make use of DM, especially its geographically distributed architecture, namely, building a cooperative and location-based AI system.

Specifically, it is suggested that the four types of information in DM–road maps, static, dynamic and prediction information–shall be extended to include in historical data, which has shown their superiority in motion prediction in chapter 6. To be pointed out, it would be better to store the historical data in the edge server of DM, in which an AI-based prediction model is learned and by which an AI-based prediction service is provided as what is done in chapter 5.

7.2.2 Application in Safety

Safety is an important application area of middle-term motion prediction. Besides the CCWS developed in chapter 5, some interesting and vital applications shall be developed. With longer distance range, some present ADAS applications can be upgraded through our middle-term prediction algorithm, such as ultra-distance rear-end collision detection and ultra-distance lane departure warning. Especially, they shall be developed cooperatively by using DM.

7.2.3 Application in Security

Latency, dropout and offline harm traffic system's security in IoV environment. Predicted vehicle motion/trajectory naturally tolerates these problems. The longer time range of middle-term motion prediction can effectively enhance the robustness of the system towards latency, dropout and offline problems, as what is reported in chapter 5. Therefore, middle-term vehicle motion prediction can be used to improve the system's survivability in security area.

On the other hand, the uncertainties that associated with future vehicle state are reduced by our middle-term vehicle motion prediction algorithms; they thus can be used to provide a confidence evaluation for some systems, such as fake vehicle state detection for vehicle security application.

7.2.4 Application in Traffic Management

Currently, most traffic management systems are macroscopic because it is difficult to obtain accurate and long-term vehicle motion profiles in microscopic viewpoint. In macroscopic traffic system, vehicles are regarded as flow, hence, it is hard to manage a single or several vehicles microscopically. DM and middle-term motion prediction make microscopic traffic management possible. A typical application is vehicle merging system that helps two vehicles merge in one road safely and comfortably. With higher accuracy and longer range of middle-term motion prediction, the performance of such microscopic traffic management systems can be further improved and new applications can be developed.

7.2.5 Application in Autonomous Driving

Middle-term vehicle motion prediction is helpful for AD system. On the one hand, it can enhance the AD system's situation awareness ability due to it makes some unknown future situation being knowable. On the other hand, it can offer some crucial information for AD system, such as TTC and CPs of a upcoming collision, and these information can help the AD system re-plan safe maneuvers

to avoid a collision. In fact, all the algorithms proposed in this dissertation are implemented based on ROS and C++, and they can be directly used by *Autoware*.

BIBLIOGRAPHY

Abboud, K., Omar, H. A., & Zhuang, W. (2016). Interworking of dsrc and cellular network technologies for v2x communications: A survey. *IEEE transactions on vehicular technology*, *65*(12), 9457–9470.

Altché, F., & de La Fortelle, A. (2017). An lstm network for highway trajectory prediction. In *2017 ieee 20th international conference on intelligent transportation systems (itsc)* (pp. 353–359).

AnisKoubaa. (2019). *Services - ros wiki.* Retrieved from http://wiki.ros.org/Services

Araniti, G., Campolo, C., Condoluci, M., Iera, A., & Molinaro, A. (2013). Lte for vehicular networking: a survey. *IEEE communications magazine*, *51*(5), 148–157.

Autoware. (2022). *Autoware - github.* Retrieved from https://github.com/autowarefoundation/autoware

Azure. (2023). *Microsoft azure.* Retrieved from https://azure.microsoft.com/

Baumann, U., Guiser, C., Herman, M., & Zollner, J. M. (2018). Predicting ego-vehicle paths from environmental observations with a deep neural network. In *2018 ieee international conference on robotics and automation (icra)* (pp. 4709–4716).

Chang, S.-H., Lin, C.-Y., Hsu, C.-C., Fung, C.-P., & Hwang, J.-R. (2009). The effect of a collision warning system on the driving performance of young drivers at intersections. *Transportation research part F: traffic psychology and behaviour*, *12*(5), 371–380.

Gao, Y., Liu, X., & Dong, W. (2017). A multiple vehicle sensing approach for collision avoidance in progressively deployed vehicle networks. In *2017 ieee 25th international conference on network protocols (icnp)* (pp. 1–10).

Gauss, C. F. (1857). *Theory of the motion of the heavenly bodies moving about the sun in conic sections: A translation of gauss's" theoria motus." with an appendix.* Little, Brown.

Gelb, A., et al. (1974). *Applied optimal estimation.* MIT press.

Gindele, T., Brechtel, S., & Dillmann, R. (2015). Learning driver behavior models from traffic observations for decision making and planning. *IEEE Intelligent Transportation Systems Magazine*, *7*(1), 69–79.

Gómez, A. E., Glaser, S., Alayli, Y., Neto, A. d. M., & Wolf, D. F. (2016). Cooperative collision warning for driving assistance. In *2016 ieee 19th international conference on intelligent transportation systems (itsc)* (pp. 990–997).

Hafner, M. R., Cunningham, D., Caminiti, L., & Del Vecchio, D. (2013). Cooperative collision avoidance at intersections: Algorithms and experiments. *IEEE Transactions on Intelligent Transportation Systems*, *14*(3), 1162–1175.

Hermes, C., Wohler, C., Schenk, K., & Kummert, F. (2009). Long-term vehicle motion prediction. In *2009 ieee intelligent vehicles symposium* (pp. 652–657).

Houenou, A., Bonnifait, P., Cherfaoui, V., & Yao, W. (2013). Vehicle trajectory prediction based on motion model and maneuver recognition. In *2013 ieee/rsj international conference on intelligent robots and systems* (pp. 4363–4369).

Hussein, A., Marín-Plaza, P., Garcia, F., & Armingol, J. M. (2017). Autonomous cooperative driving using v2x communications in off-road environment. In *2017 ieee 20th international conference on intelligent transportation systems (itsc)* (pp. 1–6).

Jeong, D., Baek, M., & Lee, S.-S. (2017). Long-term prediction of vehicle trajectory based on a deep neural network. In *2017 international conference on information and communication technology convergence (ictc)* (pp. 725–727).

Jiang, R., Xu, H., Gong, G., Kuang, Y., & Liu, Z. (2022). Spatial-temporal attentive lstm for vehicle-trajectory prediction. *ISPRS International Journal of Geo-Information*, *11*(7), 354.

JochenSprickerhof. (2015). *rosbag - ros wiki.* Retrieved from http://wiki.ros.org/rosbag

Joerer, S., Segata, M., Bloessl, B., Cigno, R. L., Sommer, C., & Dressler, F. (2013). A vehicular networking perspective on estimating vehicle collision probability at intersections. *IEEE Transactions on Vehicular Technology*, *63*(4), 1802–1812.

Juan, S. H., & Cotarelo, F. H. (2015). Multi-master ros systems. *Institut de Robotics and Industrial Informatics*, 1–18.

Julier, S. J. (2002). The scaled unscented transformation. In *Proceedings of the 2002 american control conference (ieee cat. no. ch37301)* (Vol. 6, pp. 4555–4559).

Julier, S. J., & Durrant-Whyte, H. F. (2003). On the role of process models in autonomous land vehicle navigation systems. *IEEE transactions on robotics and automation*, *19*(1), 1–14.

Julier, S. J., & Uhlmann, J. K. (1997). New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition vi* (Vol. 3068, pp. 182–193).

Julier, S. J., Uhlmann, J. K., & Durrant-Whyte, H. F. (1995). A new approach for filtering nonlinear systems. In *Proceedings of 1995 american control conference-acc'95* (Vol. 3, pp. 1628–1632).

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems.

Karagiannis, G., Altintas, O., Ekici, E., Heijenk, G., Jarupan, B., Lin, K., & Weil, T. (2011). Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions. *IEEE communications surveys & tutorials*, *13*(4), 584–616.

Kato, S., Takeuchi, E., Ishiguro, Y., Ninomiya, Y., Takeda, K., & Hamada, T. (2015). An open approach to autonomous vehicles. *IEEE Micro*, *35*(6), 60–68.

Kato, S., Tokunaga, S., Maruyama, Y., Maeda, S., Hirabayashi, M., Kitsukawa, Y., ... Azumi, T. (2018). Autoware on board: Enabling autonomous vehicles with embedded systems. In *2018 acm/ieee 9th international conference on cyber-physical systems (iccps)* (pp. 287–296).

Kawasaki, A., & Tasaki, T. (2018). Trajectory prediction of turning vehicles based on intersection geometry and observed velocities. In *2018 ieee intelligent vehicles symposium (iv)* (pp. 511–516).

Kenney, J. B. (2011). Dedicated short-range communications (dsrc) standards in the united states. *Proceedings of the IEEE*, *99*(7), 1162–1182.

Kim, J., & Kum, D. (2017). Collision risk assessment algorithm via lane-based probabilistic motion prediction of surrounding vehicles. *IEEE Transactions on Intelligent Transportation Systems*, *19*(9), 2965–2976.

Lee, M., Jo, K., & Sunwoo, M. (2017). Collision risk assessment for possible collision vehicle in occluded area based on precise map. In *2017 ieee 20th international conference on intelligent transportation systems (itsc)* (pp. 1–6).

Lefebvre, T., Bruyninckx, H., & De Schutter, J. (2004). Kalman filters for non-linear systems: a comparison of performance. *International journal of Control*, *77*(7), 639–653.

Lefèvre, S., Vasquez, D., & Laugier, C. (2014). A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH journal*, *1*(1), 1–14.

Lin, L., Li, W., Bi, H., & Qin, L. (2021). Vehicle trajectory prediction using lstms with spatial–temporal attention mechanisms. *IEEE Intelligent Transportation Systems Magazine*, *14*(2), 197–208.

Liu, J., Cai, B., Wang, Y., & Wang, J. (2013). A lane level positioning-based cooperative vehicle conflict resolution algorithm for unsignalized intersection collisions. *Computers & Electrical Engineering*, *39*(5), 1381–1398.

Liu, R., Wang, J., & Zhang, B. (2020). High definition map for automated driving: Overview and analysis. *The Journal of Navigation*, *73*(2), 324–341.

Lu, N., Cheng, N., Zhang, N., Shen, X., & Mark, J. W. (2014). Connected vehicles: Solutions and challenges. *IEEE internet of things journal*, *1*(4), 289–299.

Lu, Y., Wang, W., Hu, X., Xu, P., Zhou, S., & Cai, M. (2022). Vehicle trajectory prediction in connected environments via heterogeneous context-aware graph convolutional networks. *IEEE Transactions on Intelligent Transportation Systems*.

Lytrivis, P., Thomaidis, G., Tsogas, M., & Amditis, A. (2011). An advanced cooperative path prediction algorithm for safety applications in vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, *12*(3), 669–679.

Lyu, F., Zhu, H., Cheng, N., Zhou, H., Xu, W., Li, M., & Shen, X. (2019). Characterizing urban vehicle-to-vehicle communications for reliable safety applications. *IEEE Transactions on Intelligent Transportation Systems*, *21*(6), 2586–2602.

Madhavan, R., Kootbally, Z., & Schlenoff, C. (2006). Prediction in dynamic environments for autonomous on-road driving. In *2006 9th international conference on control, automation, robotics and vision* (pp. 1–6).

Miller, R., & Huang, Q. (2002). An adaptive peer-to-peer collision warning system. In *Vehicular technology conference. ieee 55th vehicular technology conference. vtc spring 2002 (cat. no. 02ch37367)* (Vol. 1, pp. 317–321).

NCES. (2019). *Dynamic map 2.0 consortium.* Retrieved from http://www.nces.i.nagoya-u.ac.jp/dm2/

Ndjeng, A. N., Lambert, A., Gruyer, D., & Glaser, S. (2009). Experimental comparison of kalman filters for vehicle localization. In *2009 ieee intelligent vehicles symposium* (pp. 441–446).

Petrich, D., Dang, T., Kasper, D., Breuel, G., & Stiller, C. (2013). Map-based long term motion prediction for vehicles in traffic environments. In *16th international ieee conference on intelligent transportation systems (itsc 2013)* (pp. 2166–2172).

Polychronopoulos, A., Tsogas, M., Amditis, A. J., & Andreone, L. (2007). Sensor fusion for predicting vehicles' path for collision avoidance systems. *IEEE Transactions on Intelligent Transportation Systems*, *8*(3), 549–562.

PostGIS. (2022). *Postgis.* Retrieved from http://www.postgis.org

ROS. (2022). *Robot operating system (ros).* Retrieved from http://www.ros.org

Schmidt, J., Jordan, J., Gritschneder, F., & Dietmayer, K. (2022). Crat-pred: Vehicle trajectory prediction with crystal graph convolutional neural networks and multi-head self-attention. *arXiv preprint arXiv:2202.04488*.

Schneider, R., & Georgakis, C. (2013). How to not make the extended kalman filter fail. *Industrial & Engineering Chemistry Research*, *52*(9), 3354–3362.

Schubert, R., Adam, C., Obst, M., Mattern, N., Leonhardt, V., & Wanielik, G. (2011). Empirical evaluation of vehicular models for ego motion estimation. In *2011 ieee intelligent vehicles symposium (iv)* (pp. 534–539).

Schubert, R., Richter, E., & Wanielik, G. (2008). Comparison and evaluation of advanced motion models for vehicle tracking. In *2008 11th international conference on information fusion* (pp. 1–6).

Segata, M., Vijeikis, R., & Cigno, R. L. (2017). Communication-based collision avoidance between vulnerable road users and cars. In *2017 ieee conference on computer communications workshops (infocom wkshps)* (pp. 565–570).

Sengupta, R., Rezaei, S., Shladover, S. E., Cody, D., Dickey, S., & Krishnan, H. (2007). Cooperative collision warning systems: Concept definition and experimental implementation. *Journal of Intelligent Transportation Systems*, *11*(3), 143–155.

SENSORIS. (2019). *Sensoris.* Retrieved from https://sensor-is.org/

Seo, H., Lee, K.-D., Yasukawa, S., Peng, Y., & Sartori, P. (2016). Lte evolution for vehicle-to-everything services. *IEEE communications magazine*, *54*(6), 22–28.

Shan, M., Worrall, S., & Nebot, E. (2011). Long term vehicle motion prediction and tracking in large environments. In *2011 14th international ieee conference on intelligent transportation systems (itsc)* (pp. 1978–1983).

Shangguan, A., Xie, G., Li, J., Li, X., Ji, W., Hei, X., & Ma, W. (2019). Analyzing the collision probability changed over time at intersections for autonomous vehicle. In *2019 chinese control and decision conference (ccdc)* (pp. 5467–5471).

Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, *3*(5), 637–646.

Shladover, S. E., & Tan, S.-K. (2006). Analysis of vehicle positioning accuracy requirements for communication-based cooperative collision warning. *Journal of Intelligent Transportation Systems*, *10*(3), 131–140.

Simon, D. (2006). *Optimal state estimation: Kalman, h infinity, and nonlinear approaches*. John Wiley & Sons.

St-Pierre, M., & Gingras, D. (2004). Comparison between the unscented kalman filter and the extended kalman filter for the position estimation module of an integrated navigation information system. In *Ieee intelligent vehicles symposium, 2004* (pp. 831–835).

Tan, H.-S., & Huang, J. (2006). Dgps-based vehicle-to-vehicle cooperative collision warning: Engineering feasibility viewpoints. *IEEE Transactions on Intelligent Transportation Systems*, *7*(4), 415–428.

Tao, L., Watanabe, Y., Li, Y., Yamada, S., & Takada, H. (2021). Collision risk assessment service for connected vehicles: Leveraging vehicular state and motion uncertainties. *IEEE Internet of Things Journal*, *8*(14), 11548–11560.

Tao, L., Watanabe, Y., & Takada, H. (2022). A lightweight long-term vehicular motion prediction method leveraging spatial database and kinematic trajectory data. *ISPRS International Journal of Geo-Information*, *11*(9), 463.

Tao, L., Watanabe, Y., Yamada, S., & Takada, H. (2021). Comparative evaluation of kalman filters and motion models in vehicular state estimation and path prediction. *The Journal of Navigation*, *74*(5), 1142–1160.

Tobler, W. R. (1970). A computer movie simulating urban growth in the detroit region. *Economic geography*, *46*(sup1), 234–240.

Tran, Q., & Firl, J. (2013). Modelling of traffic situations at urban intersections with probabilistic non-parametric regression. In *2013 ieee intelligent vehicles symposium (iv)* (pp. 334–339).

Tsogas, M., Polychronopoulos, A., & Amditis, A. (2005). Unscented kalman filter design for curvilinear motion models suitable for automotive safety applications. In *2005 7th international conference on information fusion* (Vol. 2, pp. 8–pp).

Tu, L., & Huang, C.-M. (2010). Forwards: A map-free intersection collision-warning system for all road patterns. *IEEE Transactions on Vehicular Technology*, *59*(7), 3233–3248.

Van Der Horst, R., & Hogema, J. (1993). Time-to-collision and collision avoidance systems. In *Proceedings of the 6th ictct workshop: Safety evaluation of traffic systems: Traffic conflicts and other measures* (pp. 109–121).

Van Der Merwe, R., & Wan, E. A. (2001). The square-root unscented kalman filter for state and parameter-estimation. In *2001 ieee international conference on acoustics, speech, and signal processing. proceedings (cat. no. 01ch37221)* (Vol. 6, pp. 3461–3464).

VNS3. (2023). *cohesive networks vns3.* Retrieved from https://www.cohesive.net/vns3/

Wan, E. A., & Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. In *Proceedings of the ieee 2000 adaptive systems for signal processing, communications, and control symposium (cat. no. 00ex373)* (pp. 153–158).

Watanabe, Y., Sato, K., & Takada, H. (2020). Dynamicmap 2.0: A traffic data management platform leveraging clouds, edges and embedded systems. *International Journal of Intelligent Transportation Systems Research*, *18*(1), 77–89.

Weiß, C. (2011). V2x communication in europe–from research projects towards standardization and field testing of vehicle communication technology. *Computer Networks*, *55*(14), 3103–3119.

Xu, X., Liu, K., Xiao, K., Ren, H., Feng, L., & Chen, C. (2018). Design and implementation of a fog computing based collision warning system in vanets. In *2018 ieee symposium on product compliance engineering-asia (ispce-cn)* (pp. 1–6).

Xu, Y., Liu, X., Cao, X., Huang, C., Liu, E., Qian, S., ... others (2021). Artificial intelligence: A powerful paradigm for scientific research. *The Innovation*, *2*(4), 100179.

Yalamanchi, S., Huang, T.-K., Haynes, G. C., & Djuric, N. (2020). Long-term prediction of vehicle behavior using short-term uncertainty-aware trajectories and high-definition maps. In *2020 ieee 23rd international conference on intelligent transportation systems (itsc)* (pp. 1–6).

Yang, C., Shi, W., & Chen, W. (2017). Comparison of unscented and extended kalman filters with application in vehicle navigation. *The Journal of Navigation*, *70*(2), 411–431.

Yoon, S., Jeon, H., & Kum, D. (2019). Predictive cruise control using radial basis function network-based vehicle motion prediction and chance constrained model predictive control. *IEEE Transactions on Intelligent Transportation Systems*, *20*(10), 3832–3843.

Zhang, K., Feng, X., Wu, L., & He, Z. (2022). Trajectory prediction for autonomous driving using spatial-temporal graph attention transformer. *IEEE Transactions on Intelligent Transportation Systems*.

Zhao, X., Jing, S., Hui, F., Liu, R., & Khattak, A. J. (2019). Dsrc-based rear-end collision warning system–an error-component safety distance model and field test. *Transportation research part C: emerging technologies*, *107*, 92–104.

Zhuang, W., Ye, Q., Lyu, F., Cheng, N., & Ren, J. (2019). Sdn/nfv-empowered future iov with enhanced communication, computing, and caching. *Proceedings of the IEEE*, *108*(2), 274–291.

APPENDIX A

List of Publications

1. Tao, L., Watanabe, Y., Yamada, S., & Takada, H. (2021). Comparative evaluation of Kalman filters and motion models in vehicular state estimation and path prediction. The Journal of Navigation, 74(5), 1142-1160.

2. Tao, L., Watanabe, Y., Li, Y., Yamada, S., & Takada, H. (2021). Collision risk assessment service for connected vehicles: Leveraging vehicular state and motion uncertainties. IEEE Internet of Things Journal, 8(14), 11548-11560.

3. Tao, L., Watanabe, Y., & Takada, H. (2022). A Lightweight Long-Term Vehicular Motion Prediction Method Leveraging Spatial Database and Kinematic Trajectory Data. ISPRS International Journal of Geo-Information, 11(9), 463.

4. Tao, L., Zhang, P., Yan, L., & Zhu, D. (2020). Automatically Building Linking Relations between Lane-Level Map and Commercial Navigation Map Using Topological Networks Matching. The Journal of Navigation, 73(5), 1159-1178.