# An Infrastructure-based Cooperative Driving Framework for Connected and Automated Vehicles

by

Zhen Yang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Civil Engineering)
in the University of Michigan
2022

Doctoral Committee:

Professor Henry X. Liu, Chair
Associate Professor Neda Masoud
Associate Professor Siqian Shen
Professor Yafeng Yin

Zhen Yang
zhenyang@umich.edu
ORCID iD: 0000-0003-3479-8463

# DEDICATION

Through a thousand miles of darkness, may we see the light from a phoenix.

# ACKNOWLEDGMENTS

First and foremost, I want to express my gratitude to my PhD advisor Prof. Henry Liu, for his guidance, instruction, and support through every stage of my PhD journey. He has been an excellent advisor, mentor, and role model for me, in both research and life, and I feel really lucky to work with him. His valuable feedback always pushes my thinking and guides my path in research. During my PhD study, we have worked on several interesting and realistic topics, including integrated control between autonomous vehicles and traffic signals, anomaly detection, and vehicle behavior prediction, which gives me a significant advance in my job application and future career path. Except for the guidance in research, Dr. Liu also helps me with my personal and professional development. I am privileged to have had the opportunity to learn from him and work with him, and I cannot imagine having a better advisor than him.

I am also thankful to Prof. Yafeng Yin, Prof. Neda Masoud, and Prof. Siqian Shen, for their guidance and valuable comments on this dissertation. I want to share my special and sincere thanks to Prof. Yin for his instructions and appreciation when I experienced difficulties in my research.

Many thanks to Prof. Yiheng Feng for all the detailed discussion in my research and for sharing his practical experience in coding and paper writing. He guided me through several projects and it is great to work with him and get his instructions when I just started my PhD study.

My sincere thanks also go to Dr. Yan Chang for her support and recommendations in my career development. She offers me many selfless help during my job application process, and I am really grateful to her.

I would also express my big thanks to all the members and alumni who have worked with me at Michigan Traffic Lab. I would like to thank Shuo and Rusheng for providing valuable comments on my research. Special thanks to Shengyin for his support and help during my PhD study. Thanks Yan, Shihong, Wai, Xintao, Xingmin, Haowei, Haojie, Zachary, Tom, Depu, Lin, Zhengxia for all the great moments during the research collaboration, group seminars, and memorable dinner parties.

I would also like to thank my friends in the Next Generation Transportation Program, Xiaotong, Zhengtian, Daniel, Mojtaba, Alex, Amir, Ethan, and Yiyang. Special thanks to Xiaotong for introducing my wife to me during my PhD study. I am indebted to my best friends, Xinpeng, Zhuo, and Shen for coloring my life during my PhD study, especially my roommate Xinpeng for five years, my best roommate forever.

Last but not least, I owe my deepest gratitude to my parents, Gailan and Kaifeng, and my wife Sunan for their love, trust, and support, unconditionally.

# TABLE OF CONTENTS

# LIST OF FIGURES

FIGURE

# LIST OF TABLES

TABLE

# LIST OF ACRONYMS

**CAV** Connected and Automated Vehicle

**AV** Autonomous Vehicle

**CV** Connected Vehicle

**RV** Regular Vehicle

**MILP** Mixed Integer Linear Programming

**LP** Linear Programming

**CACC** Cooperative Adaptive Cruise Control

**ACC** Adaptive Cruise Control

**IDM** Intelligent Driving Model

**RV** Regular Vehicle

**GPS** Global Positioning System

**LiDAR** Light Detection And Ranging

**POV** Principle Other Vehicle

**SV** Surrounding Vehicle

**BSM** Basic Safety Message

**IMU** Inertial Measurement Unit

**MSF** Multi Sensor Fusion

**RSU** Roadside Unit

**FCW** Forward Collision Warning

**OBU** Onboard Unit

**SPaT** Signal Phasing and Timing

**V2X** Vehicle to Everything

**V2I** Vehicle to Infrastructure

**V2V** Vehicle to Vehicle

**MDP** Markov Decision Process

**IRL** Inverse Reinforcement Learning

**BN** Bayesian Network

**ADE** Average Displacement Error

**DSRC** Dedicated Short-Range Communication

**VRU** Vulnerable Road User

# ABSTRACT

Trajectory planning is a key component of the Connected and Automated Vehicle (CAV) autonomy stack. It is a challenging task to plan a trajectory for a CAV that ensures safety, mobility, and comfort, especially in complex urban scenarios. In urban driving scenarios, CAVs need to interact with different road users and follow traffic rules (e.g. turning at traffic signals, yielding at a roundabout), which increases the complexity of the trajectory planning.

One potential solution to help address this challenge is to deploy smart infrastructure, which can enhance microscopic situational awareness to support the trajectory planning of CAVs. An efficient cooperative scheme between the smart infrastructure and CAVs will not only enhance the safety and mobility performance of CAVs but also improve the overall system efficiency. Therefore, in this dissertation, a cooperative planning framework is proposed that given the past trajectories of the detected vehicles, the smart infrastructure manages to provide guidance or warning to CAVs with different applications to assist their trajectory planning.

First, an integrated control framework is proposed to optimize the traffic signals in an urban arterial and provide guidance for the trajectory planning of CAVs in a mixed traffic composition of CAVs, Connected Vehicles (CVs), and Regular Vehicles (RVs). Existing studies suffer from limitations such as assuming 100% penetration rate of the CAV, centralized formulation, or limited at an isolated intersection. Thus, a combination of centralized and decentralized integrated control framework is proposed that the smart infrastructure only provides centralized trajectory planning guidance to the CAVs, and then the CAVs will plan their detailed trajectories individually. The framework is designed for a traffic corridor under mixed traffic conditions.

Second, an anomaly detection model using learning from demonstration is proposed to identify abnormal trajectories when the localization module of a Connected Vehicle (CV) or AV is under cyber attacks (e.g. Global Positioning System (GPS) spoofing, sensor attacks). Most cyber defense methods in the literature targeting GPS spoofing attacks or remote sensor attacks require access to the physical signal receivers and may not be readily available in the real-world driving environment. Instead of investigating physical GPS or LiDAR signals, this work proposes a cyber defense model that leverages domain knowledge of transportation and vehicle engineering.

Lastly, a hierarchical Principle Other Vehicle (POV) behavior prediction framework incorporating traffic signal information is proposed to predict vehicle behaviors in urban scenarios with

interactive agents. This framework includes a discrete intention prediction module and a continuous trajectory prediction module, and a mixture of offline learning and online prediction strategies are adopted to capture the difference among human drivers. Game theory is utilized to model the interaction between agents explicitly.

In summary, this thesis proposes an infrastructure-based cooperative driving framework to provide a variety of guidance or warnings to CAVs. The framework is validated in the realistic simulation or with real-world datasets. It is expected that the cooperative driving framework can be implemented in the real world to assist the trajectory planning of the CAVs, benefiting their deployment.

# CHAPTER 1

# Introduction

## 1.1 Background

Autonomous driving is an active research area and industry focus for many years. Defense Advanced Research Projects Agency (DARPA) starts the DARPA Grand Challenge to foster the development of autonomous driving via a desert racing challenge in 2004 [101]. In 2007, DARPA Urban Challenge was organized that requires autonomous vehicles to perform a series of complicated maneuvers in traffic, including merging, passing, parking [20]. In recent years, many automotive and technology companies attempts to conduct real-world deployment of autonomous driving. In 2020, Waymo, as one of the leading companies in the autonomous driving industry, became the first company to offer autonomous public ride-hailing service in the city of Phoenix, Arizona [8]. Cruise, of which General Motors owns an 80% stake, wins the first California permit to carry paying riders in AVs [3]. In May 2022, Argo AI, whose largest shareholders are Ford and Volkswagen, launches their AV testing during daytime business hours in Miami, Austin [1].

However, most real-world deployment of AVs focus on some specific scenarios or specific regions, and production-level AV still seems a distant future. Most leading companies currently only test or deploy their AV services in a few cities or even only within the downtown area of a city. On the other hand, the current deployment of the AVs is not satisfying because of the crashes or abnormal behaviors. In 2016, a Tesla Model S involves in the first fatality crash of AVs in Florida [4]. When a truck turned left in front of the Tesla AV, the Tesla vehicle failed to brake enough and ran into the truck, shown in Figure 1.1. In April 2022, a Cruise self-driving car is pulled over by a police officer for driving without headlights at night [6]. When the police officer went back to talk to his partner, the AV started to move without permission from the police officer and was pulled over again, shown in Figure 1.2.

Given the limited progress of the AV industry, a natural question arises: what are the major challenges of autonomous driving? Different researchers may have different opinions, but two major challenges are consensus, the curse of dimensionality and safety-critical scenarios. In autonomous

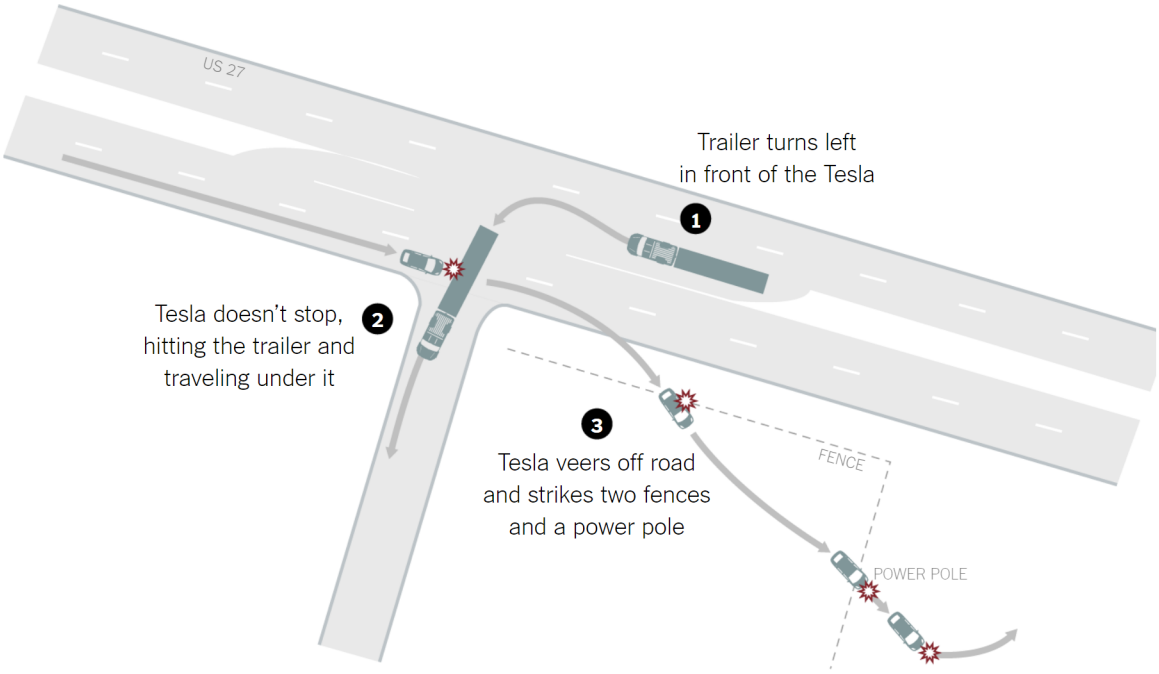



Figure 1.1: Tesla fatal crash

Figure 1.2: Cruise AV pulled by police

driving environments, different road users affect the behavior of the AV. To fully represent the complexity of the environment, the dimension of variables grows rapidly, which causes the curse of dimensionality issue [28]. How to model the environment in a computationally efficient way without losing too much optimality becomes a major concern. Besides, safety performance is currently the bottleneck of the deployment of the AVs. The crash rate of the state-of-the-art AVs is still magnitudes away from human-driven vehicle crash rate. There is still a long way to improve the safety performance of the AVs, especially under complex scenarios.

Difficulties of deployment of AVs also come from the complicated urban traffic scenarios by nature. Different from highway scenarios or open areas (e.g. parking lots), the traffic rules in urban scenarios are more complex. The AVs need to follow traffic lights at signalized intersections and obey traffic rules (e.g. yielding) at unsignalized intersections. It is not trivial to encode such rules in the autonomy stack of AVs. Besides, the AVs also interact with different road users in urban scenarios. The decision of an AV affects surrounding vehicles, and surrounding vehicles may have an influence back on the decision making of the AV. In this case, it is challenging to predict other road users' behavior and make decisions for the AVs.

Cybersecurity is another nonnegligible concern for AVs deployment. Since an AV is responsible for driving itself, it is more hazardous if the autonomous system is under cyber attack. Existing studies show that autonomous vehicles are vulnerable to cyberattacks [99, 93, 50, 83]. For example, researchers in [93] manage to adopt GPS spoofing attack to mislead the localization module of the AV so that the AV hits the road curb or goes to the wrong way. A cyber defense module is critical in the autonomous driving system to prevent the dangerous behaviors of an AV.

When it comes to cooperative driving, smart infrastructure may be helpful to address such difficulties in AVs deployment. With bird view sensors installed in the smart infrastructure, a fully observable environment can be built, and global as well as historical information of all road users can be collected. Such information can be utilized to generate guidance or security warnings to AVs to benefit their mobility or safety performance, which is the main purpose of this dissertation.

## 1.2 Connected and Automated Vehicle

Connected and Automated Vehicle (CAV)s refers to the vehicles that can communicate with other road users or smart infrastructure with connectivity, and control themselves with advanced algorithms for automation. Notice that in general, a vehicle can just be a CV that is driven by a human with communication capability or just an AV that is controlled by algorithms rather than humans without any connectivity. In this dissertation, an infrastructure-based cooperative driving framework is proposed, in which the AVs receive guidance or warnings from the smart infrastructure via connectivity. Thus, in this dissertation, it is assumed that AVs have connectivity, and AVs and

CAVs are interchangeable terminology. In this section, the connectivity and automation technologies of CAVs are introduced.

### 1.2.1 Connectivity

With Vehicle to Everything (V2X) technology, vehicles can communicate with different road users and transportation infrastructure, enhancing the safety and mobility performance of the transportation system. There are many real-world deployments of the connected vehicle environment. In 2011, the Safety Pilot Model Development project is undertaken by the U.S. Department of Transportation, to support the evaluation of Dedicated Short-Range Communication (DSRC) technology for Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) applications [18]. Built on the Safety Pilot project, in recent years, Ann Arbor Connected Vehicle Test Environment (AACVTE) becomes the world's largest real-world deployment of connected vehicles and connected infrastructure [106]. In the majority of the current deployment of the connected vehicles, the communication protocol adopts DSRC technology. The details of the DSRC standard can be found in SAE J2735 Message Set Dictionary [56].

In a CV, an Onboard Unit (OBU) is installed to broadcast Basic Safety Message (BSM)s, which includes the position, speed, acceleration, and heading information of the vehicle. At the smart infrastructure, an Roadside Unit (RSU) can be equipped to receive BSMs from CVs. At signalized intersections, RSUs can broadcast traffic signal information via Signal Phasing and Timing (SPaT) message. Other information, such as Map message, can also be provided to CVs following the DSRC standard.

With the deployment of CVs, different applications can be designed to benefit the CVs or the whole transportation system. Forward collision warning system can be designed on CVs to provide warnings to drivers when two CVs are too close to each other [47]. CVs' BSM can be adopted by the adaptive signal control system at signalized intersections to improve the mobility performance of the intelligent transportation system [30, 45].

### 1.2.2 Automation

AVs can perform specific driving tasks with automation technologies. By the definition from National Highway Traffic Safety Administration, a fully automated vehicle (i.e. level 5) drives itself, and humans are only the passengers [2]. Level 0 vehicle denotes a regular human-driven vehicle with no automation. At level 1, the vehicle can perform either longitudinal (i.e. acceleration, deceleration) or lateral (i.e. steering) driving assist features. Many commercialized vehicles with Adaptive Cruise Control (ACC) can be considered as level 1 vehicles, like the 2019 Honda Accord.

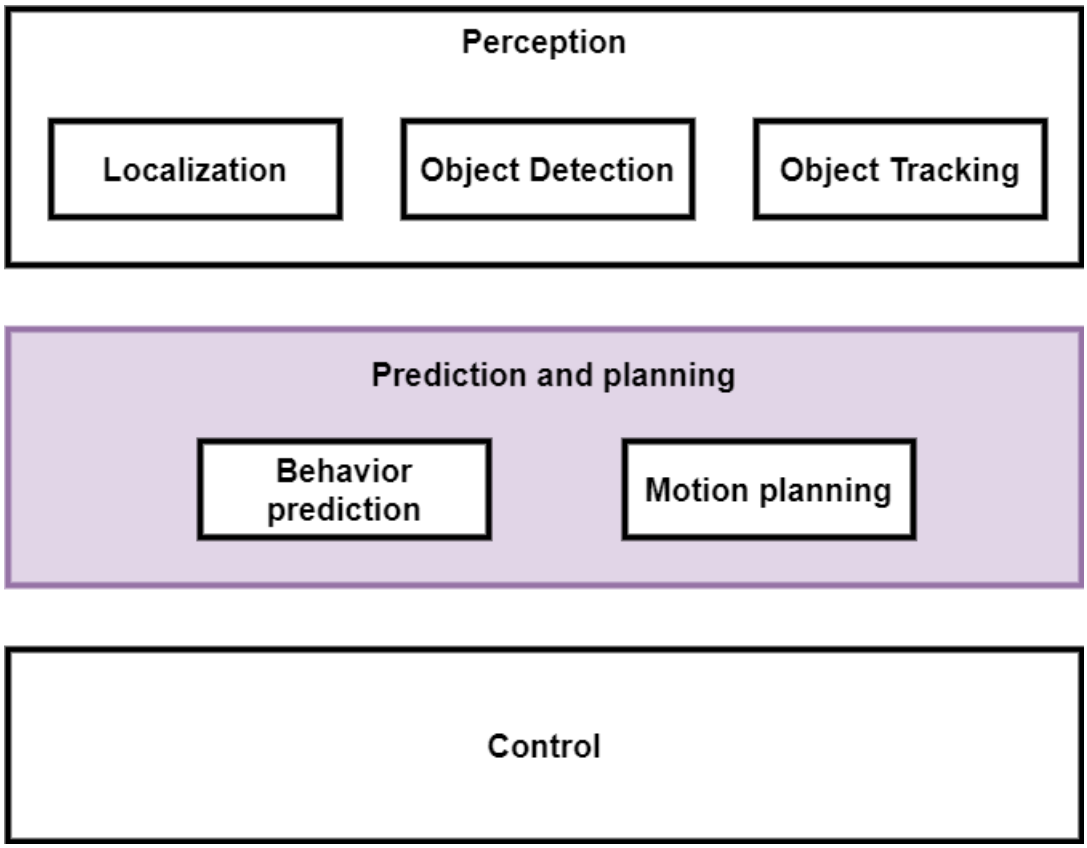



Figure 1.3: Autonomy Stack of an AV

Level 2 vehicles can provide both longitudinal and lateral driving assistance (e.g. Tesla Autopilot [7]).

Starting from level 3, there is no widely deployed commercialized AVs in the real world. Level 3 AVs can provide conditional automation that the vehicles handle all aspects of driving while the driver should be available to take over at any time. Mercedes-Benz wins the first approval for level 3 vehicles in the world [5]. Level 4 AVs can perform all driving tasks within specific service areas without any human takeover requests, and level 5 denotes fully automation in which all the driving tasks under all conditions can be performed by AVs. Numerous research and industry efforts focus on the development of level 4 and 5 autonomous driving.

Figure 1.3 illustrates the typical autonomy stack of an AV. There are three major components in the autonomy stack: perception, prediction and planning, and control.

The perception module utilizes sensors to perceive and monitor the driving environment for AVs. Common sensors equipped by AVs are cameras, Light Detection And Ranging (LiDAR)s, and radars. Although given the high price of LiDARs, most commercialized vehicles (Level 1 or 2) are equipped with cameras and radars. The perception module is responsible for a series of different tasks, in which localization, object detection, and object tracking are the most basic and

important components. Localization and mapping can locate the ego AV in global and local maps, and construct the driving environment from the object detection results [104]. Object detection can detect surrounding objects of the ego AV, including other vehicles, pedestrians, traffic signs, and road geometry. Object tracking assign the object ID to the results of the object detection, and track each object in different frames with assigned ID. More details of the perception system can be found in the surveys [14, 104, 15]. The output of the perception module is the trajectories of all agents, and each trajectory usually includes the observed position information of each agent in the map in a series of time frames, which will serve as the input of the prediction and planning module.

The prediction and planning module of the AVs is the main focus of this dissertation. With the detected object information and map information from the perception module, the behavior prediction module can predict the behavior of surrounding vehicles. The behavior prediction includes discrete intention prediction and continuous trajectory prediction [77]. Discrete intention prediction predicts the discrete intentions or maneuvers, and continuous trajectory prediction predicts a series of future positions as a predicted trajectory. The accurate behavior prediction of surrounding objects is a necessity for the safe performance of AVs, especially in complex urban scenarios with interactive agents and traffic signals. Another critical component is the motion planning module, which generates the trajectory of the ego AV in space and time in a computational efficient way. It can also be divided into two stages, discrete decision making (e.g. lane change or not) and continuous motion planning. Vehicle dynamics are usually considered at the continuous motion planning stage. The trajectory planning of the ego CAV should achieve good performance in both mobility and safety. Driving comfort and fuel economy is also important sometimes, which makes the trajectory planning process a complicated multi-object task. The trajectory planning module is also vulnerable to cyber attacks [93, 99], which may leads to serious collisions. To relieve the difficulties in the trajectory prediction and planning module, this dissertation focus on providing guidance / warnings as information to the AVs to assist their planning module.

Given the ego CAV's trajectory from the planning module, the control module is responsible for tracking the planned trajectory. During the tracking process, the role of the controller is to stabilize the planned path in the presence of tracking error or uncertainty [80]. The output of the control module is the throttle, brake, and steering of the vehicle. More details of the control techniques can be found in the surveys [80, 59].

## 1.3 Smart infrastructure and Cooperative Driving

Smart infrastructure can be defined as an interconnected sensing network that has self-awareness and self-monitoring, which can provide real-time information about the system [76]. It acquires

data in the real world and analyzes them in real time. With the sensor data, the smart infrastructure can make decisions in an autonomous way and near real time [78]. It may also provide digital information to different road users, such as vehicles, pedestrians, and cyclists. When it comes to AVs, it is expected that the smart infrastructure can provide useful guidance or warnings to AVs to assist their autonomy stack. The decision making applications (e.g. traffic signal control) in the smart infrastructure can also be integrated with autonomous driving to improve the whole transportation system efficiency.

Instead of relying on single AV intelligence, cooperative driving introduces the mechanism for multi-vehicle or vehicle-infrastructure collaboration. Connected by communication links, different traffic agents (e.g. vehicles, infrastructure) can share information with each other and make the decision jointly, targeting achieving global optimality instead of local optimality. Researchers and developers have focused on cooperative driving for a decade. In 2011, Grand Cooperative Driving Challenge is held in Netherlands to accelerate cooperative driving technology [36]. The event object is to complete the challenge quickly with V2V and V2I technology. In this dissertation, cooperative driving mainly refers to vehicle and infrastructure coordination (i.e. infrastructure-based cooperative driving). Infrastructure-based cooperative driving is the mechanism to incorporate smart infrastructure and autonomous driving to benefit the autonomy stack of AVs and improve the whole transportation system performance.

The cooperative scheme between the smart infrastructure and AVs has attracted attention from both academia and industry. The INFRAMIX project defines five levels of smart road infrastructure ([22]), from conventional road infrastructure without any information (Level E) to cooperative driving (Level A), where real-time information on vehicle movements and guidance of AVs' path planning are provided. During the upgrade of the levels, more information are provided by the smart infrastructure, such as static digital information or map support (Level D), dynamic digital information (Level C), and cooperative perception information (Level B). Federal Highway Administration deployed the CARMA project to enable further capabilities of cooperative driving automation users to interact with roadside infrastructure for research and development (R&D) purposes ([94]). The products of CARMA include CARMA Cloud, CARMA Platform, CARMA Messenger, and CARMA Street, which provide necessary software for cooperative driving research and testing.

There are also other real-world examples of cooperative driving. In 2021, BAAI-VANJEE Roadside Dataset is published, which provides a benchmark dataset for roadside perception system [119]. With birdview LiDARs and cameras installed at urban intersections and highways, LiDAR data and RGB images are captured and labeled to benefit the tasks of 2D object detection, 3D object detection, and sensor fusion in the cooperative driving research society. At a two-lane roundabout in Ann Arbor, Michigan, USA, cameras and radars are equipped at the smart infrastructure with

edge computing devices ([125]). All the road users approaching and entering the roundabout are detected and tracked, and the detected trajectories can be utilized to assist the AVs' trajectory planning ([127]).

There are several critical research questions on infrastructure-based cooperative driving. What information should be provided by the smart infrastructure to assist the CAVs? What is the communicate protocol for message delivery? How can the smart infrastructure prevent cyber attacks when delivering the message? What should be the computational structure of the cooperative driving framework? In this dissertation, the main focus is to figure out appropriate guidance or warnings that can be provided by the smart infrastructure to assist the trajectory planning of the CAVs.

## 1.4 Problem Statement

An infrastructure-based cooperative driving framework is illustrated by Figure 1.4. The key concept of this cooperative driving framework is that the smart infrastructure provides guidance or warnings to assist the CAV's trajectory planning module. The top pipeline denotes the prediction and planning module of the CAV with detected object information from on-board sensors, and the bottom pipeline represents the cooperative planning module with detected object information from infrastructure sensors. In the bottom pipeline, given the object detection results, the state estimation module estimates different road users' states and tracks their configurations in real time. The traffic signal optimization takes the object state estimation results as the input and optimizes the traffic signals. The traffic signal optimization can be integrated with CAV motion planning so that traffic signals and CAVs' trajectories can be optimized jointly, and guidance can be provided to CAV motion planning module to benefit its mobility performance. Besides, the smart infrastructure can also enhance the microscopic traffic situational awareness for CAVs by leveraging real-time and historical trajectories from a global view (i.e. bird view). Road users' behavior prediction results serve as one of the most important information of the CAV's trajectory planning module, especially for its collision avoidance. Different from the onboard sensors of CAVs, the bird view sensors installed at the smart infrastructure can observe all road users without any occlusion in a global view to provide better prediction. In addition, historical trajectories that have similar behavior patterns at a specific location (i.e. a complex signalized intersection) can be stored by the smart infrastructure, and local road users' behaviors can be better predicted with these historical trajectories. The anomaly detection module identifies abnormal trajectories caused by cyber attacks and generates warnings to the ego CAV. The collision warning module predicts if a crash or near-crash event may happen and warns the ego CAV in some high accident zones. As a supervisor, the smart infrastructure can help identify abnormal behaviors or crash events by comparing the observed behaviors of the ego CAV and of the surrounding Principle Other Vehicle (POV)s with historical

vehicle behaviors in real time. In the end, guidance and warnings can be generated from the smart infrastructure (i.e. the bottom pipeline) to serve the motion planning of CAVs.

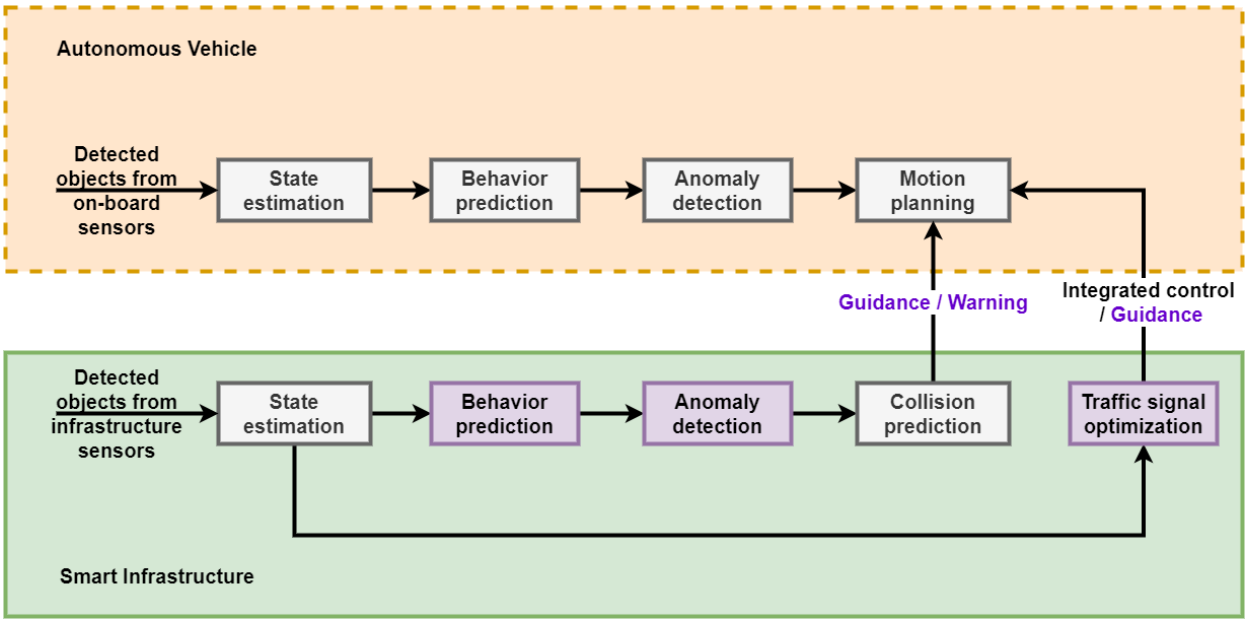



Figure 1.4: An infrastructure-based cooperative planning framework

In this dissertation, we focus on three applications (i.e. purple rectangles in Figure 1.4) of this infrastructure-based cooperative driving framework, including integrated control, anomaly detection, and POV trajectory prediction. Figure 1.5 illustrates the operating scenario of this thesis. In the figure, the red vehicle represents Regular Vehicle (RV), and the yellow vehicle represents CV. The blue vehicle denotes the ego CAV, whose trajectory planning result is shown as the blue dash line. The yellow bar in the figure denotes that the current traffic signal status is yellow. In this yellow running scenario, the RV in front of the ego CAV is a POV since its behavior may affect the trajectory planning of the ego CAV. In the integrated control application, the CAV trajectory planning (i.e. blue dash line) can be integrated with the traffic signal optimization to improve the mobility and fuel consumption performance of the whole transportation system. When it comes to the cyber security issue, a cyber attacker may attack the surrounding vehicles of the ego CAV to affect the CAV's behavior, shown as the grey dash line. The anomaly detection module can identify the abnormal vehicle trajectories and generates warnings. The POV behavior prediction module predicts the trajectories of the POV as the red dash line, and such prediction results can serve as important information to the ego CAV's trajectory planning. More details of these applications are introduced in the following paragraphs.

First, an integrated control framework is proposed to optimize the traffic signals in an urban arterial and provide guidance for the trajectory planning of CAVs in a mixed traffic composition

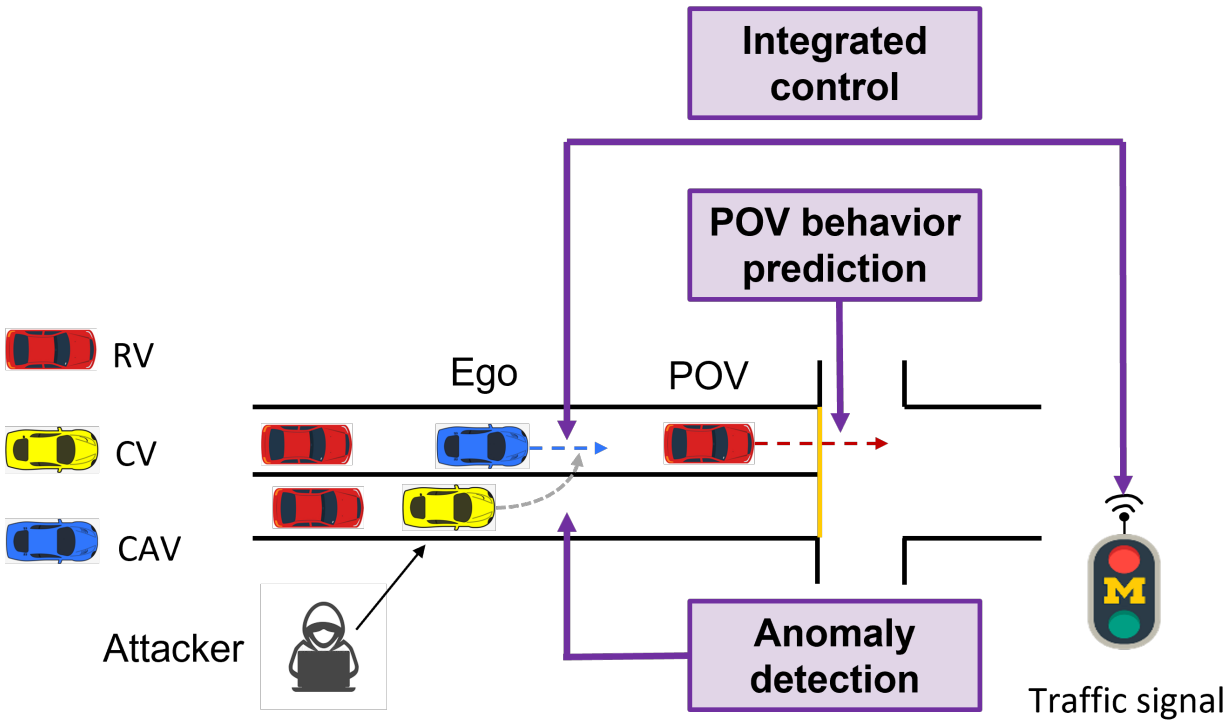



Figure 1.5: Operating scenario of this thesis

of CAVs, CVs, and RVs. The infrastructure is responsible for collecting traffic information, estimating and predicting traffic states, controlling and coordinating traffic signals, and providing high-level trajectory guidance to CAVs, with the main objective of optimizing traffic flow and improving mobility. Although great achievements have been made in the area of CAV-based intersection management, existing studies suffer from three major limitations, such as assuming 100% penetration rate of CAVs ([9], [44], [32]), centralized formulation ([68], [121], [41]), or limited at an isolated intersections ([70], [110]). This study aims to address these limitations by proposing a hierarchical and implementation-ready integrated control framework. In this work, a bi-level structure is adopted in which a centralized Corridor Coordinator is combined with decentralized intersection level controllers. In the Corridor Coordinator, offsets of coordinated phases between consecutive intersections are optimized based on link performance functions to promote two-way coordination. At each intersection, CAV trajectory guidance in terms of the time of arrival and signal parameters (i.e., green split) are jointly optimized, given offset from the Corridor Coordinator and traffic states estimated from CAVs/CVs and/or loop detector data. Optimized signal timing plans and CAV time of arrival are broadcast in the form of SPaT messages and roadside safety messages (RSMs) respectively. On the vehicle level, given the high-level guidance from the smart infrastructure, each CAV is responsible for forming ad-hoc Cooperative Adaptive Cruise Control (CACC) platoons, generating detailed vehicle trajectories, and collision avoidance with CVs

and RVs, with the main objective of ensuring safety and reducing fuel consumption and emissions.

Second, an anomaly detection model using learning from demonstration is proposed to identify abnormal trajectories when the localization module of a CV or AV is under cyberattacks (e.g. GPS spoofing, sensor attacks). Although abundant research works on the cyber defense for vehicle localization can be found in the literature ([53], [74], [88]), most defense methods in the literature targeting GPS spoofing attacks or remote sensor attacks are signal processing methods, which require access to the physical signal receivers and may not be readily available in the real-world driving environment. Instead of investigating physical GPS or LiDAR signals, the domain knowledge of transportation and vehicle engineering can be leveraged to detect the anomalies. Following this direction, this work proposes a defense model that can detect abnormal vehicle trajectories using transportation and vehicle engineering domain knowledge. The maximum entropy inverse reinforcement learning is adopted to learn the normal driving behaviors of the CV or AV from the historical trajectory dataset. To determine whether the observed CV or AV trajectories are under attack, a statistical method is developed to compare the observed trajectory with the predicted optimal trajectory based on the learned model, by training an anomaly classifier using the decision tree.

Third, a hierarchical POV behavior prediction framework incorporating traffic signal information is proposed to predict vehicle behaviors in urban scenarios with interactive agents. The urban traffic scenarios are divided into two categories, independent scenarios and interactive scenarios. A yellow light running scenario is taken as an example of the independent scenario, and a right turn scenario at the roundabout is taken as an example of the interactive scenario. The behavior prediction framework includes a discrete intention prediction module and a continuous trajectory prediction module. A Bayesian Network (BN) is adopted for the discrete intention prediction to predict the intention of the POV at each time step and the associated probability distribution. According to the results of the discrete intention prediction, continuous trajectory prediction is conducted with maximum entropy Inverse Reinforcement Learning (IRL) to produce a precisely predicted trajectory. In the interactive scenario, a potential game framework is proposed to explicitly model the interaction between vehicles. Some existing literature also adopts IRL to conduct trajectory prediction, but they only manage to learn an average driving model to make the prediction, neglecting the difference among human drivers (e.g. aggressive/mild) ([62], [98]). In this work, an average human driving model is learned by IRL offline, and during the online prediction, a driver characteristic is applied to update the model to capture the individual difference between drivers. The contribution of this work is to propose a novel hierarchical behavior prediction framework with consideration of traffic signal information and interactive agents, in which the interaction is explicitly modeled by a potential game framework. Besides, a mixture of offline learning and online prediction strategies are adopted to capture the different driving preferences

among different human drivers so that the predictor can vary to accommodate aggressive drivers or mild drivers.

## 1.5 Thesis Overview

This thesis introduces a cooperative driving framework, in which the smart infrastructure provides guidance / warnings to CAVs to assist their trajectory planning. It focuses on three applications: integrated control, anomaly detection, and POV behavior prediction. The rest of this thesis is organized as follows.

Chapter 2 introduces an integrated control framework. The smart infrastructure optimizes the traffic signals and provides high-level guidance to CAVs in a traffic corridor under mixed traffic conditions, and the CAVs plan their detailed trajectories given the guidance. The vehicle level model of the CAVs, the intersection level model of traffic signal and guidance optimization, and the corridor level traffic signal coordinator are introduced consecutively. Simulation experiments are conducted to evaluate the performance of the integrated control.

In Chapter 3, an anomaly detection method is proposed to identify abnormal trajectories when the cyberattacks of the localization module affect the trajectory planning of the CAVs. Learning from demonstration with IRL is adopted to learn the driving model of the normal driving behaviors. Then, the anomaly classifier is introduced in which a statistical method is developed to compare the observed trajectory with the predicted optimal trajectory based on the learned model. The anomaly detection method is validated on two real-world data sets for CV experiments and AV experiments, respectively.

Chapter 4 focuses on the POV behavior prediction application to predict the behaviors of the vehicles that may affect the trajectory planning of the CAVs. Both discrete intention prediction and continuous trajectory prediction of POVs are conducted, in an independent driving scenario (yellow light running scenario) and an interactive driving scenario (roundabout right turn scenario). The behavior prediction framework is evaluated in two real-world data sets collected by infrastructure sensors at urban scenarios.

Finally, the conclusions and future research directions are provided in Chapter 5.

# CHAPTER 2

# Integrated Control of Traffic Signals and CAV Trajectories

## 2.1 Introduction

### 2.1.1 Background and Related Work

Enabling technologies of CAVs bring new opportunities to the urban transportation system, especially at signalized intersections, which are considered as the bottlenecks of the traffic network. CAVs not only provide a new source of data for traffic management but also can be controlled as actuators to improve traffic flow.

Leveraging proactively broadcast data (e.g., location and speed) from CAVs as a new data source, existing studies investigate how to improve the signal control system at different scales. Different methods are developed to adjust signal timing plans dynamically based on traffic states measured from CAV data to optimize predefined performance measures (e.g., total delay, and throughput). For example, at the single intersection level, a two-level optimization problem is solved to minimize the total delay or queue length [31]. At the corridor level, a simulation-based method is introduced to predict vehicle delay with trajectory data to optimize the signals, and results show the method outperforms coordinated actuated signal control [38]. At the network level, a distributed optimization approach is proposed to control traffic signals, assuming each intersection can receive CV data and exchange information with its neighboring intersections [9].

On the other hand, as actuators, CAVs can be controlled to form compact platoons such as CACC to improve mobility [29, 27, 64]. At signalized intersections, CAVs can plan vehicle trajectories in an energy-efficient way, with the traffic signal and traffic state information (i.e., eco-driving). In [44], to obtain the optimal vehicle trajectory, a multi-stage optimal control formulation is proposed with the consideration of vehicle queue and traffic signal status, for a single CAV. [108, 112, 11] focus on eco-CACC which optimizes the trajectories of CAV platoons. Not only is the trajectory of the platoon leading CAV optimized, the platoon splitting and merging behaviors

are also considered. A field test is conducted in [11] that shows the benefits of eco-CACC.

In the past few years, researchers have proposed a new concept that integrates CAV-based signal optimization and CAV trajectory planning together, to further improve the intersection operation and reduce energy consumption. Li et al. are among the first to propose the idea of integrated optimization. In this work, CAVs follow the path optimized by signal controllers [68]. In [110], Xu et al. propose a two-level method, in which the upper level optimizes the signals and vehicle arrival time, and the lower level optimizes engine power and brake force. Yu et al. propose a comprehensive framework for the cooperative driving problem, which considers detailed signal parameters and vehicle trajectories with lane changing behavior at isolated intersections [120]. In [32], a two-stage optimization problem is formulated, in which traffic signal is optimized with dynamic programming, and vehicle trajectory is controlled based on the optimal control theory. Yu et al. [121] extend the integrated control to a corridor level. A coordinated control mechanism of CAV trajectories is developed in a centralized formulation. A Mixed Integer Linear Programming (MILP) problem is formulated to plan the complete trajectory including both longitudinal and lateral behaviors of all the CAVs through the entire corridor given origins and destinations. However, all the abovementioned studies require a 100% CAV environment. Recently, integrated optimization in mixed traffic conditions is proposed by a few researchers. [41] considers the mixed traffic of CAVs and RVs and develop a two-step control framework, in which step one optimizes the signal timing plan with consideration of vehicle trajectories, and step two designs optimal trajectories with the optimal signal plan. The study from [70] prioritizes CACC platoons at intersections to improve the overall intersection performance. The objective of signal optimization is to maximize the throughput of the intersection. Information on RVs is estimated by the location and speed of CACC vehicles, which either cruise to pass the intersection or stop at the intersection with constant deceleration.

### 2.1.2 Contribution and Organization of the Chapter

Although great achievements have been made in the area of CAV-based intersection management, existing studies suffer from three major limitations. The first limitation lies in the problem setup that in most of the studies a 100% CAV environment is assumed (e.g., [9, 44, 108, 112, 11, 68, 110, 120, 32, 121]), which requires all vehicles to be connected and highly automated (i.e., SAE Level 4 or higher). Second, usually a centralized problem is formulated (e.g., [68, 121, 41]). This requires heavily instrumented infrastructure with advanced sensors for infrastructure-based perception and high-performance computing devices for real-time computation. Neither of these can be achieved shortly. A transition period from manually driven vehicles to CAV will exist for a long time and current infrastructure with legacy sensors (e.g., loop-detectors) may even have longer

lifecycles before being replaced. Finally, most of the studies are limited at isolated intersections or simply adding up single intersections without considering coordination (e.g., [9, 70]). However, an extensive literature has shown the importance of coordination at signalized corridors.

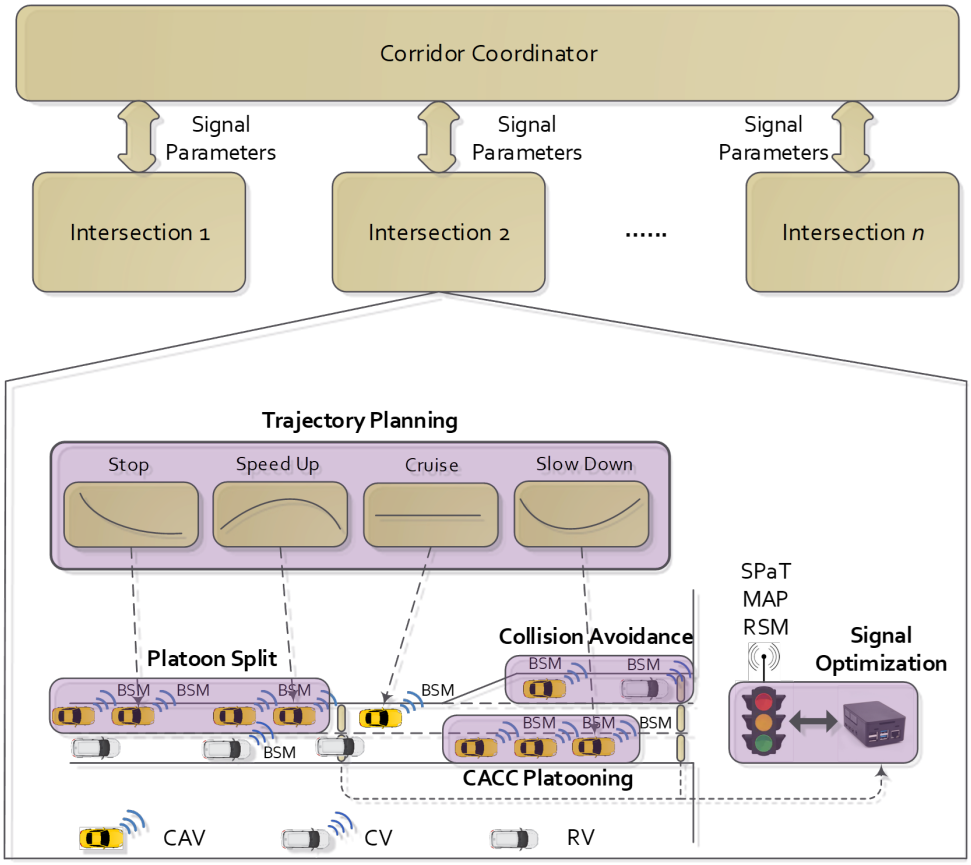



Figure 2.1: Overview of the Integrated Control Framework

This study aims to address these limitations by proposing a hierarchical and implementation-ready cooperative driving framework with a mixed traffic composition of CAVs, CVs, and RVs. In our study, CAVs refer to vehicles that proactively broadcast information and can be controlled to generate specified trajectories. CVs and RVs are driven by human drivers, which are not controllable. However, CVs proactively broadcast information to other vehicles and the infrastructure, while RVs can only be observed by infrastructure detectors at fixed locations. A conceptual overview of the proposed cooperative driving framework is shown in Figure 2.1. The infrastructure is responsible for collecting traffic information, estimating and predicting traffic states, controlling and coordinating traffic signals, and providing high-level trajectory guidance to CAVs, with the main objective of optimizing traffic flow and improving mobility. A bi-level structure is adopted in which a centralized Corridor Coordinator is combined with decentralized intersection level controllers. In the Corridor Coordinator, offsets of coordinated phases between consecutive intersec-

Table 2.1: Comparison between the proposed model and existing studies

| Literature | Traffic Signal Opt. | Vehicle Trajectory Opt. | CACC Platoon-ing | Isolated Intersec-tion | Corridor / Network Level | 100% CAV PR | Mixed Traffic Condition |
|---|---|---|---|---|---|---|---|
| [31] | X | | | X | | | X |
| [38] | X | | | | X | | X |
| [9] | X | | | | X | X | |
| [44] | | X | | X | | X | |
| [108, 112, 11] | | X | X | X | | X | |
| [68, 110] [120, 32] | X | X | | X | | X | |
| [41, 69, 54] | X | X | | X | | | X |
| [121] | X | X | | | X | X | |
| [70] | X | X | X | X | | | X |
| Our Study [115] | X | X | X | | X | | X |

tions are optimized based on link performance functions to promote two-way coordination. At each intersection, CAV trajectory guidance in terms of the time of arrival and signal parameters (i.e., green split) are jointly optimized, given offset from the Corridor Coordinator and traffic states estimated from CAVs/CVs and/or loop detector data. Optimized signal timing plans and CAV time of arrival are broadcast in the form of SPaT messages and roadside safety messages (RSMs) respectively. On the vehicle level, each CAV is responsible for forming ad-hoc CACC platoons, generating detailed vehicle trajectories, and collision avoidance with CVs and RVs, with the main objective of ensuring safety and reducing fuel consumption and emissions. Meanwhile, CAVs and CVs broadcast BSMs.

The main features of the proposed framework include 1) It is designed for mixed traffic conditions, where CAVs, CVs, and RVs co-exist on the roadway. 2) It applies to the corridor level with multiple intersections. Coordination between intersections is explicitly modeled. 3) It is implementation-ready and does not require highly automated vehicles nor heavily instrumented infrastructure. Current commercial vehicles with advanced driving assistance systems (ADAS) such as GM Super Cruise and current infrastructure with loop detectors are sufficient to deploy the proposed cooperative driving function. Other than that, only a wireless communication system (e.g., Dedicated Short Range Communication, cellular network, or hybrid) is required at both vehicle and infrastructure sides. The V2X communication network has been implemented and tested extensively in the past few years. 4) The proposed framework combines centralized and distributed control concepts, where the infrastructure provides high-level trajectory guidance to the CAVs while detailed trajectories are generated by each vehicle. This design can distribute the computational burden to achieve real-time performance, without expensive investments such as

high performance computing. The major differences between the proposed framework and some existing studies are summarized in Table 2.1.

In the following, we will first introduce the overall system structure, and data flow of the proposed framework (Section 2.2), followed by detailed modeling approaches of the three levels: vehicle-level (Sections 2.3), intersection-level (Sections 2.4), and corridor-level (Sections 2.5). Then we will show simulation experiment results and comparison to the state-of-practice intersection management method to demonstrate the improvements and benefits (Section 2.6). Finally, Section 2.7 concludes the chapter and lays out further research directions. While notations are introduced in each section, a list of complete notations is provided here for readers' convenience.

Table 2.2: Variables and Notations of Integrated Control

| Variables | Meaning | Unit |
|---|---|---|
| General Notations | | |
| $i$ | Intersection index. | |
| $j$ | Lane index. | |
| $\phi$ | Signal phase index. A lane-to-phase mapping $\phi = f(j)$ maps a lane to its corresponding phase. In the ring-barrier diagram, $\phi_{\beta\gamma\sigma}$ is used to indicate that the phase belongs to ring $\beta$, barrier $\gamma$. When $\sigma = 1$, the phase is the lead phase, and when $\sigma = 0$, the phase is the lag phase. | |
| $k$ | Vehicle group index within a lane. A CAV platoon, a single CV or a single RV is considered as a vehicle group. | |
| $s$ | Vehicle position index within a vehicle group. For a single CV or RV, $s = 1$. | |
| Vehicle level model Parameters | | |
| $v$ | Vehicle current speed | $m/s$ |
| $\Delta d$ | Distance to the front vehicle. The front vehicle is defined as the immediate downstream vehicle. | $m$ |
| $\tilde{v}$ | Speed of the front vehicle. | $m/s$ |
| $\tilde{a}$ | Acceleration of the front vehicle. | $m/s^2$ |
| $a^+_{max}$ | Maximum acceleration | $m/s^2$ |
| $a^-_{comfort}$ | Comfortable deceleration | $m/s^2$ |
| $\bar{v}$ | Average speed during the trigonometric trajectory planning period | $m/s$ |

| | | |
|---|---|---|
| $v^0$ | Initial vehicle speed of the trigonometric eco-trajectory planning period | $m/s$ |
| $a^0$ | Initial acceleration of the trigonometric eco-trajectory planning period | $m/s^2$ |
| $d^{stop}$ | Distance to the stop bar | $m$ |
| $t^{arr}$ | Time of arrival at the stop bar. | $s$ |
| $m, n$ | Model parameters. Detailed explanation can be found in [16] | |
| $jerk_{max}$ | Maximum allowed jerk | $m/s^3$ |
| $v^{max}$ | Speed limit. | $m/s$ |
| $\Gamma'$ | Green window during which a vehicle can pass the intersection | $s$ |
| $t^{leave}$ | Time interval from the start of the green to the time when the front vehicle leaves the intersection. | $s$ |
| $t^{queue}$ | Time when the vehicle arrives at the end of the queue | $s$ |
| $t^e$ | Earliest time the vehicle can arrive at the stop bar | $s$ |
| Intersection level model parameters | | |
| Decision Variables | | |
| $D_{jks}$ | Delay of the $s$th vehicle in vehicle group $k$ in lane $j$ | $s$ |
| $N_{jk}$ | Number of CAVs in vehicle group $k$ in lane $j$ | |
| $\theta_\phi$ | Green split of phase $\phi$. | $s$ |
| $g_\phi^{rem}$ | Remaining green time of phase $\phi$. | $s$ |
| $r_\phi^{rem}$ | Remaining red time of phase $\phi$. | $s$ |
| $u_{jks}^g$ | Binary variable that indicates whether the $s$th vehicle in vehicle group $k$ in lane $j$ can pass the intersection or not when the approaching phase $\phi = f(j)$ is green | |
| $u_{jks}^r$ | Binary variable that indicates whether the $s$th vehicle in vehicle group $k$ in lane $j$ can pass the intersection or not when the approaching phase $\phi = f(j)$ is red | |
| Parameters | | |
| $h_{CACC}$ | Headway between two CAVs (i.e., CACC headway) | $s$ |
| $h_{RV}$ | Headway between two regular vehicles (i.e., car-following headway) | $s$ |
| $d_j^{detect}$ | Distance from the loop-detector to the stop bar in lane $j$ | $m$ |

| | | |
|---|---|---|
| $\Delta t_{jks}^{detect}$ | Elapsed time after the $s$th vehicle in vehicle group $k$ in lane $j$ passes the loop detector | $s$ |
| $C$ | A common cycle length | $s$ |
| $g_{\phi}^{elps}$ | Green elapsed time of phase $\phi$ if it is green and 0 if the phase is red | $s$ |
| $r_{\phi}^{elps}$ | Red elapsed time of phase $\phi$ if it is red and 0 if the phase is green | $s$ |
| $t_{\phi}^{lost}$ | Lost time of phase $\phi$. | $s$ |
| $I_{\phi}^{g}$ | Binary parameter. 1 if phase $\phi$ is green, and 0 otherwise. | |
| $I_{jk}^{CACC}$ | Binary parameter. 1 if vehicle group $k$ in lane $j$ consists of CAVs and 0 otherwise | |
| $t_{i\phi}^{ge}$ | End of green time of the phase $\phi$ at intersection $i$ | $s$ |
| $N_{veh}$ | Number of vehicles at the intersection | |
| $N_{phase}$ | Number of phases at the intersection | |
| $N_{platoon}$ | Number of platoons at the intersection | |
| Corridor level model parameters | | |
| Decision Variables | | |
| $\xi_{i,i+1}$ | Offset of the coordinated phase from intersection $i$ to intersection $i+1$. | $s$ |
| $\alpha_{i\phi^*}$ | Arrival time of the first vehicle in the primary platoon of coordinated phase $\phi^*$ of intersection $i$. | $s$ |
| $D_w$ | Link performance function of scenario $w$. | $s$ |
| Parameters | | |
| $\bar{g}_{i\phi^*}$ | Average green split of coordinated phase $\phi^*$ of intersection $i$. (i.e., $\phi^* = 2$ for phase 2 and $\phi^* = 6$ for phase 6) | $s$ |
| $\bar{r}_{i\phi^*}$ | Average red duration of coordinated phase $\phi^*$ of intersection $i$. | $s$ |
| $t_{i\phi^*}^{start}$ | Start time of coordinated phase $\phi^*$ of intersection $i$. | $s$ |
| $\delta_{i\phi^*}$ | Upper bound of offset adjustment of coordinated phase $\phi^*$ of intersection $i$. | $s$ |
| $t_{i,i+1}^{travel}$ | Free flow travel time from intersection $i$ to intersection $i+1$. | $s$ |
| $q_{i\phi^*}$ | Current queue length of coordinated phase $\phi^*$ of intersection $i$. | $m$ |
| $q'_{i\phi^*}$ | Queue length at the start of the green of coordinated phase $\phi^*$ of intersection $i$. | $m$ |
| $V_{i\phi^*}$ | Average volume of coordinated phase $\phi^*$ of intersection $i$. | $vph$ |

| | | |
|---|---|---|
| $\Psi_{i\phi^*}$ | Number of vehicles in the primary platoon of coordinated phase $\phi^*$ of intersection $i$. | |
| $e_{i\phi^*}$ | Time point when the secondary platoon of coordinated phase $\phi^*$ of intersection $i$ is fully discharged. | $s$ |
| $p_{i\phi^*}$ | Platoon length in time (s) of coordinated phase $\phi^*$ of intersection $i$. | $s$ |
| $y_{i\phi^*}$ | V/C ratio of coordinated phase $\phi^*$ of intersection $i$. | % |
| $\zeta_{i\phi^*}$ | Right turning ratio of the approach containing the coordinated phase $\phi^*$ of intersection $i$. | % |
| $\xi_{i\phi^*}$ | Left turning ratio of the approach containing the coordinated phase $\phi^*$ of intersection $i$ | % |
| $N_{inter}$ | Number of intersections in the system | |

## 2.2 Framework Overview

In this section, the proposed system component diagram is described in detail. The system consists of three main components: roadway, intersection, and corridor as shown in Figure 2.2, which correspond to the three conceptual levels in Figure 2.1.

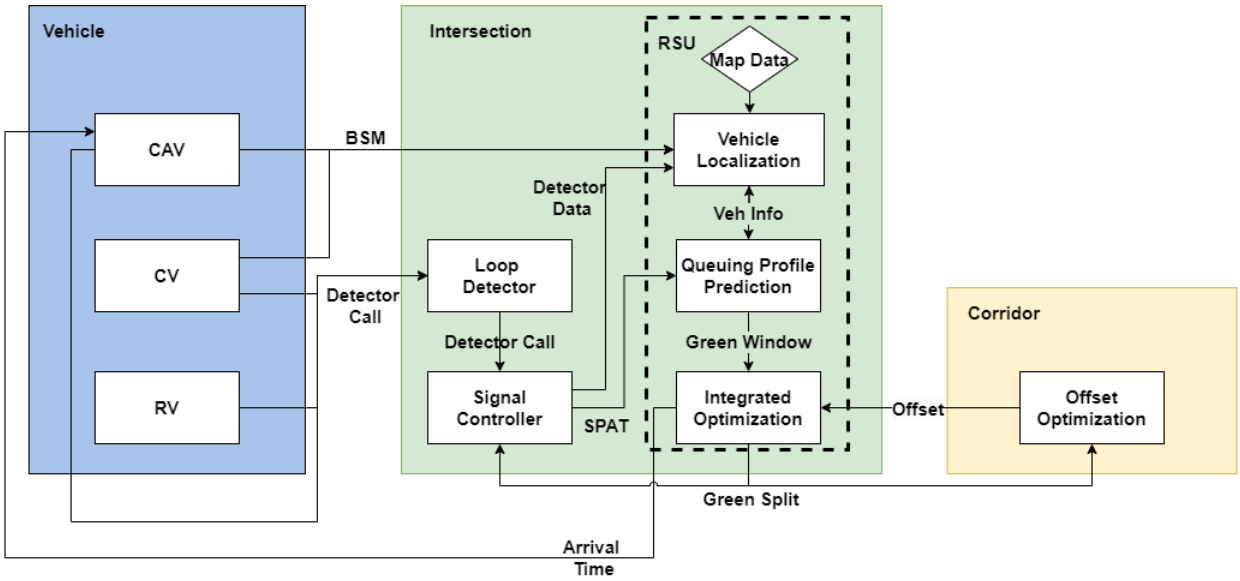



Figure 2.2: System Component Diagram

On the roadway, there are three types of vehicles: CAV, CV, and RV. All vehicles generate detector calls when they pass the loop-detectors, which are assumed to be installed at the entrance

of each link. CAVs and CVs also broadcast BSMs to enable V2V and V2I communication. The longitudinal controls of the CAV are generated by the trajectory planning models, which include eco trajectory planning, CACC platooning behaviors, and ACC. These models are introduced in section 2.3. The trajectory planning models are not responsible for lateral maneuvers. If necessary lateral maneuvers are required (e.g., turning at intersections), the system would have to give vehicle control back to the driver.

At the intersection, after the RSU receives the data (i.e., BSM and detector data), the Vehicle Localization algorithm maps the BSM and detector data on the intersection map to identify lane, approach, and approaching signal phase [30]. The Queuing Profile Prediction algorithm takes SPaT and approaching vehicle information from BSM and detectors to predict the queuing dynamics. The predicted queuing dynamics, SPaT data, and offset information from the corridor level are used to optimize the signal timing plan and the time of arrivals of CAVs, which are sent to the signal controllers to control the traffic signals and to CAV to generate speed profiles, respectively. We consider that the RSU has computation capabilities to execute the optimization model. The details of the integrated optimization at the intersection level are introduced in section 2.4. The green splits of the coordinated phases of each intersection are sent to the corridor level for offset optimization, introduced in section 2.5. The offset optimization algorithm can either reside in one of the intersections (e.g., the master intersection) or at the traffic management center (TMC). The signal controllers at the intersection broadcast SPaT data and generate loop detector data.

## 2.3 Vehicle Level Models

In this section, vehicle level models are introduced. The vehicle level models are responsible for eco-trajectory planning, ad-hoc CACC platoon formation, and collision avoidance under uncertainties. Given the time of arrival from the infrastructure side (i.e., through V2I communication), each CAV plans its own trajectory. Ad-hoc CACC platoons can be formed, and split dynamically depending on the assigned time of arrival, vehicle position in the platoon, and behaviors of uncontrolled vehicles (i.e., CVs and RVs). For the leading vehicle in a CACC platoon, a trigonometric speed profile family is adopted and one of four eco scenarios (i.e., speed up, cruise, slow down, and stop) is chosen to generate an eco speed profile. To address the uncertainties in trajectory prediction, an ACC model is applied to each CAV as a safeguard to avoid collisions. To consolidate different behaviors of a CAV, a state transition diagram is designed to represent different operating modes and their transition relations.

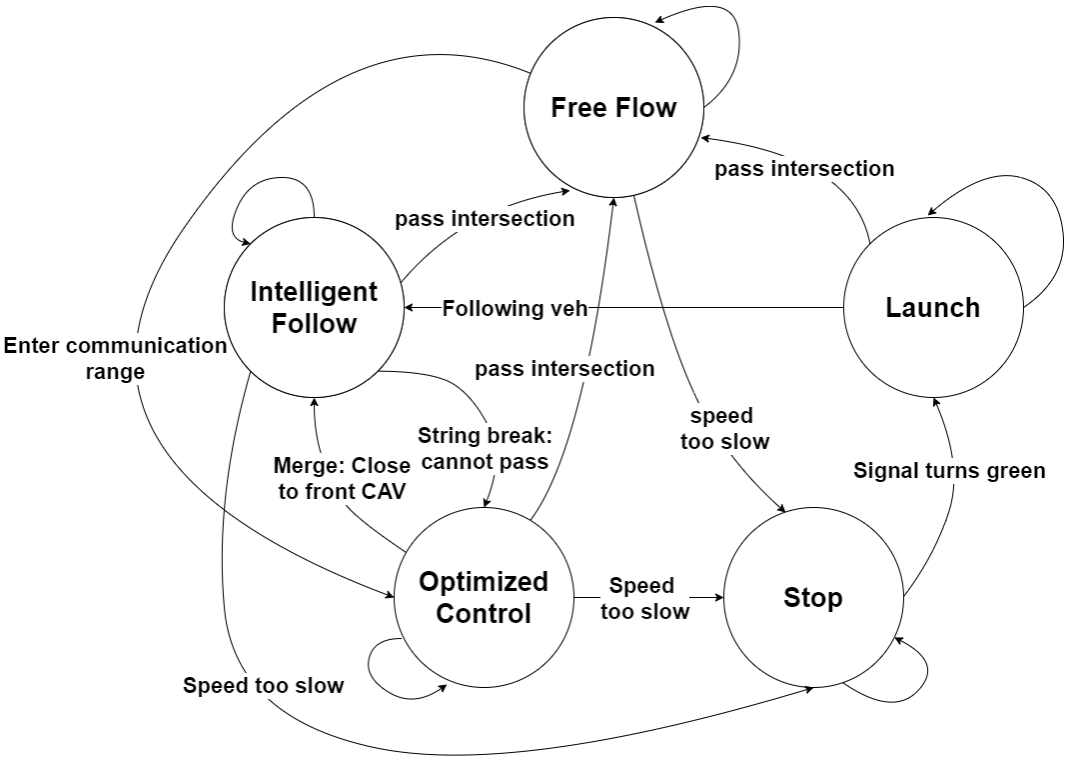
### 2.3.1 CAV States and Transitions

Five states are defined for each CAV: free flow, intelligent follow, optimized control, stop, and launch. When a CAV platoon is approaching a signalized intersection, according to different traffic and signal conditions, it may experience different operating scenarios. A CAV platoon can be formed in two ways. First, when a group of CAVs enter the network in the same lane consecutively, a platoon will be naturally formed. Second, if the distance from a CAV to its leading CAV platoon becomes small, the CAV will merge into the leading platoon. This usually happens when the leading platoon slows down for the red light, or when an RV between the CAV and the platoon makes a lane change.

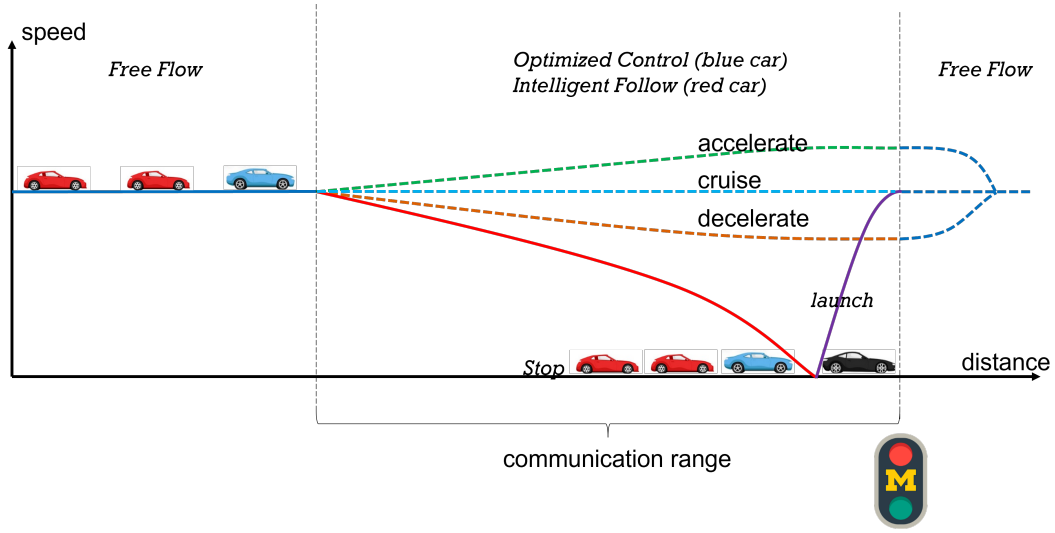
Different operating scenarios lead the CAVs to different states or transit from one state to another. The description of state transitions and corresponding triggers are shown in Figure 2.3a. The states are switched dynamically based on different criteria including vehicle speed, whether within communication range, car following distance, and traffic signal timing. To better illustrate the transitions, Figure 2.3b shows a CAV platoon of three vehicles passing the intersection as an example, which describes the most common operating scenario. Before the CAVs enter the communication range, they are in the free-flow state. When the platoon enters the communication range, the leading vehicle (blue car) switches to the optimized control state, and the following vehicles (red cars) switch to the intelligent follow state. In the optimized control state, the leading CAV can choose one of four scenarios: accelerate (green dash line), cruise (cyan dash line), decelerate (orange dash line), and plan to stop (red line). In the figure, the platoon needs to stop after an RV (i.e., black car). When a CAV stops (e.g., $v < 5mph$), it switches to the stop state. After the signal turns to green, the CAVs switch to the launch state and pass the intersection. Once the vehicle passes the intersection, it switches back to the free flow state. If a CAV doesn't need to stop, it switches directly from either optimized control or intelligent follow state to free-flow state after passing the intersection. In addition, in the intelligent follow state, if a CAV determines that its front vehicle can pass the intersection but it cannot, it switches to the optimized control state to plan a stop speed profile. Consequently, the original platoon is split into two smaller platoons. In the optimized control state or launch state, if a CAV is approaching a leading CAV platoon, it will merge into the leading platoon and switch to the intelligent follow state if the distance is smaller than a threshold (e.g., 30m).

Next, we introduce the vehicle models in each state.

In the free flow state and launch states, the vehicle is controlled by an ACC algorithm, which applies the Intelligent Driving Model (IDM) [57], shown in equation 2.1-2.2. The reference acceleration is determined by the distance to the front vehicle and the current speed of the CAV. In the equations, $\Delta d$ is the distance to the front vehicle and $d_{min}$ represents the minimum safety distance. $v^*$ is the desired speed and $a^{ref}$ is the desired acceleration. $\delta$ is a model parameter that usually is




(a) State transition diagram




(b) CAV operation scenarios

Figure 2.3: CAV Transition Diagram and Operating Scenarios

set to 4.

$$a^{ref} = a^{+}_{max}\,(1 - (\frac{v}{v^*})^{\delta} - (\frac{d^*}{\Delta d})^2) \tag{2.1}$$

$$d^* = d_{min} + vh_{ref} + \frac{v(v - \tilde{v})}{2\sqrt{a^{+}_{max}a^{-}_{comfort}}} \tag{2.2}$$

In the intelligent follow state, the CAV is a following member of a CACC platoon. The acceleration of the CAV $a^{ref}$ is calculated by a CACC model in ([103]), shown in equation 2.3, where $a^{ref_v}$ is the reference acceleration based on the speed difference and $a^{ref_d}$ is the reference acceleration based on the distance difference, calculated in equation 2.4 and equation 2.5 respectively. $d_{CACC}$ is the reference distance between two CACC vehicles. Note that in this model, "perfect following" is assumed, which ignores communication delay and string instability. The simplification is made for simulation of a large network. Readers can refer to [29, 27, 64] for more detailed CACC models.

$$a^{ref} = \min(a^{ref_v}, a^{ref_d}) \tag{2.3}$$

$$a^{ref_v} = k(v^* - v) \tag{2.4}$$

$$a^{ref_d} = k_a\tilde{a} + k_v(\tilde{v} - v) + k_d(\Delta d - d_{CACC}) \tag{2.5}$$

### 2.3.2 CACC leading vehicle trajectory planning

In the optimized control state, a trigonometric eco speed profile is generated for CACC leading vehicles. Note that a CACC platoon can consist of only one CAV. Based on the received time of arrival from the intersection controller, one of the four scenarios, shown in Figure 2.3b will be chosen. More details on how to choose the scenarios can be found in [114]. Given the terminal time $t_{arr}$ (i.e. time of arrival at the intersection), speed profiles of the other three scenarios follow the trigonometric form to reduce fuel consumption and emissions, from which analytical solutions can be derived. Collision avoidance is added to the optimized control state that when the front vehicle of a CAV is an RV or CV and the car-following distance is small (e.g. an RV makes a cut-in in front of the subject CAV), the planned eco-trajectory is interrupted, and the CAV applies the IDM, to plan its trajectory to avoid collisions. That means the CAV chooses a more conservative acceleration between the planned eco-trajectory and the IDM.

The original trigonometric eco-trajectory planning model is proposed by [16], which generates

a smooth vehicle trajectory giving terminal time (i.e., time of arrival at the intersection) to reduce fuel consumption and emissions. The terminal time ($t_{arr}$) is bounded by a green window, which defines the earliest and latest time the vehicle can arrive at the intersection, based on signal status and queue discharge time [114]. In this chapter, initial acceleration is considered, which is always assumed to be zero in the original algorithm. This original algorithm works when the CAV is cruising towards an intersection at a constant speed. However, if the trajectory planning is executed when the CAV is accelerating or decelerating or the trajectory needs to be re-planned because of an updated green window, the zero initial acceleration setting will lead to a discontinuity in the acceleration profile. To address this issue, a new segment of the trigonometric profile $v_1(t)$ is added to first bring the acceleration back to 0. Then the eco planning algorithm is applied, as shown in Eqs. 2.6-2.12.

$$v_1(t) = \frac{a_0^2}{jerk_{max}} sin(\frac{jerk_{max}}{a_0} t) + v_{init}, t \in [0, t_o) \tag{2.6}$$

$$v_2(t) = \begin{cases} v_p - v_r cos(m(t - t_o)) & t \in [t_o, t_p) \\ v_p - v_r \frac{m}{n} cos[n(t - t_o + \frac{\pi}{2n} - t_p)] & t \in [t_p, t_q) \\ v_p + v_r \frac{m}{n} & t \in [t_q, t_{arr}) \end{cases} \tag{2.7}$$

$$v_p = \frac{d^{stop} - (\frac{\pi a_0 v_{init}}{2jerk_{max}} + \frac{a_0^3}{jerk_{max}^2})}{t^{arr} - t_o} \tag{2.8}$$

$$v_r = v_p - (v_{init} + \frac{a_0^2}{jerk_{max}}) \tag{2.9}$$

$$t_o = \frac{\pi a_0}{2jerk_{max}} \tag{2.10}$$

$$t_p = t_o + \frac{\pi}{2m} \tag{2.11}$$

$$t_q = t_o + \frac{\pi}{2m} + \frac{\pi}{2n} \tag{2.12}$$

## 2.4 Intersection Level Models

The intersection controller is responsible for estimating and predicting traffic states, optimizing traffic signal parameters, and the time of arrival of CAVs. The methodology of traffic state estimation and prediction can be found in our previous work [114], which predicts the queuing dynamics

with CV and/or loop-detector data, and estimates total vehicle delay using the shockwave profile model [109] and input and output model [92]. Each vehicle is first mapped at the lane level and sorted according to its estimated distance to the stop bar. The location and speed of each vehicle are obtained directly from the BSM if it is a CV or CAV, or estimated if it is an RV, which is introduced later in equation 2.14. After the localization, the traffic state estimation algorithm utilizes shockwave models to calculate current queue length and estimate the maximum queue length and discharge time of the queue based on signal information. The prediction results not only affect the number of vehicles that will be served during green interval for each approach, but also provide a green window to the CAVs, which serves as the lower and upper boundaries of the time arrival assignments. The beginning of the green window for a CAV is defined as the time point when the CAV's front vehicle passes the intersection after the green start. The end of the green window is the same as the green end. With the estimated traffic state, a MILP problem is formulated in this section to jointly optimize traffic signal parameters and time of arrivals of CAVs.

The objective of the joint optimization is to minimize total delay and improve the efficiency of the intersection operation as shown in 2.13. $\lambda\delta_{\phi^*}$ in the objective is used as a soft constraint for dynamic offset allocation, which will be explained in the signal constraints. The decision variables include signal parameters and time of arrivals of each CAV at the intersection.

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_j \sum_k \sum_s D_{ijk} + \lambda\delta_{\phi^*} \\ \text{s.t.} \quad & D_{jks} = t_{jks}^{arr} - \frac{d_{jk1}^{stop} + (s-1)h_{CACC}v_{jk1}}{v_j^{max}} \\ & \text{Signal constraints} \\ & \text{Time of arrival constraints} \\ & \text{CACC platoon split constraints} \end{aligned} \tag{2.13}$$

Where $j$ is the lane index, and $k$ is the vehicle group index in the lane. A vehicle group can be either a platoon of CAVs, an RV, a CV, or a single CAV not in a platoon. s denotes the order of a vehicle in the vehicle group, and for a single-vehicle group, $s = 1$. $h_{CACC}$ is the CACC headway. The first constraint shows that the delay of each vehicle is equal to the total travel time minus free-flow travel time to the intersection. Notice that for the following vehicles in a CAV platoon, their free-flow travel time can be calculated based on the free-flow travel time of the leading vehicle plus the CACC headway. Other constraints include traffic signal constraints for the dual-ring barrier controller structure, time of arrival constraints that determine whether a vehicle can pass the intersection during the current cycle or next cycle, and CACC platoon split constraints that determine whether a CAV platoon should split or not. Before we introduce the details of the constraints, the estimation of $d_{jks}^{stop}$ and $v_{jks}$ is presented.

CVs and CAVs broadcast BSMs, which include accurate location and speed information (i.e. $d_{jks}^{stop}$ and $v_{jks}$) at each time step. For RVs, their locations and speeds are recorded by the entrance

loop detector when they enter the link. After RVs pass the loop detector, equation 2.14 is used to estimate the locations and speeds at each time step, assuming that the RVs are traveling with the speed limit $v_{jks}^{max}$. Where, $d_j^{detect}$ denotes the distance from the detector to the stop bar of lane $j$.

$$d_{jks}^{stop} = max[(d_j^{detect} - v_{jks}^{max}\Delta t_{jks}^{detect}, 0)] \tag{2.14}$$

### 2.4.1 Signal Constraints

Signal constraints mainly represent the standard NEMA dual-ring barrier structure, as shown in Figure 2.4.

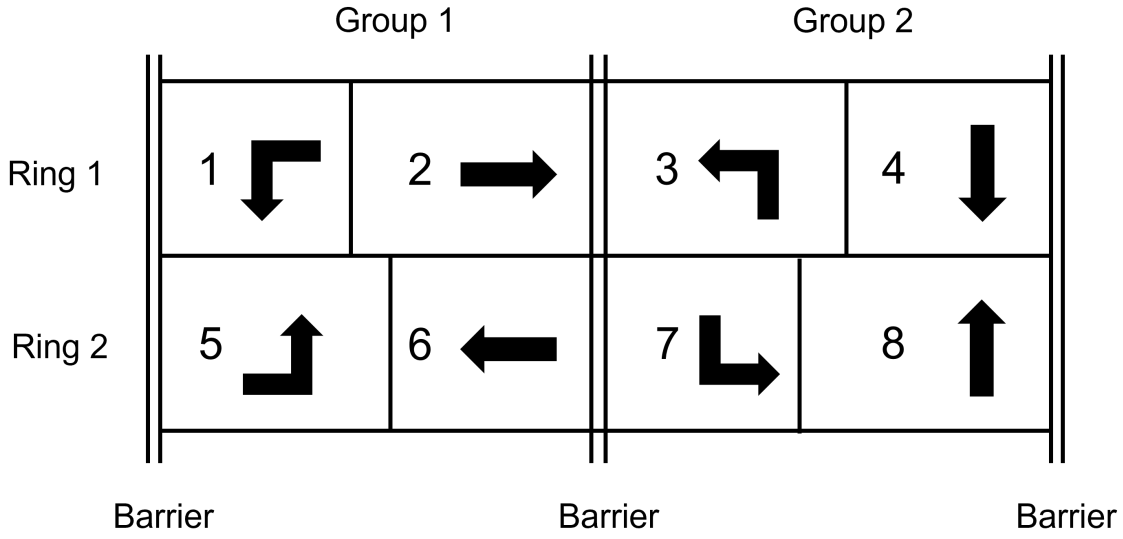



Figure 2.4: NEMA Ring Barrier Structure of Signal Timing

Equations 2.15-2.19 show the signal constraints, in which $\theta_\phi$ is the green split of phase $\phi$ and $C$ is a common cycle length. The intersections in the corridor are divided into a master intersection and other intersections. The master intersection applies equations 2.15-2.18, where the cycle length is fixed and the reference point (i.e. start of the coordinated phase) does not change. Other intersections adopt equations 2.16-2.19, in which their reference points are adjusted based on the optimized offset from the corridor coordinator. In this way, the cycle lengths of such intersections are dynamically adjusted as well, according to the assigned offsets. To better illustrate the idea, an example of the signal timing of the coordinated phase is shown in Figure 2.5. The cycle length of the master intersection is 100 seconds in this example. The coordinated phase of intersection 2 starts 3 seconds later than the coordinated phase of the master intersection, so initially, the offset is 3 seconds. After the optimization of the offset, the new optimized offset becomes 5 seconds, so the reference point of intersection 2 is adjusted accordingly. In this way, the cycle length of

intersection 2 extends to 102 seconds. Note that since the offset adjustment is small, the transition between different offsets is done within one cycle by adjusting the green split of other phases. By varying the offset and cycle length accordingly, different platoon sizes and platoon arrival times are accommodated.

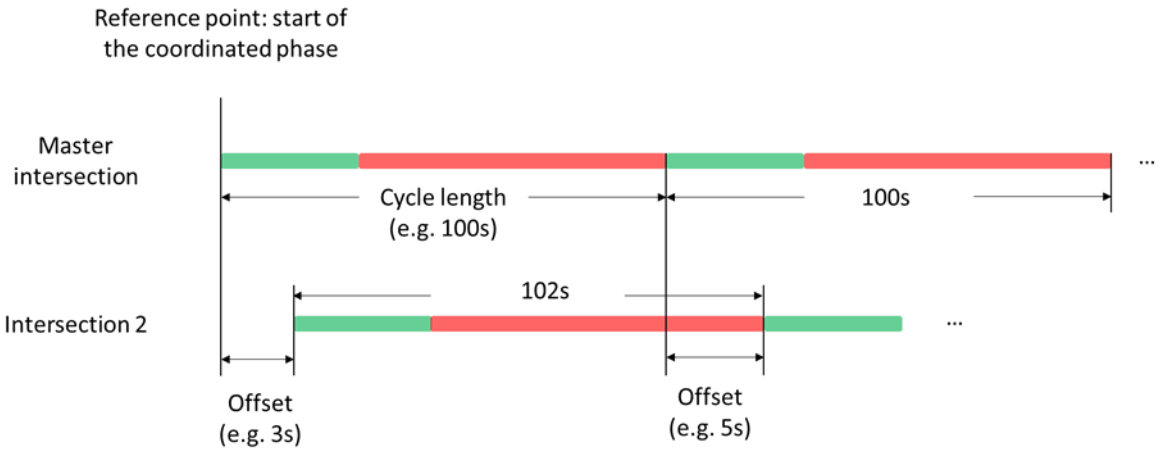



Figure 2.5: Signal Timing of the Coordinated Phase

A lane-to-phase mapping function $\phi = f(j)$ maps the lane index to 8 phases in the dual-ring structure. $g_\phi^{elps}$ denotes the elapsed green time of phase $\phi$, and if phase $\phi$ is red, $g_\phi^{elps} = 0$. $t_\phi^{lost}$ denotes the lost time of all phases, including the duration of the yellow interval and all-red interval. $I_\phi^g$ is a binary variable, which equals to 1 if phase $\phi$ is green, and 0 otherwise. Equation 2.15 shows that the summation of the green splits in each ring should be equal to the cycle length C. For each barrier, the summation of the green splits of ring 1 should be equal to the summation of the green splits of ring 2 (Eq. 2.16). Equation 2.17 shows that when phase $\phi$ is green, the summation of the remaining green time $g_\phi^{rem}$, the elapsed time $g_\phi^{elps}$, and the lost time $t_\phi^{lost}$ should be equal to the green split $\theta_\phi$. In addition, the remaining red time of phase $\phi$, $r_\phi^{rem}$ is 0. When phase $\phi$ is red, the remaining green time $g_\phi^{rem}$ is equal to 0, and the remaining red time $r_\phi^{rem}$ should be equal to the summation of the remaining green time $g_{\widetilde{\phi}}^{rem}$ of the current green phase $\widetilde{\phi}$, the lost time $t_{\widetilde{\phi}}^{lost}$, and the summation of the green split $\sum_{\bar{\phi}} \theta_{\bar{\phi}}$ of phases after $\widetilde{\phi}$ and before $\phi$ (Eq. 2.18). $\bar{\phi}$ denotes the phase(s) that turns to green before phase $\phi$ in the same cycle.

To accommodate varying traffic demands from the side street and upstream intersection, the offset is not fixed. With the offset received from the corridor-level coordinator, equation 2.19 determines the start time of the coordinated phase, which is the reference point. Taking intersection i as an example, the green start of one coordinated phase of the next cycle should be equal to the offset between intersection i-1 to intersection i (i.e. $\xi_{i-1,i}$) plus the green start of the coordinated phase of intersection $i-1$ (i.e. $t_{(i-1)\phi^*}^{start}$). $\sum_{\bar{\phi}^*} \theta_{i\bar{\phi}^*}$ is the summation of the green split from the

next phase until the coordinated phase. An adjustment bound $\delta$ is added in the objective function as a soft constraint (i.e., $\lambda\delta_{i_x\phi^*}$). As a result, the start of the coordinated phase can vary within the bound to guarantee the feasibility.

$$\sum_{\gamma}\sum_{\sigma}\theta_{\phi_{1\gamma\sigma}} = \sum_{\gamma}\sum_{\sigma}\theta_{\phi_{2\gamma\sigma}} = C \tag{2.15}$$

$$\sum_{\sigma}\theta_{\phi_{1\gamma\sigma}} = \sum_{\sigma}\theta_{\phi_{2\gamma\sigma}}, \gamma = 1, 2 \tag{2.16}$$

$$g_{\phi}^{rem} = \begin{cases} \theta_{\phi} - (g_{\phi}^{elps} + t_{\phi}^{lost}) & I_{\phi}^{g} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.17}$$

$$r_{\phi}^{rem} = \begin{cases} g_{\tilde{\phi}}^{rem} + t_{\tilde{\phi}}^{lost} + \sum_{\bar{\phi}}\theta_{\bar{\phi}} & I_{\phi}^{g} = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.18}$$

$$\xi_{i-1,\,i} + t_{(i-1)\phi^*}^{start} - \delta_{i\phi^*} \leq g_{i\tilde{\phi}}^{rem} + t_{i\tilde{\phi}}^{lost} + \sum_{\bar{\phi}^*}\theta_{i\bar{\phi}^*} \leq \xi_{i-1,\,i} + t_{(i-1)\phi^*}^{start} + \delta_{i\phi^*},\ I_{\phi^*}^{g} = 0. \tag{2.19}$$

### 2.4.2 Time of Arrival Constraints

The time of arrival constraints are mainly used to determine whether a vehicle can pass the intersection during the current cycle or the next cycle, assuming the intersection is not oversaturated. The details of the constraints are shown in equations 2.20-2.27. Two binary variables, $u_{jks}^{g}$ and $u_{jks}^{r}$, are introduced for different phase states. $u_{jks}^{g} = 1$, when the current phase of lane $j$ is green, and the vehicle can pass the intersection during the cycle, and 0 otherwise. $u_{jks}^{r} = 1$, when the current phase of lane j is red, and the vehicle can pass the intersection during the upcoming green phase, and 0 otherwise. Notice that the constraints with $u_{jks}^{g}$ (Eqs. 2.20-2.21) are only valid when the phase of lane $j$ is green (i.e. $I_{f(j)}^{g} = 1$), and the constraints with $u_{jks}^{r}$ (Eqs. 2.22-2.23) are only valid when the phase of lane $j$ is red (i.e. $I_{f(j)}^{g} = 0$).

When the signal is green, if the time of arrival of the vehicle $t_{jks}^{arr}$ is less than the remaining time of the current green phase, the vehicle can pass the intersection during the current green phase (equation 2.20). Otherwise, the time of arrival should be greater than the summation of the remaining green time $g_{f(j)}^{rem}$, the red duration of other phases, and the discharge time of the queue $t_{jks}^{leave}$ (equation 2.21). Similarly, when the signal is red if the time of arrival $t_{jks}^{arr}$ is less than $r_{f(j)}^{rem} + \theta_{f(j)}$, the vehicle can pass the intersection during the next green phase (equation 2.23), and the arrival time should be greater than $r_{f(j)}^{rem} + t_{jks}^{leave}$ (equation 2.24). Otherwise, the vehicle has to

wait for another cycle (equation 2.22). It is assumed that the traffic demand is under-saturated and no vehicle has to stop twice at an intersection (i.e., the arrival time should be less than $r^{rem}_{f(j)} + C$)). Equation 2.25 shows that the arrival time of a group leading vehicle (either CACC platoon leading vehicle or, RV/CV) should be greater than the free-flow travel time, and equation 2.26 shows that the arrival time of the group leading vehicle should be greater than the arrival time of its front vehicle $t^{arr}_{j(k-1)N_{j(k-1)}}$ plus the RV headway $h_{RV}$. Equation 2.27 shows that the discharge time of the queue $t^{leave}_{jks}$ should be equal to the number of vehicles in the queue times the headway ($h_{RV}$ or $h_{CACC}$).

$$M(1 - u^g_{jks}) \geq I^g_{f(j)}(t^{arr}_{jks} - g^{rem}_{f(j)}) \geq -Mu^g_{jks} \tag{2.20}$$

$$Mu^g_{jks} \geq I^g_{f(j)}(g^{rem}_{f(j)} + \left(C - \theta_{f(j)}\right) + t^{leave}_{jks} - t^{arr}_{jks}) \geq -M\left(1 - u^g_{jks}\right) \tag{2.21}$$

$$M\left(1 - u^r_{jks}\right) \geq \left(1 - I^g_{f(j)}\right)\left(t^{arr}_{jks} - \ r^{rem}_{f(j)} - C\right) \geq -Mu^r_{jks} \tag{2.22}$$

$$Mu^r_{jks} \geq \left(1 - I^g_{f(j)}\right)\left(\ r^{rem}_{f(j)} + \theta_{f(j)} - t^{arr}_{jks}\right) \geq -M\left(1 - u^r_{jks}\right) \tag{2.23}$$

$$t^{arr}_{jks} \geq \left(\ r^{rem}_{f(j)} + t^{leave}_{jks}\right)(1 - I^g_{f(j)}) \tag{2.24}$$

$$t^{arr}_{jk1} \geq \frac{d^{stop}_{jk1}}{v^{max}_{ij}} \tag{2.25}$$

$$t^{arr}_{jk1} \geq t^{arr}_{j(k-1)N_{j(k-1)}} + h_{RV}, \forall k > 1 \tag{2.26}$$

$$\begin{aligned} t^{leave}_{jks} \ &= \left(1 - I^g_{f(j)}\right)\left(kh_{RV}u^r_{jkN_{jk}} + \textstyle\sum_{k' \leq k}\sum_{s < N_{jk}} h_{CACC}u^r_{jk's}\right) \\ &+ I^g_{f(j)}\left(kh_{RV}(1 - u^g_{jkN_{jk}}) + \textstyle\sum_{k' \leq k}\sum_{s < N_{jk}} h_{CACC}(1 - u^g_{jk's})\right) \end{aligned} \tag{2.27}$$

### 2.4.3 CACC platoon split constraints

CACC platoon split constraints determine whether a CACC platoon needs to split or not based on the remaining green time, shown in equation 2.28 - 2.29. When the signal is green ($I^g_{f(j)} = 1$) and all CAVs in the platoon can pass the intersection during the current green phase ($u^g_{jks} = 1$), the platoon does not split and $t^{arr}_{jks} = t^{arr}_{jk(s-1)} + h_{CACC}$. Otherwise, the string splits and based on equation 2.21, the new platoon's arrival time should fulfill $t^{arr}_{jks} \geq g^{rem}_{f(j)} + \left(C - \theta_{f(j)}\right) + t^{leave}_{jks}$.

Similarly, when the signal is red ($I^g_{f(j)} = 0$), if all the CAVs in the platoon can pass the intersection during the upcoming green phase ($u^r_{jks} = 1$), then the platoon does not split, and $t^{arr}_{jks} = t^{arr}_{jk(s-1)} + h_{CACC}$. Otherwise, the platoon splits and based on equation 2.24, the new platoon's arrival time should fulfill $t^{arr}_{jks} \geq r^{rem}_{f(j)} + t^{leave}_{jks}$. Equation 2.29 shows that for the platoon leading CAV, the arrival time should be greater than the earliest arrival time given the acceleration/deceleration, and jerk limits in the trigonometric eco-trajectory planning.

$$\begin{aligned} &t^{arr}_{jk(s-1)} + h_{CACC} + I^g_{f(j)} M\left(1 - u^g_{jks}\right) + \left(1 - I^g_{f(j)}\right) M\left(1 - u^r_{jks}\right) \geq \\ &t^{arr}_{jks} \geq t^{arr}_{jk(s-1)} + h_{CACC} - I^g_{f(j)} M\left(1 - u^g_{jks}\right) - \left(1 - I^g_{f(j)}\right) M\left(1 - u^r_{jks}\right), s > 1 \end{aligned} \tag{2.28}$$

$$t^e_{jk1} - \left(1 - I^{CACC}_{jk}\right) M \leq t^{arr}_{jk1} \tag{2.29}$$

After the optimization is executed, the arrival time of each platoon leading CAV and whether the platoon needs to split are sent to the vehicle level. If the platoon does not split, the leading vehicle follows the optimized arrival time to generate its speed profile (Section 2.3.2), and the following vehicles stay in the intelligent follow state with the desired CACC headway. If the platoon needs to split, the vehicle at the split position receives its new time of arrival from the intersection controller. This vehicle serves as the leading CAV of a new platoon and switches the state to optimized control. It then generates a new speed profile based on the assigned arrival time.

## 2.5 Corridor Level Models

The corridor coordinator determines the offsets for the coordinated phases of each intersection, promoting two-way coordination. To reduce the computational burden, all vehicles in the coordinated phase are considered as one platoon and a link performance function is proposed to calculate the total delay of the vehicles in the platoon. The link performance function was first introduced in [35], but was utilized offline, without the consideration of vehicle arriving from the side street. In [17], Beak et al. implemented the link performance function in a real-time fashion. However, only one-way coordination was considered, which makes the problem much easier. In this chapter, two-way real-time coordination is proposed using the link performance function.

Figure 2.6 illustrates a three-intersection corridor with two-way coordination, and we take the eastbound through movement of intersection $i$ as an example to introduce two types of platoons for the coordinated phase. The secondary platoon, denoted by a yellow rectangle, consists of vehicles from the side streets of the upstream intersection $i-1$. The number of vehicles in the secondary platoon is calculated by the turning ratio and volume from the side street of intersection $i-1$.

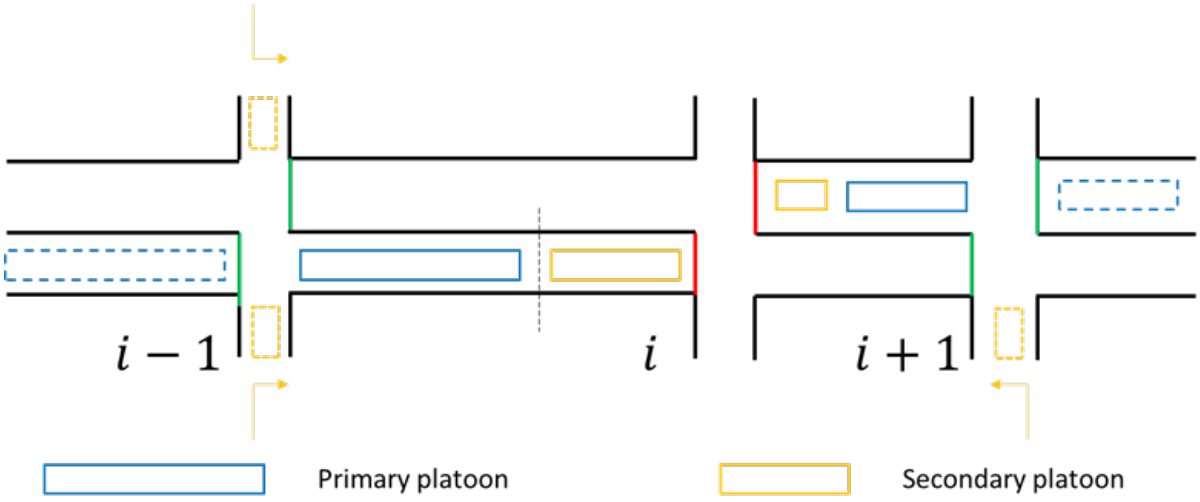



Figure 2.6: Platoons for the Coordinated Phase

The primary platoon (the blue rectangle) consists of the vehicles from the coordinated phase of the upstream intersection $i-1$. When the signal of the coordinated phase at intersection $i-1$ is red (i.e. $I^g_{\phi^*}=0$), the number of vehicles in the primary platoon $\Psi_{i\phi^*}$ of intersection $i$ can be estimated by the first case of equation 2.30, which is the summation of the current queue length ($q_{(i-1)\phi^*}$) and the estimated arriving vehicles based on the average historical demand ($V_{(i-1)\phi^*}$) of the coordinated phase. When the signal of intersection i-1 is green, the primary platoon is estimated by the second case in equation 2.30, in which $h'$ is the estimated saturation headway. When the penetration rate of the CAV is low (i.e. $<75\%$), we consider $h'=h_{RV}$. Otherwise, we consider $h'=h_{CACC}$ since CACC platoons are more likely to be generated. The primary platoon is estimated by the summation of the queue length at the start of the green $q'_{(i-1)\phi^*}$ and the demand $V_{(i-1)\phi^*}$ times the remaining green time after the queue (i.e. $q'_{(i-1)\phi^*}$) discharges. As a result, the link performance function of intersection $i$ is the total delay of the primary and secondary platoons from intersection $i-1$. Note that right turning vehicles at intersection $i-1$ and left turning vehicles at intersection i are subtracted from the total number of vehicles when calculating $\Psi_{i\phi^*}$ in equations 2.30. Fixed turning ratios (i.e. right turning ratio $\zeta_{(i-1)\phi^*}$ and left turning ratio $\xi_{i\phi^*}$, calibrated from the historical data, are assumed when estimating the lengths of primary and secondary platoons.

$$\Psi_{i\phi^*}=\begin{cases}\left(q_{(i-1)\phi^*}+\left(C-r^{elps}_{(i-1)\phi^*}\right)V_{(i-1)\phi^*}\right)(1-\zeta_{(i-1)\phi^*})(1-\xi_{i\phi^*}) & I^g_{\phi^*}=0\\ \left(q'_{(i-1)\phi^*}+\left(\bar{g}_{(i-1)\phi^*}-q'_{(i-1)\phi^*}h'\right)V_{(i-1)\phi^*}\right)\left(1-\zeta_{(i-1)\phi^*}\right)\left(1-\xi_{i\phi^*}\right) & \text{otherwise}\end{cases} \tag{2.30}$$

After the estimation of the platoon lengths, the link performance function is calculated as the total delays of the primary platoon. Notice that only delay from the primary platoon is considered

because the main purpose of coordination is to enhance the progression of coordinated phases. Figure 2.7 shows six different scenarios of link performance function calculation. In the figure, the notations are simplified, and the coordinated phase $\phi^*$ of the intersection $i$ is used for illustration. Take the second case as an example, the x-axis represents time, and the y-axis denotes the number of vehicles arrived at the intersection. The zero point of the x-axis is moved to be the start of green for better illustration. Then $-r$ is the red start and $g$ is the green end, which represent a whole cycle. $e$ is the time point when the secondary platoon is fully discharged after the signal turns to green. The length of the primary platoon (black dashed rectangle) is represented in time, and its duration $p$ is assumed to equal to the green time of the coordinated phase of the upstream intersection. Based on the calculation of the number of vehicles in the primary platoon (equation 2.30), the average headway $h$ (height of the dashed rectangle) in the figure can be calculated as $h = \Psi/p$. After the first vehicle of the primary platoon arrives at the intersection at time point $\alpha$, the primary platoon starts to accumulate delays. After the secondary platoon is fully discharged, the primary platoon starts to discharge from the intersection with the saturation flow rate. At $t_0 = e + p \times h/s$, the entire queue at the intersection is fully discharged, so the vehicles in the primary platoon that arrives later than $t_0$ do not experience any delay. The total delay of the primary platoon in the coordinated phase can be represented by the shaded area, which is the link performance function.

In subfigures 1, 2 and 3, the arrival time of the primary platoon $\alpha$ is earlier than $e$. In subfigure 1, the last vehicle in the primary platoon arrives at the intersection earlier than the time point when the queue is fully discharged (i.e. $p + \alpha \leq t_0$). Notice that although the shape of the shaded area in the two cases (i.e. $p + \alpha \leq e, p + \alpha > e$) of subfigure 1 looks different, the link performance function is the same ($LPF_1$). In subfigure 2, the last vehicle arrives at the intersection later than $t_0$ but earlier than the end of the green, while in subfigure 3, part of the platoon arrives at the intersection after the end of the green (i.e. $p + \alpha > g$), which causes extra delay in the next cycle (i.e., delay starts to accumulate at $-r$). In subfigures 4, 5 and 6, the arrival time of the primary platoon is later than $e$. In subfigure 4, the last vehicle arrives at the intersection earlier than the end of the green (i.e. $p + \alpha < g$), so no vehicle in the primary platoon experiences any delay. In subfigure 5 and 6, part of the platoon arrives at the intersection later than the end of the green, which causes extra delay, similar to subfigure 3. In subfigure 5, the first vehicle in the primary platoon arrives later than $t_0$, and in subfigure 6, the first vehicle arrives earlier than $t_0$. Based on the different situations, the link performance functions $LPF_1$ - $LPF_6$ under different scenarios are shown in equation 2.31 - 2.36.

$$LPF_1 = \frac{p^2 h(y-1)}{2} - ph\left(\alpha - e\right), \alpha \leq e \text{ and } p + \alpha \leq t_0 \tag{2.31}$$

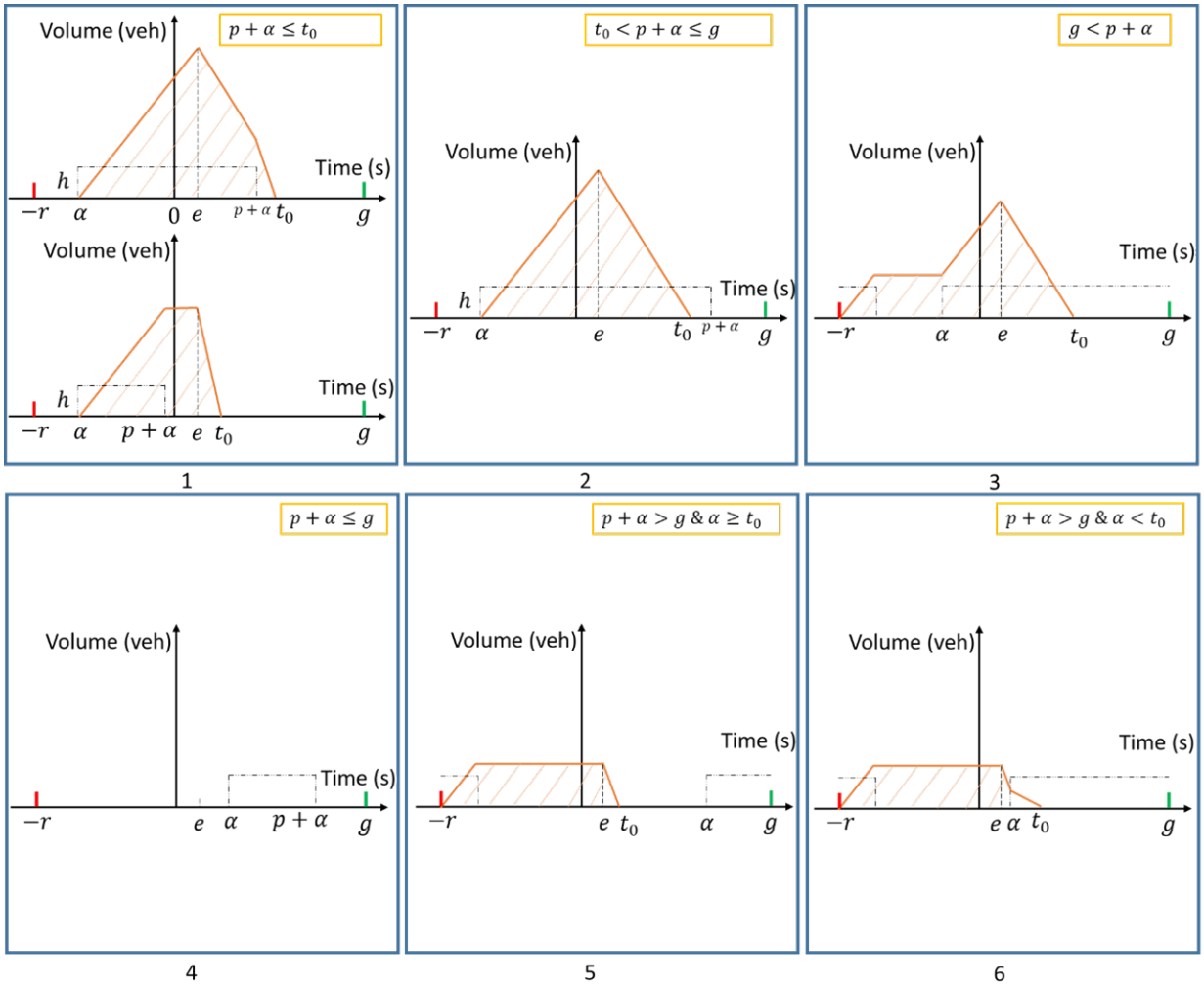



Figure 2.7: Link Performance Function of Different Scenarios

$$LPF_2 = \frac{h\left(\alpha - e\right)^2}{2(1-y)}, \alpha \leq e \text{ and } t_0 < p + \alpha \leq g \tag{2.32}$$

$$LPF_3 = \left(g + r - p\right)\left(p - g + \alpha\right)h + \frac{h\left(p - g + e\right)^2}{2(1-y)}, \alpha \leq e \text{ and } g < p + \alpha \tag{2.33}$$

$$LPF_4 = 0, \alpha > e \text{ and } p + \alpha \leq g \tag{2.34}$$

$$LPF_5 = \frac{h\left(p - g + \alpha\right)^2\left(y - 1\right)}{2} + \left(p - g + \alpha\right)h\left(r + \alpha\right), \alpha \geq t_0 \text{ and } p + \alpha > g \tag{2.35}$$

$$LPF_6 = \left(g + r - p\right)\left(p - g + \alpha\right)h + \frac{h\left(p - g + e\right)^2}{2(1-y)}, e < \alpha < t_0 \text{ and } p + \alpha > g \tag{2.36}$$

Notice that all the link performance functions are either linear or quadratic, it is easy to prove the convexity of the whole function. By aggregating the pieces of each link performance function (equation 2.31 - 2.36), we get a complete link performance function in terms of the arrival time of the first vehicle of the primary platoon (i.e. $\alpha$), as shown in Figure 2.8. In the left subfigure, the platoon length plus the discharge time of the secondary platoon $e$ is less than the green split $g$. When $\alpha$ is between $e$ and $-p$ , the vehicles in the primary platoon do not experience any delay. In the right subfigure, when $p + e > g$, the vehicles must experience some delays regardless of arrival time.

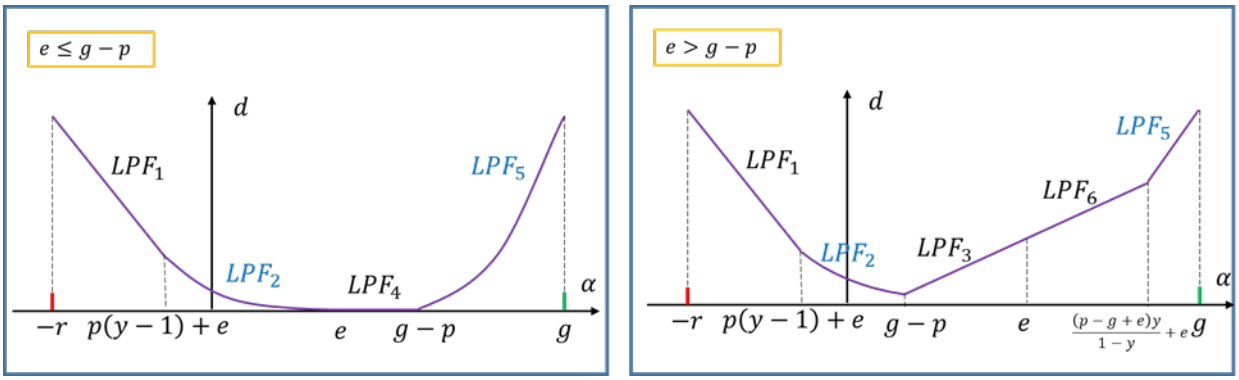



Figure 2.8: Link Performance Function in terms of the Arrival Time of the First Vehicle

By replacing the quadratic function with its linear approximation, the whole link performance

function can be expressed as a piecewise linear function. The mathematical optimization problem for the corridor-level coordinator is shown in 2.37. $\bar{g}_{i\phi^*}$ and $\bar{r}_{i\phi^*}$ denotes the average green duration and average red duration of coordinated phase $\phi^*$ of intersection $i$, based on the optimized green durations and red durations of all previous cycles from the intersection level. $t_{i-1,i}^{travel}$ is the free flow travel time from intersection $i-1$ to intersection $i$, which can be offline calibrated from the historical data. $\alpha_{i\phi^*}$ and $\xi_{i-1,i}$ are decision variables that denote the arrival time of the first vehicle in the primary platoon and the offset of the coordinated phase from intersection $i-1$ to intersection $i$, respectively. The objective is to minimize the delay of all the coordinated phases, which is estimated by the link performance functions. To formulate a convex optimization problem, the piecewise linear function can be represented by the constraints $\widetilde{D}_{i\phi^*} \geq LPF_w$ for each scenario $w$ equivalently. The offset between intersection $i-1$ and intersection $i$ is equal to the travel time on the link minus the arrival time. The arrival time should be between the start of the red and the end of green in each cycle. For two-way coordination, $\xi_{i-1,i} = -\xi_{i,i-1}$. Thus, the problem is formulated as a linear programming problem as shown in Eq. 2.37.

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_i \sum_{\phi^*} \widetilde{D}_{i\phi^*} \\ \text{s.t.} \quad & \widetilde{D}_{i\phi^*} \geq LPF_w, \ \ \forall w \\ & \xi_{i-1,i} = t_{i-1,i}^{travel} - \alpha_{i\phi^*}, \ \forall i \\ & -\bar{r}_{i\phi^*} \leq \alpha_{i\phi^*} \leq \bar{g}_{i\phi^*}, \ \forall i \\ & \xi_{i-1,i} = -\xi_{i,i-1}.i = 2 \dots N \end{aligned} \tag{2.37}$$

## 2.6 Numerical Studies

### 2.6.1 Implementation Procedure

Simulation experiments are conducted on a desktop with an Intel 3.4 GHz CPU with 4 cores and 16 GB memory. VISSIM [84] is used as the simulation environment, which replaces the roadway component in Figure 2. DriverModel.dll API is used to implement the CAV trajectory planning models. The API is also used to generate BSMs for CAVs and RVs. When CAVs have to make a lane change to realize their predefined routes, VISSIM's internal lane-changing model is executed. In addition, virtual controllers in VISSIM are used to replace the real controllers in the intersection component. The overall simulation structure remains the same.

A flow chart of the simulation implementation is shown in Figure 2.9. The purple blocks denote the CAV trajectory planning models, which generates CAV speed profiles and control the vehicles every 0.1 seconds. The green blocks denote the algorithms in the Intersection Controller module, which are executed every 3 seconds. A rolling horizon scheme is applied where the signal timing is generated for one cycle but only the first 3 seconds are implemented. The algorithms are written

in C++ and Gurobi 8.1.0 [79] is applied to solve both the MILP and Linear Programming (LP) problems. We use $N_{veh}$ to denote the number of vehicles, $N_{phase}$ to denote the number of phases, and $N_{platoon}$ to denote the number of platoons. In the MILP, the number of continuous variables is $2N_{veh}+3N_{phase}+2$, and the number of binary variables is $2N_{veh}$. The number of integer variables is $2N_{platoon}$. The number of constraints is upper bounded by $9N_{veh}+3N_{platoon}+3N_{phase}+4$, including $3N_{phase}+4$ signal constraints, $7N_{veh}+2N_{platoon}$ time of arrival constraints and $2N_{veh}+N_{platoon}$ CACC platooning constraints. Notice that $N_{platoon} = o(N_{veh})$, and the size of the problem is determined by traffic volume (i.e., $N_{veh}$) and CAV penetration rate (i.e., $N_{platoon}$). The MILP problem of the intersection level model can be solved within 0.1 seconds with 0.1% gap to the optimal solution under 100% CAV penetration rate. The yellow blocks denote the Corridor Coordinator model, which optimizes the offset every cycle. In the corridor level LP problem, there are $5N_{inter}$ continuous variables, where $N_{inter}$ denotes the number of intersections in the system. The number of constraints is upper bounded by $8N_{inter}$. The problem can be solved within 0.01 seconds with 0.1% gap to the optimal solution.

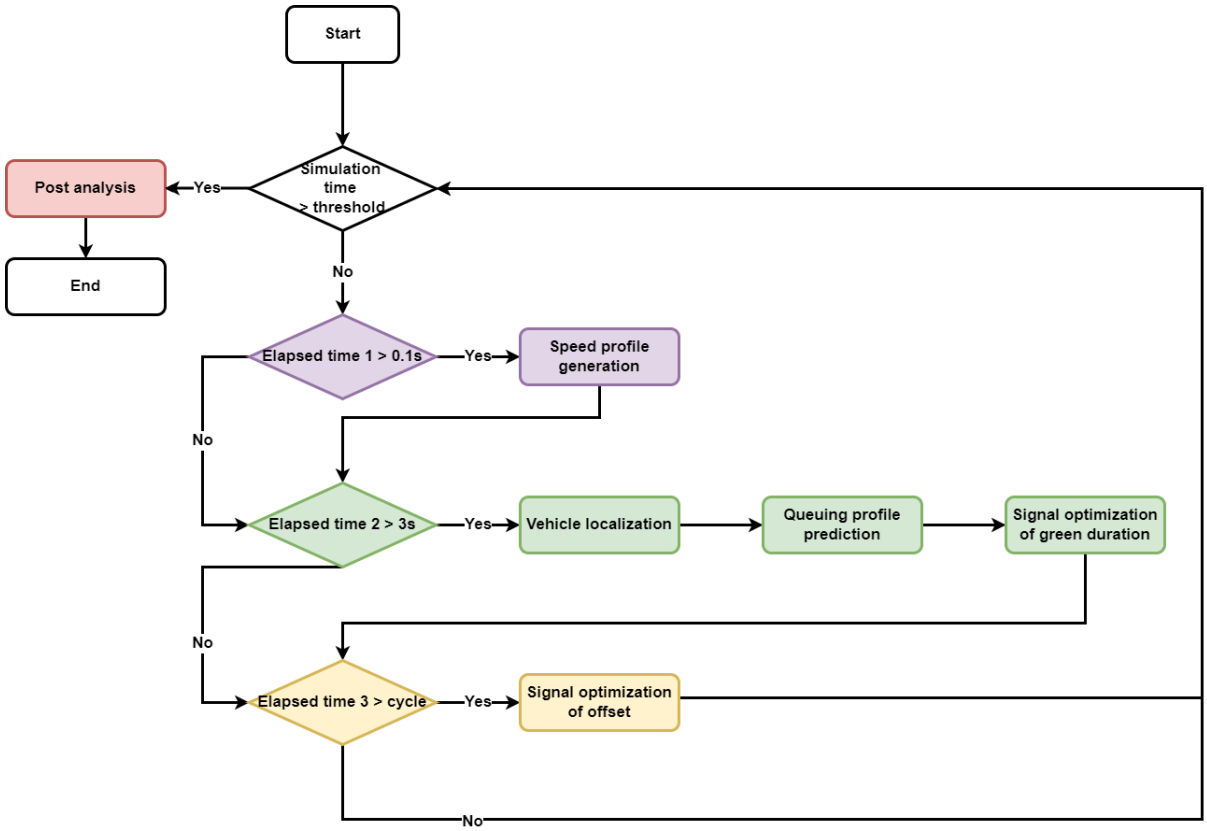



Figure 2.9: Implementation Flow Chart

### 2.6.2 Simulation Experiments

A simulation model of the Plymouth Rd corridor, in Ann Arbor, Michigan is built in VISSIM (Figure 2.10). The Plymouth corridor consists of six intersections, from Barton Dr. on the west to the Green Rd. on the east, which are indexed by 1-6. The stretch of the Plymouth Rd is about 2.2 miles and has two lanes for each direction which is one of the busiest commuting routes, serving US23 to the North campus of the University of Michigan and downtown Ann Arbor. Some crossing roadways are major arterials that carry a large volume of traffic and others are side streets with less traffic demand. The volumes and turning ratios at each intersection in the simulation are calibrated with the real-world traffic data collected from afternoon peak hour (4:00pm-5:00pm). Coordinated actuated signal control is considered as the baseline for comparison, which is optimized by VISTRO [13]. Figure 11 shows the coordination diagram of the baseline signal control. The green shaded area denotes the green waves, which indicates good coordination patterns in both westbound (Figure 2.11 (a)) and eastbound (Figure 2.11 (b)). The index of each intersection is shown at both sides of the figure for westbound and eastbound respectively. Notice that at least one stop is unavoidable due to two-way coordination.

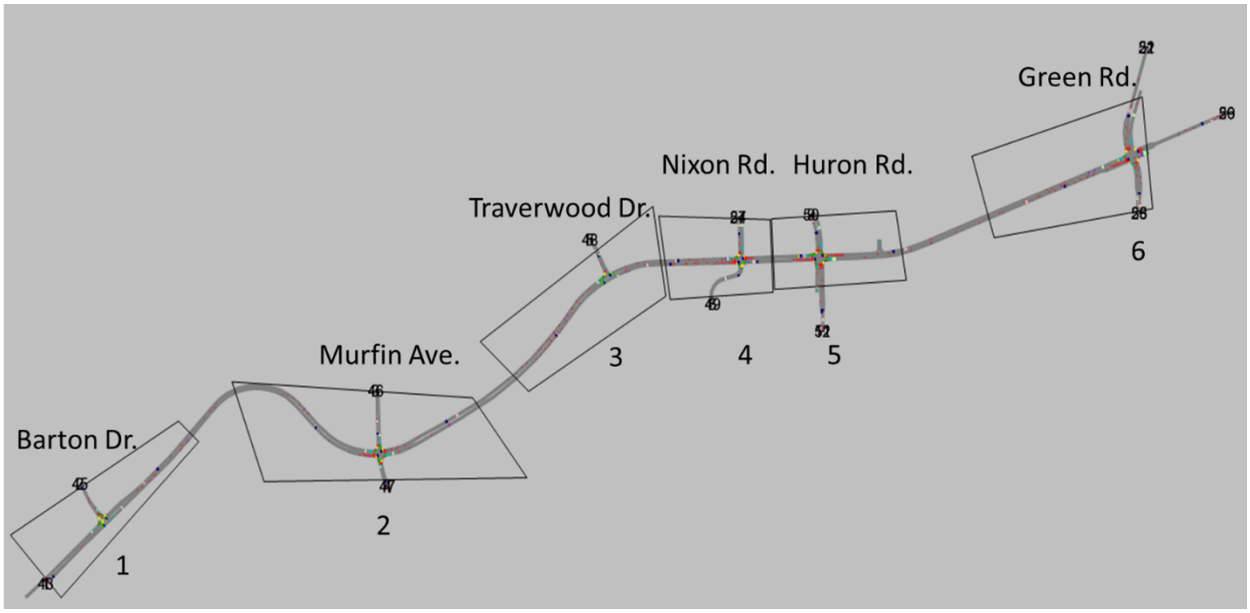



Figure 2.10: Plymouth Corridor VISSIM Simulation Model

Other critical parameters used in the simulation experiments are summarized in Table 2.3.

Table 2.3: Critical Parameters in Simulation Setup

| Parameter Name | Value |
|---|---|

| Speed Limit | 35 mph (west of Huron Rd); 45 mph (east of Huron Rd) |
|---|---|
| Minimum Green Time | 5s |
| Maximum Green Time | 100s |
| Yellow Interval | 4s |
| All-red Clearance Interval | 2s |
| Gap out Time (in actuated control) | 3 s |
| CACC headway ($h_{CACC}$) | 0.9 s |
| RV/CV headway ($h_{RV}$) | 1.8 s |

Two series of simulation experiments are conducted under different combinations of mixed traffic conditions. In the first series of experiments, there are only CV and CAV, with varying penetration rates of CAVs: 0%, 5%, 25%, 50%, 75%, and 100%. In the second series of experiments, all the CVs are replaced with RVs and the same set of CAVs penetration rates are considered. Notice that the first series has 100% connectivity, the main objective is to analyze the system performance under different levels of controllability by varying the penetration rate of the CAVs. In the second series, the connectivity is not 100% and the states of the RVs need to be estimated by loop detectors. Changing the penetration rate of the CAVs leads to the variations of both the controllability and the connectivity. The duration of each experiment is 2100 (sec) with 300 (sec) warm-up time, and each experiment is repeated with 5 random seeds. After each simulation run, all vehicle trajectories are recorded and sent to the post-analysis to calculate the fuel consumption and emissions by the MOVES model [58]. The experiment results are presented below.

Figure 2.12 shows the simulation results at the network level, in which all vehicles in the simulation are taken into consideration. The left three subfigures ((a) – (c)) show the results of the first series of experiments (CAV + CV), while the right three subfigures ((d) – (f)) show the results of the second series of experiments (CAV + RV). For both series of experiments, with the increasing penetration rates of CAVs, the benefits from eco-trajectory planning are more significant. The average vehicle speed increases when there are more CAVs, because the eco-trajectory planning helps more CAVs pass the intersection without stop, also leading to the reduction of the number of stops. In terms of fuel consumption, since the speed profiles of the CAV are smoother and unnecessary acceleration and deceleration are avoided, the fuel consumption and emissions decrease as the penetration rate of CAV increases.

More detailed data can be found in Table 2.4. Note that the comparison between the baseline and 0% CAV shows the benefits of the corridor-level adaptive signal control, where no CAV time of arrival and trajectory are optimized. For the series with CVs, the total delay decreases by 14% (from 74.84s to 64.40s) and the fuel consumption decreases by 6.8% (from 4131KJ/mile to 3867KJ/mile). The results indicate that the performance can be improved greatly by CV based

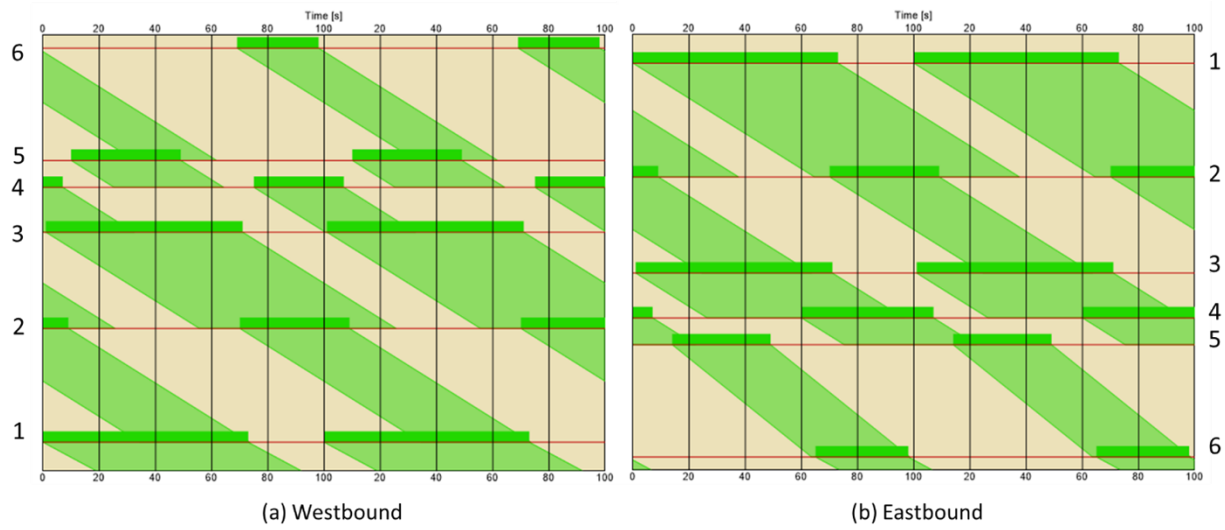



Figure 2.11: Coordination Diagram of the Coordinated Actuated Signal Control (Baseline)

coordinated adaptive control. If the CVs are replaced by the RVs (series II), the total delay decreases by 0.6% (from 74.84s to 74.39s) and the number of stops decreases by 1.9%. The fuel consumption decreases by 3.8% (from 4131KJ/mile to 3972KJ/mile). The results indicate that even loop-detector based coordinated adaptive control outperforms the coordinated actuated control (baseline), but the improvement is not as significant as CV based coordinated adaptive control. When the penetration rate of the CAV is 100%, the delay further decreases by 33.0% (to 49.99s) and fuel consumption decreases by 7.4% (to 3824KM/mile) by comparing to the baseline because the CACC platoon has much shorter headways.

Comparing the second series of experiments with the first series, the high penetration rate of CAV leads to lower benefits in the second series in all performance indexes. The reason for this phenomenon is that the states of RVs are estimated from loop-detector data, which is not as accurate as the BSMs broadcast from the CVs. Therefore, the errors in traffic state estimation lead to suboptimal solutions in the signal optimization. Meanwhile, the inaccurate estimation of queuing dynamics also causes inaccurate time of arrivals for the CAVs and further influences the eco-trajectory planning. If the generated time of arrival is later than the optimal time of arrival, the green time is not fully utilized. On the other hand, the CAV's planned trajectory may be interrupted by its leading vehicle, which results in more fuel consumption and emissions.

Figure 2.13 shows the simulation results of the mobility performance of two intersections. Most intersections have similar performances as the intersection of Green Rd. (subfigure c and d). When the penetration rate of CAV increases, the mobility improves in terms of the total vehicle delay, total stop delay, and the number of stops. At the intersection of Traverwood Dr. (subfigure a and b), mobility is getting worse when the penetration rate of CAV is low. However, since this is a

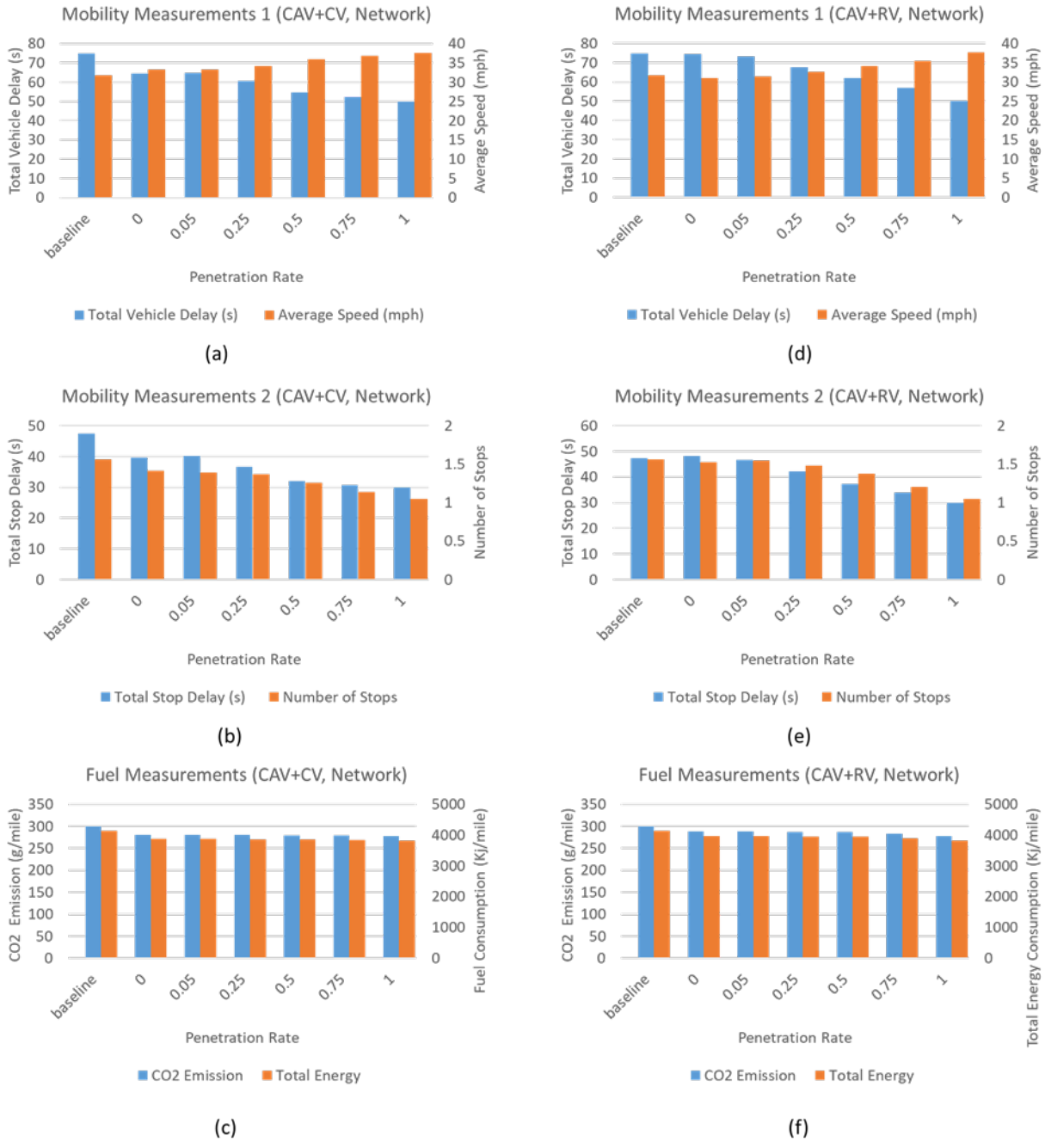



Figure 2.12: Simulation Results of Network Level

Table 2.4: Mobility and Fuel Measurements in Network Level

| Penetration rate | | baseline | 0 | 0.05 | 0.25 | 0.5 | 0.75 | 1 |
|---|---|---|---|---|---|---|---|---|
| Delay (s) | CAV+CV | 74.84 | 64.40 | 64.65 | 60.66 | 54.67 | 52.14 | 49.99 |
| | CAV+RV | 74.84 | 74.39 | 73.18 | 67.51 | 61.92 | 56.90 | 49.99 |
| Number of stops | CAV+CV | 1.56 | 1.42 | 1.39 | 1.37 | 1.26 | 1.14 | 1.05 |
| | CAV+RV | 1.56 | 1.53 | 1.55 | 1.48 | 1.37 | 1.20 | 1.05 |
| Fuel consumption (KJ/mile) | CAV+CV | 4131 | 3867 | 3868 | 3860 | 3852 | 3837 | 3824 |
| | CAV+RV | 4131 | 3972 | 3969 | 3951 | 3950 | 3898 | 3824 |

T-intersection, and the demand for this intersection is not high, the total delay and the number of stops is much less than other intersections, so the performance doesn't impact the entire network a lot.

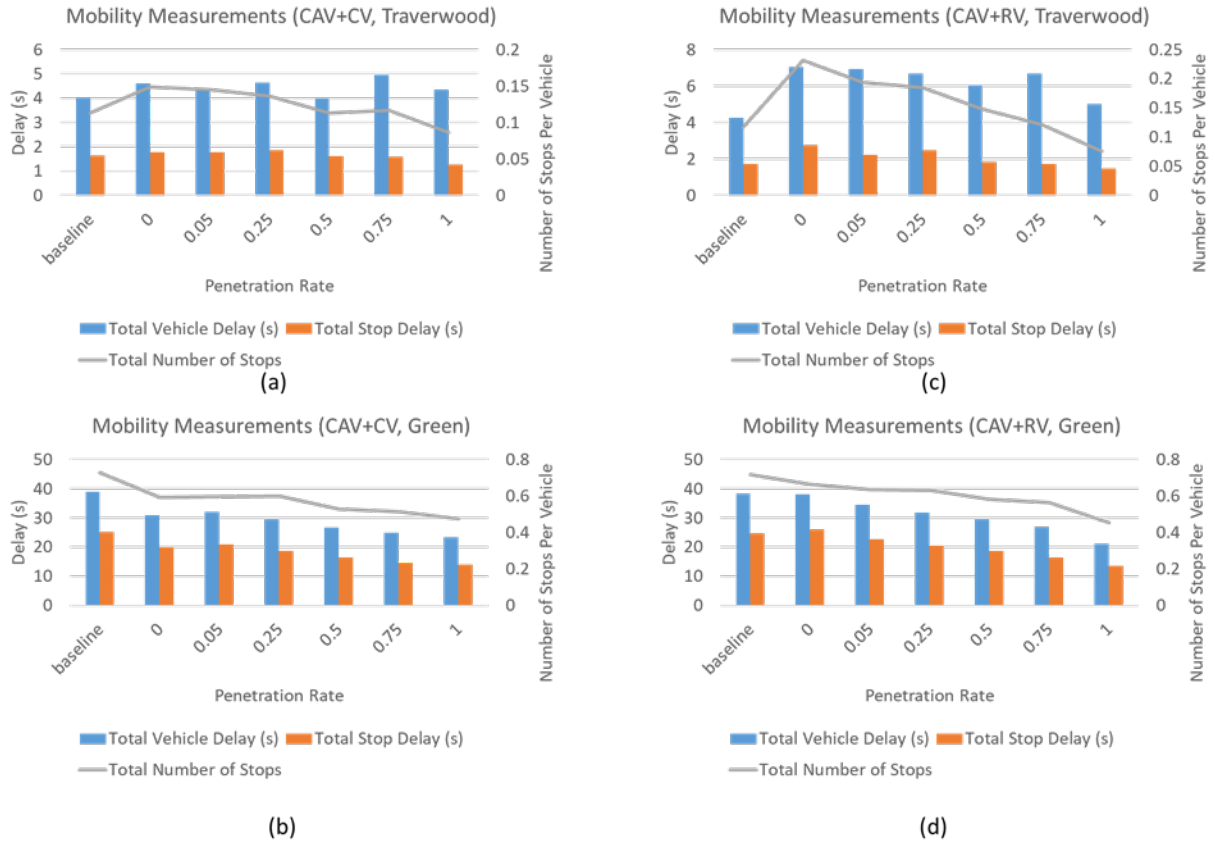



Figure 2.13: Mobility Results of Intersection Level

### 2.6.3 Sensitivity Analysis

One advantage of the proposed framework is that at the corridor level, the offset of each intersection is optimized each cycle to accommodate volume fluctuations. A series of sensitivity analysis is performed to further analyze the impact of dynamic offset optimization.

In the sensitivity analysis, the simulation period is divided into three intervals (300 - 900s, 900 - 1500s, 1500 - 2100s), plus the warm-up time. In the second interval (900 – 1500s), the traffic volume of the whole network is increased by 0%, 5%, or 10%. The traffic volumes of the first and third intervals remain unchanged. To analyze the benefits from the dynamic offset optimization, the baseline experiments use a fixed offset optimized by VISTRO in the corridor level model, and the intersection level model remains the same. All the vehicles in the sensitivity analysis are CVs, so that vehicle trajectories are not optimized.

Figure 2.14 shows the improvement of dynamic offset optimization in terms of mobility and fuel economy, comparing to the baseline. The x-axis denotes the volume fluctuation percentage described above, and the y-axis denotes the percentage of improvement. It can be seen from the figure that the benefits of dynamic offset optimization increase with higher volume fluctuations. When the volume fluctuation is 10%, the dynamic offset optimization further reduces 5.3%, 5.0% and 1.1% of the number of stops, delays and fuel consumption respectively, comparing to the baseline with fixed offsets.

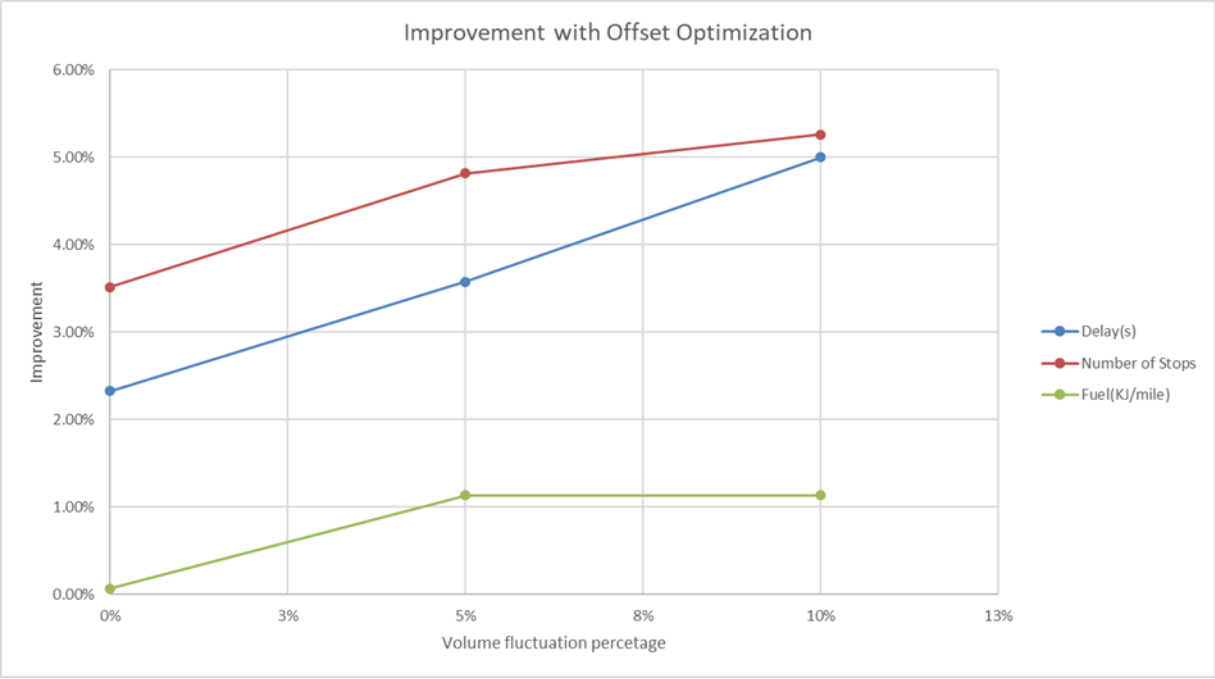



Figure 2.14: Improvement with Offset Optimization

Figure 2.15 shows the offset variations of each intersection cycle by cycle. The dashed lines de-

note the baseline fixed offsets, and the solid lines with circles are the optimized offsets. Comparing to intersection 5 and 6, the offset of the first four intersections does not change a lot. In other words, for these four intersections, the coordination is always good. This can also be validated from the coordination diagram as shown in Figure 2.11. Notice that the coordination between the fourth and the fifth intersection is sacrificed, for accommodating the coordination between other intersections. Thus, the variation of intersection 5 does not affect intersection 4 much. For intersections 5 and 6, when the volume does not fluctuate during the first interval (cycles 1 - 6), the offsets decrease and increase in the same trend. Although the volume starts fluctuating at cycle 7, the whole network has not been influenced by the volume fluctuation a lot and the offset between intersection 5 and 6 remains 45s. The offset starts changing after several cycles, which keeps increasing to 49s because the travel time becomes longer due to the larger volume. The additional mobility and fuel benefits through offset optimization in Figure 2.15 mainly come from intersections 5 and 6, since they are the major intersections in the corridor and carry large volumes.

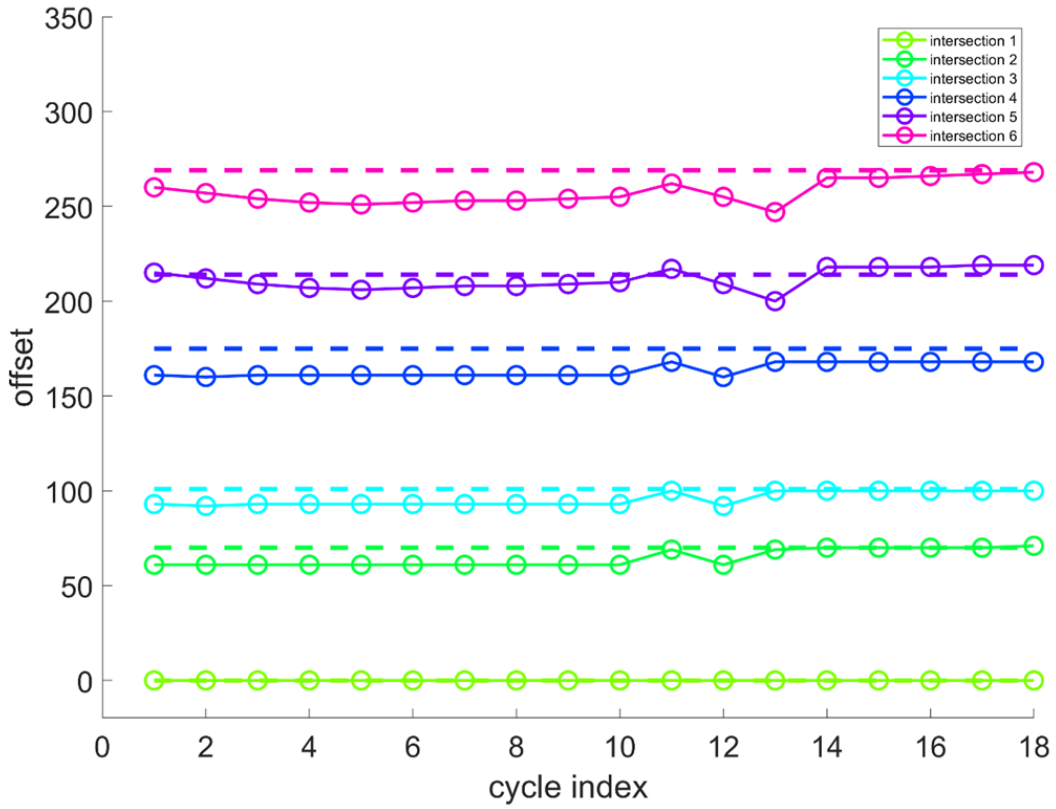



Figure 2.15: Offset Variations at Different Intersections

## 2.7 Conclusions and Further Research

This chapter proposes a cooperative driving framework for arterial corridors in a mixed traffic condition of RV, CV, and CAV. In the vehicle level models, a state transition diagram is designed to accommodate different CAV maneuvers under different operating scenarios. A trigonometric speed profile is applied for eco-trajectory planning with consideration of non-zero initial accelerations. In the intersection level models, a MILP problem is formulated to optimize traffic signals and CAV time of arrivals with the objective of minimizing total delay, given the offset from the corridor coordinator. In the corridor level models, the link performance function is adopted to calculate the total delay of the coordinated phases. Link performance functions are derived under different arrival patterns to estimate the vehicle delay. With the approximation to piecewise linear link performance functions, the offset optimization problem can be formulated as an LP problem, which is easy to solve. Simulation experiments of an arterial corridor have been performed, using real-world traffic data. Results show that the total delay decreases by 14% and the fuel consumption decreases by 6.8% due to CV-based coordinated adaptive control with dynamic offsets. When the CAV comes into play, the total delay and fuel consumption further decrease as the penetration rates of the CAV increase. When the penetration rate of CAVs is 100%, the total delay and fuel consumption reduction raises to 33% and 7.4% respectively. A sensitivity analysis of volume fluctuation has been conducted, which shows the benefits of the dynamic offset optimization at the corridor level.

In future work, the lateral behavior of CAVs needs to be explicitly modeled for the cooperation between CAVs on different lanes in the vehicle level model. The optimization at the intersection level not only provides the time of arrival of the CAVs as the guidance but may also provide lane change guidance (e.g., lane change location and time). This cooperative driving framework can also be extended to a larger transportation network, where the route choice decisions are integrated.

# CHAPTER 3

# Anomaly Detection for the Localization of Connected and Automated Vehicles

## 3.1 Introduction

### 3.1.1 Background and Related Work

CVs and AVs benefit the transportation system from multiple aspects including reducing crashes, improving mobility and sustainability. In both types of vehicles, the localization module, from which the vehicle knows its global and local positions in the driving environment, plays a critical role in information sharing and vehicle navigation. For example, the BSM broadcast by CVs contain vehicle location and motion data for a wide range of applications [73, 111, 114], and AVs utilize the localization results for trajectory planning[86]. Among all sensors that participate in localization, the GPS receiver is the most important one that obtains global positions. Commercial level GPS receivers can achieve the accuracy of 1 meter, and with dual-frequency GPS units, survey-grade GPS has the accuracy of a few centimeters[21]. Besides GPS, LiDAR locators and Inertial Measurement Unit (IMU) are also implemented and tested on AVs [34, 95, 55] for localization purposes. It is critical to ensure that the localization module is accurate, reliable, and highly secure since inaccurate localization results will significantly jeopardize AV trajectory planning and CV safety applications and may cause catastrophic consequences such as crashes.

Unfortunately, existing studies show that vehicle localization module is vulnerable to various types of cyberattacks. GPS spoofing attack is an emerging issue in modern GPS applications. The GPS spoofing attack generates fabricated GPS signals and interferes with the GPS receivers, which can degrade the performance of the localization system. The fake GPS signal usually has a higher strength to mislead the GPS receiver [99]. The practicality of the GPS spoofing attack has been proved in both research [102, 93] and real-world applications [49, 50]. In addition to the GPS spoofing attack, attacks targeting other sensors can also impact the vehicle localization. For example, Petit et al. attacked the LiDAR by injecting false reflected light, and the LiDAR falsely

detected a fake wall [83]. Although such LiDAR sensor attacks do not directly target the localization module, misinterpretation of the surrounding environment will degrade the performance of the LiDAR locator, which is one input source to the localization module. Usually, MSF algorithms are considered as one defense method against sensor attacks since it is highly unlikely that all sensors are compromised at the same time. However, a recent study from Shen et al. managed to construct an MSF attack method, which misleads the sensor fusion algorithms by only spoofing the GPS channel [93].

In general, anomaly detection can be applied to defend GPS spoofing attacks, which can be divided into two categories, node centric detection and data centric detection [105]. In the **node centric detection**, it examines the patterns in the behavior of specific nodes at a protocol level, which usually does not consider data semantics. Signatures are adopted to identify if the sender of the messages is malicious. The node centric detection can be further classified as behavioral or trust based. The behavioral mechanism checks the packet header and meta message information to detect the anomaly. Common behavioral mechanisms include watchdog [46] and flooding detection [43]. Trust-based mechanisms aggregate the trust of a node and distribute the trust among nodes to filter the malicious nodes. Trust-based mechanisms are usually vulnerable to Sybil attacks.

Different from node centric mechanisms, **data centric detection** mainly focus on data semantics, which can also be categorized into two groups, consistency-based and plausibility-based [105]. The consistency-based method examines the relations between packets to identify the anomaly of the newly received data. A cooperative approach can be adopted to analyze the information from multiple agents to identify conflicting messages [123, 40].

Plausibility-based methods filter out the packets according to the numerical plausibility value contained by the data received, which can be utilized to detect attacks with Sybil nodes. A majority of the plausibility-based detection focus on signal-based method [53, 88, 74]. The main shortage of the signal-based methods is generalizability. For example, the method proposed for the anomaly detection of a GPS-only localization system may not be suitable for an MSF localization system. Another plausibility-based detection method is prediction-based. The prediction-based methods focus on predicting the behavior of vehicles and compare the prediction with the observations. Kalman filter based approach is the most common method in this direction [97, 96, 51], in which the future trajectory of the vehicle is predicted with a Kalman filter. Other than only predicting the positions of vehicles, vehicle dynamics can be also integrated into the prediction-based mechanisms. For example, in [117], vehicle dynamics are considered to predict the bounding box of the vehicles. The prediction-based methods can be viewed as driving model-based methods that make predictions of the vehicles to detect the anomaly. Such driving model-based methods may not be generalized to different driving scenarios (e.g. highway / urban).

### 3.1.2 Contribution and Organization of the Chapter

Contributions of this chapter are three-fold:

1. **We propose a generic detection framework to detect anomalies in the localization module of CV/AV using learning from demonstration.** The proposed detection framework directly examines the results of the localization module, regardless of the mechanism of the localization module and different attack types. Thus, it is generic and can be applied to detect various types of attacks. The learning from demonstration framework can also be applied in different driving scenarios as long as the corresponding demonstrated trajectories are available.

2. **The proposed method integrates domain knowledge in detecting cyber attacks at the application level.** The methodology adopted for learning from demonstration leverages transportation and vehicle domain knowledge to learn the driving policy through real-world demonstrations. Compared with other learning based approaches, our proposed framework requires much fewer data in the training stage. To our knowledge, this is the first work that utilizes learning from demonstration with domain specific knowledge for abnormal trajectory detection.

3. **The proposed detection method has low requirements for implementation.** For AV deployment, the anomaly detection requires onboard sensors and digital map information. Such onboard sensors and digital map information are standard for all the AV configurations [15, 99]. For infrastructure side deployment, no other infrastructure sensors are needed. The required connected vehicle environment has been implemented and tested extensively in the past few years [18, 39].

The rest of the chapter is organized as follows: we first present the framework overview (Section 3.2). In Section 3.3, the threat models are presented, and the methodology of the anomaly detection model is introduced in Section 3.4. In section 3.5 and 3.6, the proposed model is validated on the AV threat model and CV threat model, respectively. Section 3.7 extends the experiments on the AV threat model to adaptive attacks. Section 3.8 concludes this chapter and lays out future research directions.

## 3.2 Framework Overview

In this chapter, a new prediction based method is proposed, which combines model-driven and data-driven approaches for GPS spoofing attack detection, and is proved to have better generalizability in the experiments. The central hypothesis is that if the data in the GPS signal is compromised, the resultant information sharing from CVs or trajectory planning from AVs will be impacted, which generates abnormal driving behaviors (i.e., abnormal vehicle trajectories). Following this direction, transportation and vehicle domain knowledge is applied with the learning

from demonstration framework. This method can be deployed in both AVs and at transportation infrastructure.

Figure 3.1 illustrates the concept of the proposed anomaly detection method. For the AV deployment, illustrated by the yellow block, the anomaly detection module is located before the trajectory planning module. Three types of information are used as the input to the detection module. First, information of the AV's POVs captured by onboard sensors. The POVs are defined as nearby vehicles that may influence the behaviors of the AV (e.g., a leading vehicle in the same lane). Second, a digital map that contains roadway geometry information. Third, the localization results provided by the localization module. In the anomaly detection module, the normal driving behavior of the AV is represented by a computational-efficient driving model, which can be learned from the historical trajectories of the AV. The normal driving behavior is then compared with the trajectories from the localization module to detect the anomaly.

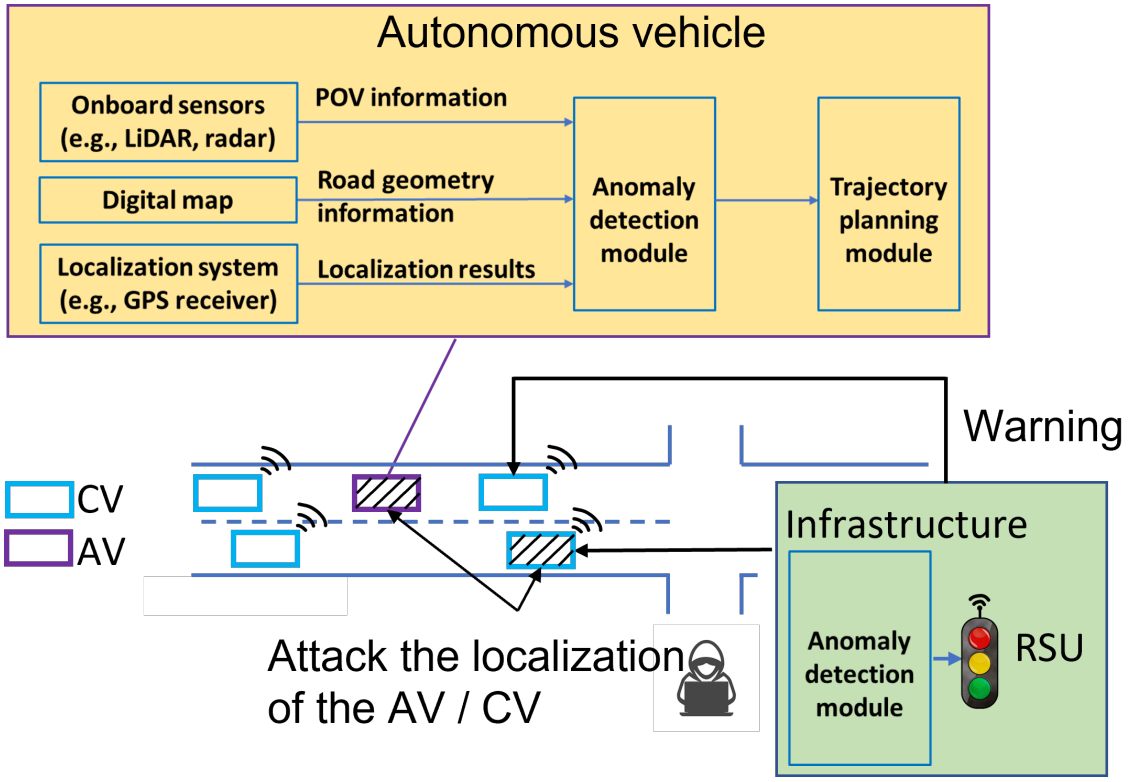



Figure 3.1: Concept of abnormal trajectory detection

For infrastructure deployment, a CV environment is assumed. In figure 3.1, the traffic scenario below the yellow block illustrates the anomaly detection concept at the infrastructure side. CVs broadcast their localization results in the form of BSMs. The infrastructure is equipped with RSU to collect BSMs from the CVs and learns normal driving behaviors. When a CV is under GPS spoofing attack, it broadcasts BSMs with falsified data elements such as location and speed. The

infrastructure compares the learned normal driving behavior and the received CV trajectory to detect the anomaly and send warnings to the victim CV and nearby vehicles. Notice that in this case, we assume that the infrastructure does not have other sensors (e.g., cameras) to cross validate the integrity of the communication messages.

The most important component in the proposed anomaly detection framework is learning normal driving behaviors. Toward this end, we apply the learning from demonstration framework, in which an agent can learn expert behaviors with demonstrations (i.e., examples). The demonstrations are state-action pairs collected from a teacher when he/she performs certain tasks. In this work, learning from demonstration is implemented to learn a computational-efficient CV/AV driving model in different driving scenarios. After collecting a sufficient number of historical trajectories as the demonstrations, maximum entropy inverse reinforcement learning is adopted to derive the optimal driving policy (i.e., reward function). The learned driving policy is used to generate a predicted optimal trajectory, which is then compared with the observed trajectory to identify whether the observed trajectory is under attack or not. A statistical method is developed to measure the dissimilarities between the observed trajectory with the predicted optimal trajectory. With appropriate features that captures such dissimilarities, a decision-tree classifier is adopted to differentiate normal trajectories and trajectories under attack.

The proposed detection method is evaluated with two threat models. The first threat model aims at attacking the MSF based localization model of an AV. The goal of the attack is to generate lateral deviations to the original trajectory to make the subject AV hit the road curb or drive in the wrong direction. The second threat model aims at attacking the Forward Collision Warning (FCW) application on CVs, which utilizes localization results from GPS to trigger the warnings. Experiments are conducted on two real-world datasets, KAIST [52] and NGSIM [10]. Experiment results show that the proposed model has a good performance in both offline detection and online detection with low false positive and false negative rates. Further adaptive attack study confirms the robustness of the model in detecting more stealthy attacks with reduced magnitude.

## 3.3 Threat model

### 3.3.1 Autonomous Vehicle Threat Model

A real-world MSF attack conducted on the Baidu Apollo system is applied as the threat model for the AV anomaly detection [93]. In this study, a GPS spoofing attack towards the MSF-based localization system of AVs is designed, shown in Figure 3.2. At the top, the original MSF algorithm takes the input from GPS, IMU, and LiDAR to generate localization results. In the attack scenario, the GPS channel is spoofed by the FusionRipper algorithm proposed in [93], which can

successfully mislead the MSF localization algorithm. The FusionRipper algorithm consists of two phases: vulnerability profiling and aggressive spoofing. In the vulnerability profiling phase, the attacker performs a constant GPS spoofing attack and observes the localization results from the MSF system to profile when the vulnerable periods appear (i.e., lateral deviation $\geq 0.295$ m on urban roads). After the vulnerable period is identified, the aggressive spoofing phase starts in which the attacker performs exponentially aggressive spoofing to quickly induce large lateral deviations. Two attack goals that cause safety hazards are considered, off-road (i.e., hitting road curbs) attack and wrong-way (i.e., driving to the opposite direction of the road) attack, and both attack goals are achieved by large lateral deviations. The off-road attack requires less lateral deviation (0.895m for urban roads) in the localization results to succeed than the wrong-way attack (1.945m for urban roads).

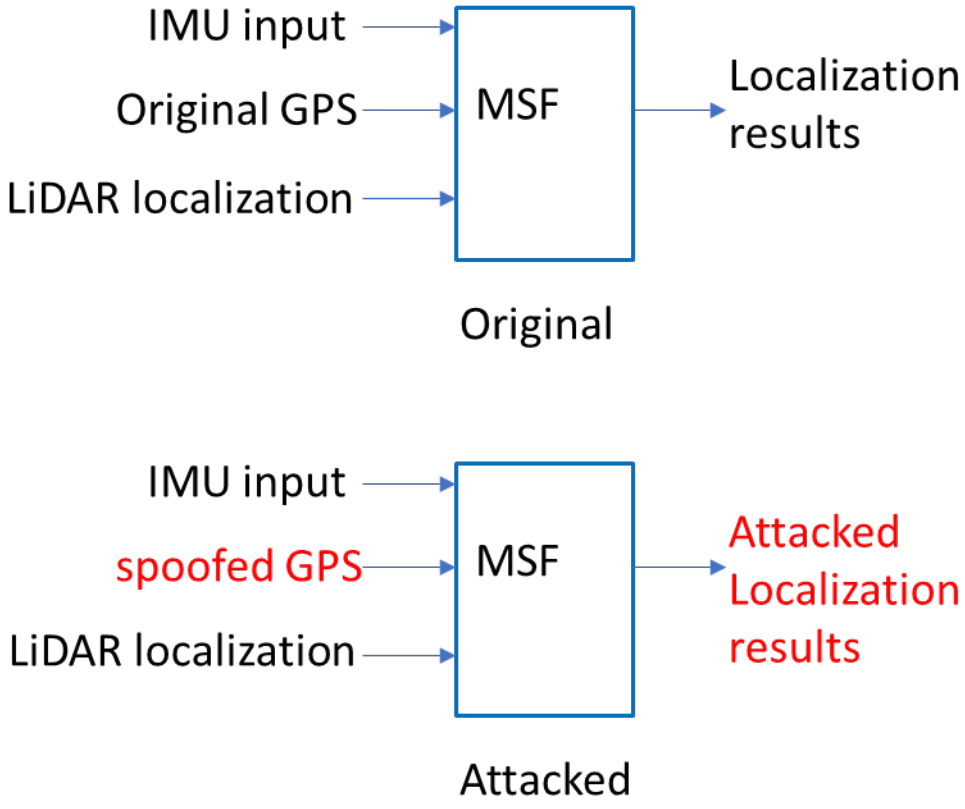



Figure 3.2: Threat model on the AV MSF System

### 3.3.2 Connected vehicle threat model

For anomaly detection in the CV environment, we choose a specific threat model towards the FCW system, which is an important CV safety application. The FCW application intends to warn

the driver of the CV in case of a potential rear-end collision with its leading vehicle in the same lane. The application uses data (i.e., BSMs) received from other vehicles to determine if a forward collision is imminent [113]. In this work, it is assumed that the FCW system only relies on the CV messages (BSM) received from its OBU to trigger the warning. The vehicle is not equipped with other sensors such as radar or camera. This setting has been introduced and validated in a lot of works [47, 113, 63]. The main reason for such a setting is that it is more vulnerable than vehicles equipped with other sensors (e.g., radar), and thus becoming favorable targets for attackers.

The goal of the attack is to generate falsified BSMs through GPS spoofing to trigger the FCW or even automatic brake of a CV. Figure 3.3 illustrates the concept of the attack, in which the blue rectangles denote the normal CV trajectory from BSMs. The red rectangles represent the BSM trajectory under attack, and the yellow rectangles denote the trajectory of the victim CV. The values within the rectangles denote timestamps, where at time $t_0$, the attack starts. At time $t_2$, the falsified BSM trajectory triggers the FCW or automatic brake of the yellow CV. Since the CV under attack is driven by a human driver, the falsified BSMs can't be detected by the vehicle (or driver) itself. This is the reason why the anomaly detection module needs to be deployed at the infrastructure side, where trajectories of all CVs can be investigated.

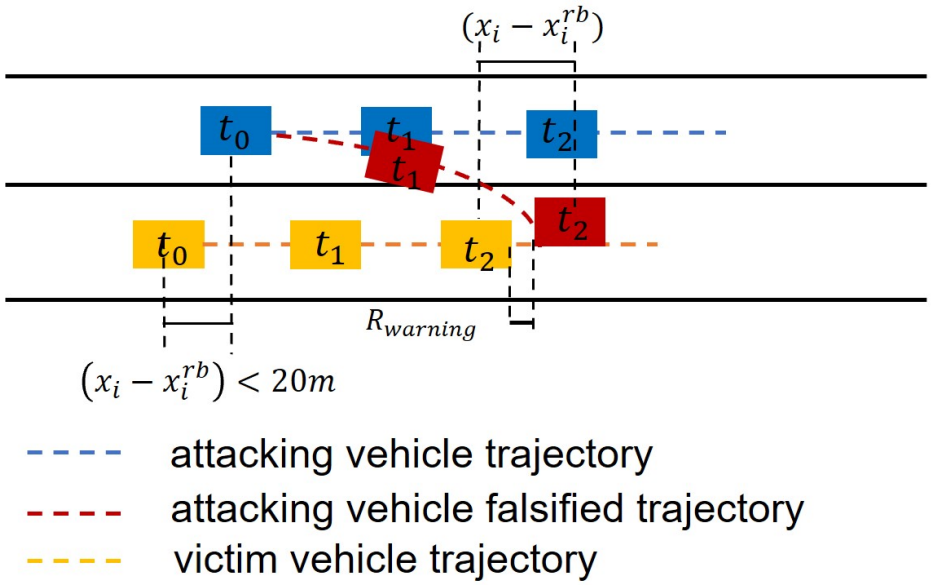



Figure 3.3: Threat model on forward collision warning system

To generate feasible attack trajectories, the attack model is formulated as an optimization problem similar as in [48, 118]. The objective function contains two parts 1) trigger the FCW of the victim vehicle; and 2) generate a smooth vehicle trajectory that is close to the real driving behavior. To achieve the first goal, future trajectory of the victim vehicle needs to be predicted. A constant

speed model is applied to predict the location of the victim vehicle at $t_2$. Based on the predicted location, the first part of the objective function tries to minimize both longitudinal distance and lateral distance between the generated falsified trajectory and the victim vehicle while keeping the two vehicles from overlapping with each other. The second part of the objective function contains driving features such as minimizing acceleration, minimizing heading change rate, and minimizing lateral speed.

The attack starts when the longitudinal distance between the attacking vehicle and the victim vehicle is less than 20m, as shown in 3.3. A rolling horizon scheme is applied, where the prediction is repeated every 0.3 seconds to minimize the prediction error. Based on the updated victim vehicle state, the attack model generates falsified trajectories for the planning horizon. The attack success criterion is based on the Honda FCW logic [60]. The Honda logic includes a warning algorithm and an avoidance algorithm. The attack succeeds when the longitudinal distance between the falsified trajectory and the victim vehicle is small enough to trigger the warning (Equation 3.1) or braking (Equation 3.2 2). In the equations, $v_F$ is the speed of the following vehicle. $v_L$ is the speed of the leading vehicle. $\dot{R}$ is the speed difference between the leading and following vehicle. $\tau_1$ is the reaction time. $\tau_2$ is the time to reach the safety gap when both leading and following vehicles engaging emergency brakes. $\alpha_1$ and $\alpha_2$ are the deceleration for braking. The suggested values are $\tau_1 = 0.5s$, $\tau_2 = 1.5s$, $\alpha_1 = 7.8m/s^2$, $\alpha_2 = 7.8m/s^2$.

$$R_{warning} = f(\dot{R}) = -2.2\dot{R} + 6.2 \tag{3.1}$$

$$R_{braking} = \begin{cases} -\tau_2\dot{R} + \tau_1\tau_2\alpha_1 - 0.5\alpha_1\tau_1^2, v_F \geq 11.67(m/s) \\ \tau_2 v_F - 0.5\alpha_1(\tau_2 - \tau_1)^2 - \frac{v_L^2}{2\alpha_2}, v_F < 11.67(m/s) \end{cases} \tag{3.2}$$

## 3.4 Defense Methodology

This section presents the anomaly detection method to identify the AV/CV attacks introduced in the previous section. There are two major challenges. **Challenge 1: real-time detection.** The operation of AV/CV is highly safety-critical. Therefore, it is vital to detect abnormal or hazardous driving behaviors in time. However, some GPS spoofing attacks can be stealthy (e.g., the first phase of the AV threat model in Section 3.3.1) while some can achieve the attack goal in a few seconds (e.g., the CV threat model in Section 3.3.2), which all pose great challenges in the detection model design. **Challenge 2: validity on different threat models.** Different threat models may cause different abnormal driving behaviors and a generic detection method is needed. It would be more meaningful if the anomaly detection method is effective under different types of cyberattacks.

### 3.4.1 Defense framework

Figure 3.4 illustrates the anomaly detection framework consisting of two steps. On the left side (i.e., offline learning), learning from demonstration is adopted to learn the driving model via maximum entropy inverse reinforcement learning, using historical trajectories. Besides, a decision tree is trained with both historical trajectories and known attack trajectories by three features (objective ratio, normality score, and trajectory displacement). The trained models are applied in the online detection step as shown on the right side of the figure. When observing a trajectory from the localization module or from the CV, its initial state and environment state are utilized in the learned driving model to generate a predicted optimal trajectory, which is then compared with the observed trajectory in terms of the three features. The results are fed into the trained decision tree classifier, which will finally decide whether the vehicle is under attack or not.

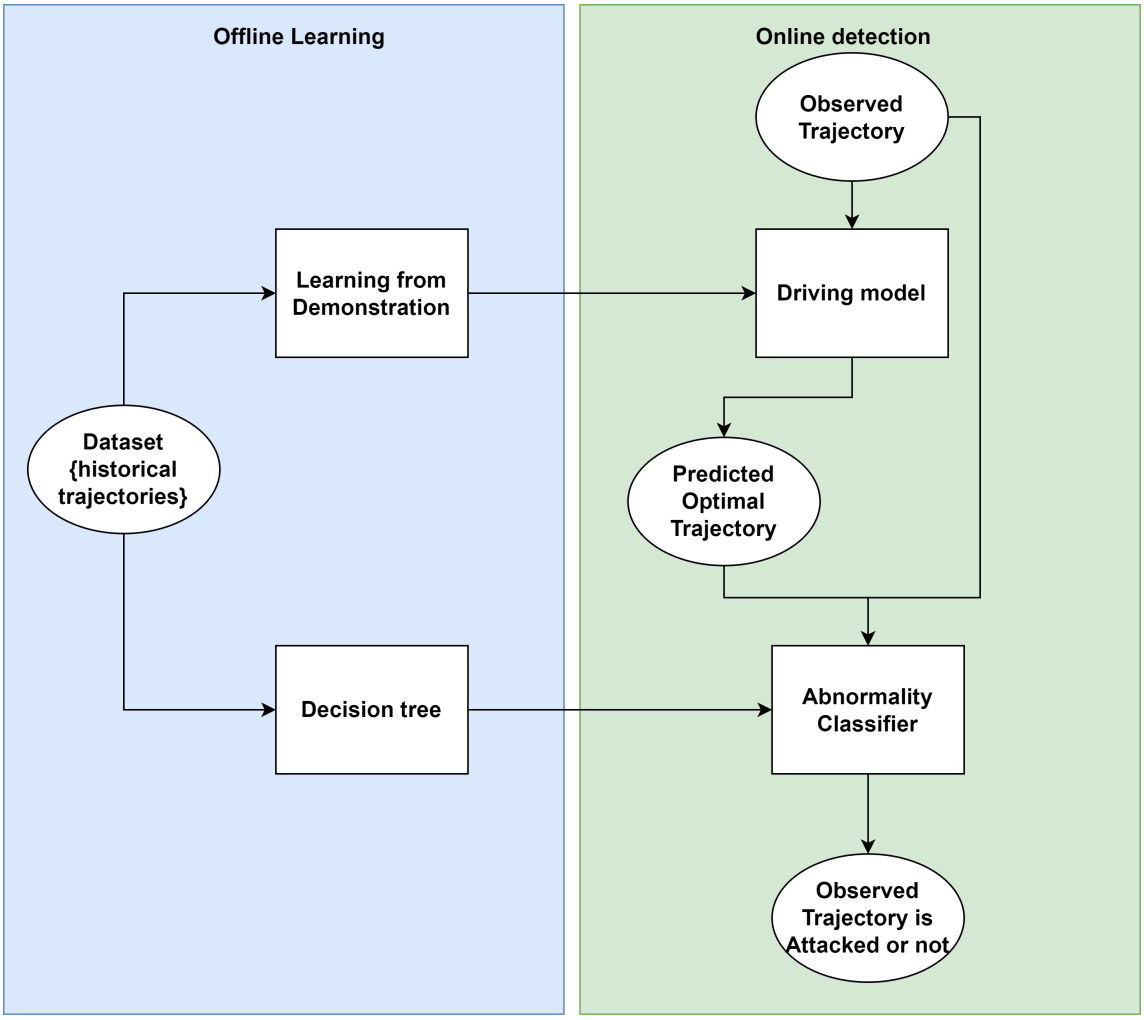



Figure 3.4: Anomaly detection framework

### 3.4.2 Learning from demonstration

A general trajectory generation problem can be formulated as an optimization problem, shown in Equation 3.3. The objective of the optimization problem is the utility function of the driving behavior, in which $\theta$ is the weight vector associated with different driving utilities. **s** is the decision variable of the optimization problem, which denotes the trajectory, a vector of trajectory points $s_i$. Each trajectory point $s_i$ at time step $i$ can be represented by ($x_i$,$y_i$,$v_i$,$a_i$,$\psi_i$ ), in which $x_i$ and $y_i$ is the longitudinal and lateral coordinate, respectively, and $\psi_i$ is the heading angle of the vehicle, between the longitudinal axis of the vehicle and the longitudinal direction of the road. $v_i$ denotes the speed of the vehicle, and $a_i$ denotes the acceleration. **u** represents the initial condition and environment states, which serve as the input parameters for the optimization problem. The initial condition includes the initial position ($x_0$,$y_0$), initial speed $v_0$, initial acceleration $a_0$, and initial heading angle $\psi_0$. The environment states include the longitudinal coordinate and the lateral coordinate of the leading vehicle. $f(\mathbf{s}, \mathbf{u})$ is a mapping function that maps the trajectory to a feature vector, which can be different under different maneuvers. The details of the features and vehicle dynamic constraints are introduced next.

$$\begin{aligned} \min_{\mathbf{s}} \quad & \theta^T f(\mathbf{s}, \mathbf{u}) \\ \text{s.t.} \quad & \text{vehicle dynamic constraints} \end{aligned} \tag{3.3}$$

*Vehicle dynamic constraints*

The vehicle dynamic constraints represent the kinematics of vehicle motion, shown in Equations 3.4,3.5,3.6,3.7, where $\tau$ is the step size. Equation 3.4 reflects the relationship between the longitudinal coordinate change and the heading angle, and similarly, Equation 3.5 reflects the relationship between the lateral coordinate change and the heading angle. Equation 3.6 shows the relationship between the heading angle rate $\psi$ and the heading angle. Equation 3.7 shows the vehicle dynamics between the velocity and the acceleration.

$$x(i+1) = x(i) + v(i)cos(\psi(i) + \psi_r(i))\tau \tag{3.4}$$

$$y(i+1) = y(i) + v(i)sin(\psi(i) + \psi_r(i))\tau \tag{3.5}$$

$$\dot{\psi}(i) = \frac{(\psi(i+1) - \psi(i))}{\tau} \tag{3.6}$$

$$a(i) = \frac{v(i+1) - v(i)}{\tau} \tag{3.7}$$

*Feature vector* The feature vector represents a desired driving policy, which is a linear combination of multiple driving features. In our proposed model, seven features are designed to describe the driving policy including both longitudinal and lateral behaviors. The following provides detailed

descriptions of the features, where $N$ represents the total number of data points in a trajectory.

(1) Speed limit: $f_1 = \frac{1}{N}\sum_i (v_i - v^{lim})^2$. This feature measures the difference between the speed at each time step $v_i$ and the speed limit $v^{lim}$, which models the driving incentive of approaching the desired speed (i.e., speed limit).

(2) Acceleration: $f_2 = \frac{1}{N}\sum_i a_i^2$. It is the summation of the square of the acceleration at each time step, which represents the smoothness of driving behaviors.

(3) Car following: $f_3 = \frac{1}{N}\sum_i \frac{1}{\min(d_i, d_i/v_i)^2}$. $d_i$ is the distance to the leading vehicle at time step i. $min(d_i, \frac{d_i}{v_i})$ chooses the smaller value between the distance and time headway. When the vehicle moves in free flow, the time headway makes an impact. When the vehicle is about to stop, the distance to the leading vehicle makes an impact. This feature models the car-following behavior of the vehicles.

(4) Lateral acceleration: $f_5 = \frac{1}{N}(a_i \sin(\psi_i))^2$. It is the summation of the square of the lateral acceleration at each time step, which measures the smoothness of lateral driving behaviors.

(5) Heading angle: $f_5 = \frac{1}{N}\sum_i \psi_i^2(1 - I^{lanechange})$. $\psi_i$ is the heading angle at time step i. $I^{lanechange}$ is the indicator of lane change, which is 1 if the heading angle between the longitudinal axis of the vehicle and the longitudinal direction of the road is larger than a threshold.

(6) Heading rate: $f_6 = \frac{1}{N}\sum_i(\dot{\psi}_i)^2$. $\psi_i$ is the heading angle change rate at time step i.

(7) Heading rate rate: $f_7 = \frac{1}{N-1}\sum_i(\dot{\psi}_{i+1} - \dot{\psi}_i)^2$. This feature measures the change rate of the heading angle rate. Features 5-7 represent the smoothness of the heading angle to measure the smoothness of lateral driving behaviors of the vehicle.

*Maximum Entropy Inverse Reinforcement Learning*

Before solving the optimization problem, the weight vector $\theta$ needs to be determined, which balances the driving features in the feature vector. It is usually difficult to specify proper weights, which represents the desired driving policy. In this study, we apply maximum entropy inverse reinforcement learning to determine the weight vector $\theta$. Considering the vehicle trajectory planning as a Markov Decision Process (MDP) with a discounted cost as Equation 3.8, in which $\gamma$ is the discounted factor and $r$ is a reward function. If the discounted factor is taken as 1, then the total return is $\theta^T f(s, u)$, for each trajectory $s$. The goal of the inverse reinforcement learning is to find the weight vector $\theta$ that maximizes the log-likelihood function $L(\theta)$, shown in Equation 3.9. $D$ is the demonstration trajectory dataset collected, including $m$ trajectories. $P(s_j|\theta, u_j)$ is the probability of trajectory $s_j$ given parameter $\theta$ and the initial condition as well as the environment state of trajectory $s_j$ (i.e., $u_j$), so when maximizing $L(\theta)$, the likelihood of using weight $\theta$ to generate all trajectories in the dataset is maximized. When the policy of the MDP is the maximum entropy policy [126], $P(s_j|\theta, u_j)$ can be written as Equation 3.10.

$$discountedcost = \sum_{i=0}^{N-1} \gamma^i r(s_i) \approx \theta^T f(s,u) \tag{3.8}$$

$$L(\theta) = \frac{1}{m} \sum_{s_j \in D} lnP(s_j|\theta, u_j) \tag{3.9}$$

$$p(s_j|\theta, u_j) = \frac{e^{-\theta^T f(s_j,u_j)}}{\sum_{s_k \in C_j} e^{-\theta^T f(s_k,u_j)}} \tag{3.10}$$

In this way, the gradient of $L(\theta)$ can be calculated as Equation 3.11, in which $\tilde{f} = \frac{1}{m}\sum_{s_j \in D} f(s_j, u_j)$denotes the empirical feature vector. Thus, the gradient of $L(\theta)$ is the difference between the expected feature vector with respect to weight $\theta$ and the empirical feature vector calculated from the dataset (i.e., observations). Furthermore, the expected feature vector can be approximated by the feature vector of the most likely trajectory.

$$\begin{aligned} \bigtriangledown_\theta L(\theta) &= \frac{1}{m} \sum_{s_j \in D} E_{p(s_j|\theta,u_j)}[f(s_j,u_j)] - \tilde{f} \\ &\approx \frac{1}{m} \sum_{s_j \in D} f(argmin_{s_j} \theta^T f(s_j,u_j)) - \tilde{f} \end{aligned} \tag{3.11}$$

With the gradient of the log-likelihood function, the pseudo-code of the maximum entropy inverse reinforcement learning algorithm can be summarized in the algorithm below, given a set of demonstration trajectories $D = \{s_1, \ldots, s_m\}$.

**Algorithm 1** Maximum Entropy Inverse Reinforcement Learning

1: Compute the empirical feature vector over all demonstrations $\tilde{f}_0 = \frac{1}{m}\sum_{s_j \in D} f(s_j, u_j)$ . Normalize the feature vector. The normalized feature vector is denoted as $\tilde{f}$
2: Initialize every entry of the weight vector $\theta$.
3: **while** $\frac{1}{m}\sum_{j=1} f(s_j^\theta, u_j) - \tilde{f} > threshold$ **do**
4: **for** For each demonstrated trajectory in the dataset **do**
5: Fix the initial condition and the environment states and optimize the trajectory using equation 3.3. The optimized trajectories are denoted as $s_1^\theta, ..., s_m^\theta$.
6: **end for**
7: The gradient can be calculated as $\bigtriangledown_\theta L(\theta) = \frac{1}{m}\sum_{j=1} f(s_j^\theta, u_j) - \tilde{f}$.
8: Update the parameter vector: $\theta(k+1) = \theta(k) + \gamma \bigtriangledown_\theta L(\theta)$, in which $\gamma$ is the learning rate.
9: **end while**

### 3.4.3 Anomaly Classifier

To differentiate normal trajectories from abnormal ones, the difference between the observed trajectory and predicted optimal trajectory should be measured quantitatively by some statistics. In this work, three statistical features are adopted. The first statistical feature is the maximum value of the objective ratio $OR_t$ of all the trajectory points until time step $t$, calculated by Equation 3.12. $OR_t$ represents the ratio between the summation of the objective value of the observed trajectory (i.e., $\sum_{\tau=1}^{t}\ observed\ objective_\tau$) and the summation of the objective value of the predicted optimal trajectory via the learned model (i.e., $\sum_{\tau=1}^{t}\ optimal\ objective_\tau$) at time step $t$. It measures how different the observed trajectory is from the optimal trajectory.

$$OR = \max_{1...t} OR_t = \frac{\sum_{\tau=1}^{t}\ observed\ objective_\tau}{\sum_{\tau=1}^{t}\ optimal\ objective_\tau} \tag{3.12}$$

The second statistical feature adopted is the max value of the normality score $NS_t$ of all the observed trajectory points until time step $t$, calculated by Equation 3.13. $NS_t$ measures the variation of the objective value of the observed trajectory. $objective_t$ denotes the objective value of the observed trajectory at time step $t$. $objective\ mean_{1...t}$ is the mean objective value of all the observed time steps until $t$, $objective\ std_{1...t}$ is the standard deviation of the objective value of all the observed time steps until $t$.

$$NS = \max_{1...t} NS_t = \frac{objective_t - objective\ mean_{1...t}}{objective\ std_{1...t}} \tag{3.13}$$

The last statistical feature is the maximum value of the average displacement error $ED_t$ with the prediction horizon of $T$ between the observed trajectory and optimized trajectory at time step $t$, calculated by Equation 3.14. The average displacement error at time step $t$ (i.e. $ED_t$) can be calculated by measuring the average point-wise Euclidean distance between the observed trajectory $(x^{obs}, y^{obs})$ and the predicted trajectory $(x^{pred}, y^{pred})$ within the prediction horizon $T$. This feature captures the difference between the observation and the optimization results in terms of the Euclidean distance in the 2-D space.

$$\begin{aligned} ED &= \max\nolimits_{1...t} ED_t \\ &= \tfrac{1}{T}\sum_{i=t}^{t+T}\sqrt{(x_i^{obs} - x_i^{pred})^2 + (y_i^{obs} - y_i^{pred})^2} \end{aligned} \tag{3.14}$$

With three statistical features defined as the input, a decision tree classifier is applied to differentiate the abnormal trajectories from normal trajectories. More information of the decision tree classifier can be found in [87].

Figure 3.5 illustrates an example of the decision tree classifier. The internal nodes, denoted by white rectangles, represent tests on attributes in the dataset at these nodes. A test splits the dataset

at the parent node into two datasets as the child nodes, and the test is determined by maximizing the information gain from the parent node to the child nodes. If the entropy is chosen to calculate the information gain, the information gain is the reduction in entropy before and after the test. The calculation of entropy is shown in Equation 3.15, in which $K$ is the total number of class labels and $p_i$ is the frequency of label $i$ in the dataset. In this way, the entropy at each node reflects the uncertainty of the dataset. For example, if all the data within a dataset have the same label, then the entropy of this dataset is zero and there is no uncertainty. By maximizing the information gain from a parent node to child nodes, the reduction of the entropy is maximized and the datasets for the children nodes become less uncertain (i.e., more data have the same label), which is the goal of classification. Each leaf node, denoted by a colored ellipse, represents a classification result that all the data in such a leaf node are classified with the majority label. In the example, the first test checks if the statistical feature of $OR$ is larger than $a$, and the dataset is split into two datasets. If the condition is true, the second test of normality score will be applied to the dataset in the left branch, and differentiate the abnormal trajectories from the normal trajectories. Similarly, in the right branch, the test of $ED$ is applied for the classification.

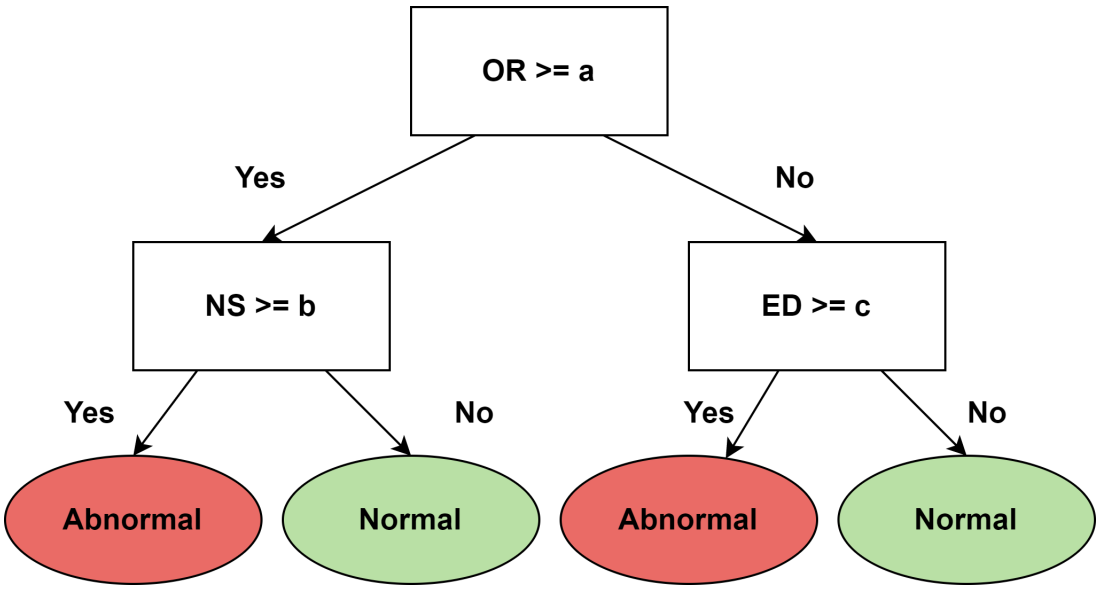



Figure 3.5: Decision tree classifier

$$Entropy = \sum_{i=1}^{K} p_i \log \frac{1}{p_i} \tag{3.15}$$

## 3.5 Detection Model Evaluation on AV Threat Model

The AV threat model (i.e., FusionRipper [93]) is implemented on the KAIST Complex Urban dataset [52], which is an AV driving dataset in both urban and highway driving scenarios based on the Apollo system. Figure 3.6 illustrates a sample trajectory (in red) that consists of both urban and highway driving scenarios in the KAIST dataset. The dataset provides raw data from LiDAR, stereo camera, GPS, and IMU. The FusionRipper algorithm takes them as the input and applies the MSF module in Apollo to obtain the compromised localization results. Specifically, lateral deviations are added to the original trajectory data. In this study, the deviated trajectory generated by FusionRipper is considered as the trajectory under attack. Meanwhile, we extract the original AV trajectories in the data as ground truth.

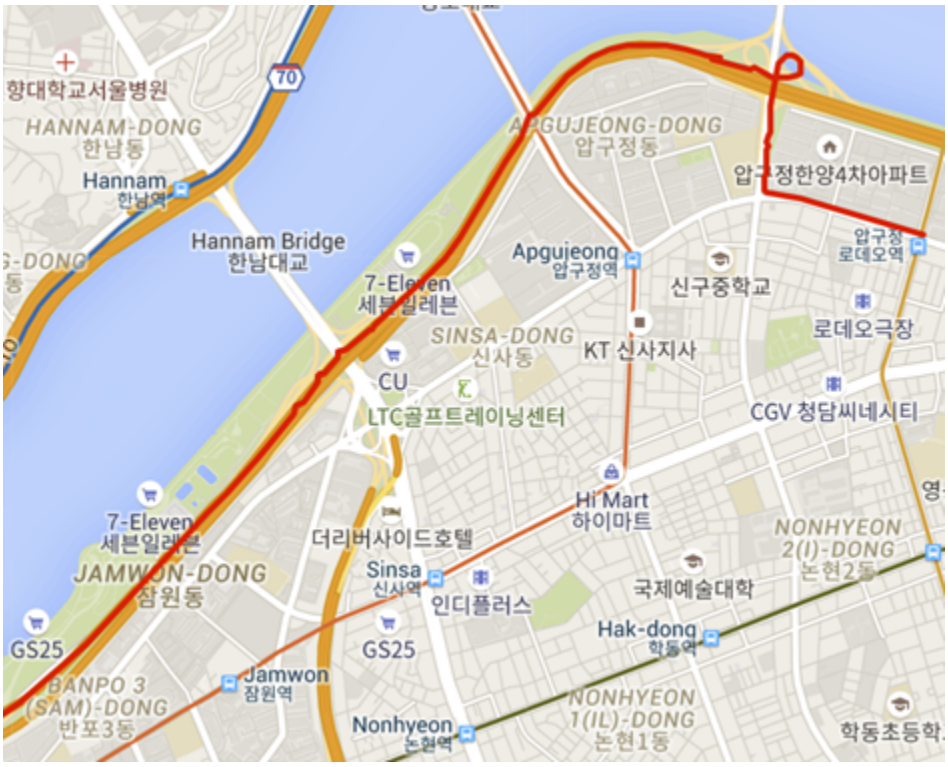



Figure 3.6: A sample trajectory in KAIST dataset

### 3.5.1 Data processing

Before applying the proposed detection method, the original KAIST trajectory data set needs to be processed to calculate two additional data elements (road orientation and car-following distance) that are needed for the proposed learning from demonstration model. The heading angle of the trajectory is the relative angle between the longitudinal axis of the vehicle and the longitudinal di-

rection of the road, but the road orientation is not included in the raw data. To calculate the relative heading angle, vehicle trajectories are allocated to the closest road segment on the OpenStreetMap [42], and then the road orientation is extracted from the waypoints of the OpenStreetMap. In the feature vector, the distance to the leading vehicle is also required to calculate the car-following distance, which is extracted from the raw images from the forward-facing stereo cameras installed on the vehicle, using the YOLO (You only look once) algorithm [85]. The disparity of the detected vehicle between the left and right stereo camera is obtained from the images to calculate the distance to the leading vehicle using triangulation. Interpolation is applied when the front vehicle is missing. Figure 3.7 presents an example of the detected leading vehicle denoted by the yellow rectangle, and distance measurement in meters denoted by the red number. With the road orientation and distance to the leading vehicle processed, all the features in the learning from demonstration model can be calculated.

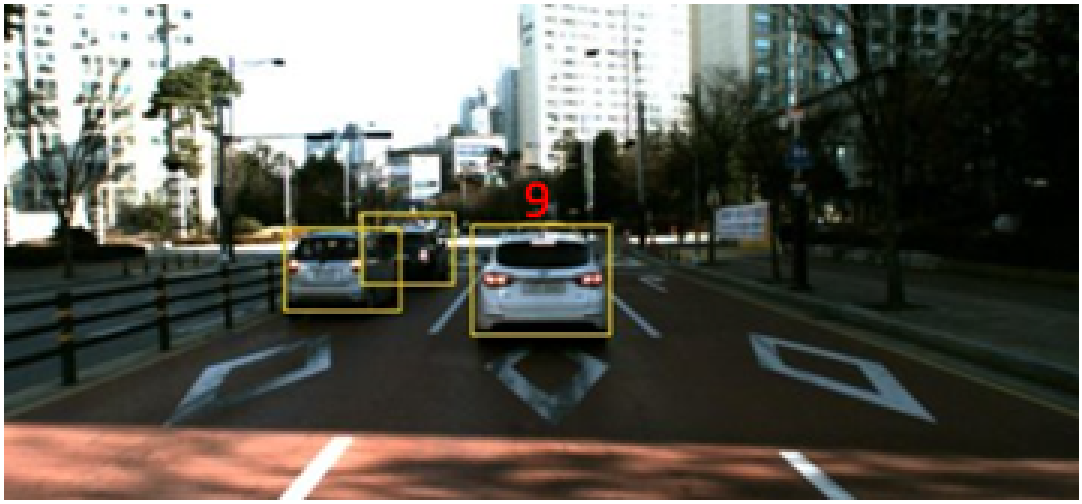



Figure 3.7: Vehicle detection and distance measurement result

### 3.5.2 Experiment setting

In total, 78 ground truth trajectories are obtained from the KAIST dataset. The ground truth trajectories include both in-lane driving and lane changing cases. We deliberately include the lane changing cases in the training dataset because the FusionRipper attack aims to create abnormal lateral deviations of a trajectory to achieve the off-road or wrong-way attack goal. It is critical to differentiate between a normal lane changing process, which also includes lateral deviations, and the lateral deviations caused by the attack. 88 attacked trajectories from the FusionRipper attack and all ground truth trajectories are utilized in the experiments for detection.

In the experiments, we mainly evaluate the proposed detection method against the off-road

attack, which requires less lateral deviation and thus is more difficult for the detection model. Two types of detection mechanisms, namely offline detection and online detection, are designed and evaluated. In the offline detection, the detection is performed after the full trajectory of the vehicle is observed. In the online detection mode, the anomaly classifier checks the trajectory every 0.5 seconds until classified as abnormal or reaching the end of the attack. The online detection mode is designed to detect the abnormal trajectories in real-time as soon as possible, which is critical to the safety performance of the AVs, but also more challenging.

### 3.5.3 Experiment results

Figure 3.8 illustrates the performance of the learning from demonstration model, by comparing a ground truth trajectory (i.e., green curve) with its corresponding predicted optimal trajectory from the learned model (i.e., blue curve). Subfigure (a) shows the position profile and subfigure (b) shows the speed profile. The prediction horizon is 2 seconds, indicating that the learned model takes an accurate position (i.e., the same as the ground truth position profile) and the corresponding speed as the input every 2 seconds. The generated trajectory is very close to the ground truth trajectory. To quantitatively measure the difference between the learned model and the ground truth trajectory, the average Displacement Error (ADE) between the ground truth trajectory and the predicted optimal trajectory is calculated as 0.76m. The calculation of the ADE is based on the of $ED_t$ in Equation 3.14. The ADE is less than 1m, which indicates that the predicted optimal trajectory fits the ground truth very well.

Based on the learned model, the value of the objective function of both observed trajectories and predicted optimal trajectories (by solving Equation 3.3) can be calculated by evaluating $\theta^T f(s, u)$, in which $\theta$ is optimized by the maximal entropy inverse reinforcement learning model. Intuitively, if the vehicle is not under attack, the value of the objective function calculated from the observed trajectory should be close to the value calculated from the predicted optimal trajectory. If the vehicle is under attack, then the two values should deviate from each other. Figure 3.9 illustrates a comparison of the objective values of the attacked trajectory, ground truth trajectory, and predicted optimal trajectory. The red curve is the objective of the attacked trajectory, and the green curve denotes the objective of the ground truth trajectory. The blue curves in both subfigures denote the objective value of the predicted optimal trajectory. In subfigure (a), when the attack is about to succeed (i.e., time $\geq$ 14 s), the objective value of the attacked trajectory deviates from the optimal value significantly. By comparing subfigures (a) and (b), the objective of the ground truth trajectory is much closer to its corresponding optimal objective value than the attacked trajectory.

Figure 3.10 shows the 3D (subfigure (a)) and 2D (subfigure (b)) scatter plots of the decision tree classification result. The red dots represent the attacked trajectories, and the black dots rep-

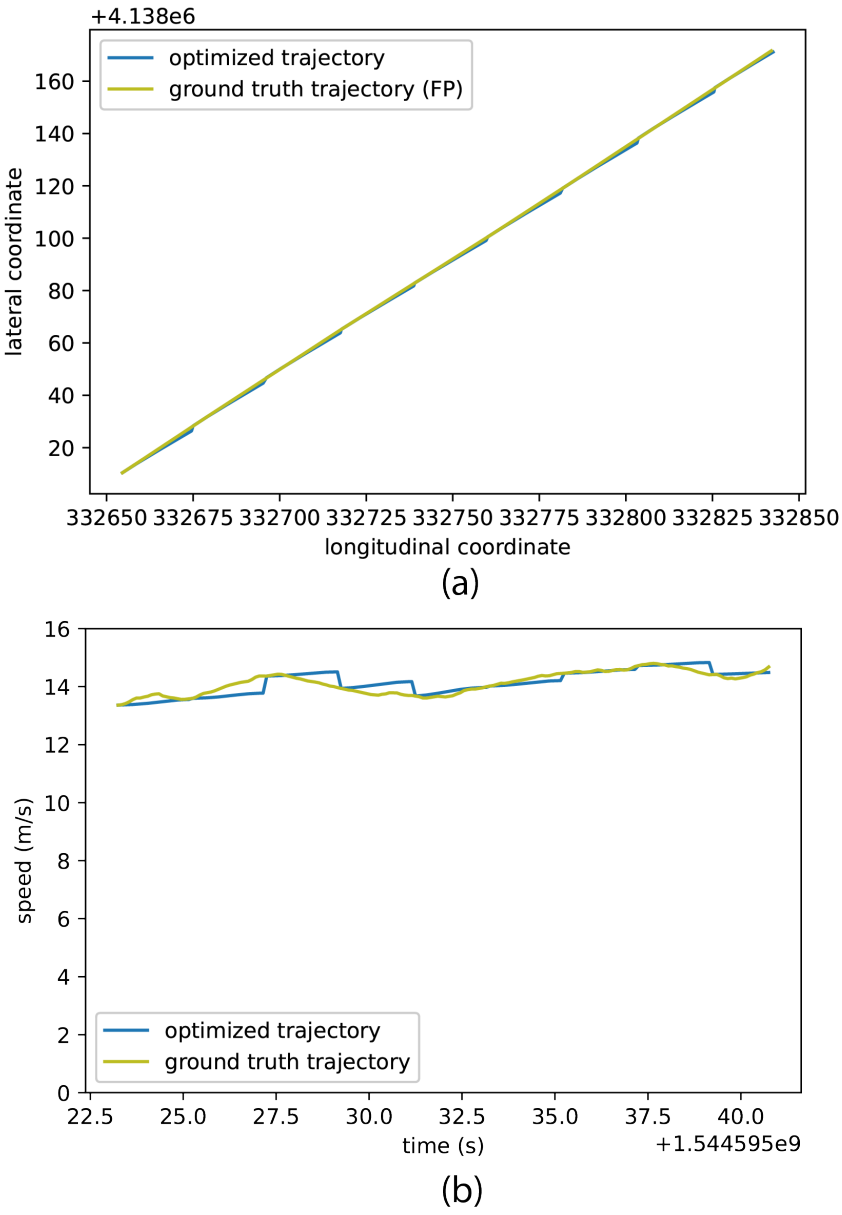



Figure 3.8: Comparison between a ground truth trajectory and a predicted optimal trajectory ((a): trajectory profile in 2-D space. (b): speed profile)

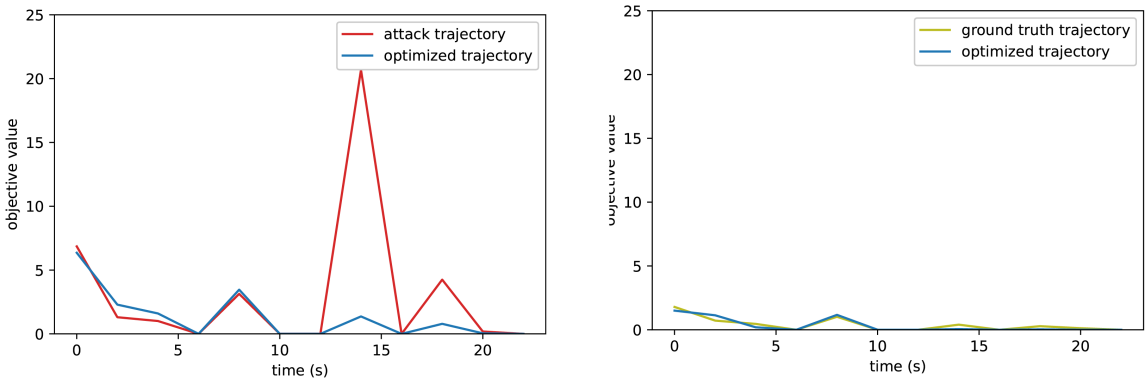
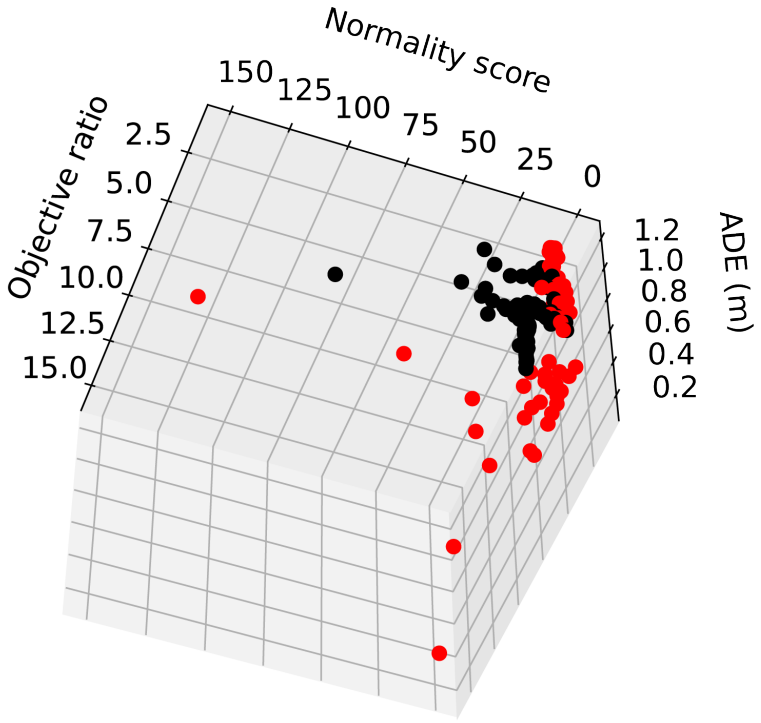


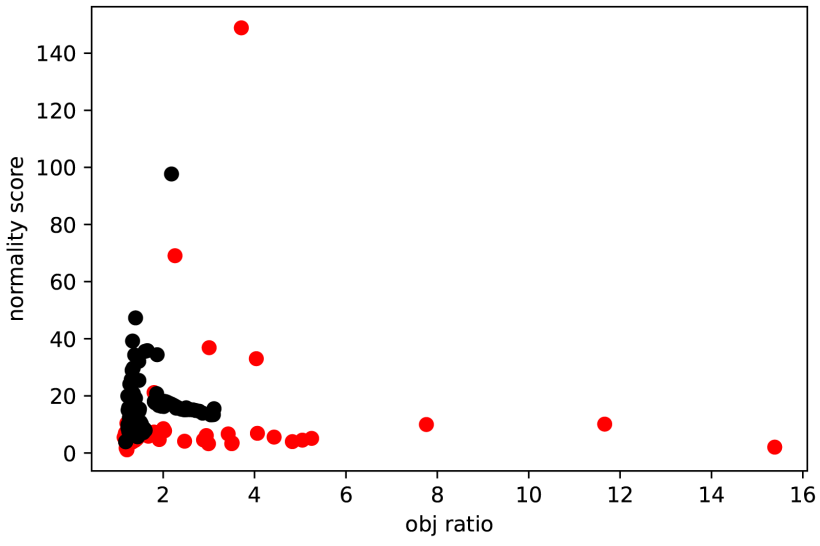

Figure 3.9: Objective value comparison of the attacked trajectory and the ground truth trajectory ((a): objective value comparison of an attacked trajectory (b): objective value comparison of a ground truth trajectory)

resent the ground truth trajectories. In the 3D scatter plot, three axis represent the three statistical features introduced in section 3.4.3, which are objective ratio, normality score, and average displacement error. With three features, the normal trajectories in the ground truth can be separated from the attacked trajectories, as shown in the 3D scatter plot. Subfigure (b) shows the distribution of trajectories if the average displacement error feature is removed. The classifier is difficult to differentiate since the normal trajectories and the attacked trajectories are mixed together. The scatter plots illustrate that the choice of these three statistical features is appropriate.

Next, we show the results of the offline and online detection respectively. In the offline detection, false positive (Type I error) indicates that a ground truth trajectory is classified as an attacked trajectory. On the contrary, false negative (Type II error) means that an attacked trajectory is not identified correctly. The false positive rate of offline detection is 8.7% (2/23), and the false negative rate is 3.7% (1/27). Figure 3.11 illustrates a false positive case and a false negative case. In subfigure (a) and (b), green curves represent the ground truth trajectories in the dataset, and the blue curves denote the optimized trajectories, respectively. Subfigure (a) shows the false positive case in the 2-D space, in which the optimized trajectory does not fit the ground truth very well, compared to a true positive case in subfigure (b) where the optimized trajectories almost overlap with the ground truth trajectory. The reason for the unsatisfying fitting in the false positive case is that the road orientation at this road segment calculated from the map data is not accurate. Thus, the optimized trajectory tries to follow the road direction within the prediction horizon and deviates from the ground truth trajectory. Such an issue may be resolved by applying an HD map for road orientation calculation, rather than utilizing the OpenStreetMap as in our study. Subfigure (c) shows the heading rate profile of the false negative case that an attacked trajectory is misclassi-




(a)




(b)

Figure 3.10: Scatter plot of the classification problem ((a): scatter plot with three features. (b): scatter plot with two features.)

fied. The red curve denotes the heading rate profile of the attacked trajectory, and the blue curve denotes the heading rate profile from a ground truth trajectory. The heading rate is a key feature in the learned driving model, and in this case, the heading rate of the attacked trajectory is similar to the heading rate of the ground truth trajectory, which makes the attacked trajectory difficult to be identified.

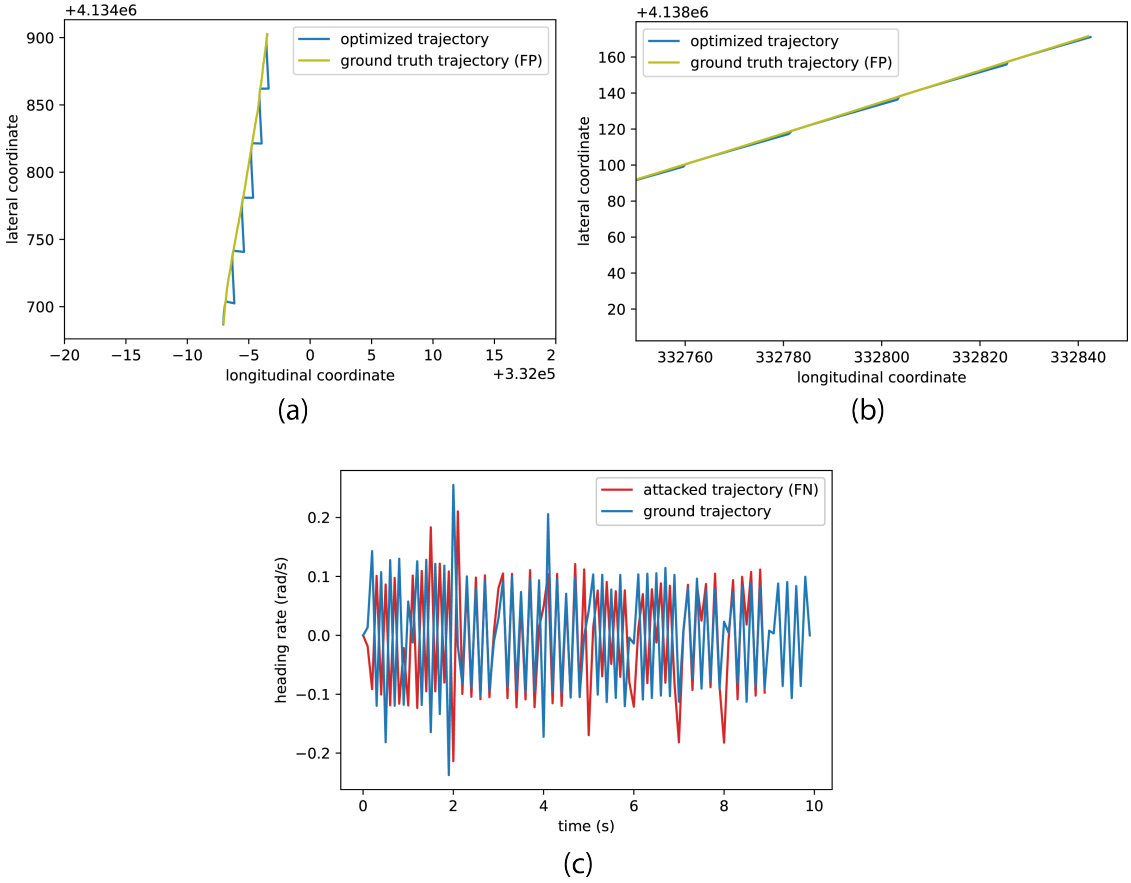



Figure 3.11: Misclassification examples of the KAIST experiments ((a): trajectory profile of the FP case. (b): baseline trajectory profile for the FP case. (c) heading rate profile of the FN case)

Figure 3.12 further shows the heading rate profile comparison, which reveals the reason why the attacked trajectories can be differentiated from the ground truth trajectories. The red curve denotes the heading rate profile of the attacked trajectory, and the green curve denotes the heading rate profile of the ground truth trajectory. The blue curves in both subfigures denote the heading rate profiles of the corresponding predicted optimal trajectories. Notice that the heading rate profiles of the predicted optimal trajectories fluctuate every 2 seconds since the prediction horizon is 2 seconds. In subfigure (a), the heading rate profile of the attacked trajectory has larger fluctuations compared to the predicted optimal trajectory. The ground truth trajectory, on the contrary, has

Table 3.1: Performance of online detection

| FP | FN | Mean attack success time (s) | Mean detection time (s) | Mean time to attack success (s) |
|---|---|---|---|---|
| 2/23 | 1/23 | 28.7 | 12.7 | 16.0 |

smaller values and small fluctuations. Such differences are reflected in the objective function value, which is one feature in the classification model.

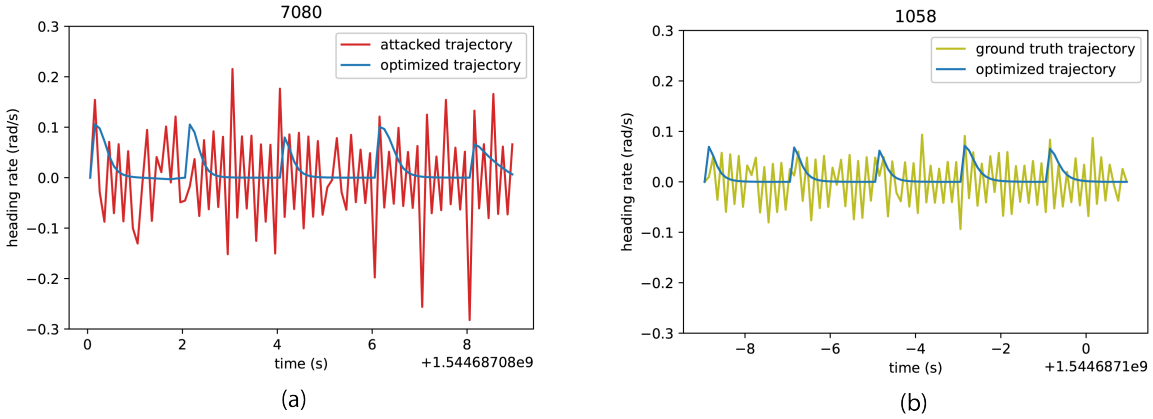



Figure 3.12: Heading rate profiles of the attacked trajectory and the ground truth trajectory ((a): attacked trajectory. (b): ground truth trajectory)

In the online detection, the anomaly classifier checks the trajectory every 0.5 seconds. The performance of the online detection is shown in Table 3.1. The false positive rate is 8.7% (2/23), and the false negative rate is 3.7% (1/27), which is the same as the offline detection results. For online detection, it is important to identify the attacked trajectory as early as possible, but at least before the attack succeeds. Therefore, we further calculate the mean detection time to compare it with the mean attack success time. The mean success time of the off-road attack is 28.7 s, and the mean detection time is 12.7 s. The time to attack success is defined as the time duration from the success time of the detection to the success time of the attack, which measures how early the attack can be detected before attack success. In the online detection, the attacked trajectories can be identified 16.0 s before the attack success time in average.

Figure 3.13 further illustrates the detection time (i.e., blue bars) in the online detection, compared to the duration of attack phases one and two. The green bars denote the duration of phase one (i.e., vulnerability profiling), and the red bars denote the attack success time, which is the end of phase two (i.e., aggressive spoofing). In general, except for the false negative case 7109, the

detection time of all other test cases is not longer than the attack success time, which indicates that the anomaly classifier can successfully detect the spoofing attack before it achieves the attack goal. For most of the test cases, the detection time is even less than the duration of the vulnerability profiling phase, which leaves sufficient time for applying further mitigation strategies.

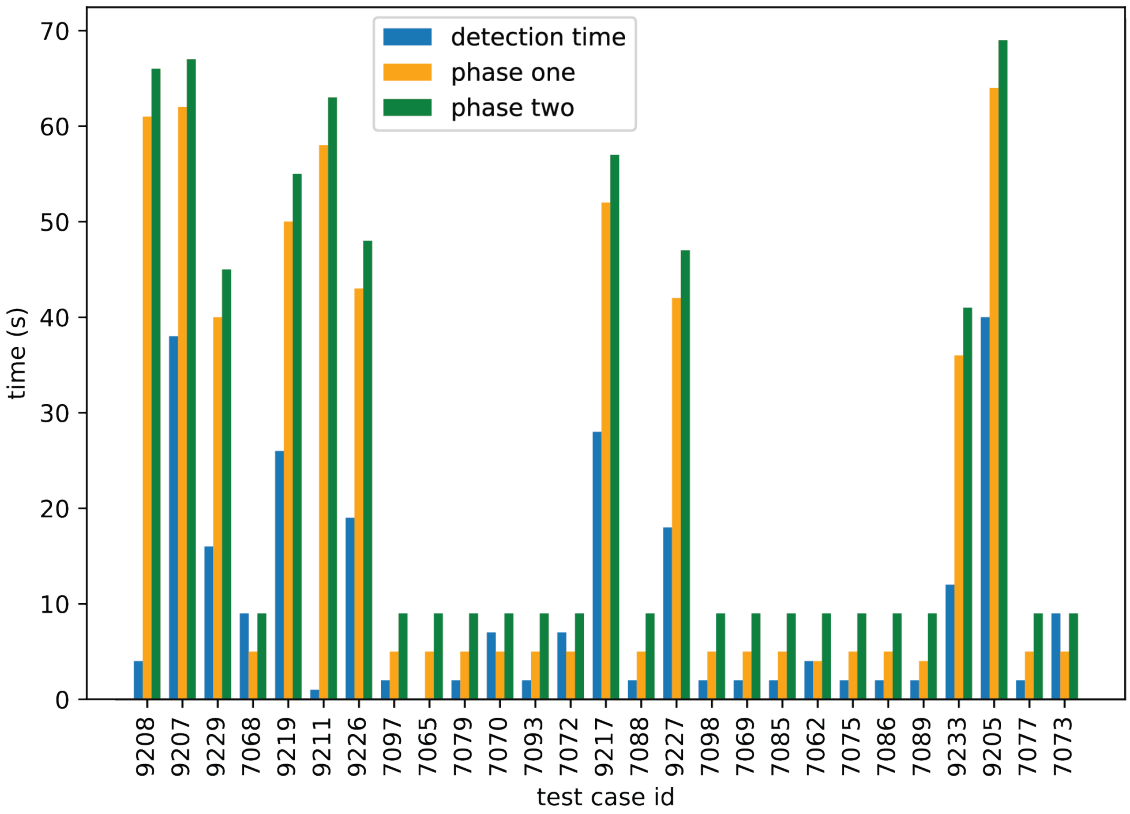



Figure 3.13: Detection time in online anomaly detection for KAIST experiments

## 3.6 Detection Model Evaluation on CV Threat Model

In the previous section, the proposed anomaly detection model is evaluated against the AV threat model. In order to further demonstrate the scalability of the model, another experiment is conducted using the Next Generation Simulation (NGSIM) Lankershim Blvd. dataset. The dataset includes vehicle trajectory data at 10 Hz on a bidirectional urban signalized arterial, captured by cameras mounted on the roof of surrounding buildings [10]. Different from the KAIST dataset in which all trajectories are generated from a single AV, the NGSIM dataset includes vehicle trajectories from different human drivers with a variety of driving behaviors. Validation of the proposed anomaly detection on the NGSIM dataset is necessary to show the effectiveness of the algorithm in a CV environment with human driven vehicles.

In this experiment, 166 ground truth trajectories from the NGSIM trajectory dataset are extracted, including 114 trajectories with no lane change behavior, and 52 trajectories with lane change behavior. For 114 trajectories without lane change behavior, the attack model illustrated in Section 3.3.2 is executed to generate the attacked trajectories. Overall, there are 280 trajectories (166 ground truth trajectories + 114 attacked trajectories) in the experiments, and 2/3 of them are used as the training data, 1/3 of them are used as the testing data. Compared to the threat model for AV, the CV threat model is much more aggressive with a much short attack duration, which greatly increases the difficulty for both offline and online detection.

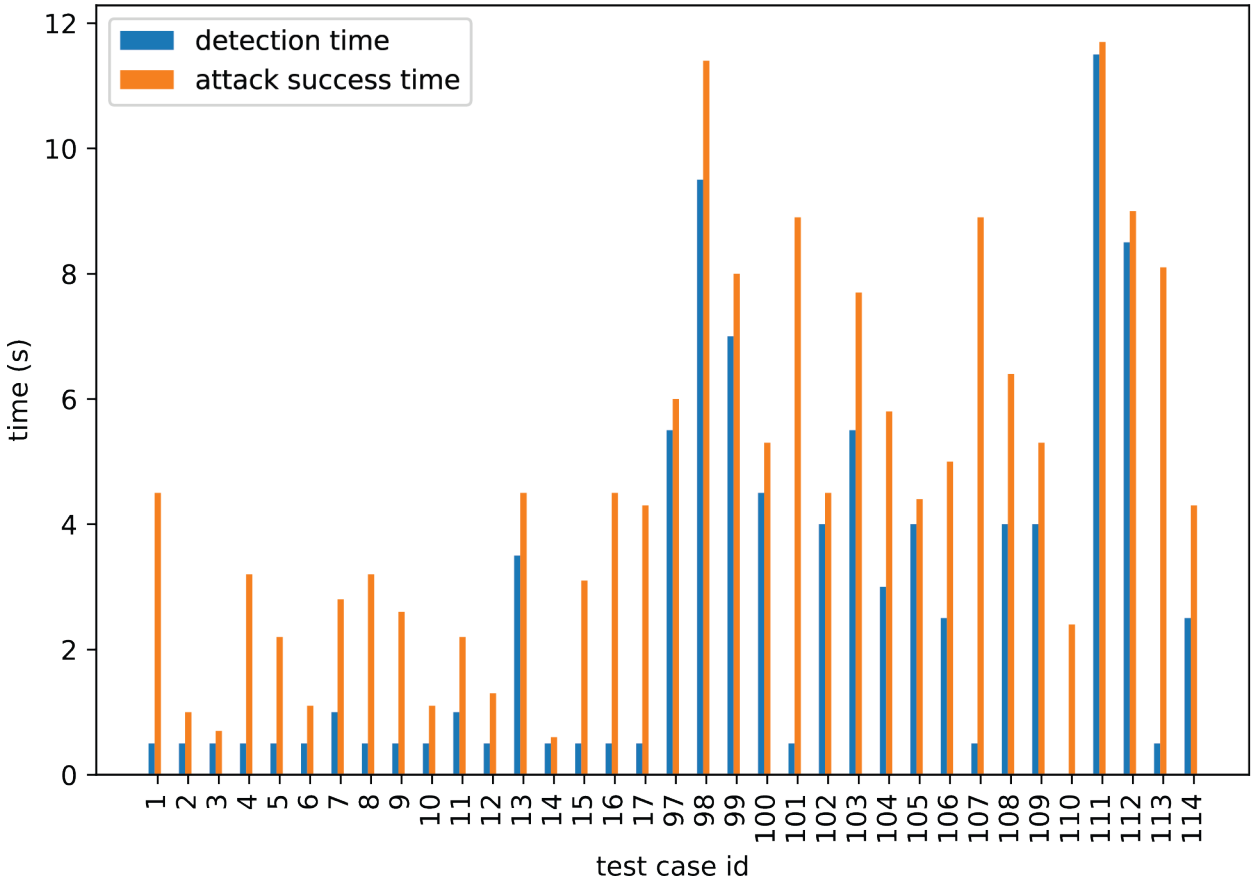



Figure 3.14: Detection time in online anomaly detection for NGSIM experiments

Similar to the KAIST experiments, both offline and online detection is conducted. In the offline detection, the false positive rate is 0% (0 out of 49), and the false negative rate is 0% (0 out of 35). Overall, the anomaly detection model shows a very good performance in the offline detection. In the online detection, the anomaly classifier checks the trajectory every 0.5 seconds until classified as abnormal or reaching the end of the attack. Overall, the false positive rate is 4.1% (2 out of 49), and the false negative rate is 2.9% (1 out of 35). Figure 3.14 shows the relationship between the detection time (blue bar) and the attack success time (orange bar) case by case. Other than

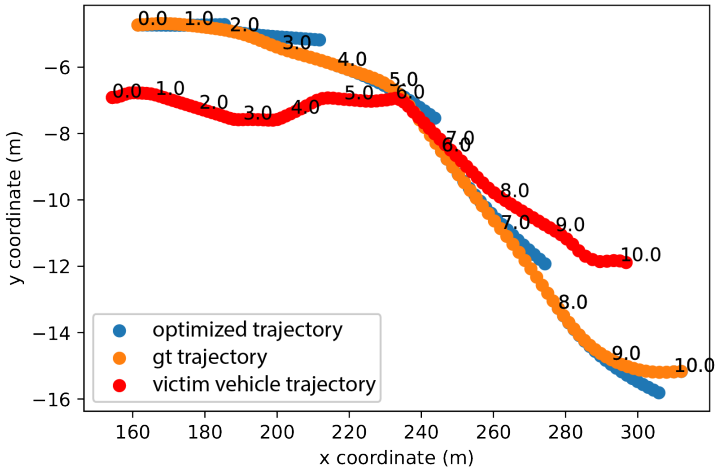
case 110, all the cases can be detected before the attack success time, and the average detection time is 2.6 seconds after the attack starts. The average attack success time is 4.7 seconds, and the average time to attack success is 2.1 seconds. Thus, even with a relatively short attack duration, the proposed anomaly detection still manages to identify the attacked trajectories.

Figure 3.15 shows two misclassification examples of a false positive (FP) case (subfigure (a)) and a false negative (FN) case (subfigure (b)) in the online detection. Subfigure (c) shows a normal lane change trajectory compared with the false negative case in subfigure (b). The number in the figure denotes the timestamp in second. For the FP case, the yellow trajectory is a normal trajectory in the NGSIM dataset, and for the FN case, the yellow trajectory is the attacked trajectory generated by the proposed attack method. The blue trajectories denote the optimized trajectories generated by the learned driving model. The red trajectories in the figure represent the trajectory of the victim vehicle. In the FP case, the ground truth trajectory performs a lane change, and the distance is very close to the victim vehicle at timestamp 5.0. In the real world, some vehicles may perform very aggressive lane changes and affect surrounding vehicles, which confuses the detection model as an FP case. In the FN case, the attacked trajectory successfully triggers the victim vehicle's warning in just 2 seconds, and the attacked trajectory is similar to a normal lane change with a smooth lateral trajectory profile shown in subfigure (c). Thus, it is difficult to identify such an attacked trajectory.

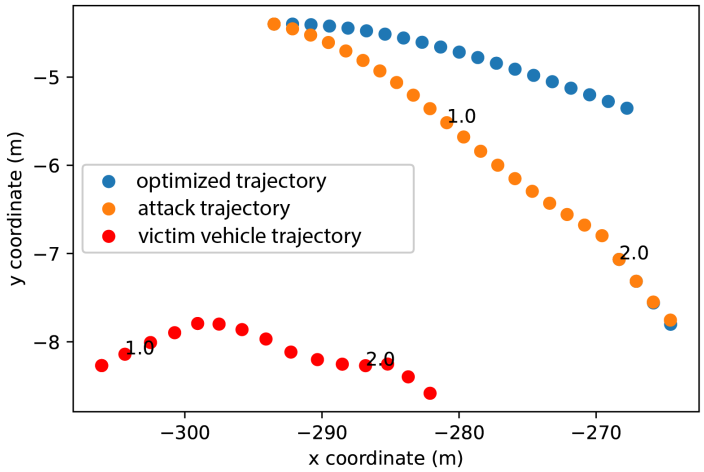
To further evaluate the generalizability of the proposed anomaly detection framework, a generalization experiment is conducted to cross-validate the effectiveness of the anomaly detector with both KAIST and NGSIM datasets. The driving model learned from the KAIST dataset is adopted in the anomaly detection of the NGSIM experiment, and all the other anomaly detection settings are the same as the original experiments in this section. In this case, the false positive rate is 8.2% (4/49), and the false negative rate is 2.9 % (1/35). The performance does not degrade too much, which proves the generalizability of the framework.

## 3.7 Detection on Adaptive Attack

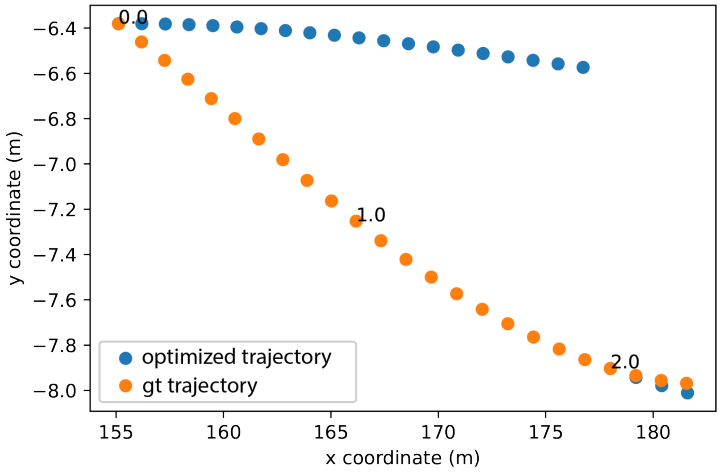
To further evaluate the capability of the proposed anomaly detection model, an adaptive attack method is designed and implemented. In the AV/CV threat model, the key idea is to add lateral deviations to the original trajectory, which either causes the subject AV hit the roadside, or leads to the emergency behavior of the victim vehicle. To make the attack more stealthy and difficult to be detected, the adaptive attack reduces the magnitude of the lateral deviations added to the ground truth trajectory, and we use the adaptive ratio to represent the significance of the magnitude reduction, ranging from 0 to 1. Figure 3.16 shows an example of the adaptive attack implemented on the KAIST dataset. The green curve denotes the ground truth trajectory that is not attacked (i.e.,




(a) A FP case




(b) A FN case




(c) Baseline for the FN case

Figure 3.15: Misclassification examples of the NGSIM experiments ((a): FP case. (b): FN case. (c) baseline for the FN case)

adaptive ratio = 0). The blue curve denotes the original attack trajectory that is evaluated in section 3.5 (i.e., adaptive ratio = 1). The orange curve denotes the trajectory under adaptive attack, with the adaptive ratio of 0.5. Notice that the lateral deviations of the orange trajectory are half of the lateral deviations of the original attack trajectory, w.r.t. the ground truth trajectory. In this way, as the adaptive ratio decreases, the adaptive attack trajectories become more and more similar to the ground truth trajectory. The adaptive attack trajectories with a very small adaptive ratio (e.g., 0.1) can be very close to the ground truth trajectories, which are very difficult to identify.

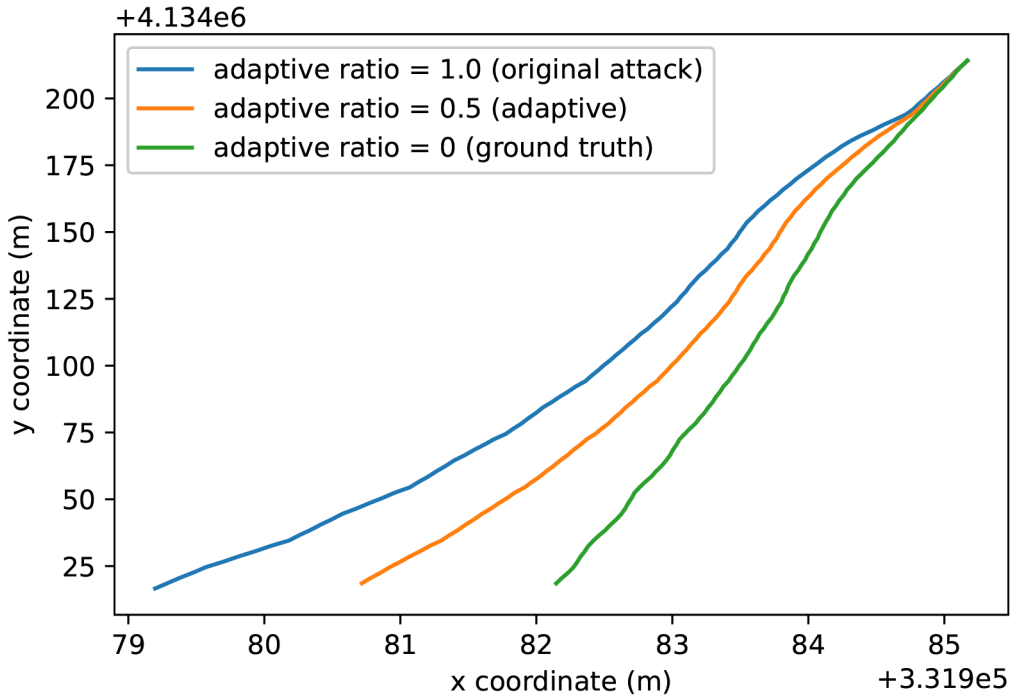



Figure 3.16: Adaptive attack example on the KAIST dataset

The adaptive attack is implemented on the KAIST dataset with the adaptive ratios of 0.8, 0.5, and 0.2. The experiment setting is the same as in section 3.5, and the driving models and the decision tree classifier are also the same. Table 3.2 shows the anomaly detection performance on the adaptive attack. When the adaptive ratio is 0.8, the performance of the anomaly detection degrades a little, with a false negative rate of 2/27. Nonetheless, the detection results are still satisfying, and most of the attacked trajectories can be identified correctly. When the adaptive ratio is 0.5, the performance of the anomaly detection is the same as the performance with the adaptive ratio of 0.8. When the adaptive ratio is 0.2, the false negative rate become 10/27. However, in this case, the adaptive attack trajectories are very close to the ground truth trajectories. In the urban scenario, the success criterion of the off-road attack is 0.895 m [93]. With the adaptive ratio of

Table 3.2: Detection performance on the adaptive attack

| Adaptive ratio | False positive | False negative |
|---|---|---|
| 0.8 | 2/23 | 2/27 |
| 0.5 | 2/23 | 2/27 |
| 0.2 | 2/23 | 10/27 |

0.2, the final lateral deviation w.r.t. the ground truth trajectory is only 0.179 m. In such cases, the subject AV is still driving within the original lane. Although the proposed model fails to detect some attack trajectories, the consequence is not hazardous.

## 3.8 Discussion and Conclusion

In this chapter, an anomaly detection model using learning from demonstration is proposed to detect GPS spoofing attacks towards the localization system of the CAVs. Maximum entropy inverse reinforcement learning is applied to learn the normal driving model. The learned driving model is then utilized to generate optimal vehicle trajectories which are compared with the observed vehicle trajectories using a decision tree classifier to determine whether the observed trajectories are under attack. The proposed detection method is evaluated in two realistic GPS spoofing attacks on AV and CV, respectively.

In both AV and CV experiments, the proposed anomaly detection method can identify most of the abnormal trajectories before the attacks succeed. Such experiment results validate the generality of the proposed anomaly detection model. Notice that although in this chapter, the anomaly detection model is only validated by GPS spoofing attack experiments, we do not utilize any specific feature of GPS signals in the anomaly detection model. In other words, the proposed model has the potentials to detect a variety of sensor attacks on the localization system as well. The reason is that the key concept of the proposed model is to compare normal versus abnormal driving behaviors. Thus, it is not sensitive to the input types or states of the localization system. As long as the driving behaviors are affected by certain cyber attacks, the proposed method can be applied to detect the anomaly.

One limitation of the proposed method is that it can be only applied to detect known attacks, because the decision tree classifier requires attacked trajectories as training data. To extend the proposed method to detect unknown attacks and to be more generic, we will explore one-class classification methods [82] where only the ground truth trajectories are needed for training a single classifier. Another limitation of this chapter is that it mainly focuses on the detection of GPS spoofing attacks without proposing defense solutions. In future work, we will investigate corresponding mitigation strategies. For example, when a trajectory is identified as abnormal, its autonomous

driving functions can be temporarily suspended until the vendor has upgraded the security system. In the online detection, a warning can be sent to the trajectory planning module of the AV to choose safe maneuvers (e.g., stop) or directly ask the driver to takeover. In a CV environment, the certificate of the attacked vehicle could be revoked so that the messages sent from this particular CV will be discarded by other vehicles.

# CHAPTER 4

# Priciple Other Vehicle Behavior Prediction

## 4.1 Introduction

### 4.1.1 Background

In the area of AVs, road users' behavior prediction is one of the most critical and complicated tasks. Considering the safety and efficiency performance, AVs not only have to acquire an accurate estimation of the current state of surrounding vehicles but also need to predict their future behaviors [77]. POVs, i.e., those vehicles whose behavior directly affects the ego AV's trajectory planning need to be predicted with high accuracy. An inaccurate behavior prediction of a POV may lead to severe outcomes.

Signalized intersections are common yet complicated urban scenarios in which traffic signals heavily affect POVs' behavior. Behavior prediction in such scenarios is a particularly challenging task, as it needs to incorporate the Signal Phase and Timing (SPaT) information to perform an accurate prediction. Only a few research efforts can be found in the literature that incorporate traffic signal information into behavior prediction models [23, 26, 116].

Another difficulty associated with POV behavior prediction in complex urban scenarios is the presence of interactive agents. A typical example is when a POV traverses a roundabout and interacts with other surrounding vehicles, it is difficult to consider the behavior prediction of the POV independently in a distributed fashion. Therefore, modeling the interaction between vehicles in an explicit and interpretable way should be emphasized in POV behavior prediction. Additionally, in many scenarios the behavior prediction problem should be solved jointly for multiple agents, leading to computational tractability issues that need to be further addressed.

Heterogeneity of driving behavior of different human drivers is another critical issue that needs to be addressed. Some drivers tend to drive more 'aggressive' while others may be more cautious. Thus, a single predictive model that predicts drivers' behavior in a uniform way might do well in the simulation yet not perform well in the real world. A well-designed behavior prediction model should be capable to adapt itself in real-time to capture the driving attitude of the driver.

Our POV behavior prediction model includes two stages: discrete intention prediction and continuous trajectory prediction. Discrete intention prediction involves the task of predicting the discrete maneuvers (e.g., pass / yield) that a POV intends to conduct. Continuous trajectory prediction aims at predicting a series of future positions of the POV given its predicted intention. Note that both discrete intention prediction and continuous trajectory prediction of POVs contribute to the trajectory planning of AVs. Discrete intention prediction of POVs could affect the discrete decision making of AVs. For example, when an AV decides whether it should change lanes, the pass or yield intention prediction of the POV in the AV's target lane becomes relevant. Similarly, continuous trajectory prediction of POVs affects trajectory planning of AVs to avoid collisions between AVs and POVs.

In this work, a hierarchical POV behavior prediction framework is proposed that incorporates traffic signal information and considers interactive agents in complex urban scenarios. Two urban scenarios, yellow light running scenarios at signalized intersections and right turn scenarios at a roundabout, are chosen as two example use cases that best represent the importance of traffic signal information and interactions in POV behavior prediction. The framework can be applied at the roadside smart infrastructure using roadside sensors (e.g., bird-view cameras). It can also be used by CAVs by detecting surrounding vehicles using their onboard sensors and receiving SPaT information from the smart infrastructure. A BN approach is adopted for the discrete intention prediction. The continuous trajectory prediction is conducted using maximum entropy IRL. More specifically, an average prediction model is learned offline by IRL. During the online prediction, a driver characteristic parameter is applied to update the average model and capture the individual differences between drivers. The potential game is adopted to explicitly formulate the interaction among vehicles in the online prediction process.

### 4.1.2 Contribution and Organization of the Chapter

The contributions of this work are three-fold:

**Proposing a novel hierarchical behavior prediction framework with consideration of traffic signal information and interactive agents:** This work presents a POV behavior prediction framework that combines both discrete intention prediction and continuous trajectory prediction by incorporating traffic signal information and considering interactive agents, in complex urban scenarios.

**Adopting a mixture of offline learning and online prediction strategies:** Learning an average human driving model from a static data set cannot represent the different driving preferences and attitudes across drivers, which is essential for accurate trajectory prediction. To address this issue, we combine offline learning of the average driving behavior with IRL and online updating of driver

characteristics with sampling methods to better predict the behaviors of different drivers.

**Explicitly modeling the interactions between vehicles:** The interactions between agents are modeled in a game theoretical approach using a potential game. The Nash Equilibrium of the potential game is guaranteed.

The rest of the chapter is organized as follows: we first present the related work in Section II, followed by the scenario overview (Section III). The methodologies for the discrete intention prediction and continuous trajectory prediction are introduced in Section IV and Section V, respectively. Next, the numerical experiments of the yellow light running scenario and the right turn scenario are presented in Section VI and Section VII, respectively. Finally, Section VIII concludes the chapter and lays out further research directions.

## 4.2 Literature review

**BN** is a widely adopted methodology in the existing literature for discrete intention prediction. Schulz et al. apply a dynamic BN to predict whether a vehicle will go straight, turn left or turn right at an intersection [90]. Schreier et al. build a comprehensive BN to perform discrete intention prediction and identify irrational driving behaviors in urban scenarios [89]. However, traffic signal information is neglected in these studies. Chen et al. use BN to predict red-light-running behaviors, utilizing the trajectory information during the yellow light [23]. Since the intention prediction is conducted only once, after the yellow light elapses 3 seconds, the discrete intention prediction results cannot be utilized for trajectory prediction during the yellow light.

For continuous trajectory prediction, **IRL** has attracted much attention since it is explainable and interpretable. Ziebart et al. propose the idea of maximum entropy inverse reinforcement learning [126], and Levine and Koltun are among the first to apply the IRL to predict human driving behaviors [62]. Sun et al. propose a hierarchical IRL formalism for probabilistic trajectory prediction in lane changing scenarios [98]. Schwarting et al. define Social Value Orientation (SVO) when applying IRL to quantify human driving social preferences and to capture the difference between human drivers [91], which is validated in a highway merging scenario. However, to the best of our knowledge, no research works on adopting IRL in complex urban scenarios exists in the literature, and this chapter aims to fill in this research gap.

**Game theory** is commonly utilized to explicitly model the interaction between vehicles [65, 33, 91, 122, 124, 72]. Zhang et al. adopt a Stackelberg game framework to predict the surrounding vehicle's behavior and plan the ego vehicle's behavior [124]. Stackelberg game is a leader-follower game that can be applied to a two-player scenario by nature, and the bi-level optimization formulation creates computational complexity. In [91], a multi-agent Stackelberg game is formulated, and the multi-level optimization problem is formulated as a joint optimization problem

to address the computational complexity. However, they do not present mathematical proof to establish the equivalency of the joint optimization problem and the multi-level optimization problem. Additionally, in some interactive scenarios, it is difficult to find the leader and the follower in the game. In [72], a potential game framework is applied in the trajectory planning problem, in which all vehicles are treated similarly without specifying leaders and followers. This framework can be generalized to multi-agent scenarios. There are a number of existing studies that have adopted the potential game for trajectory planning[72, 25]. However, to the best of our knowledge, *no existing study has adopted the potential game for the trajectory prediction problem.*

**Deep learning** is another widely adopted methodology for behavior prediction in autonomous driving applications [77]. A variety of deep learning methodologies have been adopted for vehicle behavior prediction. In [12, 128, 81], researchers adopt Long Short Term Memory (LSTM) network to conduct sequence to sequence learning [100] for the trajectory prediction tasks. The key idea of applying the sequence to sequence learning in trajectory prediction tasks is to encode the traffic information of historical observations and decode the future trajectories sequentially. To better explore and represent the observed trajectory information, convolutional neural networks (CNN) ([24, 61]) and graph neural networks (GNN) ([67, 66, 75]) are utilized and integrated with LSTM. Transformer, as a fast developing sequence to sequence learning technique, has also been applied in the trajectory prediction with good performance [127, 19]. A major drawback of applying the deep learning methodologies is that historical observations of each trajectory are required. For example, to predict the vehicle trajectory in 3 seconds, the historical observations of two seconds for such a vehicle may need to be provided to the deep learning neural networks. However, sometimes the ego autonomous vehicle may only interact with a surrounding vehicle for a short period of time, which limits the usage of the deep learning techniques. Different from the deep learning techniques, the method proposed in this chapter does not require historical observations of surrounding vehicles of the autonomous vehicle.

A few studies in the literature address the behavioral differences between human drivers when conducting trajectory prediction. In [91], a Social Value Orientation (SVO) is defined to capture the difference between human drivers. However, in the SVO map defined in this literature, human drivers may be completely selfless and aim to maximize other drivers' utilities to their own detriment, which may not be realistic in the real world. In [107], driver types are clustered to better model the driver behaviors in order to predict and compensate for the tracking error in Advanced Driver-Assistance Systems (ADAS) systems. This method is validated in a highway merging scenario. Despite the existing work in the literature, more research efforts are required to investigate the value of developing customized trajectory prediction methods that can capture the behavioral differences between human drivers, especially in complex urban scenarios.

## 4.3 Scenario overview

In general, the real-world driving scenarios can be categorized into two categories, independent scenarios and interactive scenarios. Consider a POV whose behavior may be affected by its Surrounding Vehicle (SV)s. Here, the POV refers to the principle other vehicle that may affect the behavior of the ego autonomous vehicle, and the SV refers to the surrounding vehicle that may affect the behavior of the POV. In the independent scenarios, the POV does not affect the behaviors of SVs, and therefore the SVs' behaviors can be considered as a set of environmental constraints for POV behavior prediction. In the interactive scenarios, the POV also affects the behavior of the SVs, and consequently the behavior prediction of the POV and the SVs should be conducted jointly. In this work, two driving scenarios are taken as example use cases to showcase both independent scenarios and interactive scenarios.

### 4.3.1 independent scenario - yellow light running

The independent scenario example in this work is the yellow light running scenario, which illustrates that the traffic signal plays an important role in POV behavior prediction. Figure 4.1 shows the yellow light running scenario, in which the yellow vehicle is the ego AV. The red vehicle denotes the POV whose behavior needs to be predicted, and is influenced by the blue SV and the traffic signal information. To predict the behavior of the POV, first the discrete intention prediction model is utilized to predict whether the POV chooses to pass or stop. According to the prediction of the POV's intention, the continuous trajectory prediction model is implemented to predict the detailed trajectory of the POV (red curve).

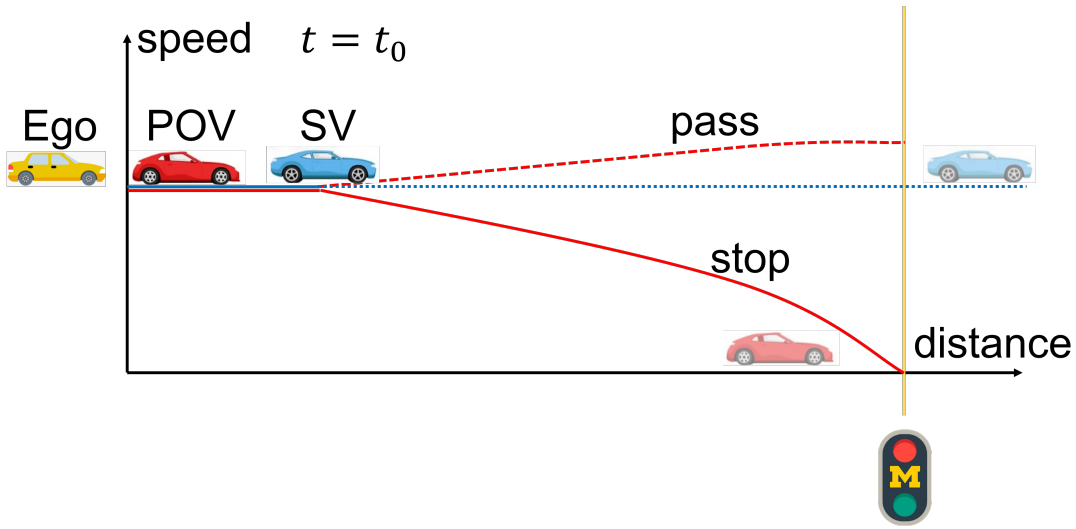



Figure 4.1: Yellow light running scenario

### 4.3.2 Interactive scenario - right turn

The interactive scenario example in this work is the right turn scenario at a roundabout, in which a POV and an SV are interacting with each other. Figure 4.2 illustrates the right turn scenario at a roundabout, where the yellow vehicle denotes the ego AV. In this scenario, the red right-turn POV's behavior needs to be predicted, accounting for the fact that it interacts with the SV within the roundabout. Similarly, the discrete intention prediction here predicts whether the POV yields to the SV, and the continuous trajectory prediction is conducted jointly for the POV and SV using a potential game to predict their detailed trajectories. Notice that although a right turn scenario is taken as an example of the interactive scenario, the methodology proposed in this work can be generalized to other pair-wise interactive scenarios (e.g., highway merging, unprotected left turn) and multi-agent interactive scenarios.

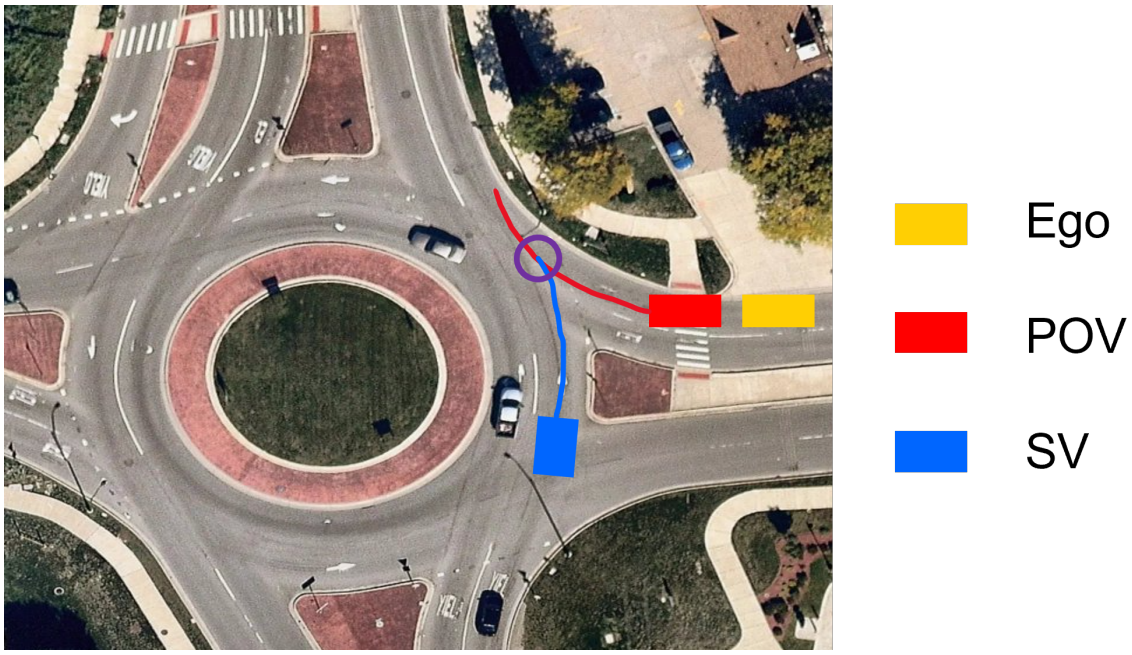



Figure 4.2: Right turn scenario

## 4.4 Discrete intention prediction

The discrete intention prediction focuses on predicting **maneuver-level** decisions. For example, in the yellow light running scenario mentioned in section 4.3.1, the goal is to predict whether the POV chooses to pass or stop at each time step during the yellow light. Such a task can be treated as a classification problem, in this work, a BN is adopted as the classifier.

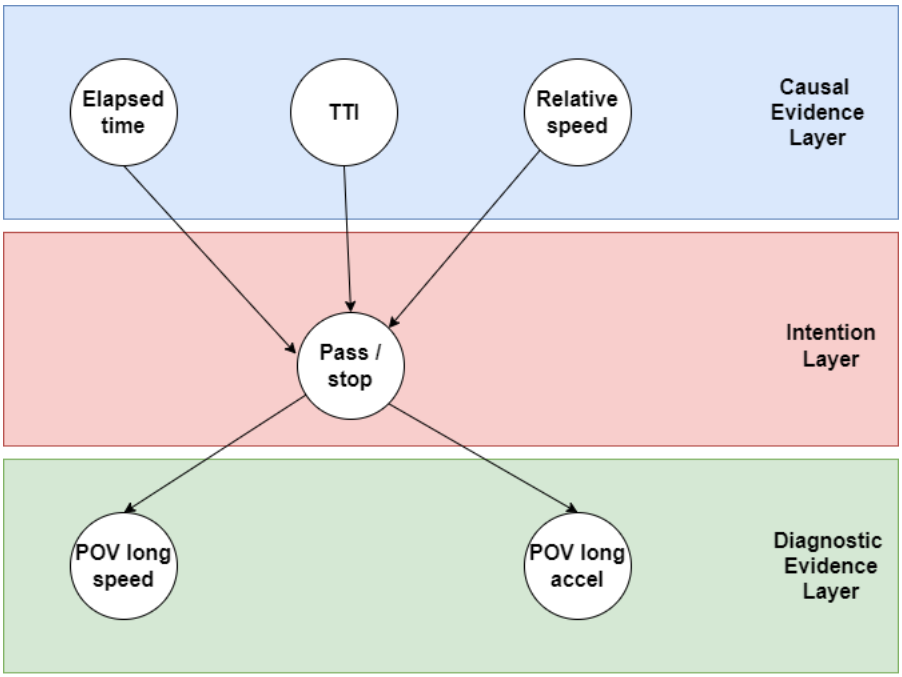
A BN can represent the dependencies between different random variables. Figure 4.3 illustrates the BN adopted for the yellow light running scenario mentioned in section 4.3.1 and the right turn scenario mentioned in section 4.3.2. Although the attributes of different scenarios are different, the networks share the same structure. The BN includes three layers: the causal evidence layer, the intention layer, and the diagnostic evidence layer. The intention layer is the middle layer, which includes the intention random variable whose probability distribution will be predicted with the BN. Causal evidence and diagnostic evidence are both observations. The causal evidence is related to the cause of the intention, usually observed from the environment, and the diagnostic evidence concern the outcome of the intention as observed from the POV. The goal of discrete intention prediction is to calculate the probability of intention ($INT$) given all causal evidence ($CE = \{CE_1, \cdots, CE_N\}$) and diagnostic evidence ($DE = \{DE_1, \cdots, DE_M\}$) as observations $P(INT|CE_1, ..., CE_N, DE_1, ..., DE_M)$. With the BN, the joint distribution of all the random variables (i.e. $P(INT, CE_1, ..., CE_N, DE_1, ..., DE_M)$) can be represented using a number of conditional distributions, shown in equation (4.1). The learning process of the BN is to calibrate the conditional distribution of $P(INT|CE_1, ..., CE_N)$ and $P(DE_\sigma|INT)$ from the dataset.

$$\begin{aligned} & P(INT|CE_1, ..., CE_N, DE_1, ..., DE_M) \\ &= \frac{P(INT, CE_1, ..., CE_N, DE_1, ..., DE_M)}{\sum_{INT} P(INT, CE_1, ..., CE_N, DE_1, ..., DE_M)} \\ &= \frac{P(INT|CE_1, ..., CE_N)\Pi_{\sigma=1}^{M} P(DE_\sigma|INT)}{\sum_{INT} P(INT|CE_1, ..., CE_N)\Pi_{\sigma=1}^{M} P(DE_\sigma|INT)} \end{aligned} \tag{4.1}$$

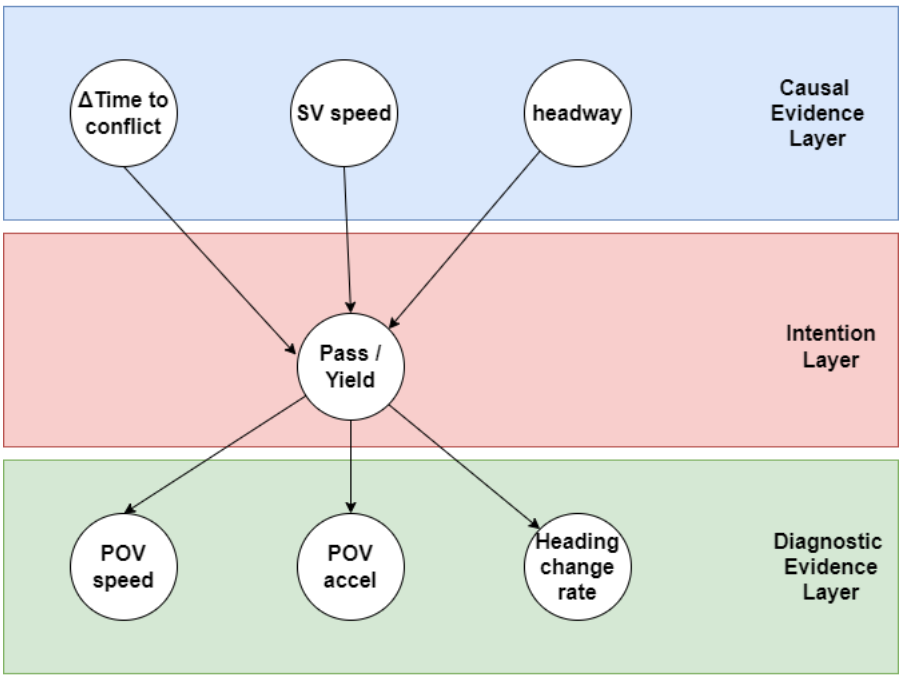
In the yellow light running scenario mentioned in section 4.3.1, the causal evidence includes elapsed yellow time, time to intersection (TTI), and relative speed to the SV. The elapsed yellow time reflects the traffic signal information. The diagnostic evidence includes longitudinal speed and longitudinal acceleration. In the right turn scenario mentioned in section 4.3.2, the causal evidence includes $\Delta$ time to conflict, which represents the absolute difference between the time to the conflict point (i.e. the purple circle in Figure 4.2) of the POV and the SV. The time to the conflict point is calculated by assuming a constant speed of the vehicle. The speed of the SV and the POV headway are also considered as the causal evidence. In the diagnostic evidence layer, during the right turn process, both the longitudinal and lateral action of the POV need to be considered. Thus, the POV speed, acceleration, and heading change rate are included as diagnostic evidence.

## 4.5 Continuous trajectory prediction

In the continuous trajectory prediction module, a generic vehicle trajectory optimization problem is formulated to model the trajectory planning process, as shown in equation (4.2). In this optimization problem $\boldsymbol{s}$ is the decision variable, which denotes the trajectory as a sequence of points $s_t$.




(a)




(b)

Figure 4.3: Bayesian network for discrete intention prediction ((a) yellow light running; (b) right turn)

Each trajectory point $s_t$ at time step $t$ can be represented by $(x_t, y_t, v_t, a_t, \psi_t, w_t)$, in which $x_t$ and $y_t$ are longitudinal and lateral coordinates, respectively. $\psi_t$ is the heading angle of the vehicle, and $w_t$ is the heading change rate. $v_t$ denotes the speed of the vehicle, and $a_t$ denotes the acceleration. $\tau$ denotes the step size between two trajectory points. Here, $\boldsymbol{p}$ represents the initial condition and environment states, which serve as input parameters to the optimization problem. For example, $\boldsymbol{p}$ may include the surrounding vehicles states $\boldsymbol{s}_{-i}$, where $i$ denotes the index of the vehicle. The objective function is a weighted sum of a series of driving features (i.e. driving objectives) $\boldsymbol{f}(\boldsymbol{s}, \boldsymbol{p})$. Each driving feature is a mapping function that maps the trajectory to a feature vector, which can be different in different scenarios. The constraints in the optimization problem represent vehicle dynamics. The details of the driving feature vector will be introduced in the appendix. The goal of the IRL is to learn the weight vector $\boldsymbol{\theta}$ associated with the feature vector $\boldsymbol{f}(\boldsymbol{s}, \boldsymbol{p})$.

$$
\begin{aligned}
&\min_{\boldsymbol{s}} \boldsymbol{\theta}^T \boldsymbol{f}(\boldsymbol{s}, \boldsymbol{p}) \\
&\text{s.t. } x_{t+1} = x_t + v_t \tau \cos \psi_t \\
&y_{t+1} = y_t + v_t \tau \sin \psi_t \\
&v_{t+1} = v_t + a_t \tau \\
&\psi_{t+1} = \psi_t + w_t \tau \\
&x_0, y_0, v_0, \psi_0 = x_{start}, y_{start}, v_{start}, \psi_{start}
\end{aligned}
\tag{4.2}
$$

Similar to Section 3.4.2, maximum entropy inverse reinforcement learning is adopted to learn the driving model represented by the weight vector in equation 4.2. The details of the derivation and the algorithm can be found in Section 3.4.2.

During the offline learning process, the trajectory data set is divided into subsets, where each subset includes trajectories with the same discrete intention. For example, in the roundabout right-turn scenario, vehicle trajectories are divided into three subsets, including the trajectory subset of the through movement vehicles within the roundabout (i.e., the SV in Figure 4.2), the trajectory subset of the right turn vehicles with the intention of yielding, and the trajectory subset of the right turn vehicles with the intention of passing. Note that the through movement vehicles have the right of the way, so in normal driving scenarios they do not need to yield to the POV. Thus, we only consider one intention for the through movement SVs. After dividing the trajectories into subsets, the offline learning process with IRL is conducted for each subset separately to learn different weight sets $\theta$ for each subset.

### 4.5.1 Potential game

Potential game is a special class of game, in which the Nash Equilibrium can be guaranteed. In a multi-agent environment, assume that each agent $i$ has a utility function $U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \boldsymbol{\theta}_{\boldsymbol{i}}^T \boldsymbol{f}_{\boldsymbol{i}}(\boldsymbol{s}, \boldsymbol{p})$.

In the potential game, a potential function $F$ for all agents can be found such that the change of the utility function of agent $i$ caused by the deviation of an action of agent $i$ (i.e. $\boldsymbol{s}_i \rightarrow \boldsymbol{s}_i'$) is equal to the same amount of change in the total potential function $F$ for all agents. The mathematical form of the definition is shown in equation (4.3).

$$U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - U_i(\boldsymbol{s}_i', \boldsymbol{s}_{-i}) = F(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - F(\boldsymbol{s}_i', \boldsymbol{s}_{-i}). \forall \boldsymbol{s}_i, \boldsymbol{s}_i', \boldsymbol{s}_{-i} \tag{4.3}$$

To construct a potential function for the trajectory prediction problem, the utility functions should be designed wisely and careful attention needs to be paid to the learning process. Notice that the utility function is a linear combination of different driving features, and to construct the potential function, the driving features are divided into two categories. We define the driving features that only involve the states of the agent itself as the independent features, which can be denoted as $f_i^q(\boldsymbol{s}_i)$. For example, the driving feature of approaching the desired speed is an independent feature, because the value of such a driving feature only depends on the speed profile of the agent itself. The driving features that involve both the state of the agent itself and other agents' states can be denoted as $f_i^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$, which are defined as interactive features. For example, the collision avoidance feature is an interactive feature because it involves both the state of the agent itself and other surrounding agents' states.

**Lemma 4.5.1.** When the utility function of each agent $i$ only consists of independent driving features $f_i^q(\boldsymbol{s}_i)$. The potential function can be constructed as the summation of the utility functions of all agents.

*Proof.* $U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \sum_q \theta_i^q f_i^q(\boldsymbol{s}_i), \forall i$
$F(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \sum_i U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \sum_i \sum_q \theta_i^q f_i^q(\boldsymbol{s}_i)$
Then, $U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - U_i(\boldsymbol{s}_i', \boldsymbol{s}_{-i}) = \sum_q \theta_i^q (f_i^q(\boldsymbol{s}_i) - f_i^q(\boldsymbol{s}_i')), \forall \boldsymbol{s}_i, \boldsymbol{s}_i', \boldsymbol{s}_{-i}$
Since all features are independent features, the deviation of $\boldsymbol{s}_i$ only affects $f_i^q(\boldsymbol{s}_i)$. $F(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - F(\boldsymbol{s}_i', \boldsymbol{s}_{-i}) = \sum_q \theta_i^q (f_i^q(\boldsymbol{s}_i) - f_i^q(\boldsymbol{s}_i')) = U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - U_i(\boldsymbol{s}_i', \boldsymbol{s}_{-i}), \forall \boldsymbol{s}_i, \boldsymbol{s}_i', \boldsymbol{s}_{-i}$

□

**Lemma 4.5.2.** When the utility function of each agent $i$ only consists of interactive driving features $f_i^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$, i.e. $U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \sum_r \theta_i^r f_i^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$. The potential function can be constructed with $f_i^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$ under following conditions.
(i) The interactive driving features satisfy the symmetric property: $f_i^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = f_{-i}^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i}), \forall -i$ involved in $f^r$. Thus, we can denote $f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = f_i^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = f_{-i}^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$
(ii) The weight associated with $f_i^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$ and $f_{-i}^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$ should be equal for agent $i$ and agent $-i$. The weight can be denoted as $\theta^r$
If the conditions are satisfied, the potential function can be constructed as $F(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) =$

$\sum_r \theta^r f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$. Notice that in this case, the interactive features $f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$ only appear once in the potential function.

*Proof.* $F(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \sum_r \theta^r f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$
According to condition (i)(ii), $U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - U_i(\boldsymbol{s}'_i, \boldsymbol{s}_{-i}) = \sum_r \theta^r_i(f^r_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - f^r_i(\boldsymbol{s}'_i, \boldsymbol{s}_{-i})) = \sum_r \theta^r(f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - f^r(\boldsymbol{s}'_i, \boldsymbol{s}_{-i})), \forall \boldsymbol{s}_i, \boldsymbol{s}'_i, \boldsymbol{s}_{-i}$
$F(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - F(\boldsymbol{s}'_i, \boldsymbol{s}_{-i}) = \sum_r \theta^r(f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - f^r(\boldsymbol{s}'_i, \boldsymbol{s}_{-i})) = U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) - U_i(\boldsymbol{s}'_i, \boldsymbol{s}_{-i}), \forall \boldsymbol{s}_i, \boldsymbol{s}'_i, \boldsymbol{s}_{-i}$
□

**Lemma 4.5.3.** In the general cases, the utility function of each agent $i$ consists of both independent features $f^q_i(\boldsymbol{s}_i)$ and interactive features $f^r_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$, i.e. $U_i(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \sum_q \theta^q_i f^q_i(\boldsymbol{s}_i) + \sum_r \theta^r f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$. The potential function should be constructed as $F(\boldsymbol{s}_i, \boldsymbol{s}_{-i}) = \sum_i \sum_q \theta^q_i f^q_i(\boldsymbol{s}_i) + \sum_r \theta^r f^r(\boldsymbol{s}_i, \boldsymbol{s}_{-i})$.

The proof of Lemma V.3 is the combination of the proof of Lemma V.1 and V.2. Notice that the difficult design in the potential game formulation is condition (ii) in Lemma V.2, because the weight vector is actually learned by maximum entropy inverse reinforcement learning. In this work, the only interactive driving feature is the collision avoidance feature. In this case, the learning process for each agent can be conducted separately. In the end, the weights of the collision avoidance feature for different agent $i$ can be adjusted to the same value by applying $\alpha_i \theta$ as the actual weight. Here, $\alpha_i$ is a zoom factor to regulate the weight vector, which does not change the optimization problem for each agent.

With the potential game formulation, some good properties can be achieved:

(a) The Nash Equilibrium can be guaranteed by directly optimizing the potential function $F$. The definition of the Nash Equilibrium is:

**Definition 4.5.1** (Nash equilibrium)**.** In the $n$ agents game, for a utility maximization problem, the strategies $(\boldsymbol{s}^*_1, ..., \boldsymbol{s}^*_n)$ are a Nash Equilibrium if for each agent $i$, $\boldsymbol{s}^*_i$ is the best response to the strategies of all other agents $-i$, $(\boldsymbol{s}^*_1, ..., \boldsymbol{s}^*_{i-1}, \boldsymbol{s}^*_{i+1}, ..., \boldsymbol{s}^*_n)$: $U_i(\boldsymbol{s}^*_i, \boldsymbol{s}^*_{-i}) \geq U_i(\boldsymbol{s}'_i, \boldsymbol{s}^*_{-i}), \forall \boldsymbol{s}_i$ [37]

(b) It is appropriate and computational tractable to directly optimize the potential function $F$ rather than solve the best response dynamics. The best response dynamics is a typical method to solve the multi-agent game, in which each agent solves its own utility maximization problem given other agents' actions. The best response dynamics usually take a long time to converge, which does not satisfy the efficiency requirements of the behavior prediction problem. When solving the optimization problem of the potential function $F$ just once, the computational efficiency can be improved significantly. **More importantly, different from the multi-agent trajectory planning problem, the behavior prediction problem for multiple agents is a centralized problem by**

**nature.** No matter whether the behavior prediction framework is implemented at the smart infrastructure or the AV, the centralized predictor needs to predict the behavior of all the surrounding POVs. It is more appropriate to adopt the potential game framework for the behavior prediction problem.

### 4.5.2 Online prediction with driver characteristic

In the offline learning process with IRL described in Section 3.4.2, the trajectories are divided into subsets according to the vehicle movement and the discrete intention. Within each subset, it still includes vehicle trajectories from different human drivers. In this way, after the offline learning process, the learned weight $\boldsymbol{\theta}$ for each subset can only represent an average human driving model.

However, drivers may have different driving preferences in the real world. Some human drivers are aggressive and prefer the mobility driving features, while others are cautious and consider the safety and smoothness driving features more. It is unreasonable to utilize an average driving model to predict different human drivers' driving behaviors. To address this issue, a driver characteristic $\lambda$ is designed in the online prediction framework. The online prediction model is shown in equation (4.4), in which we categorize the driving features into two categories, the mobility features and the safety / smoothness features. The driver characteristic $\lambda$ is adopted to balance the weight of the mobility features and the safety / smoothness features. The $\lambda$ is initialized as 0.5 at time $t_0$ when a trajectory is observed at the beginning. During the online prediction process, assume that the trajectory has been observed from time $t_0$ to the current time $t_c$. Different values of $\lambda$ are sampled and the optimization problem in equation (4.4) is solved from $t_0$ to $t_c$ to get optimized trajectories corresponding to each $\lambda$. The optimized trajectories from $t_0$ to $t_c$ are compared with the observed trajectory to find the $\lambda_c$ whose corresponding optimized trajectory fits the observation best. Then, such $\lambda_c$ will be adopted for the trajectory prediction in the next prediction window.

$$\begin{aligned} &min_{\boldsymbol{s}} \lambda \boldsymbol{\theta}_{\boldsymbol{mob}}^T \boldsymbol{f}_{\boldsymbol{mob}}(\boldsymbol{s}, \boldsymbol{p}) + (1-\lambda) \boldsymbol{\theta}_{\boldsymbol{safe}}^T \boldsymbol{f}_{\boldsymbol{safe}}(\boldsymbol{s}, \boldsymbol{p}) \\ &s.t. C(\boldsymbol{s}, \boldsymbol{u}) \leq 0 \end{aligned} \tag{4.4}$$

Overall, during the online prediction process, a potential game is formulated by constructing a potential function described in Section 4.5.1. The potential function is optimized to predict the future trajectories for each agent (i.e. SV and POV). Given the prediction results of SV and POV, for each vehicle separately, the driver characteristic is sampled to capture the driving preferences of each human-driven vehicle. The new driver characteristic $\lambda_i$ for each agent $i$ will be adopted in the potential function to make future predictions.

### 4.5.3 Features in the continuous trajectory prediction

The driving features adopted in the continuous trajectory prediction are introduced here. Notice that in section 4.5.1, the driving features are categorized as independent features and interactive features. In section 4.5.2, the driving features are categorized as mobility features and safety / smoothness features. When introducing each feature in this section, the category of each feature is also introduced.

1. Desire speed: $\frac{1}{N}\sum_t (v_t - v^{des})^2$. At each time step $t$, the vehicle tries to approach the desire speed. [Independent][Mobility].

2. Acceleration smoothness: $\frac{1}{N}\sum_t a_t^2$. Vehicle usually minimizing the control input of acceleration. [Independent][Safety/Smoothness].

3. Collision avoidance: $\frac{1}{N}\sum_t \frac{1}{(x_{it}-x_{-it})^2+(y_{it}-y_{-it})^2}$. Vehicle tends to avoid the collision from other vehicles. Notice that this feature is symmetric to agent $i$ and agent $-i$. [Interactive][Safety/Smoothness].

4. Heading smoothness: $\frac{1}{N}\sum_t (\psi_t - \rho_t)^2$. Here, the heading $\psi_t$ denotes the relative vehicle heading w.r.t. the road orientation $\rho_t$. This feature indicates the vehicle follows the road geometry. [Independent][Safety/Smoothness].

5. Lateral acceleration: $\frac{1}{N}\sum_t (a_t \sin(\psi_t))^2$. Similar to acceleration smoothness feature, vehicle usually minimizes the control input.[Independent][Safety/Smoothness].

6. Stop: $\frac{1}{N}\sum_t ((x_t - x_{stop})^2 + (y_t - y_{stop})^2)$. If the vehicle chooses to stop, the vehicle try to approach the stopping position. [Independent][Mobility].

## 4.6 Experiments of yellow light running scenario

The Next Generation Simulation (NGSIM) dataset of Lankershim Boulevard [10] is adopted to validate the behavior prediction framework in the yellow light running scenario mentioned in section 4.3.1. In the NGSIM dataset, vehicle trajectories in a traffic corridor are captured by cameras mounted on the roof of surrounding buildings. The road geometry of Lankershim Boulevard is shown in Fig. 4.4. A total of 30 minutes of trajectory data are included in the full dataset, and traffic signal data associated with vehicle trajectory data are also available in the dataset. The trajectory collecting frequency is 10 Hz.

### 4.6.1 Experiments for discrete intention prediction

In the experiments of the discrete intention prediction of the yellow light running scenario, 361 trajectories are extracted from the NGSIM dataset, which experiences the yellow light. 2/3 of the trajectories are in the training set, and 1/3 of the trajectories are in the test set. The trajectory

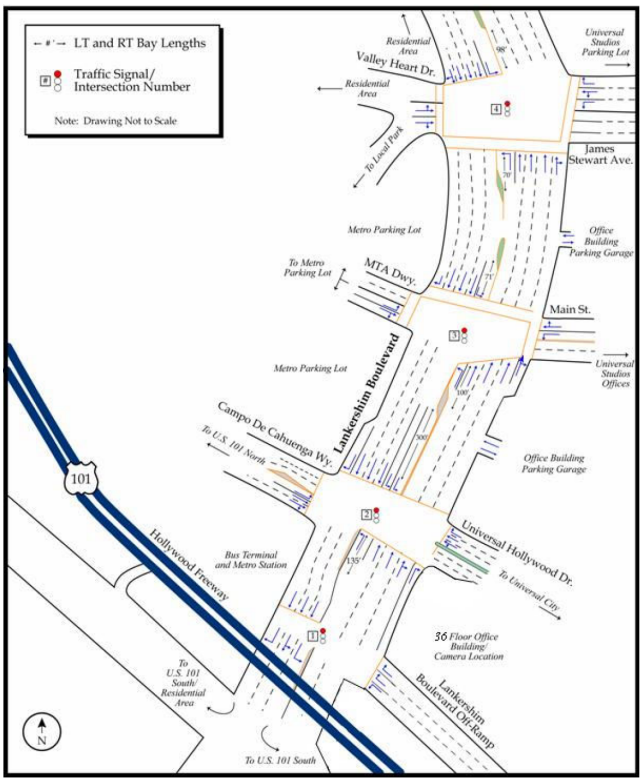



Figure 4.4: Road geometry of Lankershim Boulevard

points are manually labeled that if at the end of the yellow light, the distance to the stop bar of the vehicle is larger than a threshold, all the trajectory points of such a trajectory are labeled as stop. Otherwise, the trajectory points are labeled as pass.

To evaluate the performance of the BN, a naive predictor is developed that if the maximum travel distance within the remaining yellow time is larger than the distance to stop bar, then the prediction is pass [71]. In other words, if the vehicle is able to pass the intersection during the yellow light, the naive predictor thinks that it should pass.

The accuracy of the Bayesian network is 91.2%, while the accuracy of the naive predictor is 83.2%. It indicates that in the real world, although some vehicles can pass the intersection during the yellow light, the drivers still choose to stop for safety concerns. Notice that the discrete intention prediction is conducted trajectory-point-wise, which means that the prediction is made for each trajectory point (every 0.1 s) during the yellow phases. For some cases, the discrete intention prediction is not correct at the beginning of the yellow phase, but it gets corrected as time elapses.

Fig. 4.5 shows two specific cases that are representative (subfigure (a)-(c) and subfigure (d)-(f)) in the yellow light scenarios. The top two subfigures (i.e. (a) and (d)) are the probability-time diagrams, in which the red curves denote the probability variation of stop and the green curves

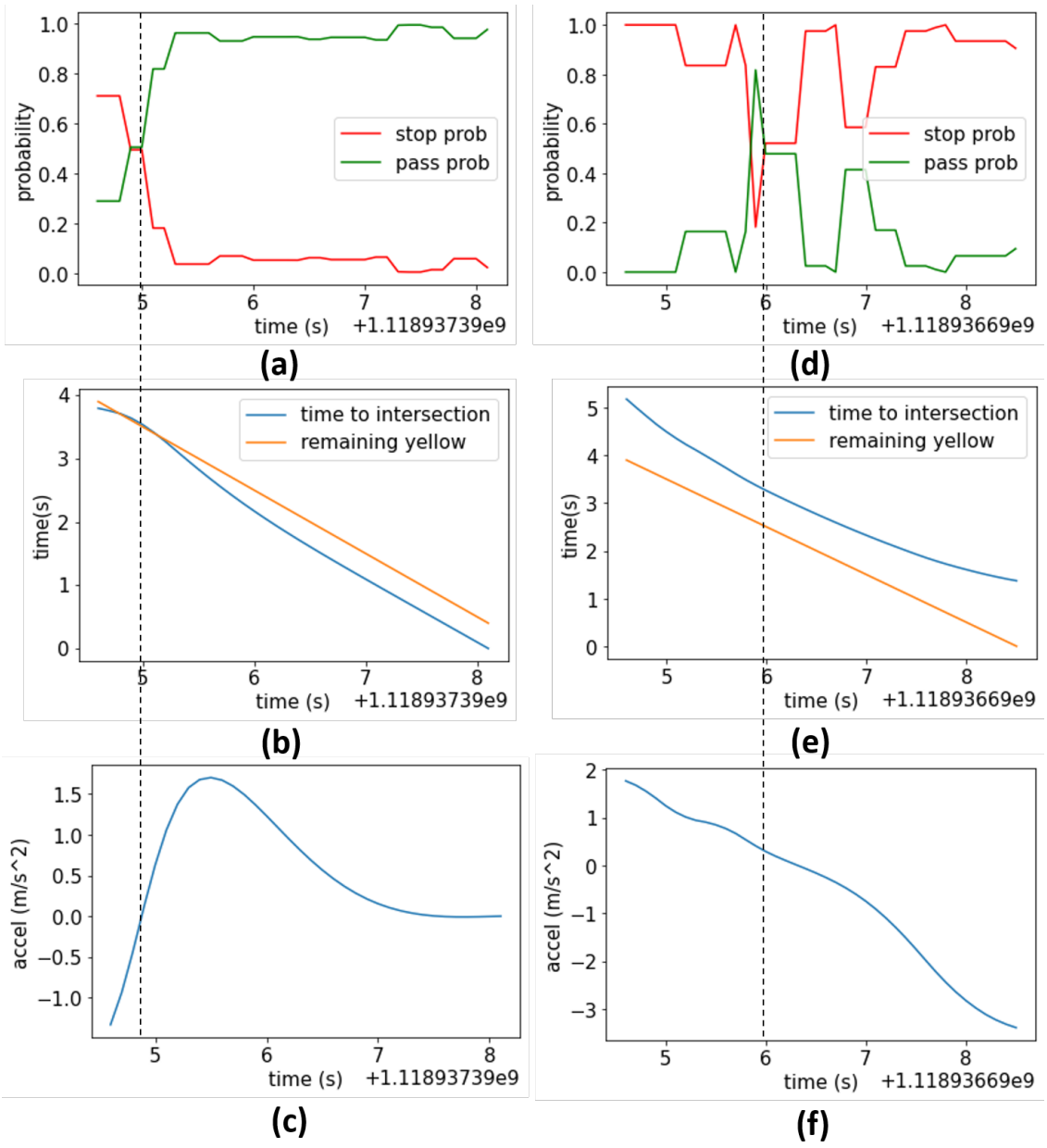



Figure 4.5: Case study of discrete intention prediction of yellow light running scenario

denote the probability variation of pass. At each time step, the summation of the probability of stop and probability of pass should always equal 1. To analyze the probability variation in these two specific cases, the profile of some key features such as remaining time, time to intersection, and longitudinal acceleration are shown in subfigures (b), (c), (e), and (f). The middle two subfigures (i.e. (b) and (e)) show the relationship between the remaining yellow time (yellow line) and time to intersection (blue curve). When the time to intersection is larger than the remaining yellow time, the vehicle is far away from the intersection and is less likely to have the intention of pass. The bottom two subfigures (i.e. (c) and (f)) are acceleration profiles. In the first case (subfigure (a)-(c)), the probability of stop is larger than the probability of pass. The reason for that is the acceleration at the beginning of the yellow is negative, which indicates the driver may want to stop when the yellow just starts. However, since the time to intersection is quite close to the remaining yellow time, the vehicle can actually pass the intersection. Thus, the vehicle starts to accelerate, and the prediction result changes to pass. In the second case (subfigure (d)-(f)), the prediction results fluctuate in the middle of the yellow time. Subfigure (e) shows that at the beginning, the vehicle is far away from the intersection (i.e. time to intersection is much larger than the remaining yellow time). However, the driver thinks the vehicle can pass the intersection and accelerates, so the time to intersection is getting closer and closer to the remaining yellow time. Because of that, the probability of stop goes down at the beginning. Later on, the driver realizes that it is impossible to pass the intersection without violating the red light because the distance to the intersection is still too far, then the vehicle starts to decelerate and the probability of stop rises back.

Table 4.1 shows the transferability performance of the BN. Cross validation experiments are conducted in which the trajectories at three intersections are utilized. For each experiment in the table, the trajectories at two intersections will be utilized to train the BN, and the trajectories at the remaining intersection are in the test set. In the table, the performance of the BN is compared with the Naive predictor. In all cross validation experiments, the performance of the BN outperforms the naive predictor, which is consistent with the overall performance.

Table 4.1: Cross validate of the Bayesian Network

| Accuracy | Expriment 1 | Experiment 2 | Experiment 3 |
|---|---|---|---|
| BN | 92.2% | 90.7% | 88.2% |
| Naive | 77.3% | 85.0% | 81.6% |

### 4.6.2 Experiments for continuous trajectory prediction

In the continuous trajectory prediction section, 60 training trajectories and 60 testing trajectories are extracted from the NGSIM dataset. In the training set, half of the trajectories pass the intersec-

tion during the yellow light, while the others choose to stop. All the testing trajectories experience the yellow light, whose intentions are unknown.

The continuous trajectory prediction is conducted for 3-second prediction windows and updated every 0.5 seconds. The Average Displacement Error (ADE) is chosen as the error measurement. The calculation of the ADE is shown in (4.5), which measures the average Euclidean distance between the observed trajectory and the predicted trajectory in the prediction window. A naive predictor is designed that assumes constant speed and heading in the prediction window. The prediction accuracy of our framework is 0.5 m on average for all testing trajectories and the prediction accuracy of the naive predictor is 1.75 m.

$$ADE = \frac{1}{N}\sum_{i=0}^{T}\sqrt{(x_i^{obs} - x_i^{pred})^2 + (y_i^{obs} - y_i^{pred})^2} \tag{4.5}$$

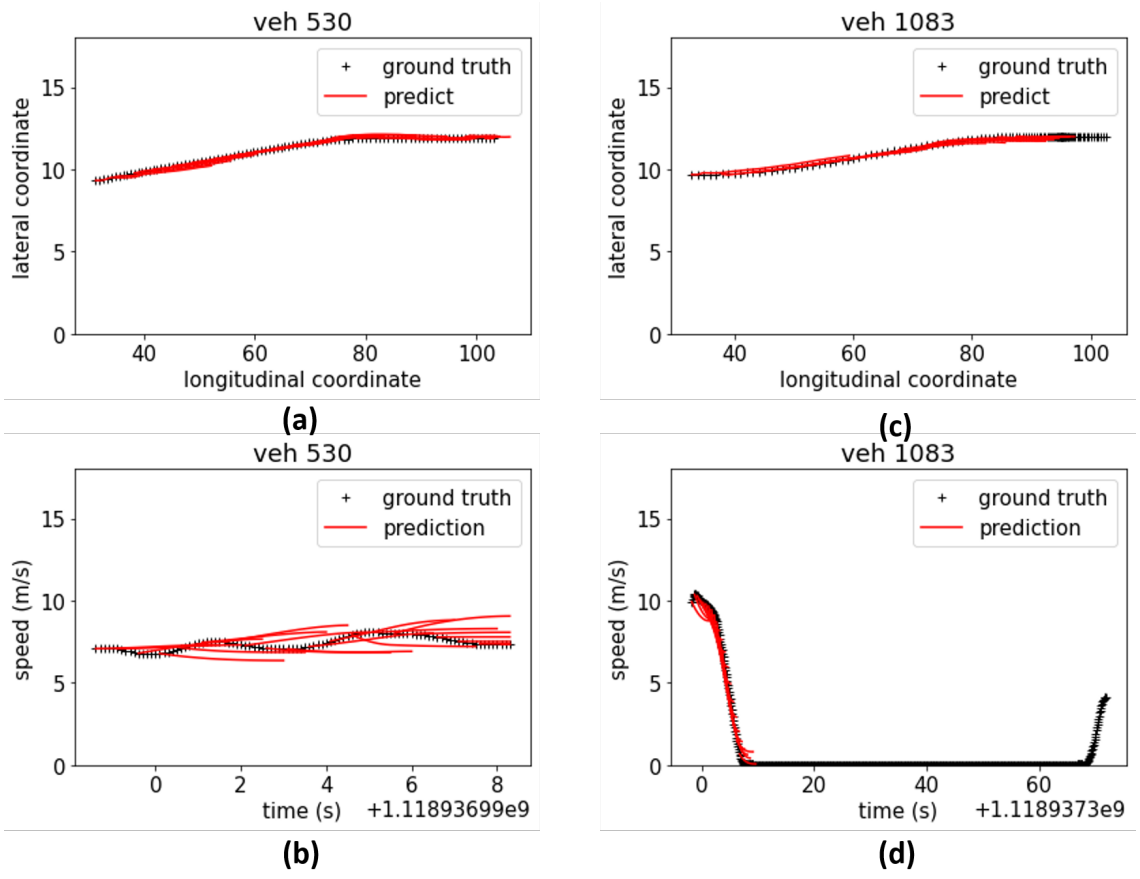



Figure 4.6: Case study of continuous trajectory prediction of yellow light running scenario

Fig. 4.6 shows the continuous trajectory prediction of two cases. In all four subfigures, black curves with crosses denote the actual trajectory, and the red curves denote the prediction conducted

every 0.5 seconds with a 3-second prediction horizon. Subfigure (a) and (b) shows the position profile and speed profile respectively of the pass sub-scenario, and subfigure (c) and (d) show the position profile and speed profile respectively of the stop subscenario. In both sub-scenarios, the prediction is quite close to the ground truth, in terms of the position profile and speed profile.

Different from other literature that only evaluates the average performance among all testing trajectories, we also evaluate the overall accuracy distribution for all testing trajectories. Figure 4.7 shows the error distribution of the predictor proposed in this work, In all test cases, the prediction error in a 3-second horizon is less than 1 meter.

Notice that in the experiments of continuous trajectory prediction, it is not assumed that the upper-level discrete intention prediction is perfectly accurate. The impact of the discrete intention prediction is analyzed for each testing trajectory. When the intention prediction is accurate, the average ADE is 0.49 m. With wrong intention prediction, the average ADE is 0.58 m. Such results are consistent with the accuracy of discrete intention prediction since over 90% of the time the discrete intention prediction is correct.

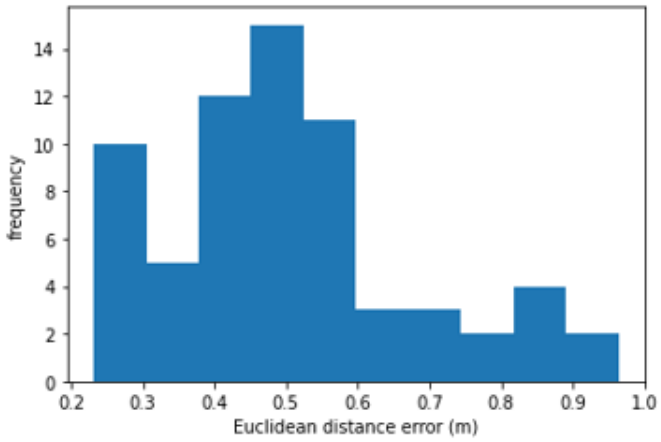



Figure 4.7: Prediction error distribution of the yellow light running scenario

## 4.7 Experiments of right turn scenario

To validate the behavior prediction framework in the right turn scenario mentioned in section 4.3.2, a trajectory dataset from a two-lane roundabout in Ann Arbor, Michigan is adopted in the experiment [125]. The roundabout is at the intersection of the State St. and Ellsworth Rd., where infrastructure sensors (i.e. cameras, radars) are installed at the four corners. In this way, all the vehicle trajectories approaching and entering this roundabout can be detected and collected in a 24/7 manner [127]. The frequency of recording the trajectories is 2.5 Hz. Figure 4.2 shows the

geometry of this roundabout, and the right turn vehicle trajectories and through movement vehicle trajectories at the outer lane of the roundabout are extracted from one day at the northeast corner for this experiment, as illustrated by the figure. Figure 4.8 shows the overview of the trajectory data at the two-lane roundabout, in which the x-axis denotes the longitude, and the y-axis denotes the latitude.

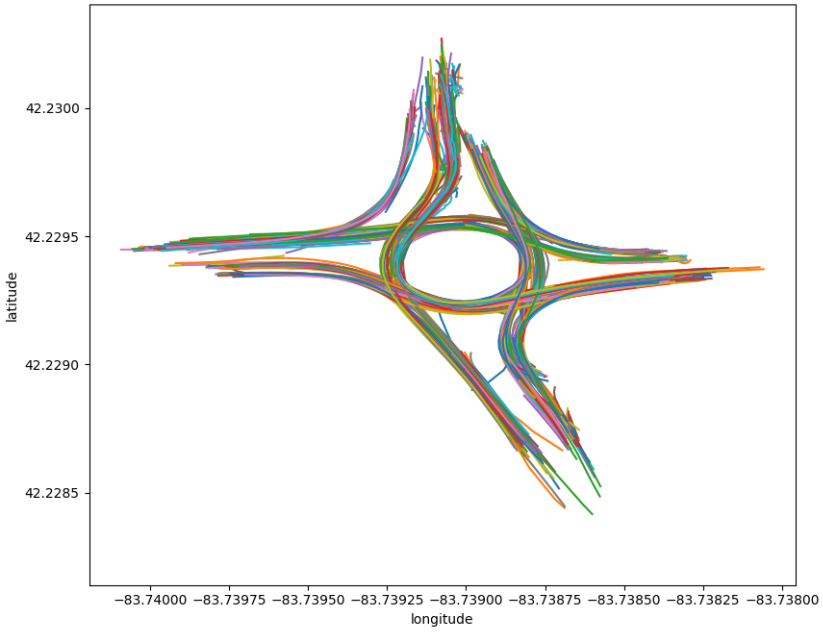



Figure 4.8: Trajectory overview at the two-lane roundabout

### 4.7.1 Experiments for discrete intention prediction

Before applying the BN on the trajectory dataset, the intention of each trajectory point at each timestamp is manually labeled. Figure 4.9 shows the labeling criteria. For a trajectory, if at some moment its speed is lower than a threshold ($v_{stop}$), then before its speed reaches the threshold ($t < t_1$), the trajectory points are labeled as yield, illustrated by trajectory 1 in the figure. After it accelerates and its speed exceeds $v_{stop}$, the intention of those trajectory points are labeled as pass. For a trajectory that always has a speed that is larger than the stop speed $v_{stop}$ (e.g. trajectory 2 in the figure), all its trajectory points are labeled as pass.

A naive predictor is designed as the baseline. It assumes that if $\Delta$ time to conflict is smaller than a threshold, the right turn POV chooses to yield. As is discussed in the discrete intention

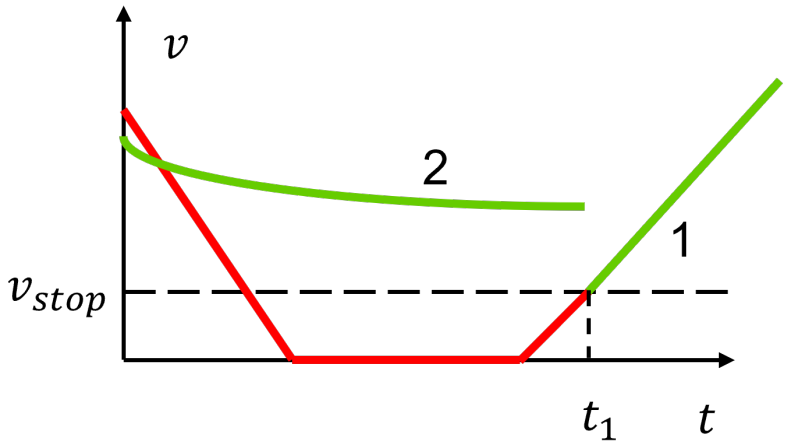



Figure 4.9: Intention labeling criteria

prediction section, if the right-turn vehicle follows the right of the way, it should yield the through movement vehicle when the time difference to the conflict point is small.

Overall, 191 vehicle trajectories that have more than 2 seconds of interaction (i.e. more than 5 trajectory points) with the through movement vehicle are adopted in the experiment. 2/3 of them are utilized for training, and 1/3 are utilized for testing. The detection rate of the BN is 87.7%, while the naive predictor has the prediction accuracy of 80.9%.

Figure 4.10 shows two case studies (i.e. subfigure (a)-(c), subfigure (d)-(f)) of the discrete intention prediction. In the figure, the blue trajectories denote the through movement vehicles within the roundabout. The green and red trajectories denote the entrance vehicles, and the green represents the intention of pass, while the red denotes the intention of yield. The first column shows the ground truth intention of the right-turn vehicles, and the second column shows the prediction results of the right-turn vehicles. The last column shows the probability variation w.r.t. time. In the first case study, the prediction of the initial three trajectory points is pass, while the ground truth is yield. The reason is that the initial speed is large (i.e. the distance between two trajectory points is large), and the prediction result is pass. Later, as the vehicle decelerates, the prediction goes to yield, and after that the prediction results are accurate. In the second case, two POVs, denoted by circle and rectangle respectively, interact with the through movement vehicle. Both right-turn vehicles have the intention of pass. The prediction results of the first vehicle are accurate, but for the second vehicle, the initial prediction of the vehicle is yield. The reason is that the initial speed of the second vehicle is low. In other words, the second vehicle decelerates when approaching the roundabout, and its speed is close to the stop speed. In such a corner case, the prediction results predict wrongly at the beginning, but it becomes accurate after a few time steps.

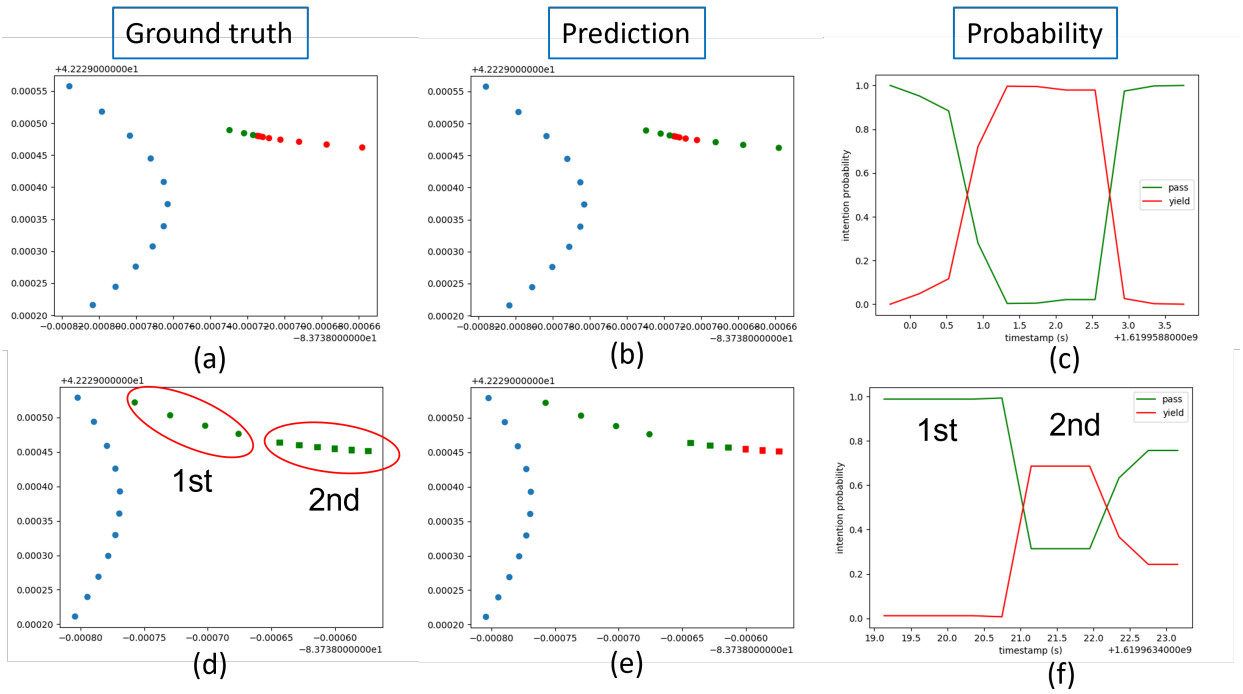


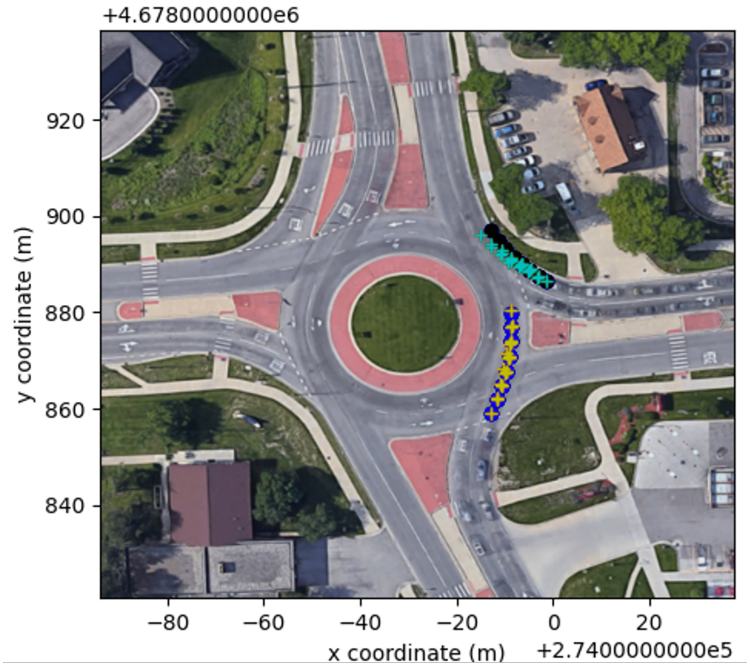

Figure 4.10: Case study of intention prediction at the roundabout

### 4.7.2 Experiments for continuous trajectory prediction

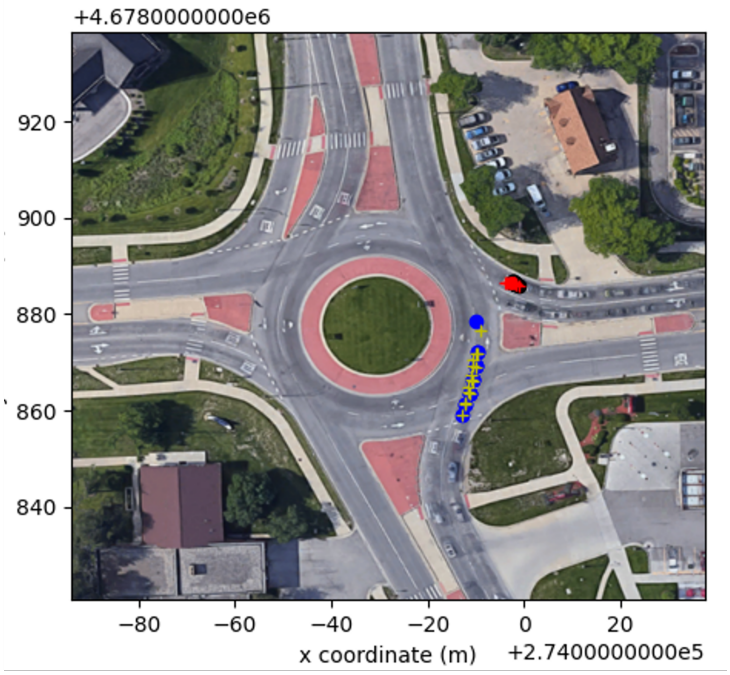
In the experiments for continuous trajectory prediction, 2224 right-turn trajectories are extracted from the trajectory dataset collected at the two-lane roundabout. Each right-turn vehicle (i.e. POV) has at least two seconds of interaction with the through movement vehicle (i.e. SV) in the roundabout. 1605 trajectories are in the training set and 619 trajectories are in the testing set. The testing set includes trajectories of different days under different operating scenarios (e.g. different weather, weekday / weekend). The prediction is conducted every 0.4 seconds, and the prediction horizon in this experiment is 2 seconds.

The ADE is adopted as the prediction accuracy measurement, the same as in the experiment for the right turn scenario mentioned in section 4.3.2. The average ADE of POVs is 0.51 m, and the average ADE of SVs is 0.93 m. To further analyze the performance, the prediction accuracy is evaluated for the pass or yield right turn vehicles separately. When the right turn vehicles choose to pass, the prediction accuracy is 0.76 m, and the ADE of the corresponding interactive SV is 0.87 m on average. When the right turn vehicles choose to yield, the prediction accuracy is 0.44 m, and the ADE of the corresponding interactive SV is 0.94 m on average. Notice that when the right turn vehicles choose to yield, they usually have relatively low speed profiles, and the prediction accuracy when the vehicles travel at low speed is higher.

Figure 4.11 illustrates the trajectory prediction results for a pass scenario and a yield scenario




(a) Pass




(b) Yield

Figure 4.11: Case study of trajectory prediction at the roundabout

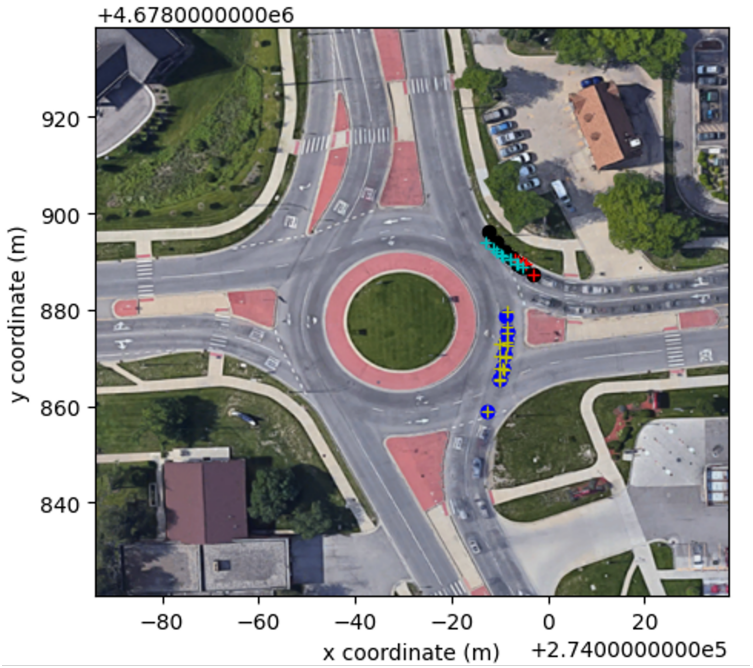
in subfigure (a) and (b) respectively. In the figure, the x-axis and the y-axis denote the local x coordinate and local y coordinate, respectively. The blue dots denote the ground truth trajectory of the SV, and the black dots denote the trajectory of the POV. The yellow crosses represent the trajectory prediction results of SV. In both cases, the trajectory prediction of the SV fits the ground truth well. The cyan crosses denote the trajectory prediction of the POV when the discrete intention prediction is pass, and the red crosses denote the trajectory prediction of the POV when the discrete intention prediction is yield. In these two cases, the discrete intention prediction is correct, and the continuous trajectory prediction of the POV is very similar to the ground truth POV trajectories.

Notice that in the continuous trajectory prediction, it is not assumed that the discrete intention prediction is perfectly accurate. An intuitive question would be: how much does the wrong discrete intention prediction affect the continuous trajectory prediction? Figure 4.12 illustrates two corner cases when the discrete intention prediction is not accurate at some moment. In subfigure (a), the actual intention of the right turn vehicle is pass as the ground truth. However, the initial discrete intention prediction is yield, which leads to the continuous trajectory prediction results as the red crosses. Although such continuous trajectory prediction results follow the trend of the ground truth trajectory, the speed prediction of the POV deviates from the ground truth and predicts the vehicle to slow down. Later, the discrete intention prediction changes to pass at the second time step and the continuous trajectory prediction fits the ground truth well. In subfigure (b), the actual intention of the right turn vehicle is yield, but the initial discrete intention prediction is pass. In this case, it is even better than case (a) because the vehicle already has a low speed. The error induced by the wrong discrete intention prediction is smaller in the continuous trajectory prediction.

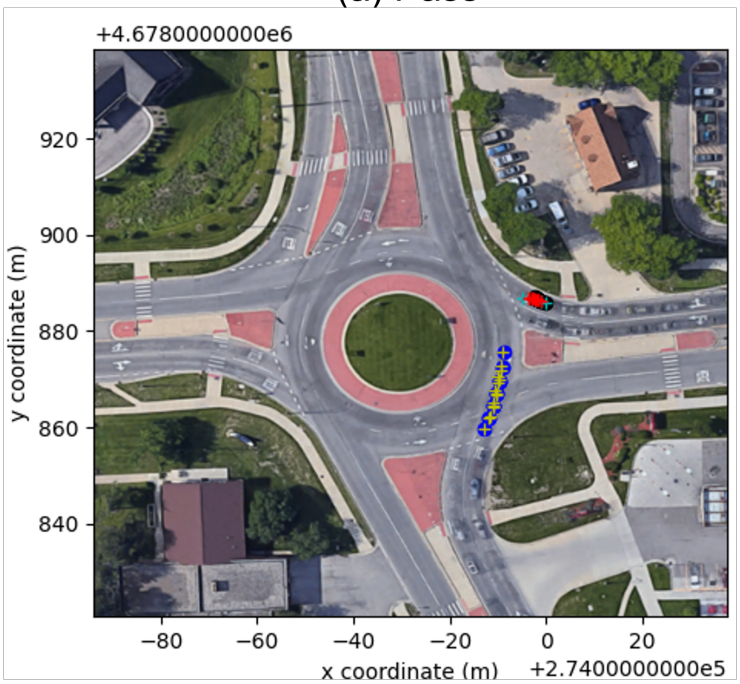
In subfigure (b) of Figure 4.11 and subfigure (a) of Figure 4.12, there are frame missing issues in the SV trajectories. The blue dot trajectories miss one frame in each case. However, the predictor proposed in this work still manages to predict the SV trajectory accurately after the missing frame. It illustrates the robustness of the predictor in this work.

## 4.8 Conclusion

In this chapter, a hierarchical behavior prediction framework is proposed that incorporates traffic signal information and considers interactive agents in urban scenarios. The framework has two stages: discrete intention prediction and continuous trajectory prediction. The discrete intention prediction is conducted with BN. Given the discrete intention prediction results, the continuous trajectory prediction is conducted by applying the maximum entropy inverse reinforcement learning and the potential game theory. In the offline learning process, IRL is adopted to learn the average driving model and the base predictor. In the online prediction process, a driver characteristic is designed to capture the difference among different human drivers, and the potential game




(a) Pass




(b) Yield

Figure 4.12: Corner cases of trajectory prediction at the roundabout

is formulated to explicitly model the interaction among vehicles.

The behavior prediction framework is validated with two experiments: the yellow light running scenario and the right-turn scenario, which best represents the importance of the traffic signal information and the interactions among the vehicles. In the yellow light running scenario, the ADE as the error measurement is 0.5 m on average for the three-second prediction horizon. In the right turn scenario, the ADE of the POV trajectory prediction is 0.51 m on average for the two-second prediction horizon. In both experiments, the connection between the discrete intention prediction and the continuous trajectory prediction is analyzed in detail.

In future work, a crash / near-crash event predictor may be developed following the framework of this work. In this work, the normal POV driving behaviors are predicted, which are the majority in the real-world trajectories. However, the corner rare events are also very important because they significantly affect the safety performance of autonomous vehicles. A crash or near-crash event prediction framework may be developed by comparing the observed trajectory with the predicted trajectory using the prediction framework in this work, to identify the abnormal crash or near-crash trajectories.

# CHAPTER 5

# Summary and Future Directions

## 5.1 Summary of the thesis

With high expectation of AV from academia and industry, the actual progress of AV deployment is not satisfying because of a variety of challenges, especially in complex urban scenarios. To address the challenges in AV's autonomy stack, smart infrastructure can play an important role in a cooperative driving approach. In this thesis, a cooperative driving framework is proposed to provide guidance or warnings to support the trajectory planning of the AVs.

First, an integrated control framework is proposed that optimizes the traffic signals and provide high-level guidance for CAV's trajectory planning. Although existing studies make good achievements in CAV based intersection management, they are constrained by three drawbacks: assumption of 100 % penetration rate of CAVs, centralized formulation, and isolated intersection. In this study, we overcome these drawbacks by proposing a decentralized approach for integrated control in a traffic corridor under mixed traffic conditions. Instead of controlling the CAVs by a centralized controller, our framework provides high-level guidance to CAVs to assist their trajectory planning, and the detailed trajectories are generated by the CAVs themselves. The integrated control framework is validated in microscopic traffic simulation with realistic settings from a real-world traffic corridor. The simulation shows the mobility and fuel economy performance improvement of the integrated control framework, and more benefits are achieved with higher penetration rate of CAVs.

Second, an anomaly detection module using learning from demonstration is designed to identify abnormal trajectories caused by cyber attacks on the localization module of CVs and AVs. Most existing defense methods for the localization module are signal processing methods, which requires physical access to the signal receivers in the localization module. In this work, instead of investigating the signal receivers, the vehicle and transportation domain knowledge is adopted to identify the anomalies. A learning from demonstration method is utilized to learn the normal driving model from the historical trajectory datasets of CVs or AVs. To determine whether the observed trajectory is abnormal or not, a statisitical method is developed to quantify the difference

between the observed trajectory and optimized trajectory generated by the learned driving model, by training an anomaly classifier of a decision tree. The anomaly detection method is evaluated by two real-world datasets, for AV and CV experiments separately. In both experiments, the false positive rate and false negative rate are quite satisfying, and most of the attacks can be identified before the attack success time. To further explore the capability of the anomaly detection, an adaptive attack is designed to reduce the influence of the attack, making it more stealthy. Under the adaptive attack with more challenging conditions, the anomaly detection method proposed still has a reasonable performance.

Third, a novel hierarchical POV behavior prediction framework is developed, incorporating traffic signal information in complex urban scenarios with interactive agents. POV refers to the vehicles that may affect the behavior of the ego AV. The behavior prediction is divided into two stages, discrete intention prediction and continuous trajectory prediction. At the discrete intention prediction stage, the discrete maneuvers or intentions of the POV is predicted with Bayesian networks. Given the discrete intention prediction results, the continuous trajectory prediction focuses on predicting a series of future positions of the POV. The continuous trajectory prediction is conducted with maximum entropy inverse reinforcement learning. To explicitly model the interactions between vehicles, a potential game framework is adopted in this work. Different from other literature that develops one predictor for different human-driven vehicle behavior prediction, in this work a mixture of offline learning and online prediction strategies are designed, and driver characteristics are utilized in online prediction to capture the difference among human drivers (e.g. aggressiveness). The behavior prediction framework is validated in an independent scenario (i.e. yellow light running scenario) and an interactive scenario (i.e. roundabout right turn scenario). Two real-world datasets are adopted to evaluate the performance of the proposed predictor, and average displacement error of the prediction is low in both experiments.

In summary, this thesis presents a series of applications of the smart infrastructure to assist the trajectory planning of CAVs by providing guidance or warnings. The proposed methods address the challenges of CAVs' autonomy stack, which pave the way for the development and deployment of CAVs in the near future.

## 5.2 Future directions

This thesis not only develops several critical applications in an infrastructure-based cooperative driving framework, but also points out several future research directions.

**Safety critical prediction**. In Chapter 3, an anomaly detection framework is proposed to identify abnormal trajectories caused by cyber attacks. Notice that the framework proposed is generalizable to different cyber attacks since the key idea is to learn the normal driving model and to

compare the observed trajectory to the normal driving behaviors. An intuitive question is whether this method can be adopted to identify other misbehavior, like crash or near-miss events. In a crash or near-miss event, some human-driven vehicles behave abnormally, which should be detected in advance so that the nearby CAVs will not be affected. In the safety critical prediction, smart infrastructure can also serve an important role to monitor the microscopic traffic situation. It may predict the crash events in advance and provide warnings to CAVs. However, it is a more difficult question that anomaly detection for cyber attacks since in a crash scenario, the vehicle may only have one or two seconds of misbehavior. It is a critical yet challenging task to predict the crash in advance, which worth more research efforts.

**Prediction and planning integration** In Chapter 4, a hierarchical behavior prediction is proposed to predict POVs' behavior for CAVs. In the current autonomy stack of autonomous driving industry, the prediction and planning are separate modules. The behavior prediction serves as the upstream component of trajectory planning. However, in recent years, more and more researchers realize the impact of the trajectory planning on the trajectory prediction. In an interactive scenario, the trajectory planning of the ego CAV affects the trajectory prediction of surrounding POVs, and the prediction results of POVs affect CAV trajectory planning back. Research efforts start to focus on jointly solving the trajectory prediction and planning problem. In this case, how to jointly integrate the trajectory prediction at the smart infrastructure with bird view cameras and the trajectory planning at the CAVs become a new research question. It is easier to model the interactions at a global view from the smart infrastructure during the trajectory prediction process, but some wise designs are required to integrate prediction and planning at the CAV.

**Decision fusion** In this thesis, a series of applications of cooperative driving are proposed, in which guidance or warnings are provided by the smart infrastructure to assist CAVs' trajectory planning. In Chapter 2, the guidance of time of arrival is integrated into CAV's trajectory planning as an example. However, for other guidance or warnings derived from Chapter 3 and 4, the integration between the guidance or warning and the CAVs' trajectory planning is not modeled explicitly in this thesis. The integration between the guidance and CAVs' decision making process can be considered as a decision fusion problem. Several research questions are critical in the decision fusion area. When should a CAV take the guidance or warnings from the smart infrastructure? How can the guidance and warnings participate in CAVs' trajectory planning algorithm? Real world testing and evaluation are required for the decision fusion of CAVs before the deployment.

**Large scale cooperative driving** This thesis introduces a cooperative driving framework focusing on isolated intersections or a traffic corridor. When it comes to large-scale smart infrastructure deployment at city level, many research questions are worthy of exploration. The learning from demonstration method (i.e. data-driven) methods proposed in this work should be generalized to large scale transportation network via transfer learning. City-level cooperative driving may also

provide planning guidance at higher level, like route choice guidance for CAVs. Similar to Chapter 2, the computational tasks should be distributed between the traffic management center at the city and the edge computing system at each intersection appropriately to balance the computational efficiency and the optimality of the whole system. It is expected to have more research, industry, and government efforts on the large scale cooperative driving for CAVs.

# Bibliography


[1] Argo ai is testing fully driverless vehicles in miami and austin. https://www.theverge.com/2022/5/23/23137834/argo-ai-driverless-test-av-miami-austin. Accessed: 2022-05-23.

[2] Automated vehicles for safety. https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety.

[3] Gm's cruise wins first california permit to carry paying riders in driverless cars. https://www.reuters.com/business/autos-transportation/self-driving-cars-gms-cruise-could-soon-carry-paying-riders-san-francisco-2022-06-02/. Accessed: 2022-06-02.

[4] Inside the self-driving tesla fatal accident. https://www.nytimes.com/interactive/2016/07/01/business/inside-tesla-accident.html. Accessed: 2016-07-12.

[5] Mercedes-benz wins world's first approval for level 3 autonomous cars: What's that mean? https://www.slashgear.com/782536/mercedes-benz-wins-worlds-first-approval-for-level-3-autonomous-cars-whats-that-mean/. Accessed: 2022-02-28.

[6] Self-driving car pulled over in san francisco, confounding police. 'ain't nobody in it.'. https://www.usatoday.com/story/tech/2022/04/13/self-driving-car-california-police/7302081001/. Accessed: 2022-04-13.

[7] Tesla autopilot - wikipedia. https://en.wikipedia.org/wiki/Tesla_Autopilot.

[8] Waymo is expanding its driverless program in phoenix. https://techcrunch.com/2022/05/18/waymo-is-expanding-its-driverless-program-in-phoenix/. Accessed: 2022-05-18.

[9] SMA Bin Al Islam and Ali Hajbabaie. Distributed coordinated signal timing optimization in connected transportation networks. *Transportation Research Part C: Emerging Technologies*, 80:272–285, 2017.

[10] Vassili Alexiadis, James Colyar, John Halkias, Rob Hranac, and Gene McHale. The next generation simulation program. *Institute of Transportation Engineers. ITE Journal*, 74(8):22, 2004.

[11] Mohammed Hamad Almannaa, Hao Chen, Hesham A Rakha, Amara Loulizi, and Ihab El-Shawarby. Field implementation and testing of an automated eco-cooperative adaptive cruise control system in the vicinity of signalized intersections. *Transportation research part D: transport and environment*, 67:244–262, 2019.

[12] Florent Altché and Arnaud de La Fortelle. An lstm network for highway trajectory prediction. In *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)*, pages 353–359. IEEE, 2017.

[13] PTV America. Ptv vistro user manual. *Ptv Ag*, 2:07, 2014.

[14] Eduardo Arnold, Omar Y Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. A survey on 3d object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3782–3795, 2019.

[15] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert Systems with Applications*, 165:113816, 2021.

[16] Matthew Barth, Sindhura Mandava, Kanok Boriboonsomsin, and Haitao Xia. Dynamic eco-driving for arterial corridors. In *2011 IEEE forum on integrated and sustainable transportation systems*, pages 182–188. IEEE, 2011.

[17] Byungho Beak, K Larry Head, and Yiheng Feng. Adaptive coordination based on connected vehicle technology. *Transportation Research Record*, 2619(1):1–12, 2017.

[18] Debby Bezzina and James Sayer. Safety pilot model deployment: Test conductor team report. *Report No. DOT HS*, 812(171):18, 2014.

[19] Manoj Bhat, Jonathan Francis, and Jean Oh. Trajformer: Trajectory prediction with local self-attentive contexts for autonomous driving. *arXiv preprint arXiv:2011.14910*, 2020.

[20] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.

[21] Sean Campbell, Niall O'Mahony, Lenka Krpalcova, Daniel Riordan, Joseph Walsh, Aidan Murphy, and Conor Ryan. Sensor technology in autonomous vehicles: A review. In *2018 29th Irish Signals and Systems Conference (ISSC)*, pages 1–4. IEEE, 2018.

[22] Anna Carreras, Xavier Daura, Jacqueline Erhart, and Stefan Ruehrup. Eu-tp1488. 2018.

[23] Xiqun Chen, Lingxiao Zhou, and Li Li. Bayesian network for red-light-running prediction at signalized intersections. *Journal of Intelligent Transportation Systems*, 23(2):120–132, 2019.

[24] Nachiket Deo and Mohan M Trivedi. Convolutional social pooling for vehicle trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1468–1476, 2018.

[25] Gilberto Díaz-García, Luis Burbano, Nicanor Quijano, and Luis Felipe Giraldo. Distributed mpc and potential game controller for consensus in multiple differential-drive robots. In *2019 IEEE 4th Colombian Conference on Automatic Control (CCAC)*, pages 1–6. IEEE, 2019.

[26] Wenchao Ding and Shaojie Shen. Online vehicle trajectory prediction using policy anticipation network and optimization-based context reasoning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9610–9616. IEEE, 2019.

[27] Shuo Feng, Haowei Sun, Yi Zhang, Jianfeng Zheng, Henry X Liu, and Li Li. Tube-based discrete controller design for vehicle platoons subject to disturbances and saturation constraints. *IEEE Transactions on Control Systems Technology*, 28(3):1066–1073, 2019.

[28] Shuo Feng, Xintao Yan, Haowei Sun, Yiheng Feng, and Henry X Liu. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature communications*, 12(1):1–14, 2021.

[29] Shuo Feng, Yi Zhang, Shengbo Eben Li, Zhong Cao, Henry X Liu, and Li Li. String stability for vehicular platoon control: Definitions and analysis methods. *Annual Reviews in Control*, 47:81–97, 2019.

[30] Yiheng Feng. *Intelligent traffic control in a connected vehicle environment*. PhD thesis, The University of Arizona, 2015.

[31] Yiheng Feng, K Larry Head, Shayan Khoshmagham, and Mehdi Zamanipour. A real-time adaptive signal control in a connected vehicle environment. *Transportation Research Part C: Emerging Technologies*, 55:460–473, 2015.

[32] Yiheng Feng, Chunhui Yu, and Henry X Liu. Spatiotemporal intersection control in a connected and automated vehicle environment. *Transportation Research Part C: Emerging Technologies*, 89:364–383, 2018.

[33] Jaime F Fisac, Eli Bronstein, Elis Stefansson, Dorsa Sadigh, S Shankar Sastry, and Anca D Dragan. Hierarchical game-theoretic planning for autonomous vehicles. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9590–9596. IEEE, 2019.

[34] Yanbin Gao, Shifei Liu, Mohamed M Atia, and Aboelmagd Noureldin. Ins/gps/lidar integrated navigation system for urban and indoor environments using hybrid scan matching algorithm. *Sensors*, 15(9):23286–23302, 2015.

[35] Nathan H Gartner, John DC Little, and Henry Gabbay. Optimization of traffic signal settings by mixed-integer linear programming: Part i: The network coordination problem. *Transportation Science*, 9(4):321–343, 1975.

[36] Andreas Geiger, Martin Lauer, Frank Moosmann, Benjamin Ranft, Holger Rapp, Christoph Stiller, and Julius Ziegler. Team annieway's entry to the 2011 grand cooperative driving challenge. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1008–1017, 2012.

[37] Robert S Gibbons. *Game theory for applied economists*. Princeton University Press, 1992.

[38] Noah J Goodall, Brian L Smith, and Byungkyu Park. Traffic signal control with connected vehicles. *Transportation Research Record*, 2381(1):65–72, 2013.

[39] Deepak Gopalakrishna, Vince Garcia, Ali Ragan, Tony English, Shane Zumpf, Rhonda Young, Mohamed M Ahmed, Fred Kitchener, Nayel Urena Serulle, Eva Hsu, et al. Connected vehicle pilot deployment program phase 1, concept of operations (conops), icf/wydot [phase 2 update]. Technical report, United States. Department of Transportation. Intelligent Transportation . . . , 2020.

[40] Jyoti Grover, Manoj Singh Gaur, Vijay Laxmi, and Nitesh Kumar Prajapati. A sybil attack detection approach using neighboring vehicles in vanet. In *Proceedings of the 4th international conference on Security of information and networks*, pages 151–158, 2011.

[41] Yi Guo, Jiaqi Ma, Chenfeng Xiong, Xiaopeng Li, Fang Zhou, and Wei Hao. Joint optimization of vehicle trajectories and intersection controllers with connected automated vehicles: Combined dynamic programming and shooting heuristic approach. *Transportation research part C: emerging technologies*, 98:54–72, 2019.

[42] Mordechai Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive computing*, 7(4):12–18, 2008.

[43] Ali Hamieh, Jalel Ben-Othman, and Lynda Mokdad. Detection of radio interference attacks in vanet. In *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*, pages 1–5. IEEE, 2009.

[44] Xiaozheng He, Henry X Liu, and Xiaobo Liu. Optimal vehicle speed trajectory on a signalized arterial with consideration of queue. *Transportation Research Part C: Emerging Technologies*, 61:106–120, 2015.

[45] Larry Head, Steve Shladover, and Ann Wilkey. Multi-modal intelligent traffic signal system. *University of Arizona*, pages 32–36, 2012.

[46] Jorge Hortelano, Juan Carlos Ruiz, and Pietro Manzoni. Evaluating the usefulness of watchdogs for intrusion detection in vanets. In *2010 IEEE International Conference on Communications Workshops*, pages 1–5. IEEE, 2010.

[47] Gavin Howe, Guogang Xu, Dick Hoover, Devin Elsasser, and Frank Barickman. Commercial connected vehicle test procedure development and test results–forward collision warning. Technical report, 2016.

[48] Shihong Ed Huang, Yiheng Feng, and Henry X Liu. A data-driven method for falsified vehicle trajectory identification by anomaly detection. *Transportation Research Part C: Emerging Technologies*, 128:103196, 2021.

[49] Todd E Humphreys, Brent M Ledvina, Mark L Psiaki, Brady W O'Hanlon, Paul M Kintner, et al. Assessing the spoofing threat: Development of a portable gps civilian spoofer. In *Proceedings of the 21st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2008)*, pages 2314–2325, 2008.

[50] GNSS Inside. Tesla model s and model 3 prove vulnerable to gps spoofing attacks, research from regulus cyber shows, 2019.

[51] Attila Jaeger, Norbert Bißmeyer, Hagen Stübing, and Sorin A Huss. A novel framework for efficient mobility data verification in vehicular ad-hoc networks. *International Journal of Intelligent Transportation Systems Research*, 10(1):11–21, 2012.

[52] Jinyong Jeong, Younggun Cho, Young-Sik Shin, Hyunchul Roh, and Ayoung Kim. Complex urban dataset with multi-level sensors from highly diverse urban environments. *The International Journal of Robotics Research*, 38(6):642–657, 2019.

[53] Ashkan Kalantari and Erik G Larsson. Statistical test for gnss spoofing attack detection by using multiple receivers on a rigid body. *EURASIP Journal on Advances in Signal Processing*, 2020(1):1–16, 2020.

[54] Md Abdus Samad Kamal, Tomohisa Hayakawa, and Jun-ichi Imura. Development and evaluation of an adaptive traffic signal control scheme under a mixed-automated traffic scenario. *IEEE Transactions on Intelligent Transportation Systems*, 21(2):590–602, 2019.

[55] Jonathan Kelly and Gaurav S Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *The International Journal of Robotics Research*, 30(1):56–79, 2011.

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

[57] Arne Kesting, Martin Treiber, and Dirk Helbing. Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1928):4585–4605, 2010.

[58] John Koupal, Mitch Cumberworth, Harvey Michaels, Megan Beardsley, and David Brzezinski. Design and implementation of moves: Epa's new generation mobile source emission model. *Ann Arbor*, 1001(48):105, 2003.

[59] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):712–733, 2020.

[60] K Lee and Huei Peng. Evaluation of automotive forward collision warning and collision avoidance algorithms. *Vehicle System Dynamics*, 43:735–751, 11 2005.

[61] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 336–345, 2017.

[62] Sergey Levine and Vladlen Koltun. Continuous inverse optimal control with locally optimal examples. *arXiv preprint arXiv:1206.4617*, 2012.

[63] Liang Li, Guangquan Lu, Yunpeng Wang, and Daxin Tian. A rear-end collision avoidance system of connected vehicles. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 63–68. IEEE, 2014.

[64] Shengbo Eben Li, Yang Zheng, Keqiang Li, and Jianqiang Wang. An overview of vehicular platoon control under the four-component framework. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 286–291. IEEE, 2015.

[65] Sisi Li, Nan Li, Anouck Girard, and Ilya Kolmanovsky. Decision making in dynamic and interactive environments based on cognitive hierarchy theory, bayesian inference, and predictive control. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2181–2187. IEEE, 2019.

[66] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving. *arXiv preprint arXiv:1907.07792*, 2019.

[67] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. Grip: Graph-based interaction-aware trajectory prediction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3960–3966. IEEE, 2019.

[68] Zhuofei Li, Lily Elefteriadou, and Sanjay Ranka. Signal control optimization for automated vehicles at isolated signalized intersections. *Transportation Research Part C: Emerging Technologies*, 49:1–18, 2014.

[69] Xiao Joyce Liang, S Ilgin Guler, and Vikash V Gayah. A heuristic method to optimize generic signal phasing and timing plans at signalized intersections using connected vehicle technology. *Transportation Research Part C: Emerging Technologies*, 111:156–170, 2020.

[70] Hao Liu, Xiao-Yun Lu, and Steven E Shladover. Traffic signal control by leveraging cooperative adaptive cruise control (cacc) vehicle platooning capabilities. *Transportation research part C: emerging technologies*, 104:390–407, 2019.

[71] Henry X Liu and Wenteng Ma. A virtual vehicle probe model for time-dependent travel time estimation on signalized arterials. *Transportation Research Part C: Emerging Technologies*, 17(1):11–26, 2009.

[72] Mushuang Liu, Ilya Kolmanovsky, H Eric Tseng, Suzhou Huang, Dimitar Filev, and Anouck Girard. Potential game based decision-making frameworks for autonomous driving. *arXiv preprint arXiv:2201.06157*, 2022.

[73] Ning Lu, Nan Cheng, Ning Zhang, Xuemin Shen, and Jon W Mark. Connected vehicles: Solutions and challenges. *IEEE internet of things journal*, 1(4):289–299, 2014.

[74] Ryuga Matsumura, Takeshi Sugawara, and Kazuo Sakiyama. A secure lidar with aes-based side-channel fingerprinting. In *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, pages 479–482. IEEE, 2018.

[75] Xiaoyu Mo, Yang Xing, and Chen Lv. Heterogeneous edge-enhanced graph attention network for multi-agent trajectory prediction. *arXiv preprint arXiv:2106.07161*, 2021.

[76] Risako Morimoto. Estimating the benefits of effectively and proactively maintaining infrastructure with the innovative smart infrastructure sensor system. *Socio-Economic Planning Sciences*, 44(4):247–257, 2010.

[77] Sajjad Mozaffari, Omar Y Al-Jarrah, Mehrdad Dianati, Paul Jennings, and Alexandros Mouzakitis. Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[78] Robert Ighodaro Ogie, Pascal Perez, and Virginia Dignum. Smart infrastructure: an emerging frontier for multidisciplinary research. *Proceedings of the institution of civil engineers-smart infrastructure and construction*, 170(1):8–16, 2017.

[79] Gurobi Optimization. Inc.,"gurobi optimizer reference manual," 2015. 2014.

[80] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.

[81] Seong Hyeon Park, ByeongDo Kim, Chang Mook Kang, Chung Choo Chung, and Jun Won Choi. Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1672–1678. IEEE, 2018.

[82] Pramuditha Perera, Poojan Oza, and Vishal M Patel. One-class classification: A survey. *arXiv preprint arXiv:2101.03064*, 2021.

[83] Jonathan Petit, Bas Stottelaar, Michael Feiri, and Frank Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and lidar. *Black Hat Europe*, 11(2015):995, 2015.

[84] AG PTV. Ptv vissim 7 user manual.(pp. 195). *PTV AG, Karlsruhe, Germany*, 2014.

[85] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[86] Tyler GR Reid, Sarah E Houts, Robert Cammarata, Graham Mills, Siddharth Agarwal, Ankit Vora, and Gaurav Pandey. Localization requirements for autonomous vehicles. *arXiv preprint arXiv:1906.01061*, 2019.

[87] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[88] Erick Schmidt, Nikolaos Gatsis, and David Akopian. A gps spoofing detection and classification correlator-based technique using the lasso. *IEEE Transactions on Aerospace and Electronic Systems*, 56(6):4224–4237, 2020.

[89] Matthias Schreier, Volker Willert, and Jürgen Adamy. An integrated approach to maneuver-based trajectory prediction and criticality assessment in arbitrary road environments. *IEEE Transactions on Intelligent Transportation Systems*, 17(10):2751–2766, 2016.

[90] Jens Schulz, Constantin Hubmann, Julian Löchner, and Darius Burschka. Interaction-aware probabilistic behavior prediction in urban environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3999–4006. IEEE, 2018.

[91] Wilko Schwarting, Alyssa Pierson, Javier Alonso-Mora, Sertac Karaman, and Daniela Rus. Social behavior for autonomous vehicles. *Proceedings of the National Academy of Sciences*, 116(50):24972–24978, 2019.

[92] Anuj Sharma, Darcy M Bullock, and James A Bonneson. Input–output and hybrid techniques for real-time prediction of delay and maximum queue length at signalized intersections. *Transportation Research Record*, 2035(1):69–80, 2007.

[93] Junjie Shen, Jun Yeon Won, Zeyuan Chen, and Qi Alfred Chen. Drift with devil: Security of multi-sensor fusion based localization in high-level autonomous driving under {GPS} spoofing. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 931–948, 2020.

[94] Saeid Soleimaniamiri, Xiaopeng Shaw Li, Handong Yao, Amir Ghiasi, Govind Vadakpat, Pavle Bujanovic, Taylor Lochrane, John Stark, Katherine Blizzard, David Hale, et al. Cooperative automation research: Carma proof-of-concept transportation system management and operations use case 1-basic arterial travel–stop-controlled intersections. Technical report, United States. Federal Highway Administration, 2021.

[95] Andrey Soloviev. Tight coupling of gps, laser scanner, and inertial measurements for navigation in urban environments. In *2008 IEEE/ION Position, Location and Navigation Symposium*, pages 511–525. IEEE, 2008.

[96] Hagen Stübing, Jonas Firl, and Sorin A Huss. A two-stage verification process for car-to-x mobility data based on path prediction and probabilistic maneuver recognition. In *2011 IEEE Vehicular Networking Conference (VNC)*, pages 17–24. IEEE, 2011.

[97] Hagen Stubing, Attila Jaeger, Christoph Schmidt, and Sorin A Huss. Verifying mobility data under privacy considerations in car-to-x communication. In *17th ITS World CongressITS JapanITS AmericaERTICO*, 2010.

[98] Liting Sun, Wei Zhan, and Masayoshi Tomizuka. Probabilistic prediction of interactive driving behavior via hierarchical inverse reinforcement learning. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2111–2117. IEEE, 2018.

[99] Xiaoqiang Sun, F Richard Yu, and Peng Zhang. A survey on cyber-security of connected and autonomous vehicles (cavs). *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[100] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[101] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, et al. Stanley:

The robot that won the darpa grand challenge. *Journal of field Robotics*, 23(9):661–692, 2006.

[102] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. On the requirements for successful gps spoofing attacks. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 75–86, 2011.

[103] Bart Van Arem, Cornelie JG Van Driel, and Ruben Visser. The impact of cooperative adaptive cruise control on traffic-flow characteristics. *IEEE Transactions on intelligent transportation systems*, 7(4):429–436, 2006.

[104] Jessica Van Brummelen, Marie O'Brien, Dominique Gruyer, and Homayoun Najjaran. Autonomous vehicle perception: The technology of today and tomorrow. *Transportation research part C: emerging technologies*, 89:384–406, 2018.

[105] Rens Wouter van der Heijden, Stefan Dietzel, Tim Leinmüller, and Frank Kargl. Survey on misbehavior detection in cooperative intelligent transportation systems. *IEEE Communications Surveys & Tutorials*, 21(1):779–811, 2018.

[106] Xingmin Wang, Shengyin Shen, Debra Bezzina, James R Sayer, Henry X Liu, and Yiheng Feng. Data infrastructure for connected vehicle applications. *Transportation Research Record*, 2674(5):85–96, 2020.

[107] Ziran Wang, Xishun Liao, Chao Wang, David Oswald, Guoyuan Wu, Kanok Boriboonsomsin, Matthew J Barth, Kyungtae Han, BaekGyu Kim, and Prashant Tiwari. Driver behavior modeling using game engine and real vehicle: A learning-based approach. *IEEE Transactions on Intelligent Vehicles*, 5(4):738–749, 2020.

[108] Ziran Wang, Guoyuan Wu, Peng Hao, Kanok Boriboonsomsin, and Matthew Barth. Developing a platoon-wide eco-cooperative adaptive cruise control (cacc) system. In *2017 ieee intelligent vehicles symposium (iv)*, pages 1256–1261. IEEE, 2017.

[109] Xinkai Wu and Henry X Liu. A shockwave profile model for traffic flow on congested urban arterials. *Transportation Research Part B: Methodological*, 45(10):1768–1786, 2011.

[110] Biao Xu, Xuegang Jeff Ban, Yougang Bian, Wan Li, Jianqiang Wang, Shengbo Eben Li, and Keqiang Li. Cooperative method of traffic signal optimization and speed control of connected vehicles at isolated intersections. *IEEE Transactions on Intelligent Transportation Systems*, 20(4):1390–1403, 2018.

[111] DianGe Yang, Kun Jiang, Ding Zhao, ChunLei Yu, Zhong Cao, ShiChao Xie, ZhongYang Xiao, XinYu Jiao, SiJia Wang, and Kai Zhang. Intelligent and connected vehicles: Current status and future perspectives. *Science China Technological Sciences*, 61(10):1446–1471, 2018.

[112] Hao Yang, Hesham Rakha, and Mani Venkat Ala. Eco-cooperative adaptive cruise control at signalized intersections considering queue effects. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1575–1585, 2016.

[113] Tangtao Yang, Yi Zhang, Jiwen Tan, and Tony Z Qiu. Research on forward collision warning system based on connected vehicle v2v communication. In *2019 5th International Conference on Transportation Information and Safety (ICTIS)*, pages 1174–1181. IEEE, 2019.

[114] Zhen Yang, Yiheng Feng, Xun Gong, Ding Zhao, and Jing Sun. Eco-trajectory planning with consideration of queue along congested corridor for hybrid electric vehicles. *Transportation Research Record*, 2673(9):277–286, 2019.

[115] Zhen Yang, Yiheng Feng, and Henry X Liu. A cooperative driving framework for urban arterials in mixed traffic conditions. *Transportation research part C: emerging technologies*, 124:102918, 2021.

[116] Zhen Yang, Rusheng Zhang, and Henry X Liu. A hierarchical behavior prediction framework at signalized intersections. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 515–521. IEEE, 2021.

[117] Chaitanya Yavvari, Zoran Duric, and Duminda Wijesekera. Vehicular dynamics based plausibility checking. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8. IEEE, 2017.

[118] Jun Ying and Yiheng Feng. A full vehicle trajectory planning model for urban traffic control based on imitation learning. *Transportation Research Record (To Appear)*, 2022.

[119] Deng Yongqiang, Wang Dengjiang, Cao Gang, Ma Bing, Guan Xijia, Wang Yajun, Liu Jianchao, Fang Yanming, and Li Juanjuan. Baai-vanjee roadside dataset: Towards the connected automated vehicle highway technologies in challenging environments of china. *arXiv preprint arXiv:2105.14370*, 2021.

[120] Chunhui Yu, Yiheng Feng, Henry X Liu, Wanjing Ma, and Xiaoguang Yang. Integrated optimization of traffic signals and vehicle trajectories at isolated urban intersections. *Transportation Research Part B: Methodological*, 112:89–112, 2018.

[121] Chunhui Yu, Yiheng Feng, Henry X Liu, Wanjing Ma, and Xiaoguang Yang. Corridor level cooperative trajectory optimization with connected and automated vehicles. *Transportation Research Part C: Emerging Technologies*, 105:405–421, 2019.

[122] Hongtao Yu, H Eric Tseng, and Reza Langari. A human-like game theory-based controller for automatic lane changing. *Transportation Research Part C: Emerging Technologies*, 88:140–158, 2018.

[123] Kamran Zaidi, Milos B Milojevic, Veselin Rakocevic, Arumugam Nallanathan, and Muttukrishnan Rajarajan. Host-based intrusion detection for vanets: a statistical approach to rogue node detection. *IEEE transactions on vehicular technology*, 65(8):6703–6714, 2015.

[124] Qingyu Zhang, Reza Langari, H Eric Tseng, Dimitar Filev, Steven Szwabowski, and Serdar Coskun. A game theoretic model predictive controller with aggressiveness estimation for mandatory lane change. *IEEE Transactions on Intelligent Vehicles*, 5(1):75–89, 2019.

[125] Rusheng Zhang, Zhengxia Zou, Shengyin Shen, and Henry X. Liu. Design, implementation, and evaluation of a roadside cooperative perception system. *101st Transportation Research Board (TRB) Annual Meeting 2022*, 2022.

[126] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[127] Zhengxia Zou, Rusheng Zhang, Shengyin Shen, Gaurav Pandey, Punarjay Chakravarty, Armin Parchami, and Henry X. Liu. Real-time full-stack traffic scene perception for autonomous driving with roadside cameras. *2022 IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[128] Alex Zyner, Stewart Worrall, and Eduardo Nebot. Naturalistic driver intention and path prediction using recurrent neural networks. *IEEE transactions on intelligent transportation systems*, 21(4):1584–1594, 2019.