Model-free Learning and Interaction-aware Control for Safe Autonomous Driving

by

Kaiwen Liu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in The University of Michigan
2023

Doctoral Committee:

Professor Anouck R. Girard, Co-Chair
Professor Ilya V. Kolmanovsky, Co-Chair
Associate Professor Necmiye Ozay
Professor Jing Sun

Kaiwen Liu

kwliu@umich.edu

ORCID iD: 0000-0002-9542-8004

To my family.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my advisors, Prof. Anouck Girard and Prof. Ilya Kolmanovsky, for providing me the opportunity to pursue a PhD and for unwaveringly supporting me throughout my academic journey. Working with them has been an absolute privilege and joy. They provided me with invaluable guidance and support at every step of my research. In fact, it was under their tutelage that I first delved into the field of controls when I was still an undergraduate. I still recall the countless hours they spent teaching and mentoring me during the early stage of my journey. Both Prof. Girard and Prof. Kolmanovsky are exceptional mentors and serve as outstanding role models for me. It was their guidance and support that makes me resilient to challenges and stress and makes my PhD journey full of joy and passion, and I cannot express my gratefulness enough to them.

I would like to thank my committee members, Prof. Jing Sun and Prof. Necmiye Ozay for their invaluable support and encouragement. Their expert insight and constructive feedback have been instrumental in enhancing the quality of my research and dissertation writings. I am truly fortunate to have had the opportunity to work with such outstanding scholars.

I would like to express my deep appreciation to my collaborators at Ford, Dr. Eric Tseng, Dr. Dimitar Filev, Dr. Suzhou Huang, and Dr. Malladi Bharaniprabha, for the incredible opportunity to work with them. Their invaluable insights and practical expertise have significantly shaped and enhanced my research. I would also like to thank collaborators at Automotive Research Center, Dr. Denise Rizzo for her invaluable contribution to my research, especially on topics related to ground vehicles and off-road scenarios. I am also grateful to my mentors and colleagues at Motional, Dr. Eric Wolff and Dr. Dmitry Yershov, for their guidance and support during my internship. Working with them over the summer provided me with a deeper understanding of the current state of the art in machine learning and autonomous driving, and it was a truly enriching experience.

I would like to thank financial support from Rackham Predoctoral Fellowship

and Leinweber Space Fellowship, which enable me to concentrate on my dissertation research and enhance my research skills.

I am also fortunate to have had the opportunity to work with numerous talented PhD students and postdocs in our group, and I am grateful for their company and inspiration. In particular, I want to express my sincere appreciation to Dr. Nan Li, who played an invaluable role as a "shadow" advisor especially during my early years as a PhD student. His guidance, support, and friendship have been instrumental in shaping my research and making my PhD journey a meaningful and enjoyable one.

Lastly, I would like to thank my family for their support throughout my journey. Their unconditional love, encouragement, and belief in me have been a constant source of strength and inspiration, and I am deeply grateful for all that they have done. Without them, I would not have been able to accomplish all that I have.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES viii

LIST OF TABLES x

ABSTRACT xi

CHAPTER

I. Introduction 1

1.1 Background and Motivations 1
1.2 Challenges and Research Gaps 3
1.2.1 Interaction-awareness for safe road sharing with other road users 3
1.2.2 Ensuring safety in uncertain or unpredictable situations 6
1.3 Summary of Contributions 7
1.4 Dissertation Overview 10

II. Interaction-aware Control for Autonomous Driving with Applications to Forced Merge 13

2.1 Introduction 13
2.2 Models and Control Strategy Descriptions 16
2.2.1 Control architecture 16
2.2.2 Vehicle dynamics 17
2.2.3 Traffic dynamics 17
2.2.4 Reward function 18
2.2.5 Selecting trajectories as vehicle actions 19
2.2.6 Model predictive control strategy 22
2.3 Game-Theoretic Model for Vehicle Cooperation Behaviors and Explicit Representation Using Imitation Learning 23

2.3.1 Leader-follower game-theoretic model 23
2.3.2 Explicit representation of leader-follower game policy through imitation learning 26
2.4 Decision Making under Cooperation Intention Uncertainty . . 29
2.4.1 Estimation of interacting vehicle's cooperation intention 29
2.4.2 Control strategy for multi-vehicle interactions . . . 31
2.5 Simulation and Validation Results 34
2.5.1 Interacting vehicles driven by leader/follower 35
2.5.2 Interacting vehicles driven by intelligent driver model 37
2.5.3 Interacting vehicles following traffic data 39
2.6 Summary 44

III. Model-free Learning to Avoid Constraint Violations for Non-safety Critical Systems 46

3.1 Introduction 46
3.2 Problem Formulation 48
3.3 Learning Explicit Reference Governor 50
3.3.1 Enforcing constraints using ERG 51
3.3.2 Learning algorithm for enforcing constraints 53
3.4 Applications and Results 55
3.4.1 Constrained control of a delivery robot 55
3.4.2 Electric vehicle velocity control and battery management 57
3.4.3 Vehicle rollover avoidance 60
3.5 Summary 63

IV. Mode-free Learning to Avoid Constraint Violations for Safety Critical Systems 65

4.1 Introduction 65
4.2 Problem Formulation 68
4.3 Learning Reference Governor 71
4.3.1 Reference Governor 72
4.3.2 Safe learning algorithm 73
4.3.3 Theoretical properties 76
4.4 Ground Vehicle Rollover Avoidance 83
4.4.1 Application to a reduced-order linear model 84
4.4.2 Application to a high-fidelity CarSim model 86
4.5 Fuel Truck Dynamic Model under Sloshing Effects 89
4.5.1 Equivalent trammel pendulum model of liquid sloshing 89
4.5.2 Tank truck dynamic model 92
4.6 Fuel Truck Rollover Avoidance 96

4.6.1 Validation of modeling liquid sloshing using the trammel pendulum . . . 96
4.6.2 Simulation results of tank truck dynamics . . . 97
4.6.3 Applying learning reference governor (LRG) to the tank truck . . . 99
4.6.4 Applying LRG to variable load and speed scenarios 104
4.7 Summary . . . 106

V. Conclusion and Future Work . . . 108

5.1 Conclusion . . . 108
5.2 Future Work . . . 109

APPENDIX . . . 112

BIBLIOGRAPHY . . . 116

LIST OF FIGURES

Figure

1.1 A 1950s illustration of autonomous vehicles . . . 2
1.2 Levels of driving automation defined by SAE International . . . 3
1.3 Autonomous vehicles need to share road with other road users . . . 4
1.4 Overview of this dissertation. . . . 10
2.1 The autonomous vehicle (blue) needs to merge onto the highway before the on-ramp section ends. . . . 14
2.2 Proposed control architecture for forced merge scenarios. . . . 16
2.3 A sampled 5th-order polynomial lane change trajectory. . . . 20
2.4 Sample trajectories for the merging vehicle . . . 22
2.5 Illustration of highway forced merge scenarios for validations of the LFGC when highway vehicles are controlled by leaders/followers in the game. . . . 35
2.6 Results of the proposed LFGC against other drivers following leader/follower policy with different leaders and followers combinations . . . 36
2.7 Illustration of highway forced merge scenarios for validations of the LFGC when highway vehicles are controlled by IDM. . . . 38
2.8 Results of the LFGC against other vehicles controlled by IDM with different target vehicles and desired time headways . . . 40
2.9 Top view of the highway that is used for collecting the US101 traffic data . . . 42
2.10 Smooth vehicle trajectories from the US101 traffic dataset using the Savitzky-Golay filter. . . . 43
2.11 Selection of interacting vehicles . . . 43
2.12 One merge scenario identified from the US101 traffic dataset . . . 44
2.13 An illustration of a successful merge when validating the LFGC against the US Highway 101 dataset . . . 45
3.1 Diagram of a nominal closed-loop system augmented with a learning-based ERG to enforce constraints. . . . 49
3.2 Delivery robot diagram and learning algorithm results applied to the robot . . . 58
3.3 Learning algorithm application to electric vehicle velocity control and battery management . . . 59

3.4 Reference and velocity responses of the vehicle, (a) at the beginning, and (b) at the end, of learning. 60
3.5 Utility truck *CarSim* model. 61
3.6 Learning algorithm application to vehicle rollover avoidance 62
3.7 Vehicle sine-and-dwell test corresponding to different phases of learning 63
3.8 State responses for the sine-and-dwell test corresponding to different phases of learning 64
4.1 Diagram of a nominal closed-loop system augmented with a learning-based reference governor for handling constraints. 68
4.2 Learning algorithm application to vehicle rollover avoidance based on a linear model 86
4.3 Step responses without reference governor and with reference governor before and after learning 86
4.4 Learning algorithm application to vehicle rollover avoidance based on high-fidelity *CarSim* model 87
4.5 Results of sine-and-dwell test at different learning phases 88
4.6 State responses of sine-and-dwell test at different learning phases . . 89
4.7 Diagram of the trammel pendulum model. 90
4.8 Illustration of motion of the trammel pendulum system. 91
4.9 Top view and back view of the tank truck dynamic model. 92
4.10 Illustration of the rotational dynamics of the trammel pendulum system about the roll center. 93
4.11 Validation of fuel tank model compared to FLUENT simulations . . 97
4.12 State responses for step commands in the no load scenario 98
4.13 State responses for step commands in the solid load scenario 99
4.14 State responses for step commands in the liquid load scenario 100
4.15 The LTR responses of the no load scenario (top left), solid load scenario (top right), and the liquid load scenario (bottom). 101
4.16 Learning algorithm application to fuel truck rollover avoidance . . . 102
4.17 (a) Steering angle and (b) LTR responses for the step command test. 103
4.18 (a) Steering angle and (b) LTR responses for the sine-and-dwell test. 103
4.19 The roll angle, roll rate, slip angle, yaw rate, pendulum angle, and pendulum angular velocity responses for the sine-and-dwell test. . . 105
4.20 Estimated $\overline{D}$ values at different speeds and fill ratios 106
4.21 Vehicle speed, steering angle, and LTR responses for the sine-and-dwell test when the truck is decelerating and accelerating. 107

LIST OF TABLES

Table

2.1 Intelligent driver model parameters. 39
2.2 Statistics of validating the LFGC using the US101 traffic dataset . . 42
4.1 Tank truck simulation parameters. 97

ABSTRACT

Autonomous vehicles technologies have greatly advanced in recent years, with promises of various benefits, including improved safety, efficient transportation, increased accessibility, etc. However, ensuring safety is one of the major challenges in bringing them into reality. On one hand, autonomous vehicles may need to share the road with other road users (including human-driven vehicles, cyclists, pedestrians, etc.). On the other hand, the manufacturers have to cope with multiple sources of variability in these vehicles due to part-to-part differences, aging, degradation, or even in-field modifications. This dissertation focuses on addressing these challenges by developing a behavior planner that is able to account for interaction with human drivers and model-free learning-based safety supervisors that are able to adapt to different systems or operating environments.

This dissertation first presents the design of a game-theoretic interaction-aware behavior planner. Inspired by Stackelberg Competition, predictive models for human interactions are developed based on the Leader-Follower Game, and a decision-making framework is proposed that integrates game-theoretic predictions, online estimation of other driver's uncertain interactions and optimal control with explicit safety characterization. The proposed approach is applied to forced merging scenarios, where interaction and negotiation with other drivers are typically required. A comprehensive set of simulation-based case studies and validations on naturalistic driving dataset are presented, where the proposed approach demonstrates a high success rate.

The dissertation then introduces two model-free learning algorithms to design safety supervisors suitable for non-safety critical systems and for safety critical systems. The design of the safety supervisor relies on the reference governor scheme, which is an add-on scheme to enforce pointwise-in-time state and control constraints. In non-safety critical control systems, where the violation of the constraints is not desirable but does not lead to severe consequences, such systems may initially operate with constraint violations and learn over time to avoid them through less aggressive maneuvering. In safety critical cases, in which constraint violation may lead to catastrophic consequences, systems will initially operate conservatively, and then improve their performance as they learn more about constraint boundaries and ma-

neuvers that approach the constraint boundary. The results include developments and demonstrations of novel algorithms and supporting theory for autonomous online learning to operate systems safely and non-conservatively. Several applications are considered including power management of electric vehicles, rollover avoidance of ground vehicles, and tanker truck rollover avoidance under liquid sloshing effects.

CHAPTER I

Introduction

This chapter provides an overview of the dissertation. It begins with the background information, which introduces the recent advancements in autonomous driving technology. Subsequently, the remaining challenges and research gaps that this dissertation focuses on are discussed in Section 1.2. Section 1.3 summarizes the original contributions made by this dissertation, and Section 1.4 offers an overview of each chapter.

1.1 Background and Motivations

Autonomous driving technologies have made significant advancements in recent years, promising a wide range of benefits. One of the potential benefits is improved road safety and the significant reduction of fatal accidents that involve personal injuries. Additionally, these technologies have the potential to greatly improve modern mobility by making driving more accessible to everyone, especially seniors and people with disabilities. Furthermore, they can lead to more efficient land and energy use, making transportation more environmentally friendly. Through optimal routing and speed planning, autonomous driving can reduce traffic congestion and fuel/energy consumption, providing more efficient and economic travel. By making transportation more sustainable and efficient, these technologies have the potential to transform the way we travel, making it safer, more accessible, and more environmentally-friendly.

The concept of autonomous vehicles or "driveless" cars has been brought up since 1927 [1], though it was a pure science fiction for decades, as shown in Figure 1.1. In 1958, Chrysler released the first car in the world that is equiped with cruise control [2]. Researchers and car manufacturers soon dedicated themselves to developing such advanced technologies. In 1995, VaMP, developed by researchers in Bundeswehr University of Munich and Mercedes-Benz, was one of the first fully autonomous vehicles

that are able to drive almost autonomously for 2000 (km) [3]. Since the initiation of the DARPA Grand Challenges in 2004 in the United States [4], autonomous driving has undergone rapid developments over the past 20 years. Notable examples include the start of Google self-driving car project in 2009 [5], the release of Tesla Autopilot in 2015 [6], etc.

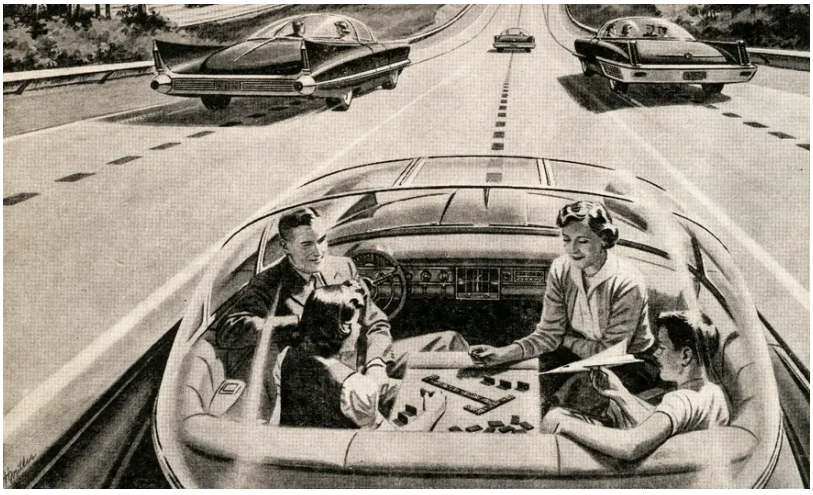

Figure 1.1: A 1950s illustration of autonomous vehicles [7]

Autonomous vehicles are not just a vision for the future, they are already present on the roads today, although not yet fully autonomous. According to SAE International [9] (and as adopted by the U.S. Department of Transportation), there are six levels of automation for autonomous vehicles, and many car manufacturers have already released vehicles with level-2 autonomy features. Examples of these features include Tesla's Autopilot, Ford's Blue Cruise, and GM's Super Cruise. In 2023, Mercedes released level-3 autonomous vehicles, which require the driver to be attentive enough to promptly take control if necessary but enable the vehicle to handle "all aspects of the driving" when engaged [10]. This is a significant step up from level-2 autonomy and demonstrates the progress that has already been made in autonomous driving technology.

In other words, autonomous vehicles are not just a concept for the future, but a reality that is already present on our roads. Many of the vehicles currently driving on the road are equipped with advanced autonomous technologies, and the trend is expected to continue to grow in the coming years.

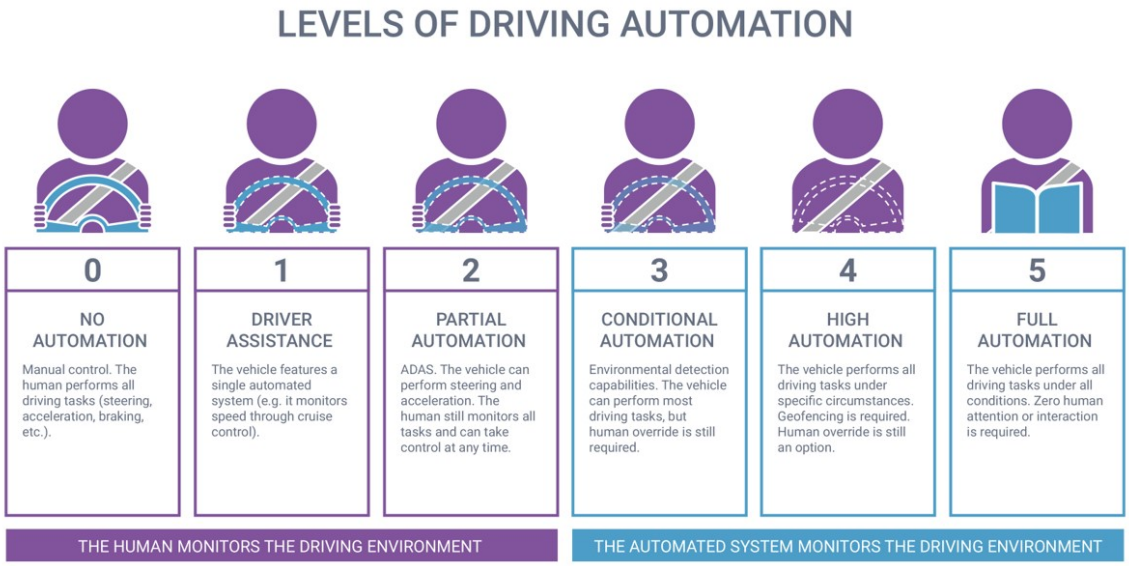

Figure 1.2: Levels of driving automation defined by SAE International [8]

1.2 Challenges and Research Gaps

While autonomous vehicles have the potential to bring many benefits, safety is always the top priority. Developing and deploying autonomous vehicles requires ensuring the safety of both the vehicles and their passengers. This is a major concern that has to be addressed by combined advances in perception, decision-making, and control.

In particular, ensuring the safety of autonomous vehicles requires significant progress in decision-making and control. These fields are essential for designing autonomous systems that can operate safely and efficiently in complex environments. As a part of my dissertation research, I am focusing on advancing these fields and tackling the challenges associated with them.

1.2.1 Interaction-awareness for safe road sharing with other road users

While level-2 and level-3 autonomous vehicles are already available in the market, the penetration of autonomous vehicles is still expected to be relatively low in the coming years. According to [12], only 12 percent of new passenger cars sold by 2030 will have level-3 or higher autonomous technologies, and this number is expected to increase to 37 percent by 2035. This means that vehicles with advanced autonomous

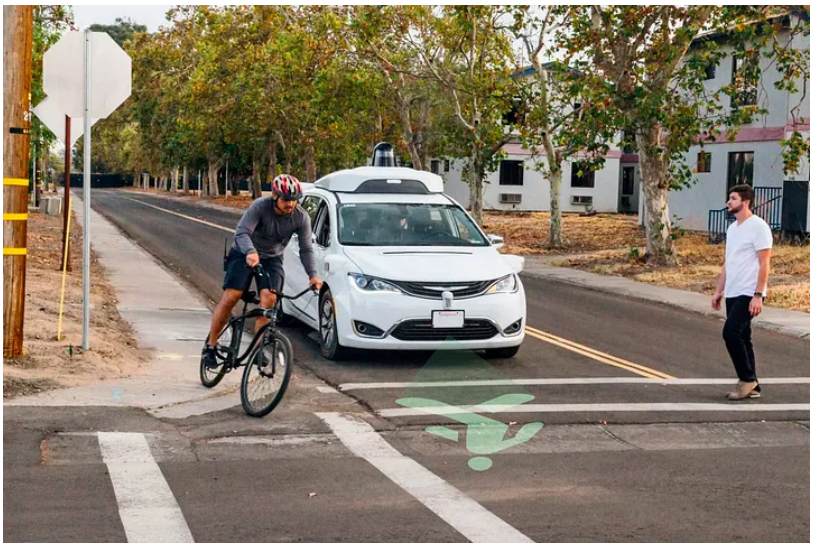

Figure 1.3: Autonomous vehicles need to share road with other road users [11].

driving technologies will need to share the road with vehicles with lower autonomy, which are mainly driven by humans. Even if all vehicles are fully autonomous and centrally coordinated, other traffic participants (such as pedestrians and cyclists) will likely not be. This presents a significant challenge to autonomous vehicle manufacturers. Autonomous vehicles need to ensure safe and efficient operations in a mixed traffic environment, where there is no direct coordination with other traffic participants, such as cases shown in Figure 1.3.

Forced merging scenarios present a particularly intense interaction between vehicles, where the merging vehicle needs to make a mandatory lane change while negotiating with other vehicles already in the target lane. Interactions with other drivers throughout the merging process is essential, as the merging ego vehicle may need to slow down and yield to other vehicles or force them to slow down and yield to it, especially towards the end of the current lane.

However, interacting with human drivers is inherently difficult. Beginner drivers may struggle during such scenarios, and learning stickers are available to assist them. Identifying and tracking other drivers' intentions over time, as well as understanding their behavior, is necessary throughout the interaction process. Autonomous vehicles developers need to account for the fact that human drivers may not follow fixed rules and exhibit varying levels of aggressiveness, distraction, and generosity.

There exists an extensive literature on modeling human driver interactions and autonomous vehicle decision-making during lane change or merging. To handle interaction uncertainties (e.g., due to varied cooperation intentions of other vehicles), the Partially Observable Markov Decision Process (POMDP) framework has been

exploited, where the uncertainties are modeled as latent variables and estimated online based on observed trajectories [13, 14, 15, 16, 17, 18, 19]. However, solving a POMDP problem with a large state and/or action space is computationally very demanding [20]. Consequently, conventional POMDP-based approaches typically only consider the interaction of the ego vehicle with one interacting vehicle at a time to minimize the state space dimension. However, in reality a merging scenario can involve simultaneous interactions with multiple vehicles. Some researchers use specific designed intention models, and online inferring other vehicle intentions [21, 22], but these methods generally do not have a comprehensive set of validations.

Reinforcement Learning (RL) is another popular approach to developing control policies [23, 24, 25, 26], especially for lane change or merge scenarios [27, 28, 29, 30]. An RL-based policy can account for the vehicle interactions in certain scenarios through training in an environment capable of representing such interactions [31, 32, 33]. In order to obtain RL driving policies that behave like human drivers, several researchers chose to use inverse RL to estimate the human's reward function for driving [34, 35, 36, 37]. To be able to model different human driver styles and/or interaction intentions, [38] incorporates cooperativeness into the intelligent driver model and [39] formulates different reward functions for different drivers and performs RL based on the models. Although RL-based approaches are appealing in terms of their potential to handle complex traffic scenarios with multi-vehicle interactions, potential drawbacks of these approaches that hinder their practical application include their lack of interpretability and explicit safety guarantees, because safety is typically only promoted through certain terms in the reward function rather than enforced through hard constraints.

To achieve more interpretable control, it has been proposed to explicitly incorporate a prediction model for vehicle interactions in the control algorithm. For instance, several researchers have leveraged deep learning methods (in particular, recurrent neural networks [40] and graphical neural networks [41]) to predict human drivers' behaviors. Multi-modal trajectory prediction methods [42, 43, 44, 45] are able to predict a distribution of possible other vehicles' future trajectories. However, such prediction models are mostly designed for open-loop predictions (i.e., they do not respond to ego vehicle's actions), and utilizing these prediction models may not account for driver's interactions to certain ego's actions within the planning horizon. Reference [46] uses a Social Generative Adversarial Network (Social GAN) to generate predictions of other vehicles' future trajectories in response to ego vehicle's actions. However, the Social GAN does not account for variations of drivers' styles and inten-

tions and needs to be trained with sufficient traffic data [47]. For the latter, it has been reported that multi-vehicle interaction scenarios in released traffic datasets are insufficient [48]. Game-theoretic methods have also been investigated for modeling vehicle interactions in lane change or merge scenarios [17, 49, 50, 51, 52, 53]. It is possible to account for varied driving styles and/or intentions with these game-theoretic methods, for instance, through modeling and online estimation of drivers' cognitive levels [50] or aggressiveness [54, 55].

1.2.2 Ensuring safety in uncertain or unpredictable situations

Ensuring the safety of autonomous vehicles involves more than just avoiding collisions with other vehicles and preventing personal injuries. It also involves operating within critical limits to prevent damage to various systems. These limits can include thermal limits to prevent overheating of the battery or other thermal systems, power limits to prevent damage to the powertrain, traction limits to avoid slipping, rollover limits to prevent rollover accidents, and more. These limits are essential for the safe operation of autonomous vehicles and must be taken into account in their development and deployment.

As a result, these critical limits require special attention during the design process, particularly in the development of the control system. However, the operation of autonomous vehicle systems can be influenced by various factors such as manufacturing variability, aging, and degradation, which can operate under uncertain environmental conditions. Additionally, in-field modifications of some systems may be entirely unpredictable during design time. Moreover, the boundary of critical limits that need to be followed may be uncertain during design time and can change based on the operating environment and system condition. Maneuvers that can lead to violations of critical limits may be unknown beforehand. These challenges significantly affect the ability of autonomous vehicle developers to ensure safety without violating critical limits during the design phase.

To address the uncertainties that arise in operating environments and uncertain limit boundaries, a common approach is to operate autonomous systems conservatively to avoid constraint violations in the worst-case "tolerance stack-up" scenario. While such conservative operation ensures safety, it can significantly limit vehicle performance and mobility. Therefore, it is essential to develop novel approaches that can balance the tradeoff between robustness and performance while ensuring the safety of autonomous systems.

A novel, emerging approach to address such constraints is to exploit learning (or

adaptation) capabilities of future autonomous vehicles and systems. Reinforcement learning (RL) [56] has been pursued in the literature to directly learn a controller. While RL has been investigated for control of systems with input constraints [57, 58], traditional RL algorithms do not address state/output constraints other than through penalty functions, and they do not provide explicit guarantees on eventual constraint enforcement. Although safe RL techniques have recently been developed to handle state/output constraints, they are typically model-based [59, 60, 61] or require a robust controller for the nominal dynamics of the system [62, 63], which require significant amount of knowledge about the system.

Model predictive control (MPC) is a common approach to enforce constraints [64, 65, 66]. For systems which do not have an accurate model, learning-based model predictive control (LMPC) has been proposed, which integrates model-based MPC with learning [67, 68, 69, 70]. Safety guarantees for LMPC have been developed in [71], however, a nominal linear model and a bound on deviations of the system dynamics is required. LMPC algorithms frequently learn/estimate a model that can represent the system dynamics, and then use this learned/estimated model to compute control signals [72]. Gaussian processes have been employed for LMPC in [73], [74], however, a theoretical guarantee of safety is missing from these approaches. Adaptive MPC approaches exist which can handle model uncertainty through online parameter estimation [75], [76]. However, such approaches which assume that the model is known except for the parameters require more knowledge of the system compared to the proposed approach.

In addition to LMPC, adaptive and learning control barrier function methods have been proposed to handle systems with parametric uncertainty [77], [78], but they rely on a known nominal system model and only ensure safety for a certain class of model uncertainty. Iterative learning control (ILC) is another technique that learns from prior experience to improve the controller performance [79]. However, the focus of ILC is typically on improving the tracking performance over a repeated operation [80]. Extensions of ILC that do not require repetitions such as in [81] still require that trajectories executed by the system are related by a time-scale transformation.

1.3 Summary of Contributions

My dissertation research addresses the challenges mentioned in Section 1.2 through the development of a novel planning/control framework that accounts for human-driven vehicles' reactions in different traffic scenarios when interacting with autonomo-

us vehicles. At the same time, in uncertain or unknown situations, a novel and emerging approach, which relies on the integration of prediction and learning/adaptation, is developed to ensure safety while not sacrificing performance. In particular, these approaches can be utilized to design new decision-making systems for autonomous vehicles where safety is explicitly accounted for either during interactions with humans or operating under unknown circumstances. Methodologically, to account for safety during interactions with human drivers, models representing interactions are developed based on the applications of the game theory and are used to predict human reactions. The decision-making process during the interaction is formulated using the model and optimal control and solved in real-time by modern techniques involving machine learning. In circumstances where the autonomous vehicle is operating in uncertain environments or conditions, learning algorithms are designed based on control theory and set theory to gradually learn to operate the system safely and non-conservatively by exploring the maximum response due to different commands.

A more detailed summary of the original contributions of my dissertation research can be summarized as follows:

1. To account for the interaction and cooperation of humans drivers during interactions, predictive models of human reactions in interactive environments are developed based on game theory. A decision-making framework is developed based on such predictive models, where the interaction uncertainties are estimated online based on observed human reactions, and safety requirements are explicitly accounted for by setting a probabilistic safety bound.

2. The game-theoretic predictive models and the associated decision-making framework is integrated with machine learning to expedite the online computations and to achieve real-time implementations. The decision framework is evaluated and verified through a comprehensive set of simulation-based case studies that include cases where other vehicles are controlled by various types of driver models and cases where their motion follows actual vehicle trajectories presented in publicly available datasets (e.g., NGSIM US101 Highway Dataset [82]). The proposed decision-making framework demonstrates a high success rate (in terms of safely achieving the goal) in these verifications and exhibits a practical potential to overcome the challenge for autonomous vehicles to interact with humans safely and effectively.

3. To ensure the safety of autonomous systems when operating in unknown circumstances and to address the performance-robustness trade-off mentioned above,

two learning schemes for systems to autonomously explore their limits are developed. These two learning schemes are suitable for different situations. One scheme starts with an aggressive design and gradually learns to avoid safety violations. The other scheme will initially operate the system conservatively and gradually improve its performance as it gains more information about the system and the environment.

4. The autonomously learning algorithms, which ensure the systems to operate safely and non-conservatively after learning, has been theoretically analyzed. The effectiveness of the learning algorithms has been verified based on several applications including power management of electric vehicles, ground vehicle rollover avoidance, and tanker truck rollover avoidance under liquid sloshing effects.

Most of the contribution outlined above have been documented and published in peer-reviewed journals or conference proceedings, including [83, 84, 85, 86, 87, 88].

Some additional contributions made during my PhD but not covered in this dissertation are as follows:

1. A C++ solver/toolbox has been developed for solving stochastic control problems in Partially Observable Markov Decision Processes (POMDPs) based on the framework proposed in [16]. The aim of this toolbox is to enable fast and real-time capabilities of the proposed algorithm. The toolbox includes the game-theoretic decision-making systems proposed in [83] as well as several other driver models [1].

2. The Leader Follower Game proposed in Chapter II can also be utilized for simulations of human reactions during complex traffic environments. Such simulations can be used to evaluate and verify different control strategies. A simulation and testing environment is established based on the Simulation of Urban Mobility (SUMO) [89], and its implementation is documented in [90].

3. The learning-based approaches proposed in Chapters III and IV have potential for broader applications to other autonomous systems beyond ground vehicles. The approaches have been successfully demonstrated in aerospace applications of spacecraft rendezvous and proximity maneuvering, as described in [91] and

[1] The toolbox is available on github and can be found at `https://github.com/kevinyt001/MPC_CPOMDP`.

[92]. These applications are safety-critical and require precise control and constraint satisfaction. The use of learning-based approaches can enable more efficient and adaptable control of these systems while still maintaining safety and constraint satisfaction.

1.4 Dissertation Overview

This dissertation is motivated by the need to create effective methods for safe and non-conservative operation of automated vehicles in mixed traffic with human participants. Our focus is on ensuring that autonomous vehicles can interact seamlessly with other road users and adapt to diverse driving conditions. With this goal in mind, we will develop advanced techniques that enable autonomous vehicles to navigate through complex traffic scenarios, make real-time decisions, and respond to unexpected events.

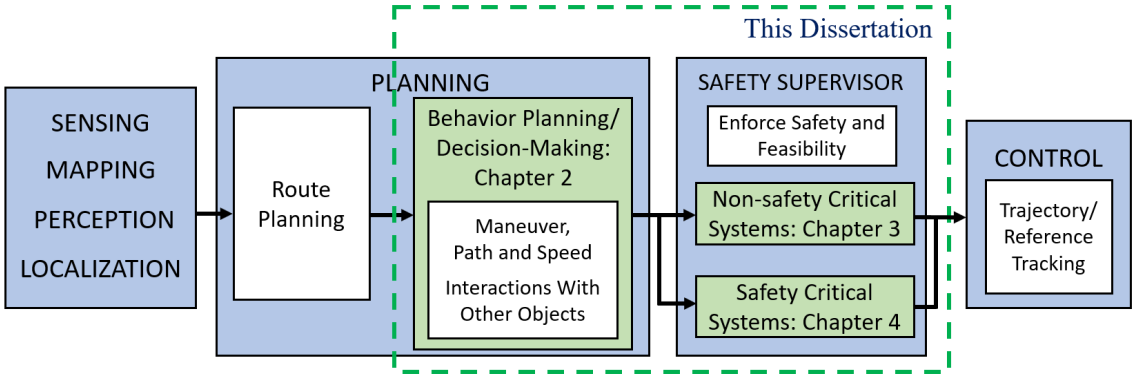

Figure 1.4: Overview of this dissertation.

The autonomous driving technology stack comprises several components [93, 94], as illustrated in Figure 1.4. This dissertation focuses on the behavior planning module and the safety supervisor module, which are critical for ensuring safe and effective operation of autonomous vehicles. Specifically, this dissertation will delve into the design, implementation, and evaluation of the proposed solutions for these modules. By advancing the state of the art in behavior planning and safety supervision, we aim to contribute to making autonomous driving a practical and trustworthy solution for the transportation industry.

Chapter II will mainly focus on the behavior planning module of the autonomous vehicles, where we present the design of interaction-aware autonomous vehicle planning and control strategy. The interaction between autonomous vehicle and other

vehicles is represented using models and concepts from game theory [95]. More specifically, we leverage the Stackelberg Competition models and make appropriate modifications to accommodate the differences between the economic markets and the actual driving situations. This chapter then considers a specific application to forced merge scenarios, which are scenarios with heavy interactions among vehicles requiring an automated vehicle to nudge its way into traffic. In this chapter, a novel game-theoretic controller, called the Leader-Follower Game Controller (LFGC), is introduced, where the interactions between the autonomous ego vehicle and other vehicles with a priori uncertain driving intentions is modeled as a partially observable leader-follower game. The LFGC estimates the other vehicles' intentions online based on observed trajectories, and then predicts their future trajectories and plans the ego vehicle's own trajectory using Model Predictive Control (MPC) to simultaneously achieve probabilistically guaranteed safety and merging objectives. To verify the performance of LFGC, we test it in simulations and with the naturalistic driving dataset provided by NGSIM [82], where the LFGC demonstrates a high success rate of 97.5% in merging.

Chapter III presents a novel approach to designing a safety supervisor that can adapt to different operating conditions or environments, with an emphasize on non-safety critical systems. This chapter relies on an emerging approach to address the above performance-robustness tradeoff, which utilizes the integration of prediction and learning/adaptation. Specifically, we propose a model-free learning algorithm that modifies the parameters of an explicit reference governor (ERG) scheme over time to avoid violations of pre-specified constraints. The ERG modifies setpoint commands to a nominal closed-loop system, and our learning algorithm adjusts the ERG parameters based on observed constraint violations during a learning phase. After the learning phase is completed, the modified ERG ensures that constraint violations are eliminated while maintaining satisfactory performance. Theoretical properties of the algorithm are analyzed, and several examples are presented to demonstrate its effectiveness. This approach is particularly useful in non-safety critical scenarios where occasional constraint violations are undesirable but do not result in catastrophic consequences.

Chapter IV focuses on designing a safety supervisor for safety-critical systems, where even occasional constraint violations may result in catastrophic consequences. In such systems, learning must be performed in a way that ensures constraint satisfaction at all times. Instead of relying on ERG as in Chapter III, this chapter integrate learning with traditional reference governor that guard the nominal sys-

tem against violation of pre-specified constraints by modifying set-point commands. More specifically, a model-free learning algorithm is developed to gradually improve the performance of the reference governor in terms of response speed while ensuring constraint satisfaction for all time. After introducing the learning algorithm and outlining its theoretical properties, this chapter investigates its applications to ground vehicle rollover avoidance and fuel truck (tank truck) rollover avoidance under sloshing effects. Through simulations based on *CarSim* (a high-fidelity vehicle dynamics simulation software) and a fuel truck model that accounts for liquid fuel sloshing effects, we show that the proposed approach can effectively protect ground vehicles and fuel trucks from rollover accidents under various operating conditions.

Finally, Chapter V provides the conclusion of this dissertation and outlines future research directions. The chapter summarizes the contributions of each chapter and discusses their significance in the context of autonomous driving. Moreover, this chapter goes beyond the conclusion and provides an extensive discussion of potential areas for future research.

CHAPTER II

Interaction-aware Control for Autonomous Driving with Applications to Forced Merge

In this chapter, we present and describe an innovative interaction-aware control strategy. In particular, we propose a novel game-theoretic controller, called the Leader-Follower Game Controller (LFGC), in which the interactions between the autonomous ego vehicle and other vehicles with a priori uncertain driving intentions is modeled as a partially observable leader-follower game. The LFGC estimates the other vehicles' intentions online based on observed trajectories, and then predicts their future trajectories and plans the ego vehicle's own trajectory using Model Predictive Control (MPC) to simultaneously achieve probabilistically guaranteed safety and driving objectives. A particular application of the proposed control strategy is to forced merge scenarios, which are typically interaction-intense, is considered and presented in this chapter.

2.1 Introduction

Advances in autonomous vehicle technologies are projected to reduce vehicle crashes and fatalities, improve mobility especially for elderly and disabled people, reduce fuel/energy consumption and emissions, and to promote more efficient land uses [96, 97, 98]. Despite these benefits, there are still many challenges that need to be addressed to deliver a highly (level 4 or level 5) autonomous vehicle [9]. One challenging scenario for both human drivers and autonomous vehicles is highway forced merge, where the merging vehicle needs to choose a proper gap in the highway traffic and potentially force the upstream traffic to slow down so that it can safely merge into that gap. Forced merge typically occurs in mandatory merge scenarios where the current lane is ending, such as at highway on-ramps. When the traffic is dense,

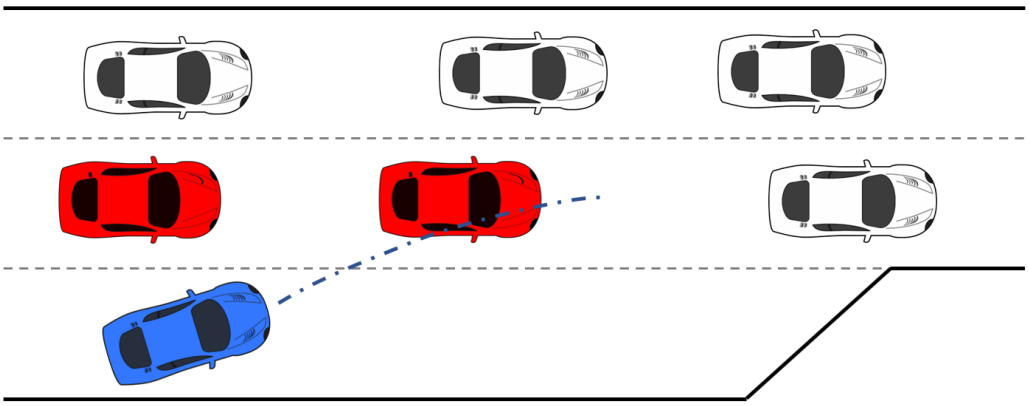

Figure 2.1: The autonomous vehicle (blue) needs to merge onto the highway before the on-ramp section ends. In dense traffic, there may not be a sufficient gap for the autonomous vehicle to merge into. In this case, the autonomous vehicle needs to force the other vehicles to cooperate and let it cut in. However, interacting vehicles that are aware of the autonomous vehicle's merging attempt (red) may choose to *proceed* or *yield* depending on their intentions.

interactions and/or cooperation between the merging vehicle and vehicles driving in the target lane are often needed. In particular, a vehicle in the target lane may choose to ignore the merging vehicle (i.e., *proceed*) and consequently the merging vehicle can only merge behind. Alternatively, the vehicle in the target lane may choose to *yield* to the merging vehicle (i.e., let the merging vehicle merge in front of it). In order to successfully merge into a busy traffic, an autonomous vehicle controller needs to appropriately respond to the intentions to proceed or yield of other vehicles. An overly conservative controller may yield to all other vehicles (including those that intend to yield to the autonomous ego vehicle) and eventually fail to merge, while an overly aggressive controller may have conflicts with the vehicles that intend to proceed and lead to vehicle crashes. Meanwhile, the decision whether to proceed or to yield to another vehicle depends not only on the traffic situation (e.g., the relative position and velocity between the two vehicles) but also on its driver's general driving style, personality, mood, etc. For instance, in a similar situation, an aggressive driver may be inclined to proceed while a cautious/conservative driver may tend to yield. This poses a significant challenge to autonomous vehicle planning and control.

In this chapter, we propose a novel high-level control algorithm, called the Leader-Follower Game Controller (LFGC), for autonomous vehicle planning and control in forced merge scenarios. In the LFGC, drivers' interaction intentions (to *proceed* or *yield*) and their resulting vehicle behaviors are represented by an explicit game-theoretic model with multiple concurrent leader-follower pairs, called a leader-follower

game [99]. To account for interaction uncertainties, the pairwise leader-follower relationships among the vehicles are assumed to be *a priori* uncertainty and modeled as latent variables. The LFGC estimates the leader-follower relationships online based on observed trajectories and makes optimal decisions for the autonomous ego vehicle using a Model Predictive Control (MPC)-based strategy. The proposed approach thus adapts to the inferred leader-follower relationship estimates to simultaneously achieve probabilistically guaranteed safety and the merging objectives. The LFGC presented in this chapter possesses superior properties in several aspects: 1) Instead of relying on discretization of the state space (and the POMDP framework) [100, 101, 102], the LFGC is designed assuming a continuous state space, which results in smoother trajectories for lower-level controllers to track and which alleviates the computational difficulty associated with discrete spaces. 2) Unlike using a small number of actions (or motion primitives) to represent vehicle behavior [103, 104, 105], the LFGC predicts and plans vehicle motion using two much larger sets of trajectories (162 trajectories for the merging ego vehicle and 81 trajectories for each of the highway interacting vehicles), which leads to finer-resolution controls and the potential for higher performance. 3) The LFGC is validated based on a comprehensive set of simulation-based test cases including cases where other vehicles are controlled by various types of driver models and cases where their motion follows real traffic data.

The contributions and novelties of the proposed LFGC and this chapter are as follows:

1. The LFGC uses a game-theoretic model for vehicle trajectory prediction while accounting for interactions and cooperation and while leading to interpretable control solutions (because the control solutions are based on model predictive control with an interpretable game-theoretic prediction model).

2. The LFGC handles interaction uncertainties due to varied cooperation intentions of other vehicles by modeling these uncertainties as latent variables and estimating them online based on observed trajectories and Bayesian inference.

3. The LFGC represents vehicle safety requirements (e.g., collision avoidance) as constraints and pursues optimization subject to satisfying an explicit probabilistic safety characterization (i.e., a user-specified probability bound of safety) in the presence of interaction uncertainties.

4. The LFGC is designed in a continuous state space setting, which can avoid space discretization associated with POMDP-based approaches (such as pursued in

[83]) or other discrete state space-based approaches. This allows LFGC to treat higher-dimension problems and to handle more complex scenarios that involve interactions with multiple vehicles.

5. The LFGC is validated based on a comprehensive set of simulation-based case studies that include cases where other vehicles are controlled by various types of driver models and cases where their motion follows actual vehicle trajectories in the NGSIM US Highway 101 dataset [82]. For the latter, the LFGC demonstrates a high success rate (in terms of safely completing merges) of 97.5%.

2.2 Models and Control Strategy Descriptions

In this section, we introduce our proposed control architecture, models to represent the vehicle and traffic dynamics and the MPC-based strategy for the ego vehicle's trajectory planning.

2.2.1 Control architecture

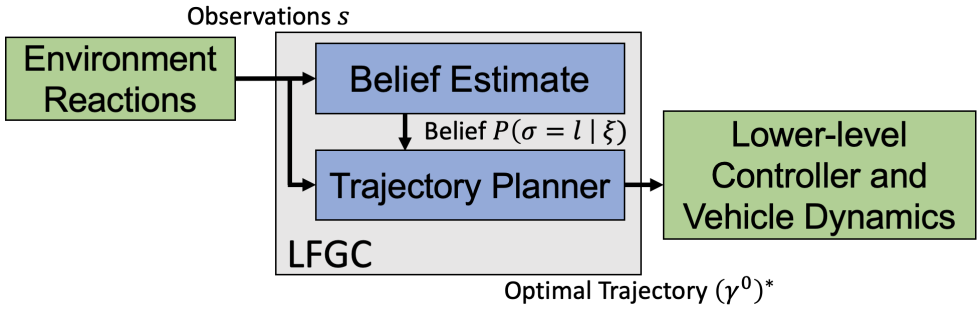

Figure 2.2: Proposed control architecture for forced merge scenarios.

Figure 2.2 shows the proposed control architecture considered in this chapter that focuses on the design of Leader-Follower Game Controller (LFGC), which consists of a belief estimator and a trajectory planner. LFGC takes observation of the environment as an input, estimates other drivers' intentions based on the leader-follower game, and utilizes game-theoretic prediction and model predictive control to generate a planned trajectory. As is common in the literature [106, 107, 108], we assume that the vehicle has an existing lower-level controller that can execute the planned trajectory respecting the vehicle dynamics.

2.2.2 Vehicle dynamics

We use the kinematic bicycle model [109] to represent the motion of each vehicle. The kinematic bicycle model is defined by the following set of continuous-time equations,

$$\begin{aligned} \dot{x} &= v\cos(\psi+\beta), \\ \dot{y} &= v\sin(\psi+\beta), \\ \dot{v} &= a, \\ \dot{\psi} &= \frac{v}{l_r}\sin(\beta), \\ \beta &= \tan^{-1}\left(\frac{l_r}{l_r+l_f}\tan\delta_f\right), \end{aligned} \tag{2.1}$$

where we have assumed only front-wheel steering δ_f and no rear-wheel steering (i.e., $\delta_r = 0$); x and y are the longitudinal and lateral positions of the vehicle; v is the speed of the vehicle; ψ and β are the yaw angle and the slip angle of the vehicle; l_f and l_r represent the distances from the CG of the vehicle to the front wheel and rear wheel axles; a is the acceleration along the direction of speed v. The control inputs are the acceleration and front-wheel steering, $u = [a, \delta_f]^T$.

While vehicle models other than (2.1) could be used, the above kinematic bicycle model (2.1) is suitable for our purpose of trajectory prediction and planning in forced merge scenarios – it can produce sufficiently accurate predictions of vehicle trajectories under given acceleration and front-wheel steering profiles [110] while it is simple and thus computationally efficient. Note also that planned trajectories are passed to the vehicle controller which tracks them while potentially relying on higher fidelity dynamic models of the vehicle.

2.2.3 Traffic dynamics

We consider a traffic scenario involving $n+1$ vehicles, including the ego vehicle, denoted by 0, and n other interacting vehicles k, $k \in \{1, \ldots, n\}$, which correspond to vehicles that are aware of the ego vehicle's merging attempt. Note that vehicles that are not aware of the merging vehicle can still be considered as the interacting vehicle, and their intents can be classified as not yielding to the merging vehicle in the actual application. For the purpose of analyzing and inferring human drivers' intentions, we assume all interacting vehicles are aware of the ego vehicle's merge. Therefore, the traffic state and its dynamics are characterized by the aggregation of all $n+1$ vehicles' states and dynamics. Specifically, we describe the traffic dynamics using the

following discrete-time model,

$$\bar{s}_{t+1} = f(\bar{s}_t, \bar{u}_t), \tag{2.2}$$

where $\bar{s}_t = (s_t^0, s_t^1, s_t^2, \ldots, s_t^n)$ denotes the traffic state at the discrete time instant t, with s_t^0 denoting the ego vehicle's state and s_t^k, $k \in \{1, \ldots, n\}$, denoting the kth interacting vehicle's state; and similarly, $\bar{u}_t = (u_t^0, u_t^1, u_t^2, \ldots, u_t^n)$ denotes the aggregation of all $n+1$ vehicles' control inputs at the time constant t. In particular, each vehicle's state s_t^k, $k \in \{0, 1, \ldots, n\}$, consists of its x and y positions, speed, and yaw angle, i.e., $s_t^k = [x_t^k, y_t^k, v_t^k, \psi_t^k]^T$, and each vehicle's control inputs are $u_t^k = [a_t^k, \delta_{f,t}^k]^T$. Accordingly, the function f in (2.2) that represents the transition of traffic state from $\bar{s}_t$ to $\bar{s}_{t+1}$ as a result of all vehicles' control inputs $\bar{u}_t$ is an aggregation of $(n+1)$-copies of the kinematic bicycle model (2.1) converted to discrete time with a specified sampling period ΔT and using the Euler method.

2.2.4 Reward function

The reward function $R(\bar{s}_t, \bar{u}_t)$ is a mathematical representation of the driving goals of the driver. Here, we start by considering the interactions between the ego vehicle and one other vehicle, i.e., $\bar{s}_t = (s_t^0, s_t^1)$ and $\bar{u}_t = (u_t^0, u_t^1)$. In this case, the traffic state is composed of the states of these two vehicles, and the reward received by the ego vehicle depends on the states and control inputs of both vehicles. Following [83], we consider

$$R\big(\bar{s}_t, u_t^0, u_t^1\big) = w^T r, \tag{2.3}$$

where $r = [r_1, r_2, r_3, r_4, r_5]^T$ and $w \in \mathbb{R}_+^5$ is a vector of weights. The reward terms $r_1, \ldots, r_5$ are defined to represent the following common considerations during driving: 1) safety (r_1, r_2), i.e., not colliding with other vehicles and not getting off the road; 2) liveness (r_3, r_4), i.e., approaching the destination; and 3) perceived safety and comfort (r_5), i.e., maintaining a reasonable separation from other vehicles.

More detailed definitions of $r_1, \ldots, r_5$ are the following,

- r_1 is an indicator for vehicle collisions. A vehicle is represented by a 7 (m) $\times$ 2.5 (m) bounding box. If the ego vehicle's bounding box overlaps with that of another vehicle, then $r_1 = -1$; $r_1 = 0$ otherwise. The weight for r_1 is chosen to be large to prioritize vehicle safety over other considerations. Note that for actual dataset validation, the bounding box size is adjusted according to the actual vehicle's dimension.

- r_2 is an indicator for getting outside road boundaries. The road boundaries represent safety and hard position limit for the autonomous merging task. The ego vehicle is considered outside road boundary if it either enters the target lane too early (before the start of the acceleration lane) or does not accomplish merge when reaching the end of the acceleration lane.

- r_3 and r_4 capture the liveness properties of a vehicle, defined according to

$$r_3 = x_0, \qquad r_4 = -\left|y_0 - y_{\mathrm{r}}\right|, \tag{2.4}$$

  where y_{r} corresponds to the center of target lane. We assign a higher weight to r_4 than r_3 to promote merging whenever appropriate.

- r_5 penalizes the ego vehicle for getting too close to other vehicles. We consider a larger, 11 (m) $\times$ 3 (m) bounding box for each vehicle, and $r_5 = -1$ if this larger bounding box of the ego vehicle overlaps with that of another vehicle; $r_5 = 0$ otherwise. This term is used to encourage the ego vehicle to maintain a reasonable separation distance from other vehicles to improve safety and comfort. Note that for actual dataset validation, the bounding box size is adjusted proportional to the actual vehicle's dimension.

2.2.5 Selecting trajectories as vehicle actions

Instead of considering a discrete set of acceleration a and steering δ_f values as in [83], we consider a sampled set of vehicle motion trajectories over a planning horizon of $T = N\Delta T$ (s) as the action space for each vehicle. Directly using vehicle motion trajectories has been widely used in the literature [111, 112]. Specifically, each trajectory is a time history of vehicle state $s_t = [x_t, y_t, v_t, \psi_t]^T$ starting from the vehicle's current state s_0. Note that the time history of control inputs $u_t = [a_t, \delta_{f,t}]^T$ corresponding to each trajectory can be computed from the vehicle dynamics model (2.1). Compared to representing vehicle motion using discrete acceleration and steering levels as in [83], the method here can lead to smoother trajectories and finer-resolution controls, which will be illustrated in Section 2.5.

For interacting vehicles driving in the target lane, we only consider their longitudinal motion, which corresponds to the assumption that these vehicles do not change lanes. Assuming $\psi = 0$ and $\delta_f = 0$, the kinematic bicycle model (2.1) for these

vehicles reduces to

$$\dot{x} = v, \quad \dot{y} = 0, \quad \dot{v} = a, \quad \dot{\psi} = 0, \quad \beta = 0. \tag{2.5}$$

In this case, a trajectory starting with a given initial condition depends only on the profile of acceleration a over $[0, T]$. In particular, at each sample time instant, we consider 81 acceleration profiles, which translates into 81 trajectories through (2.5), for each interacting vehicle k driving in the target lane, and we treat these trajectories as its admissible actions. Note that we also enforce the speed limits $v_t^k \in [v_{\min}, v_{\max}]$ when we generate these trajectories. We denote each such trajectory as $\gamma_m^k(s_0^k)$, with $m = 1, 2, \ldots, 81$, and the collection of such trajectories as $\Gamma^k(s_0^k) := \{\gamma_m^k(s_0^k)\}_{m=1}^{81}$.

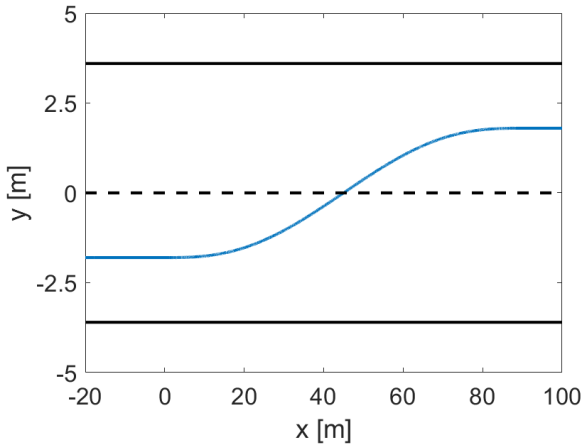

Figure 2.3: A sampled 5th-order polynomial lane change trajectory.

The merging vehicle's maneuvers include both lane keeping and lane change. Trajectories or pieces of trajectories that represent lane keeping are generated using (2.5) in a similar way as above. For a lane change, we use 5th-order polynomials to represent lane change trajectories [113], where a sample lane change trajectory is shown in Figure 2.3. Specifically, a lane change trajectory is produced by the solution to the following boundary value problem: Find the coefficients $a_1, \ldots, a_5$ and $b_1, \ldots, b_5$ such that the 5th-order polynomials

$$\begin{aligned} x(\zeta) &= a_0 + a_1\zeta + a_2\zeta^2 + a_3\zeta^3 + a_4\zeta^4 + a_5\zeta^5, \\ y(\zeta) &= b_0 + b_1\zeta + b_2\zeta^2 + b_3\zeta^3 + b_4\zeta^4 + b_5\zeta^5, \end{aligned} \tag{2.6}$$

satisfy specified initial and terminal conditions $(x_{\text{ini}}, \dot{x}_{\text{ini}}, \ddot{x}_{\text{ini}}, y_{\text{ini}}, \dot{y}_{\text{ini}}, \ddot{y}_{\text{ini}})$ and $(x_{\text{term}}, \dot{x}_{\text{term}}, \ddot{x}_{\text{term}}, y_{\text{term}}, \dot{y}_{\text{term}}, \ddot{y}_{\text{term}})$, where $(x_{\text{ini}}, \dot{x}_{\text{ini}}, \ddot{x}_{\text{ini}}, y_{\text{ini}}, \dot{y}_{\text{ini}}, \ddot{y}_{\text{ini}})$ corresponds to either the vehicle's current state or its state at the start of a lane change, and

$(x_{\text{term}}, \dot{x}_{\text{term}}, \ddot{x}_{\text{term}}, y_{\text{term}}, \dot{y}_{\text{term}}, \ddot{y}_{\text{term}})$ corresponds to the vehicle's state after the completion of a lane change. The variable ζ in (2.6) denotes continuous time. We let $\zeta = 0$ correspond to the current sample time instant and assume that 1) the vehicle can start a lane change at any sample time instant $\zeta = t\,\Delta T$, with $t = 0, \ldots, N-1$, over the planning horizon, and 2) a complete lane change takes a constant time duration of $T_{lc} = 3$ (s) [113]. Then, for the case where at the current sample time instant the vehicle is in the middle of a lane change (i.e., the vehicle started the lane change ΔT_{lc} (s) ago), $(x_{\text{ini}}, \dot{x}_{\text{ini}}, \ddot{x}_{\text{ini}}, y_{\text{ini}}, \dot{y}_{\text{ini}}, \ddot{y}_{\text{ini}})$ corresponds to the vehicle's current state and is satisfied by (2.6) at $\zeta = 0$, while $(x_{\text{term}}, \dot{x}_{\text{term}}, \ddot{x}_{\text{term}}, y_{\text{term}}, \dot{y}_{\text{term}}, \ddot{y}_{\text{term}})$ is satisfied by (2.6) at $\zeta = T_{lc} - \Delta T_{lc}$. For the case where the vehicle starts a lane change at a future sample time instant $t\,\Delta T$, $(x_{\text{ini}}, \dot{x}_{\text{ini}}, \ddot{x}_{\text{ini}}, y_{\text{ini}}, \dot{y}_{\text{ini}}, \ddot{y}_{\text{ini}})$ corresponds to the vehicle's state at the start of the lane change and is satisfied by (2.6) at $\zeta = t\,\Delta T$, while $(x_{\text{term}}, \dot{x}_{\text{term}}, \ddot{x}_{\text{term}}, y_{\text{term}}, \dot{y}_{\text{term}}, \ddot{y}_{\text{term}})$ is satisfied by (2.6) at $\zeta = t\,\Delta T + T_{lc}$. Furthermore, we allow the vehicle, when it is in the middle of a lane change, to abort the lane change at any sample time instant $\zeta = t\,\Delta T$ over the planning horizon. This represents a "change of mind" of the driver when a previously planned lane change becomes no longer feasible/safe. A trajectory for aborting a lane change is generated in a similar way as a lane change trajectory, but the terminal condition $(x_{\text{term}}, \dot{x}_{\text{term}}, \ddot{x}_{\text{term}}, y_{\text{term}}, \dot{y}_{\text{term}}, \ddot{y}_{\text{term}})$ corresponds now to the vehicle's state after its returns to its original lane. Finally, we glue together pieces of trajectories for lane keeping, lane change, and aborting lane change to construct complete trajectories over the planning horizon. This way, we obtain a total of 162 trajectories for the merging vehicle that we treat as admissible actions. Each of these trajectories is characterized by 1) whether and when to start a lane change and 2) whether and when to abort an improper lane change. Figure 2.4 illustrates a sampled set of such trajectories when the vehicle has not started a lane change and those when the vehicle is in the middle of a lane change. We denote each such trajectory as $\gamma_m^0(s_0^0)$, with $m = 1, 2, \ldots, 162$, and the collection of such trajectories as $\Gamma^0(s_0^0) := \{\gamma_m^0(s_0^0)\}_{m=1}^{162}$.

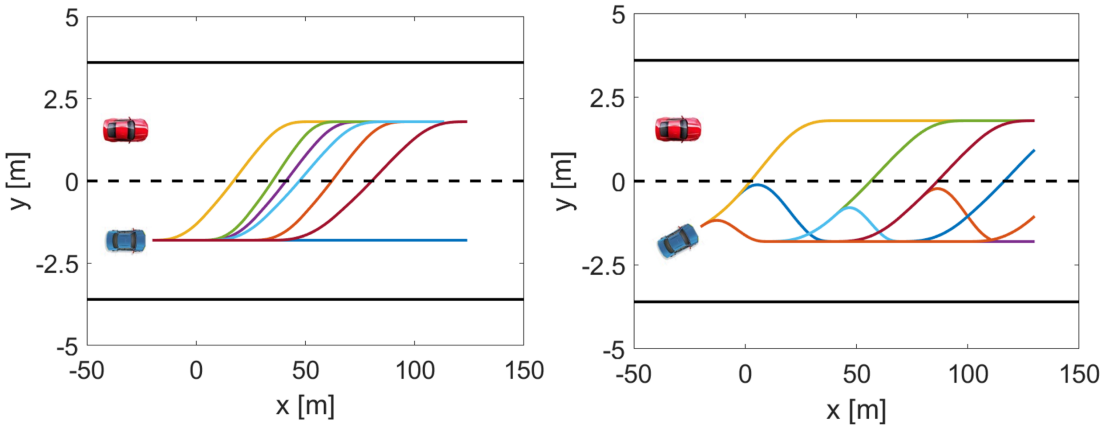

Figure 2.4: Sample trajectories for the merging vehicle. Left figure shows some sample trajectories of the ego merging vehicles before the merging starts. Right figure shows some sample trajectories of the ego merging vehicle after it starts the merging process.

We have defined the choice of a trajectory over a planning horizon for the merging vehicle as its action. Note that the time history of control inputs $u_t = [a_t, \delta_{f,t}]^T$ corresponding to each of these trajectories can be calculated according to the vehicle dynamics model (2.1). In the actual implementation, the chosen trajectory can also be commanded to a lower level vehicle motion controller, and in this chapter, it is assumed that the motion according to (2.6) is accurately realized.

2.2.6 Model predictive control strategy

We first consider an MPC-based trajectory planning strategy for the autonomous ego vehicle accounting for the presence of a single interacting vehicle: At each sample time instant t, the ego vehicle computes an optimal trajectory, $(\gamma_t^0)^*$, that maximizes its cumulative reward over the planning horizon according to

$$
\begin{aligned}
(\gamma_t^0)^* \in \underset{\gamma_t^0 \in \Gamma^0(s_t^0)}{\operatorname{argmax}} & \sum_{\tau=0}^{N-1} \lambda^\tau R\big(\bar{s}_{t+\tau}, u_{t+\tau}^0, u_{t+\tau}^1\big), \qquad (2.7)\\
\text{s.t.} \quad & \bar{s}_{t+\tau+1} = f\big(\bar{s}_{t+\tau}, u_{t+\tau}^0, u_{t+\tau}^1\big),\\
& \bar{s}_{t+\tau} \in S_{\text{safe}}, \quad \forall \tau = 1, \dots, N,
\end{aligned}
$$

where $\bar{s}_{t+\tau} = (s_{t+\tau}^0, s_{t+\tau}^1)$ represents the predicted traffic state at the discrete time instant $t+\tau$, while $u_{t+\tau}^0$ and $u_{t+\tau}^1$ represent, respectively, the predicted ego vehicle's and interacting vehicle's control inputs at $t+\tau$. The parameter $\lambda \in (0,1)$ is discounting future reward thereby prioritizing immediate reward. In (2.7), $R\big(\bar{s}_{t+\tau}, u_{t+\tau}^0, u_{t+\tau}^1\big)$

represents the reward received by the ego vehicle at $t+\tau$, which is described in Section 2.2.4, and S_{safe} represents a set of *safe* traffic states, used to enforce strict safety specifications (such as collision avoidance, road boundary constraints, etc). After an optimal trajectory $(\gamma_t^0)^*$ is obtained, the ego vehicle applies the control inputs corresponding to this trajectory, $(u_t^0)^* = [(a_t^0)^*, (\delta_{f,t}^0)^*]^T$, over one sampling period to update its state, and then repeats the above procedure at the next sample time instant $t+1$.

Note the following points: 1) The expression (2.7) corresponds to the case where the ego vehicle interacts with only one other vehicle ($k=1$). We will extend our MPC strategy to handle multiple vehicle interactions in Section 2.4.2. 2) The ego vehicle's control inputs over the planning horizon, $\{u_t^0, \ldots, u_{t+N-1}^0\}$, correspond to its planned trajectory γ_t^0 and are calculated using γ_t^0 and the vehicle dynamics model (2.1), as has been discussed in Section 2.2.5. 3) The interacting vehicle's control inputs over the planning horizon, $\{u_t^1, \ldots, u_{t+N-1}^1\}$, are unknown variables. In what follows, we introduce a game-theoretic approach that enables predictions of $\{u_t^1, \ldots, u_{t+N-1}^1\}$ in response to the ego vehicle's actions so that the MPC problem (2.7) is solvable.

2.3 Game-Theoretic Model for Vehicle Cooperation Behaviors and Explicit Representation Using Imitation Learning

In this section, we introduce the leader-follower game employed in this chapter for modeling the interaction/cooperation between the merging vehicle and vehicles driving in the target lane. In order to simplify the online computations associated with this game-theoretic model, imitation learning is utilized to derive a neural network-based explicit representation of the model, which is used online for predicting the interacting vehicles' trajectories in response to the merging ego vehicle's actions in our MPC-based trajectory planning strategy.

2.3.1 Leader-follower game-theoretic model

During a highway forced merge process, the merging vehicle (ego vehicle) interacts with other vehicles driving in the target lane, who may choose to proceed or yield to the merging vehicle depending on the traffic situation and individual driver's preference. In this chapter, we consider a game-theoretic model based on pairwise leader-follower interactions, called a leader-follower game, to represent drivers' coop-

eration intentions and their resulting vehicle behaviors. In this model, a vehicle (or, a driver) who decides to proceed before another vehicle is a leader in this vehicle pair and the one who decides to yield to another vehicle is a follower in the pair. The leader and the follower use different decision strategies. This leader-follower game-theoretic model was originally proposed in [99], where it demonstrated the ability to effectively model drivers' intentions to proceed or yield (e.g., caused by common traffic rules and etiquette) in driving through intersections scenarios. Here, we briefly review this game-theoretic model and introduce its application to our highway forced merge scenarios.

Denote the trajectories of the leader and the follower as $\gamma_{l,t} \in \Gamma_l(\bar{s}_t)$, and $\gamma_{f,t} \in \Gamma_f(\bar{s}_t)$, respectively, where $\Gamma_l(\bar{s}_t)$ and $\Gamma_f(\bar{s}_t)$ are the sets of admissible trajectories of the leader and the follower. We assume that both vehicles make decisions to maximize their cumulative rewards, denoted as $\mathbf{R}_l(\bar{s}_t, \gamma_{l,t}, \gamma_{f,t})$ and $\mathbf{R}_f(\bar{s}_t, \gamma_{l,t}, \gamma_{f,t})$, respectively, and defined according to

$$\mathbf{R}_\sigma\big(\bar{s}_t, \gamma_{l,t}, \gamma_{f,t}\big) = \sum_{\tau=0}^{N-1} \lambda^\tau R_\sigma\big(\bar{s}_{t+\tau}, u_{l,t+\tau}, u_{f,t+\tau}\big), \tag{2.8}$$

where $\sigma \in L = \{\text{leader}, \text{follower}\}$ represents the leader or follower role in the game, $R_\sigma\big(\bar{s}_{t+\tau}, u_{l,t+\tau}, u_{f,t+\tau}\big)$ is the reward function for the leader or the follower defined as in Section 2.2.4, $u_{l,t+\tau}$ and $u_{f,t+\tau}$, $\tau = 0, \ldots, N-1$, are the control inputs corresponding to $\gamma_{l,t}$ and $\gamma_{f,t}$ as described in Section 2.2.5, and λ is the discount factor.

Specifically, we model the leader's and the follower's interactive decision processes as follows:

$$\gamma_l^*(\bar{s}_t) \in \underset{\gamma_{l,t} \in \Gamma_l(\bar{s}_t)}{\operatorname{argmax}} \; Q_l(\bar{s}_t, \gamma_{l,t}), \tag{2.9}$$

$$\gamma_f^*(\bar{s}_t) \in \underset{\gamma_{f,t} \in \Gamma_f(\bar{s}_t)}{\operatorname{argmax}} \; Q_f(\bar{s}_t, \gamma_{f,t}), \tag{2.10}$$

where $\gamma_l^*(\bar{s}_t)$ (resp. $\gamma_f^*(\bar{s}_t)$) is an optimal trajectory of the leader (resp. follower) given the current traffic state $\bar{s}_t$, and Q_l and Q_f are defined as

$$Q_l(\bar{s}_t, \gamma_{l,t}) = \min_{\gamma_{f,t} \in \Gamma_f^*(\bar{s}_t)} \mathbf{R}_l(\bar{s}_t, \gamma_{l,t}, \gamma_{f,t}), \tag{2.11}$$

$$Q_f(\bar{s}_t, \gamma_{f,t}) = \min_{\gamma_{l,t} \in \Gamma_l(\bar{s}_t)} \mathbf{R}_f(\bar{s}_t, \gamma_{l,t}, \gamma_{f,t}), \tag{2.12}$$

where $\Gamma_f^*(\bar{s}_t) = \{\gamma'_{f,t} \in \Gamma_f(\bar{s}_t) : Q_f(\bar{s}_t, \gamma'_{f,t}) \geq Q_f(\bar{s}_t, \gamma_{f,t}),\ \forall \gamma_{f,t} \in \Gamma_f(\bar{s}_t)\}$.

The decision model (2.9)-(2.12) can be explained as follows: A follower represents a driver who intends to yield. Due to uncertainty about the other driver's action, the follower decides to take an action that maximizes her worst-case reward through (2.10) and (2.12). Such a "max-min" decision strategy of the follower models the yielding behavior because it assumes the other driver can take actions freely. Similarly, a leader represents a driver who intends to proceed and assumes the other driver will yield. Therefore, the leader uses the follower model to predict the other driver's action and takes an action that maximizes the leader own reward under the predicted follower's action through (2.9) and (2.11).

Note that this leader-follower strategies set-up is not symmetric. We let the follower adopt the worst-case assumptions (which is reasonable since the follower does not know the leader's action/strategy), but let the leader adopt the best-case assumptions (which imply that the follower will take the conservative worst-case strategy). The worst-case assumptions of the follower are realized by considering all possible actions of the leader during the minimization (2.12), whereas the best-case assumptions of the leader are achieved by only considering actions in $\Gamma_f^*(\bar{s}_t)$, as shown in (2.11). $\Gamma_f^*(\bar{s}_t)$ is follower's set of rational actions, which consists of all actions in the current state $\bar{s}_t$ that can maximize the worst-case reward. We purposely introduce this asymmetry to provoke aggressive actions from the leader of the game.

With the above review of the game-theoretic model originally introduced in [99], we explain below the strategy in the context of a main lane traveling vehicle making decisions in the situations where an on-ramp vehicle is likely to perform a forced merge maneuver under heavy traffic. A follower in the game represents a cautious or conservative driver, who tends to yield to an on-ramp merging vehicle. Due to uncertainty about all possible on-ramp merging vehicles' actions, the follower is assumed to take a "max-min" strategy to secure his reward in the worse-case, which is achieved through (2.10) and (2.12). A leader, on the contrary, represents an aggressive driver and may favor proceeding before the on-ramp merging vehicle. This aggressive behavior is induced by assuming the other player, the on-ramp vehicle, is being a follower and taking the worst case "max-min" strategy described by (2.9) and (2.11).

Note that while the above decision model is inspired by a Stackelberg game model [114], however, as stated in (2.9)-(2.12) it differs from a Stackelberg game model and the associated Stackelberg equilibrium. A Stackelberg game model relies on stronger assumptions such as the follower can instantaneously know and respond to the leader's action, and this is usually not the case during drivers' interactions due to the fact

that a driver's decision can only be revealed over time and is further impeded by human reaction delay. In contrast, (2.9)-(2.12) do not rely on such an assumption.

Although the asymmetric leader-follower roles in the decision model (2.9)-(2.12) are used to represent drivers' intentions and tendencies to proceed or yield, the model does not imply that a main lane vehicle will always choose to proceed when it encounters a follower on-ramp vehicle. Neither does it imply an on-ramp merging vehicle will always force a merge if it encounters a follower vehicle in the main lane. For instance, a merging vehicle may merge in front of a leader interacting vehicle in the following two situations: 1) The merging vehicle is ahead of the interacting vehicle with a sufficiently large distance to allow safe merging. 2) The merging vehicle is about to reach the end of its lane. In the second situation, getting off the road would yield a large penalty (see Section II-C), and the merging vehicle may choose to merge ahead of the interacting vehicle to avoid the large penalty as long as its merge is not 'likely' (to be quantified in a later section) to lead to a collision. The above observations clarify that the leader-follower roles in our decision model (2.9)-(2.12) are not determined simply by vehicle spatial positions (i.e., a leader is not necessarily a vehicle in front). Moreover, this model facilitates the highway merging human driver behavior we have all observed – an on-ramp vehicle may choose to merge and force the traffic in the main lane to slow down, if necessary. As the merging vehicle approaches the end of its lane, it is increasingly inclined to merge to avoid the penalty of getting off the road even if all the interacting vehicles on the target/main lane are leaders (whose drivers are inclined to proceed) and the current gaps are not large enough for a typical constant speed comfortable merge. As for the aggressive behaviors of human drivers of the main lane vehicles, the model (2.9)-(2.12) enhances the safety aspects of these leader interacting vehicles, as they will be able to predict the on-ramp vehicle's upcoming merging maneuver. Then, for their own safety and comfort, these aggressive drivers may still choose to slow down and yield despite playing a leader's role. Therefore, our leader-follower model (2.9)-(2.12) is suitable for trajectory prediction and planning in tight and competitive traffic scenarios where forced merges are commonly observed.

2.3.2 Explicit representation of leader-follower game policy through imitation learning

Based on (2.9)-(2.12), we are able to predict other vehicles' decision and trajectories given the knowledge of drivers' intentions and the current traffic state information. Hence, we can denote leader's optimal action policy as $\gamma_l^*(\bar{s}_t)$ and follower's optimal action policy as $\gamma_f^*(\bar{s}_t)$. Obtaining $\gamma_l^*(\bar{s}_t)$ and $\gamma_f^*(\bar{s}_t)$ require going through (2.9)-(2.12),

and the repeated online computations involving (2.9)-(2.12) can be time consuming. As a result, we want to explicitly represent γ_l^* and γ_f^*.

Here, $\gamma_\sigma^*(\bar{s}_t), \sigma \in L$ are maps that map current traffic state to a predicted trajectory that other vehicles will follow. These maps are determined according to (2.9)-(2.12). Instead of algorithmically determining $\gamma_l^*(\bar{s}_t)$ and $\gamma_f^*(\bar{s}_t)$, we follow [115] and exploit the use of supervised learning, more specifically, imitation learning, to represent $\gamma_\sigma^*(\bar{s}_t)$.

Imitation learning can be considered as a supervised learning problem, where an autonomous agent tried to learn a policy by observing expert's demonstrations. The expert demonstration can be generated either by a human operator or an artificial intelligent agent. In this work, we treat $\gamma_\sigma^*(\bar{s}_t)$ obtained by (2.9)-(2.12) as the expert policy.

Algorithm 2.1 Imitation learning algorithm to obtain Leader-Follower Game policies

Initialize $\hat{\gamma}_k^0$ to an arbitrary policy
Initialize dataset $\mathcal{D} = \emptyset$
for $n = 1 : n_{\max}$ **do**
    Initialize the simulation environment
    for $t = 0 : t_{\max} - 1$ **do**
        if ego vehicle fails or succeeds **then**
            Break
        end if
        if $\hat{\gamma}_k^{n-1}(\bar{s}_t) \neq \gamma_k(\bar{s}_t)$ **then**
            $\mathcal{D} = \mathcal{D} \cup (\bar{s}_t, \gamma_k(\bar{s}_t))$
        end if
        Randomly select $\gamma_0 \in \Gamma_0$
        $\bar{s}_{t+1} = f(\bar{s}_t, \gamma_0, \hat{\gamma}_k^{n-1}(\bar{s}_t))$
    end for
    Train classifier $\hat{\gamma}_k^n$ on $\mathcal{D}$
end for
$\hat{\gamma}_k = \hat{\gamma}_k^{n_{\max}}$

Mathematically, the imitation learning process can be formulated as,

$$\hat{\gamma}_\sigma \in \underset{\gamma_\theta}{\operatorname{argmin}} \, \mathbb{E}_{\bar{s} \sim \mathbb{P}(\bar{s}|\gamma_\sigma^*)} \Big(\mathcal{L}(\gamma_\sigma^*(\bar{s}), \gamma_\theta(\bar{s})) \Big), \tag{2.13}$$

where γ_θ represents a policy that is optimized with respect to and is parametrized by θ (e.g. neural network weights), $\mathcal{L}$ represents a loss function, and $\mathbb{E}_{\bar{s} \sim \mathbb{P}(\bar{s}|\gamma_\sigma^*)}(\cdot)$ is defined as,

$$\mathbb{E}_{\bar{s} \sim \mathbb{P}(\bar{s}|\gamma_\sigma^*)}(\cdot) = \int (\cdot) d\mathbb{P}(\bar{s}|\gamma_\sigma^*). \tag{2.14}$$

From (2.13) and (2.14), one important feature is that the expectation is with respect to the probability distribution $P(\bar{s}|\gamma_\sigma^*)$, which represents the distribution of $\bar{s}$ with respect to the expert policy γ_σ^*.

As investigated in [115], traditional imitation learning followed by (2.13) and (2.14) suffers from sampling bias issues because only the state that can be reached by executing the expert policy γ_σ^* will be included in the dataset, and this sampling bias can make the error between the expert policy γ_σ^* and the trained policy $\hat{\gamma}_\sigma$ propagate over time. More specifically, a bias prediction of the trained policy at one time may lead the system to reach a new state that is not in the training dataset, and the trained policy will need to make a prediction at the new state, which may cause the predicted state and actual state become farther and farther away over time.

To overcome this drawback, the "Dataset Aggregation" algorithm has been used to obtain the policy $\hat{\gamma}_\sigma$. The overall learning objective for the Dataset Aggregation algorithm can be described by,

$$\hat{\gamma}_\sigma \in \underset{\gamma_\theta}{\operatorname{argmin}} \, \mathbb{E}_{\bar{s}\sim\mathbb{P}(\bar{s}|\hat{\gamma}_\theta)}\Big(\mathcal{L}(\gamma_\sigma^*(\bar{s}), \gamma_\theta(\bar{s}))\Big), \tag{2.15}$$

$$\mathbb{E}_{\bar{s}\sim\mathbb{P}(\bar{s}|\gamma_\theta)}(\cdot) = \int (\cdot) d\mathbb{P}(\bar{s}|\gamma_\theta). \tag{2.16}$$

Intuitively, the difference between (2.13) and (2.15) is that the later optimizes the expectation with respect to the probability distribution induced by the policy γ_θ. This can effectively resolve the propagation of errors due to sampling bias, because the dataset contains information of states $\bar{s}$ reached by executing γ_θ.

Algorithm 2.1 presents how we obtain an explicit representation of the leader or follower decision policies through imitation learning. In Algorithm 2.1, n_{max} represents the total number of learning epochs and t_{max} the simulation length of each learning epoch. For each learning epoch, simulation environment is first initialized (Line 4), which means the force merge scene is created with ego vehicle and one interacting vehicle. When the ego vehicle either merges successfully or fails (means collision or fail to merge by the end of current lane), the algorithm will restart with another learning epoch (Line 6-8).

With the aid of such explicit representation of (2.9)-(2.12) through imitation learning, we are able to reduce the online computation of $\gamma_l^*(\overline{s}_t)$ and $\gamma_f^*(\overline{s}_t)$ by over 300 times, from 0.325 (s) (by going through (2.9)-(2.12)) to 0.001 (s) (by accessing the imitated policy). These timing results are based on averaging over 500 random generated $\overline{s}_t$, and the tests are performed on the computer listed in Section 2.5. The

drawback of this imitation learning approach is that it requires training of the neural network to represent the policy and a sufficient set of data for training.

2.4 Decision Making under Cooperation Intention Uncertainty

In this section, we describe the decision making algorithm, called the Leader-Follower Game Controller (LFGC), for the highway forced merge scenario under cooperation intention uncertainty. During the forced merge process, we generate an estimate of other driver's cooperation intention, as described in this section. Based on the estimate of cooperation intention, we apply the control strategy presented in (2.7) under multi-vehicle interactions settings by considering pairwise interactions.

2.4.1 Estimation of interacting vehicle's cooperation intention

According to Section 2.3, the model (2.9)-(2.12) and the imitation learning policies can be used to predict the other vehicles' decisions and future trajectories under the knowledge of their drivers' cooperation intentions. However, in a given traffic scenario, we may not know the other drivers' cooperation intentions a priori, because a driver's intention depends not only on the traffic situation (e.g., the relative position and velocity between two vehicles) but also on the driver's style/type (e.g., aggressive versus conservative). To deal with prior uncertainties in other vehicles' cooperation intentions, in what follows we describe an approach where such uncertainties are modeled as latent variables, other vehicle's cooperation intentions are estimated, and the trajectory is generated through the application of predictive control. We can model other driver's behavior based on their cooperation intentions using the leader-follower game. A yielding vehicle may have similar behavior as a follower in the game, while a proceeding (not yielding) vehicle may be modeled as a leader in the game. In this sense, we can estimate interacting vehicle's cooperation intention by estimating their leader or follower roles in the leader-follower game.

To achieve that, we consider the traffic dynamics model (2.2) and the leader or follower's optimal actions (2.9) and (2.10). From the perspective of the ego vehicle, the interacting vehicle is playing a leader-follower game with it, and the traffic dynamics model can be written as

$$\bar{s}_{t+1} = f(\bar{s}_t, u_t^0, u_t^1) = f\big(\bar{s}_t, u_t^0, (u_{\sigma,t}^1)^*(\bar{s}_t)\big), \tag{2.17}$$

where u_t^0 is the control of ego vehicle, u_t^1 is the control of the interacting vehicle and is determined by the leader-follower game, $\sigma \in L = \{\text{leader}, \text{follower}\}$ represents either leader of follower, and $(u_{\sigma,t}^1)^*(\bar{s}_t)$ is the first control input corresponding to the optimal trajectory of $\gamma_\sigma^*(\bar{s}_t)$ in (2.9) and (2.10). Now the only input to (2.17) is the control of the ego vehicle u_t^0.

However, in reality, the interacting vehicle's decision does not necessarily follow the optimal policy computed from (2.9) and (2.10). In order to account for the difference between the leader-follower policy and the actual policy of the interacting vehicle, we assume the system is propagated by (2.17) with an additive Gaussian noise, i.e.,

$$\bar{s}_{t+1} = f\big(\bar{s}_t, u_t^0, (u_{\sigma,t}^1)^*(\bar{s}_t)\big) + w, \quad w \sim N(0, W), \tag{2.18}$$

where w is the additive Gaussian noise with 0 mean and covariance W.

The ego vehicle is assumed to have a prior belief on σ, denoted as $\mathbb{P}(\sigma = l)$, with $l \in L = \{\text{leader}, \text{follower}\}$. Then based on all previous traffic states and on all actions taken by the ego vehicle,

$$\xi_t = \{\bar{s}_0, \bar{s}_1, \ldots, \bar{s}_t, u_0^0, u_1^0, \ldots, u_{t-1}^0\}, \tag{2.19}$$

the ego vehicle needs to compute or maintain a posterior belief of the interacting vehicle's leader or follower role, $\mathbb{P}(\sigma = l|\xi_t)$.

The conditional posterior belief of interacting vehicle's leader or follower's role is computed using the hybrid estimation algorithm proposed in [116].

Specifically, identification of the interacting vehicle's leader or follower role can be achieved by

$$\mathbb{P}(\sigma = l|\xi_t) = \frac{\Lambda_{l,t}}{c_t} \sum_{k \in L} \pi_{lk} \mathbb{P}(\sigma = k|\xi_{t-1}), \tag{2.20}$$

where $\mathbb{P}(\cdot|\cdot)$ is the conditional probability; π_{lk} denotes the transition probability of the interaction vehicle's role from k to l; and $\Lambda_{l,t}$ is the likelihood function of the interacting vehicle as role l, defined by

$$\begin{aligned} &\Lambda_{l,t} = \mathcal{N}(r_{l,t}, 0, W), \\ &r_{l,t} = \bar{s}_t - f(\bar{s}_{t-1}, u_{t-1}^0, (u_{l,t}^1)^*(\bar{s}_{t-1})), \end{aligned} \tag{2.21}$$

where $\mathcal{N}(r_{l,t}, 0, W)$ denotes the probability density function of the normal distribution with mean 0 and covariance W evaluated at $r_{l,t}$; and c_t is a normalization constant.

Assuming the interacting vehicle's role remains unchanged over the merge period,

i.e., $\pi_{lk} = 1$ when $l = k$ and $\pi_{lk} = 0$ when $l \neq k$, the posterior belief of the interacting vehicle's leader or follower role can be updated using

$$\mathbb{P}(\sigma = l|\xi_t) = \frac{\mathcal{N}(r_{l,t}, 0, W)\mathbb{P}(\sigma = l|\xi_{t-1})}{\sum_{k\in L}\mathcal{N}(r_{k,t}, 0, W)\mathbb{P}(\sigma = k|\xi_{t-1})}, \tag{2.22}$$

where $\mathbb{P}(\sigma = l|\xi_{t-1})$ is the previous belief of the interacting vehicle's leader or follower role. Note that the belief in (2.22) is updated from previous to posterior at the 4 (Hz) decision frequency; hence the approach still allows to account for changing roles of the interacting vehicle during the merge period.

2.4.2 Control strategy for multi-vehicle interactions

When the traffic is busy, there may exist multiple vehicles on highway that may interfere with the ego vehicle's merge, such as in the case shown in Figure 2.1. One low complexity solution would be for the ego vehicle to only consider interaction with the first vehicle, and after the first vehicle becomes farther away, the ego vehicle starts to interact with the second vehicle. However, this may slow down the estimate of later vehicles' intentions, and in the case of highway forced merge, this can lead to the ego vehicle missing an opportunity to merge.

Another solution is to interact with multiple vehicles at the same time. In this case, a model needs to be constructed to predict interacting vehicle's actions. Although the 2-player leader-follower game described in Section 2.3 can be extended to multi-player leader-follower game by considering a multi-level decision hierarchy and then solving for the Nash-equilibrium, such extensions may exponentially increase the computational time as the number of players increase. The Stackelberg equilibrium can be hard to obtain when there are more than 3 players [117]. As a result, we propose a computationally tractable approach to extend the framework to multi-vehicle interactions by considering pairwise interactions.

When there are m interacting vehicles, we consider pairwise interactions of the ego vehicle and each interacting vehicle. Then we can construct m traffic states denoted as $\bar{s}^k,\ k \in \{1, \ldots, m\}$ which contains the states of the ego vehicle and kth interacting vehicle, and the dynamic model of each $\bar{s}^k$ is given by

$$\bar{s}^k_{t+1} = f(\bar{s}^k_t, \bar{u}^k_t) = f(\bar{s}^k_t, u^0_t, u^k_t). \tag{2.23}$$

Similarly, we can denote by $\sigma^k \in L = \{\text{leader}, \text{follower}\}$ the pairwise leader or follower role of the kth interacting vehicle and by ξ^k_t the collection of all previous

pairwise traffic states and actions taken by the ego vehicle, i.e.,

$$\xi_t^k = \{\bar{s}_0^k, \bar{s}_1^k, \ldots, \bar{s}_t^k, u_0^0, u_1^0, \ldots, u_{t-1}^0\}. \tag{2.24}$$

Then we can utilize (2.22) to update the belief of each interacting vehicle's leader or follower role, $\mathbb{P}(\sigma^k = l|\xi_t^k), l \in L = \{\text{leader}, \text{follower}\}$. The MPC-based control strategy presented in (2.7) can be reformulated as

$$\begin{aligned}(\gamma^0)^* \in \underset{\gamma^0 \in \Gamma^0(s_t^0)}{\operatorname{argmax}} & \sum_{k=1}^{m} \mathbb{E}\Big\{ \sum_{\tau=0}^{N-1} \lambda^\tau R\big(\bar{s}_{t+\tau}^k, u_{t+\tau}^0, \hat{u}_{\sigma^k,t}^k(\bar{s}_{t+\tau}^k)\big) \Big| \xi_t^k \Big\}, \\ \text{s.t.} \quad & \bar{s}_{t+\tau+1}^k = f\big(\bar{s}_{t+\tau}^k, u_{t+\tau}^0, \hat{u}_{\sigma^k,t}^k(\bar{s}_{t+\tau}^k)\big), \forall k = 1, \ldots, m \\ & \sum_{k=1}^{m} \mathbb{P}\Big(\bar{s}_{t+\tau}^k \in S_{\text{safe}}, \forall \tau = 1, \ldots, N \,\Big|\, \xi_t^k\Big) \geq m - \varepsilon,\end{aligned} \tag{2.25}$$

where $\hat{u}_{\sigma,t}^k(\bar{s}_{t+\tau}^k)$ is the first control input corresponding to the trajectory of the trained policy $\hat{\gamma}_\sigma(\bar{s}_{t+\tau}^k)$ in (2.15), and $\varepsilon \in [0, 1]$ represents a (user-specified) required probability level of constraint satisfaction. Please note that we treat the selection of the trajectory (time-history of states) as the vehicle's control action, and (2.25) is solved by evaluating all pre-computed trajectories in the carefully chosen trajectory set $\Gamma^0(s_t^0)$. By carefully constructing the trajectory set as described in Section 2.2.5, we are able to ensure computational efficiency and to guarantee optimality within the pre-computed trajectory set.

The expectation in the objective function can be computed according to

$$\begin{aligned}&\mathbb{E}\Big\{ \sum_{\tau=0}^{N-1} \lambda^\tau R\big(\bar{s}_{t+\tau}^k, u_{t+\tau}^0, \hat{u}_{\sigma^k,t}^k(\bar{s}_{t+\tau}^k)\big) \Big| \xi_t^k \Big\} \\ &= \sum_{l \in L} \sum_{\tau=0}^{N-1} \lambda^\tau R\big(\bar{s}_{l,t+\tau}^k, u_{t+\tau}^0, \hat{u}_{l,t}^k(\bar{s}_{l,t+\tau}^k)\big) \mathbb{P}(\sigma^k = l|\xi_t^k),\end{aligned} \tag{2.26}$$

where $\bar{s}_{l,t+\tau}^k$ is the predicted traffic state given that the interacting vehicle's role is l,

$$\bar{s}_{l,t+\tau+1}^k = f\big(\bar{s}_{l,t+\tau}^k, u_{t+\tau}^0, \hat{u}_{l,t}^k(\bar{s}_{l,t+\tau}^k)\big) \tag{2.27}$$

and the last constraint in (2.25) can be evaluated by

$$\mathbb{P}\Big(\bar{s}_{t+\tau}^k \in S_{\text{safe}}, \forall \tau = 1, \ldots, N \,\Big|\, \xi_t^k\Big) = \sum_{l \in L} \min_{1 \leq \tau \leq N} \mathbb{I}_{S_{\text{safe}}}(\bar{s}_{l,t+\tau}^k) \mathbb{P}(\sigma^k = l|\xi_t^k), \tag{2.28}$$

where $\mathbb{I}_B(b)$ is the indicator function of b in set B.

Note that the last constraint in (2.25) enforces the following condition,

$$\mathbb{P}\Big(\bigcap_{k=1}^{m} \bar{s}_{t+\tau}^k \in S_{\text{safe}}, \forall \tau = 1, \dots, N \,\Big|\, \xi_t^k\Big) \geq 1 - \varepsilon, \tag{2.29}$$

which means that the probability of any pairwise interactions entering unsafe states (e.g., collision and out of road boundaries) is less than ε.

To derive (2.29), we first denote the event $A^k := \{\bar{s}_{t+\tau}^k \in S_{\text{safe}}, \forall \tau = 1, \dots, N | \xi_t^k\}$, then

$$\begin{aligned}\mathbb{P}\big(\bigcap_{k=1}^{m} A^k\big) &= 1 - \mathbb{P}\big(\bigcup_{k=1}^{m} (A^k)^c\big) \geq 1 - \sum_{k=1}^{m} \mathbb{P}\big((A^k)^c\big) \\ &= 1 - \sum_{k=1}^{m} (1 - \mathbb{P}(A^k)) = \sum_{k=1}^{m} \mathbb{P}(A^k) - m + 1,\end{aligned}$$

and applying the last constraint in (2.25), it follows that

$$\mathbb{P}\big(\bigcap_{k=1}^{m} A^k\big) \geq \sum_{k=1}^{m} \mathbb{P}(A^k) - m + 1 \geq 1 - \varepsilon.$$

The major differences between (2.7) and (2.25) are the following: 1) $\{u_t^1, u_{t+1}^1, \dots, u_{t+N-1}^1\}$ presented in (2.7) are unknown, while in (2.25), they are obtained based on trained policy from the imitation learning; 2) The maximization of the cumulative reward in (2.7) is changed to maximization of the expected cumulative reward in (2.25) to account for probabilistic belief about the interacting vehicle's leader/follower role; 3) The expected cumulative reward is changed to the sum of the expected reward of all pairwise interactions to account for uncertain behavior of multiple vehicles; 4) The hard constraint is changed to a probabilistic chance constraint with $\varepsilon \in [0, 1]$ being a design parameter.

The decision making algorithm proceeds as follows: At the sampling time t, the ego vehicle measures the current states of each pairwise interaction and adds them together with the previous control input to the observation vectors ξ_t^k. The belief about each vehicle's leader or follower role is updated according to (2.22) based on ξ_t^k. Then, the MPC-based control strategy (2.25) is utilized to obtain the optimal trajectory $(\gamma^0)^*$ by searching through all trajectories introduced in Section 2.2.5, and the ego vehicle applies the first control input $(u_t^0)^*$ over one sampling period to update its states. The whole procedure is repeated at the next sampling time.

Note that the control strategy (2.25) is "interaction-aware" due to the following reasons: 1) It predicts other vehicles' trajectories under varied interaction intentions based on the leader-follower game-theoretic model (2.9)-(2.12). 2) The predictions are closed-loop. Specifically, for different trajectory plans of the ego vehicle, $\gamma^0 \in \Gamma^0(s_t^0)$, the corresponding trajectory predictions of the other vehicles under certain intentions are different. This is the case because the predicted other vehicles' actions are traffic state-dependent while the predicted traffic states depend on the planned ego vehicle's trajectory. 3) The objective function in (2.25) is a conditional expectation and the constraint to represent safety is a conditional probability, both of which are conditioned on the latest estimates of other vehicles' intentions (i.e., leader or follower), $\mathbb{P}(\sigma^k = l|\xi_t^k)$. Meanwhile, the other vehicles' intentions are estimated based on their previous interaction behaviors.

2.5 Simulation and Validation Results

In this section, we present validation results of applying the proposed Leader-Follower Game Controller (LFGC) for autonomous vehicle forced merge problems. Specifically, we consider three sets of simulation validations, and in these simulations, the LFGC assumes that interacting vehicles are playing a leader-follower game with the ego vehicle and estimates their leader/follower roles in the game. We also assume that once in the mandatory lane change situation, the ego vehicle prepositions itself towards the lane marker with turn signals to declare its merge intention and starts the forced merge process. As a result, interacting vehicles are aware of the ego vehicle's merging intention and react accordingly. We first validate the LFGC with interacting vehicles controlled by leaders or followers in the leader-follower game. Then we test the LFGC versus interacting vehicles controlled by other types of drivers or actual traffic data. Specifically, we test the cases where interacting vehicles are controlled by the intelligent driver model (IDM) and where interacting vehicles are following the actual US Highway 101 traffic data present in the Next Generation Simulation website [82]. In the first two sets of tests (interacting vehicles controlled by leader-follower game and IDM), three interacting vehicles moving at high velocity are considered, and in the third set of test, a real traffic dataset corresponding to a fairly dense traffic is considered. In these simulation studies, ΔT for planning is chosen as 1 (s), and the LFGC makes decisions at a frequency of 4 (Hz). Note that our simulations are performed in MATLAB R2019a on an PC with Intel Xeon E3-1246 v3 @ 3.50 GHz CPU and 16 GB RAM.

2.5.1 Interacting vehicles driven by leader/follower

We first test our proposed LFGC when interacting vehicles are simulated and controlled by leaders/followers in the game. The scenario we considered is shown in Figure 2.5, where the autonomous ego vehicle (blue) in the acceleration lane needs to merge onto the highway before the end of acceleration lane while multiple other vehicles (red, pink, green) are currently driving on the highway. The ego vehicle starts the forced merge process by biasing towards lane markers and flashing turn signals at the moment shown in Figure 2.5. In such a scenario, the autonomous vehicle needs to interact with other vehicles to merge safely.

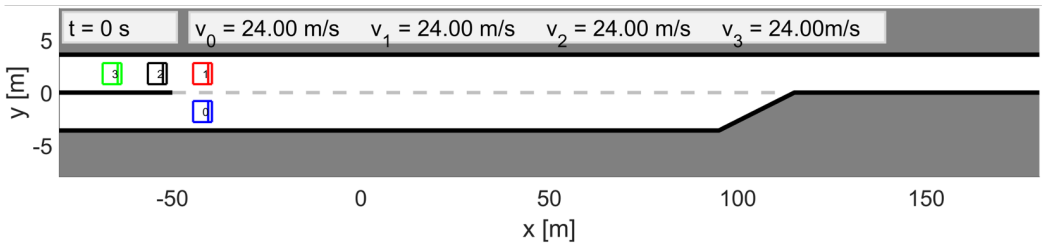

Figure 2.5: Illustration of highway forced merge scenarios for validations of the LFGC when highway vehicles are controlled by leaders/followers in the game.

For the LFGC, the planning horizon is selected as $N = 4$ and the chance constraint parameter is chosen as $\varepsilon = 0.1$. Note that a larger N may result in better long-term performance but also lead to longer computational time, while a smaller N may emphasize on immediate benefits and hence fail to merge in many scenarios. For highway forced merge considered in this chapter, N, in general, needs to be chosen such that it is longer than the duration of the lane change (i.e., $N\Delta T \geq T_{\text{lc}}$). The initial beliefs are set to $\forall k \in \{1, 2, 3\}$, $\mathbb{P}(\sigma^k = \text{leader}) = \mathbb{P}(\sigma^k = \text{follower}) = 0.5$. Figure 2.6 shows the results when the ego vehicle is interacting with different combinations of leaders and followers. In Figure 2.6, the left column ((a-1) to (d-1)) shows the ego vehicle belief about each of the other vehicles being a leader in the game, $\mathbb{P}(\sigma^k = \text{leader}), k = 1, 2, 3$. The right column shows the time history results of the ego vehicle and other vehicles' behaviors during this forced merge process.

Figure 2.6(a) shows the results when the ego vehicle is interacting with three leaders. The ego vehicle is able to capture interacting vehicle's intentions that all vehicles are more likely to be leaders in the game as shown in Figure 2.6(a-1). After obtaining this information, the ego vehicle decides to slow down after $t = 1$ (s) and waits to merge after all interacting vehicles pass. Shown in Figure 2.6(b) are the results of the ego vehicle interacting with two leaders (vehicles 1 and 2) and one

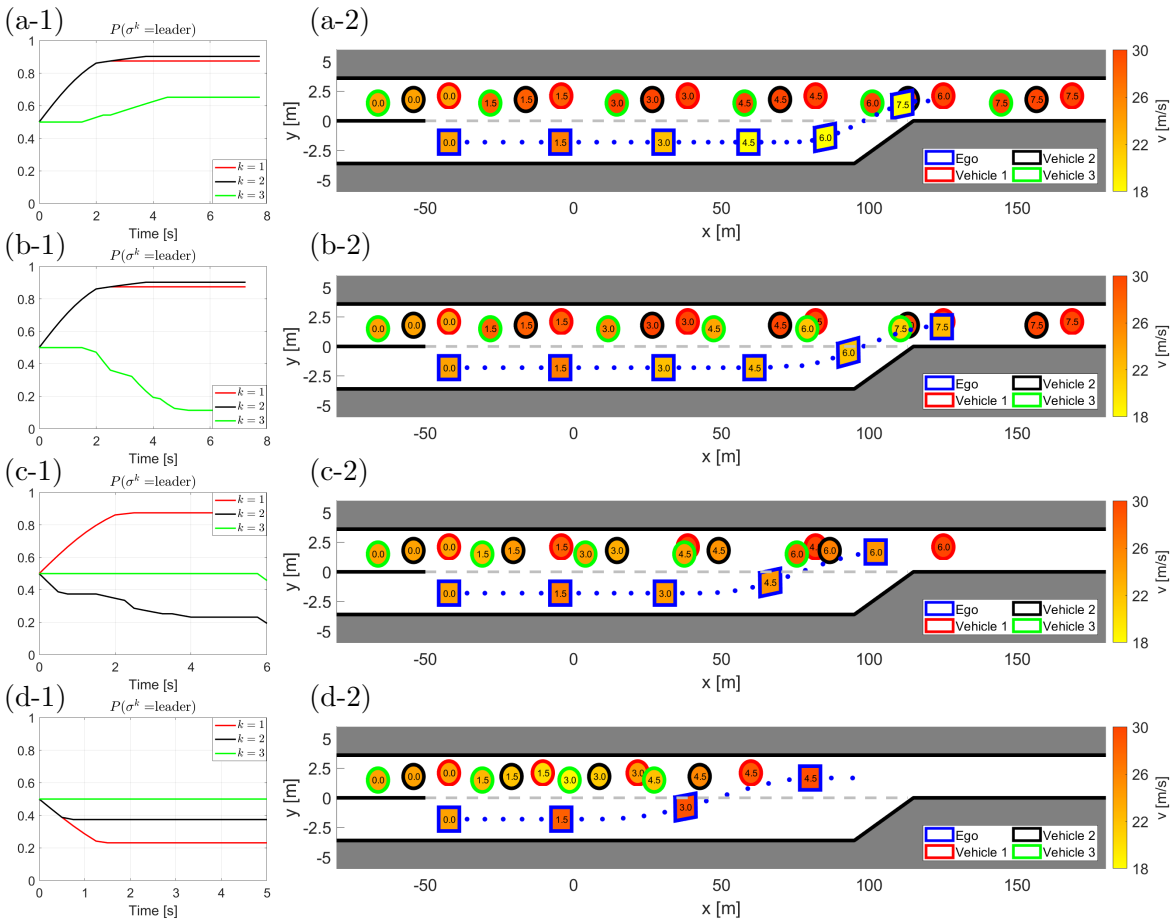

Figure 2.6: Results of the proposed LFGC against other drivers following leader/follower policy with different leaders and followers combinations: (a) three leaders; (b) one leader (vehicle 1) and two followers (vehicles 2 and 3); (c) two leaders (vehicles 1 and 2) and one follower (vehicle 3); (d) three followers. The left column (a-1) to (d-1) shows the ego vehicle's belief on other vehicles' being a leader in the game, $\mathbb{P}(\sigma^k = \text{leader}), k = 1, 2, 3$. The right column (a-2) to (d-2) shows the time history results of the ego and other vehicles behaviors during this forced merge process. Specifically, in the right column, the boundary line color of each block distinguishes different vehicles, the number in the block represents time in seconds, the color of each block describes the speed of the vehicle at that time instant, and the blue dotted line represents the ego vehicle's trajectory. Note that vehicles 1-3 share the same vertical positions, and some offsets in vertical direction are added to better distinguish them in the figure.

follower (vehicle 3). In this case, the ego vehicle observes that vehicles 1 and 2 speed up and do not yield to it, so the ego vehicle decides to slow down and merge between vehicles 2 and 3. When the ego vehicle is interacting with one leader (vehicle 1) and two followers (vehicle 2 and 3), the ego vehicle recognizes interacting vehicle's

intentions correctly as shown in Figure 2.6(c-1). Then the ego vehicle starts to slow down after $t = 1$ (s), and successfully merges between vehicles 1 and 2, which is shown in Figure 2.6(c-2). We also perform the test when the ego vehicle interacts with three followers, and the results are shown in Figure 2.6(d), where the ego vehicle observes all vehicles yielding intentions, speeds up and merges in front of all interacting vehicles. The average computational time for solving (2.25) at each time step is 0.182 (s).

For all cases shown in Figure 2.6, the initialized beliefs are the same, which means the ego vehicle does not know ahead of time whether the interacting vehicle is a leader or a follower. As a result, the ego vehicle relies on its observations to estimate interacting vehicles leader/follower role. The proposed LFGC can capture interacting vehicles' intentions and making decisions accordingly when all interacting vehicles are controlled by the leader/follower in a leader-follower game.

2.5.2 Interacting vehicles driven by intelligent driver model

The validation results shown in Section 2.5.1 assumes that other drivers make decisions based on the leader-follower game. The LFGC assumes other drivers are playing leader-follower game with the ego vehicle, estimates their roles in the game, and makes decision accordingly. This means that the environment in Section 2.5.1 behaves just as the LFGC expects. However, the actual behavior of other drivers might be different from the leader-follower game's policy. As a result, we want to further investigate how the framework responds to other types of driver models.

In this section, we employ the intelligent driver model (IDM) to control other vehicles and interact with the ego vehicle. The ego vehicle is still controlled by the LFGC and tries to estimate interacting vehicles' intentions by estimating their corresponding leader or follower roles. IDM is a continuous-time car-following model and is described by (2.30) to (2.32) [118]. The equations for IDM are

$$\dot{x} = v, \tag{2.30}$$

$$\dot{v} = a_m \left(1 - \left(\frac{v}{v_0} \right)^{\delta} - \left(\frac{\phi^*(v, \Delta v)}{\phi} \right)^2 \right), \tag{2.31}$$

where x is the longitudinal position; v is the longitudinal velocity; v_0 is the desired velocity of the vehicle; $\phi = x - x_t - l_t$ is the following distance with x_t being the position of the target vehicle and l_t being the length of the target vehicle; $\Delta v = v - v_t$ is the velocity difference of the vehicle and the target vehicle; $\phi^*(v, \Delta v)$ is obtained

according to,

$$\phi^*(v, \Delta v) = \phi_0 + vT + \frac{v\Delta v}{2\sqrt{a_m b}}, \tag{2.32}$$

where a_m, ϕ_0, T, b are parameters of the IDM model. The physical interpretation of these parameters are the maximum acceleration a_m, the minimum car following distance ϕ_0, the desired time headway T, and the comfortable deceleration b.

For the validation tests, we consider the scenario shown in Figure 2.7. In Figure 2.7, there is another vehicle ahead of all vehicles (black vehicle 4), and it is driving at a constant speed. The ego vehicle is still the same as in Section 2.5.1 and is controlled by the LFGC, which means that from the ego vehicle perspective, it is playing a leader-follower game with all interacting vehicles. For these three interacting vehicles (vehicle 1 to 3), they are controlled by IDM to follow either the front vehicle (vehicle 4) or the ego vehicle with a certain time headway T. The IDM model parameters are listed in Table 2.1. Note that the ego vehicle regards vehicle 4 as the environmental vehicle and assumes it is driving at constant speed.

Different desired time headways in IDM may reflect conservativeness of the drivers. If the interacting vehicle intends to yield to the ego vehicle, we model it to use IDM to follow the ego vehicle with certain time headway. This means each interacting vehicle has an option to follow either the front vehicle or the ego vehicle.

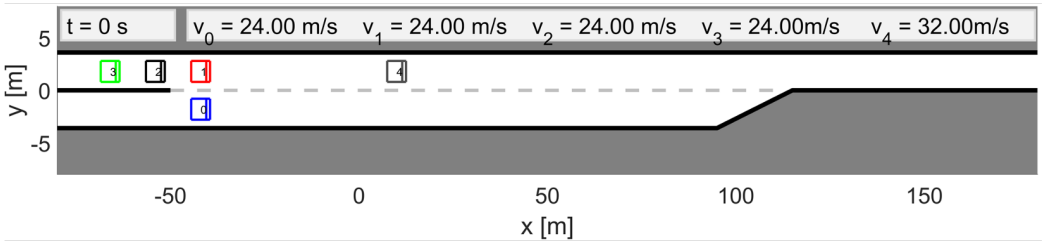

Figure 2.7: Illustration of highway forced merge scenarios for validations of the LFGC when highway vehicles are controlled by IDM.

For the LFGC, the setting is the same as in Section 2.5.1. The planning horizon is selected as $N = 4$ and the chance constraint parameter is chosen as $\varepsilon = 0.1$. The initial beliefs are set to $\forall k \in \{1, 2, 3\}$, $\mathbb{P}(\sigma^k = \text{leader}) = \mathbb{P}(\sigma^k = \text{follower}) = 0.5$. Figure 2.8 shows the results when the ego vehicle is interacting with other vehicles controlled by IDM with different target vehicles and different desired time headways.

In Figure 2.8(a), the first interacting vehicle (vehicle 1) intends to yield to the ego vehicle, so it chooses to follow the ego vehicle with 1 (s) time headway, while the last two interacting vehicles are following the front vehicles with 0.5 (s) headway.

Parameters	Values
Desired velocity v_0	32 m/s
Minimum spacing ϕ_0	2 m
Maximum acceleration a_m	4 m/s^2
Comfortable deceleration b	3 m/s^2
Acceleration exponent δ	4
Desired time headway T	0.5 to 2.5 s

Table 2.1: Intelligent driver model parameters.

From Figure 2.8(a-1), the ego vehicle thinks that vehicle 1 has a high probability of being a follower in the game and chooses to merge in front of vehicle 1 as depicted in Figure 2.8(a-2). Figure 2.8(b) shows another case where the first interacting vehicle (vehicle 1) follows the front vehicle with 0.5 (s) headway, and the second interacting vehicle intends to yield to the ego vehicle and follows the ego vehicle with 0.5 (s) headway. Then in this case, from the ego vehicle perspective, vehicle 1 has higher probability of being a leader in the game, while vehicle 2 has a higher probability of being a follower in the game, and hence the ego vehicle successfully merges in front of vehicle 2 in this case. Two other non-yield cases are shown in Figure 2.8(c) and (d). Figure 2.8(c) shows the results of all interacting vehicles following the front vehicle with 0.5 (s) headway. From the ego vehicle perspective, all interacting vehicles are more likely to be leaders in the game, so the ego vehicle successfully merges after all vehicles pass. In Figure 2.8(d), all interacting vehicles follow the front vehicle with 1.5 (s) headway. In this case, the ego vehicle finds that vehicle 2's behavior is conservative and thinks vehicle 2 has a higher probability of being a follower in the game. Hence, the ego vehicle successfully merges between vehicles 1 and 2. The average computational time for solving (2.25) at each time step is 0.198 (s).

For all cases shown in Figure 2.8, the ego vehicle starts with the same initial belief. This means that the ego vehicle does not know other drivers conservativeness (represented by desired time headway) and intentions (represented by target vehicles) a priori. The ego vehicle relies on the LFGC to estimate their intentions, make decisions accordingly and is able to merge successfully.

2.5.3 Interacting vehicles following traffic data

We have already tested the LFGC with other vehicles driven by leader/follower in the leader-follower game and by IDM models. We want to further test the controller's performance against real traffic data. Specifically, we use the US highway 101 traffic

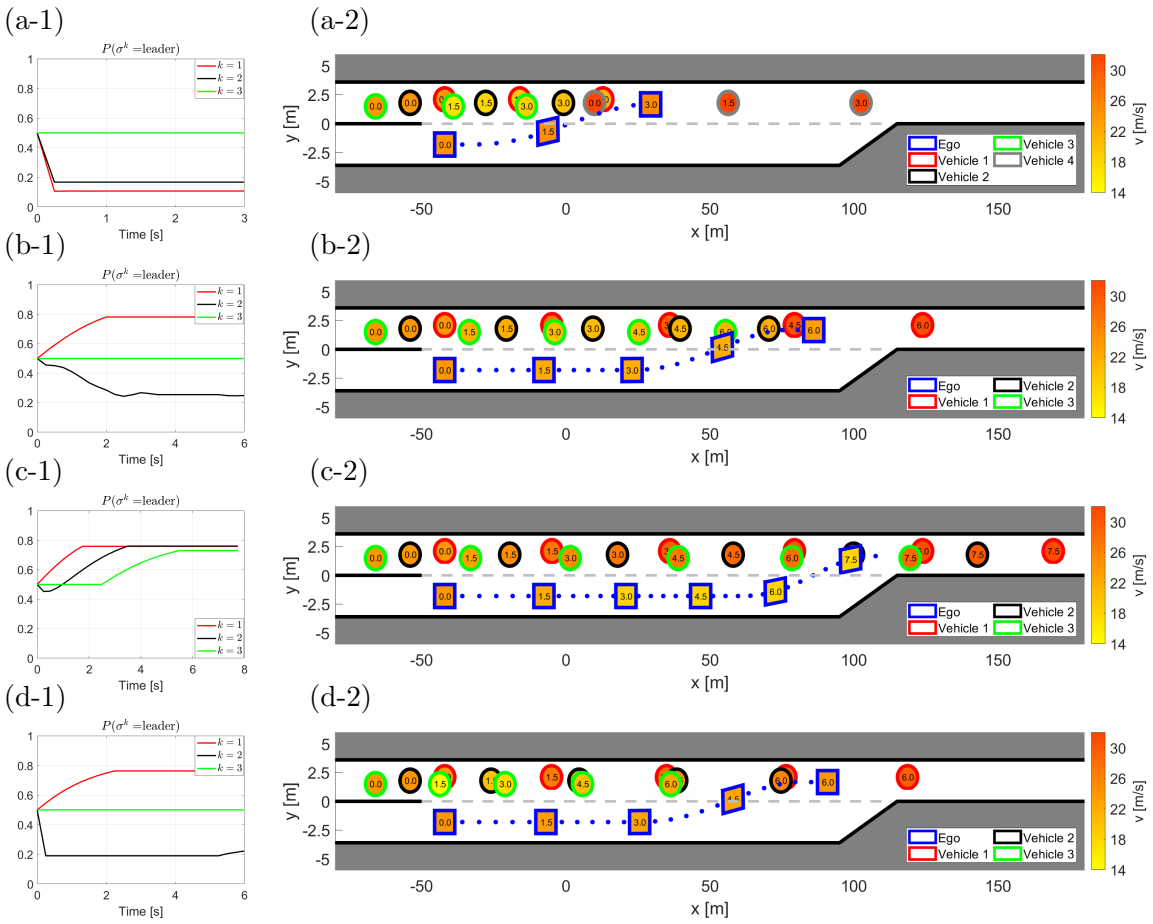

Figure 2.8: Results of the LFGC against other vehicles controlled by IDM with different target vehicles and desired time headways: (a) vehicle 1 yields (follows the ego vehicle) with time headway $T = 1$ (s) and vehicles 2 and 3 follow the front vehicle with $T = 0.5$ (s); (b) vehicle 2 yields (follows the ego vehicle) with $T = 0.5$ (s) and vehicles 1 and 3 follow the front vehicle with $T = 0.5$ (s); (c) all vehicles follow the front vehicle with $T = 0.5$ (s); (d) all vehicles follow front vehicle with $T = 1.5$ (s). The left column (a-1) to (d-1) shows the ego vehicle's belief on other vehicles' being a leader in the game, $\mathbb{P}(\sigma^k = \text{leader}), k = 1, 2, 3$. The right column (a-2) to (d-2) shows the time history results of the ego and other vehicles behaviors during this forced merge process. Specifically, in the right column, the boundary line color of each block distinguishes different vehicles, the number in the block represents time in seconds, the color of each block describes the speed of the vehicle at that time instant, and the blue dotted line represents the ego vehicle's trajectory. Note that vehicles 1-4 share the same vertical positions, and some offsets in vertical direction are added to better distinguish them in the figure. Note also that since vehicle 4 (grey boundary) drives at constant speed, the time history results of vehicle 4 is only shown in (a) but omitted in (b) to (d) for clarity.

dataset from the Next Generation Simulation (NGSIM) website [82], which is collected by the United States Federal Highway Administration and is considered as one of the largest publicly available sources of naturalistic driving data. The US highway 101 dataset has been extensively studied in the literature [119, 120, 121].

More specifically, we consider a portion of the US101 traffic dataset that contains 30 minutes of vehicles trajectories on the US101 highway. The time period ranges from 7:50 to 8:20 am, which represents the buildup of congestion around the morning peak hours. The dataset contains position and velocity trajectories as well as vehicle dimensions for around 6000 vehicles, and the information is recorded every 0.1 (s). The top view of the portion of the US101 highway that is used for collecting the data is shown in Figure 2.9. The studied section consists of five main lanes of the highway, one on-ramp to the highway, one off-ramp exiting the highway, and also one auxiliary lane that is used to merge into the highway and exit the highway.

As discussed in [122], the US101 dataset contains a significant amount of noise due to video analysis and numerical differentiation. To overcome this drawback, the Savitzky-Golay filter [123] is utilized to smooth vehicles' positions and update their corresponding velocities. The Savitzky-Golay filter performs well for signal differentiation and smoothing the US101 dataset with window length 21 [120]. One original vehicle trajectory and the corresponding smoothed vehicle trajectory are shown in Figure 2.10.

For the validation tests of the LFGC, we focus on the on-ramp and the auxiliary lane to identify all merging vehicles. After identifying the merging vehicles and the corresponding scenario, we identify the interacting vehicles according to Figure 2.11. Specifically, we consider the first vehicle in the target lane that is within 2 (s) time headway in front of the ego vehicle as the first interacting vehicle and regard the consecutive vehicles as the second and third vehicles. For all other vehicles present in the scenario, the ego vehicle will regard them as environmental vehicles and assume they drive at constant speed. One identified merging scenario is shown in Figure 2.12.

For each merging scenario, instead of letting the ego vehicle follow the traffic data, we use the LFGC to control ego vehicle's action and resulting trajectory. For all other vehicles, including interacting vehicles and environmental vehicles, they follow their original trajectories in the US101 traffic dataset. Then, the LFGC needs to estimate interacting vehicles' intentions and control the ego vehicle to merge appropriately. Interacting vehicles and environmental vehicles may interact with the merging vehicle in the actual traffic during the data collection. Note, however, that interacting vehicles' or environmental vehicles' behaviors are pre-determined by the traffic dataset

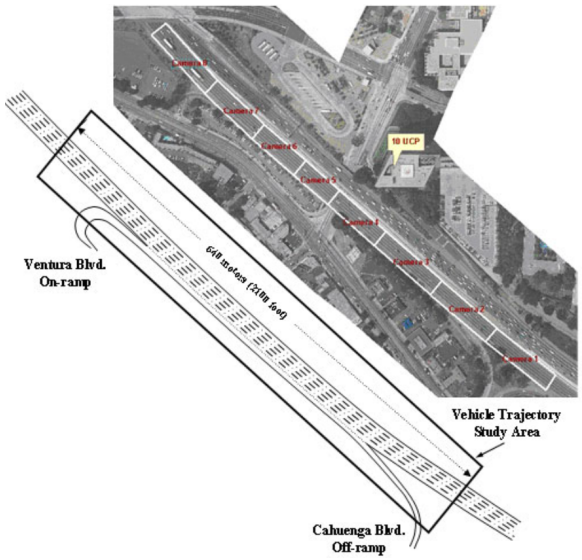

Figure 2.9: Top view of the highway that is used for collecting the US101 traffic data [82]. The section of interest includes five main lanes of the highway, one on-ramp to the highway, one off-ramp exiting the highway, and also one auxiliary lane that is used to merge into the highway and exit the highway.

and in this test do not change in response to ego vehicle's actions.

Time: 06/15/2015	7:50 to 8:05 am	8:05 to 8:20 am	Total
Number of Merges	119	79	198
Success	116	77	193
Fail to Merge	2	2	4
Collision	1	0	1
Success Rate	97.5%	97.5%	97.5%

Table 2.2: Statistics of validating the LFGC using the US101 traffic dataset. "Success" means the ego vehicle successfully merges to the target lane without any collisions. "Fail to Merge" means that the ego vehicle fails to merge by the end of the auxiliary lane. "Collide" means the ego vehicle collides with other vehicles.

Statistics of validating the LFGC based on the US101 traffic dataset are shown in Table 2.2. There are a total of 198 merge cases present in the dataset that happen from 7:50 to 8:20 am. The average computational time for solving (2.25) at each time step among all merge cases is 0.259 (s). In 193 merge cases, the LFGC successfully maneuvers the ego vehicle to merge to the target lane. The LFGC fails in 5 cases including 4 "Fail to Merge" cases and 1 "Collision" case. The reason for these failure cases is primarily due to either 1) the LFGC cannot obtain a high belief on interacting vehicles' driving intentions and hence needs to take conservative action to avoid col-

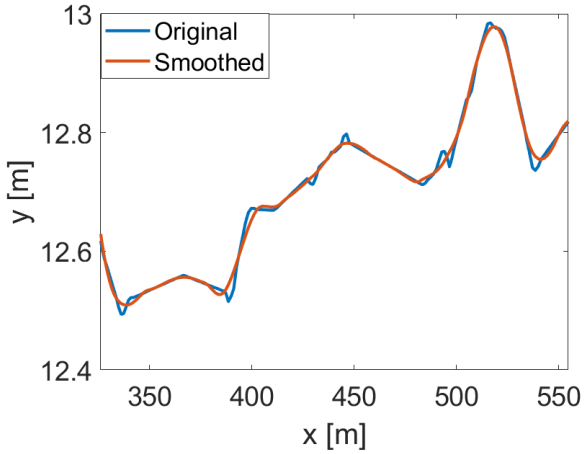

Figure 2.10: Smooth vehicle trajectories from the US101 traffic dataset using the Savitzky-Golay filter.

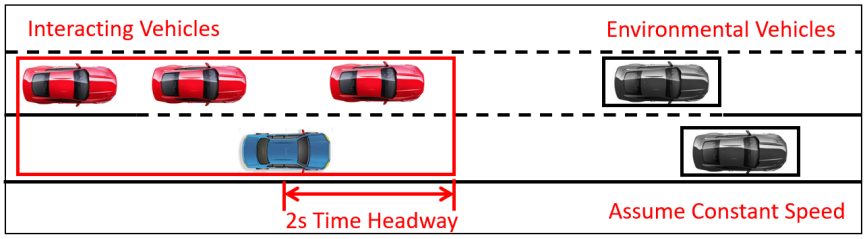

Figure 2.11: Selection of interacting vehicles: The ego vehicle (blue vehicle) considers vehicles inside the selection box (red box) as the interacting vehicles. The front end of the selection box is 2 (s) time headway in front of the ego vehicle. The first vehicle in the target lane within the selection box is regarded as the first interacting vehicle, and the following vehicles are regarded as second and third interacting vehicles. For all other vehicles on the highway, they are treated as environmental vehicles and are assumed to maintain a constant speed.

lision, or 2) the traffic is dense such that there is no safe margin for the ego vehicle to merge without intersecting with other vehicles' collision boxes. The "Collision" case is a case where the collision is caused by a vehicle behind rear-ending the ego vehicle when both vehicles are in the auxiliary lane. Note that in our validation method all surrounding vehicles are following their original trajectories in the dataset. In this collision case, the vehicle behind is not reacting to the autonomous ego vehicle that is driving in its front. This case does not represent a failure of our LFGC controller (because in reality the vehicle behind will adjust its speed to avoid rear-ending) but represents a limitation of our current validation method with traffic data, i.e., the traffic environment is not responsive to the ego vehicle's behavior. While the above highlights some limitations of the testing with recorded traffic data, such an approach

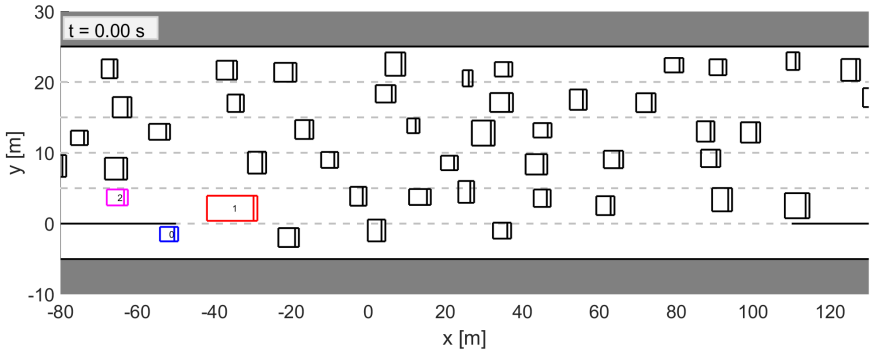

Figure 2.12: One merge scenario identified from the US101 traffic dataset. In this scenario, vehicle 0 (blue vehicle) is the merge vehicle, and we let the LFGC control vehicle 0. Based on our criteria of selecting interacting vehicles, vehicle 1 (red vehicle) and vehicle 2 (pink vehicle) are selected as the interacting vehicles, and all other vehicles (black vehicles) are regarded as environmental vehicles, which are assumed to drive at constant speed.

is common see, e.g., [53, 124], as it allows to compare the decisions with those of the human drivers.

In Figure 2.13, we present screenshots for one successful merge. In these figures, the blue vehicle is controlled by the LFGC, and the grey box represents the actual position of the ego vehicle in the dataset. All other vehicles (including red interacting vehicles and black environmental vehicles) are following their corresponding trajectories in the dataset. The ego vehicle controlled by the LFGC makes similar decisions compared to the human driver (grey box): Both the LFGC and the human driver try to speed up and merge in front of the truck (Vehicle 1) at first. However, after recognizing that the truck is more likely to proceed without yielding, the ego vehicle decides to slow down and merges after the truck. [1]

2.6 Summary

In this chapter, we proposed and presented a Leader-Follower Game Controller (LFGC) for autonomous vehicle planning and control with an application in forced merge scenarios. The LFGC treats interaction uncertainties due to different driver intentions as latent variables, estimates on-board other driver intentions, and chooses actions to facilitate the ego vehicle's merge. In particular, the LFGC is able to perform a receding horizon optimization subject to an explicit probabilistic safety char-

[1] Animations of LFGC validating against US Highway 101 dataset can be found at `https://youtu.be/_2Z9E57yTIQ`.

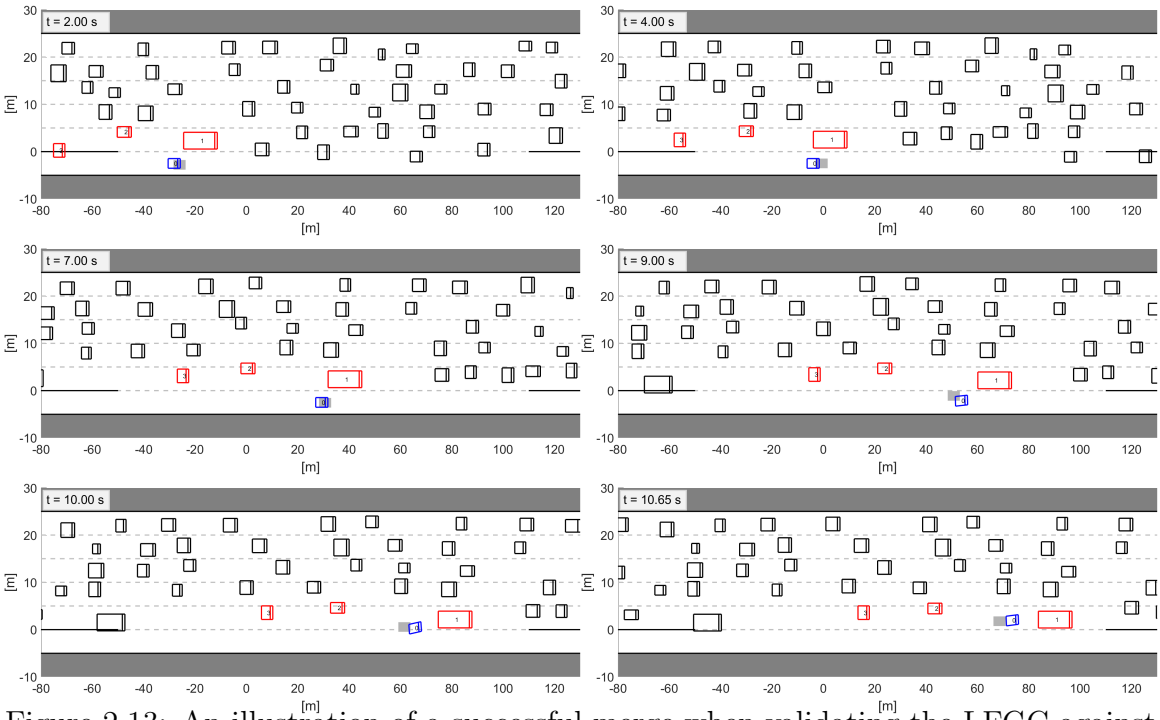

Figure 2.13: An illustration of a successful merge when validating the LFGC against the US Highway 101 dataset. The blue vehicle is the ego vehicle controlled by the LFGC, and the grey box is the position of the ego vehicle present in the dataset.

acterization i.e., subject to constraints representing vehicle safety requirements. By considering pairwise interactions of the ego vehicle and interacting vehicles, the LFGC is able to handle interactions with multiple vehicles in a computational tractable way. Finally, three sets of simulation-based validations are performed to demonstrate effectiveness of the LFGC, including scenarios that other vehicles are following leaders or followers in the game, the Intelligent Driver Model (IDM), and actual US Highway 101 data. In the first and second sets of tests, we considered three interacting vehicles demonstrating good performance against very dense traffic moving at high velocity. In the third set of tests, we considered evaluation against a real traffic dataset corresponding to fairly dense traffic.

CHAPTER III

Model-free Learning to Avoid Constraint Violations for Non-safety Critical Systems

As depicted in Figure 1.4, the behavior planner will generate certain trajectories or commands for the system to track. In order to further ensure safety, the behavior planner is followed by a safety supervisor, which serves to guard the system and potentially modify the planner's or human operator's commands to prevent safety issues. Safety does not only refer to avoiding collisions but can be broadly defined as not violating specified constraints. The emphasis of this chapter is non-safety critical systems or constraints, where constraint violations are not desirable but do not lead to severe consequences. This chapter introduces a model-free learning algorithm that over time modifies the parameters of an explicit reference governor (ERG) scheme so that violations of pre-specified constraints are avoided after a sufficiently informative learning phase. The ERG is an add-on scheme that modifies set-point commands to a nominal closed-loop system. Our learning algorithm modifies the ERG parameters based on observed constraint violations during a learning phase so as to eliminate constraint violations after learning is completed. Theoretical properties of the algorithm are analyzed and several examples that illustrate its effectiveness are presented.

3.1 Introduction

In recent years, autonomous systems, including autonomous vehicles, have seen significant advancements in technology and capabilities. These systems have the potential to revolutionize transportation and improve efficiency in a variety of industries. By removing the need for human intervention in many tasks, autonomous systems can increase productivity, reduce costs, and improve safety. As these systems may operate in uncertain environment and are subject to manufacturing variability,

aging, degradation and damage, the constraint boundaries are often uncertain and maneuvers that can cause constraint violations are often unknown a priori.

In this chapter, we propose an algorithm with the ability to learn to enforce state/output constraints. The proposed learning algorithm exploits only the knowledge of time instants at which constraints are violated to infer certain system properties and use them for enforcing constraints. In non-safety critical cases, where constraint violations are undesirable but do not lead to catastrophic consequences, the systems may operate initially with occasional constraint violations and learn over time to avoid them through reducing aggressive maneuvering by a necessary but minimum amount.

Specifically, we exploit the explicit reference governor (ERG) [125, 126, 127], which is an add-on scheme to a pre-stabilized nominal system that pre-filters commands (set-points) passed to the nominal controller to avoid potential constraint violations. Such an add-on scheme is appealing for integration with the U.S. military legacy systems as it does not need to change the existing/legacy controllers. The ERG is enhanced with learning functionality so that after learning is completed, the ERG is able to enforce constraints on a system whose model is unknown. Results characterizing convergence properties of the learning algorithm under suitable assumptions are presented, and it is shown that the algorithm can be applied to nonlinear systems with non-convex constraints.

To illustrate our learning algorithm functionality, three simulation examples are reported in this chapter. The first example represents control of a robotic arm on a moving platform with a constraint on the motor temperature, and is used to illustrate our results characterizing the algorithm's theoretical properties under given assumptions. The second example deals with velocity control and battery management of an electric vehicle with constraints on the battery temperature and the time rate of change in the battery state-of-charge. Finally, the third example concerns rollover avoidance for a ground vehicle. The second and third examples are used to illustrate the application of our algorithm to more complicated systems.

The contribution and novelties of the proposed approach in this chapter are as follows:

1. This chapter proposes a novel learning algorithm that is able to enforce system constraints and guarantee constraint satisfaction after a sufficiently informative learning phase. The ERG, which is leveraged by the learning algorithm, is an add-on scheme to the close-loop system. This algorithm learns from constraint violation occurrences and is suitable for non-safety critical systems.

2. Assumptions and supporting theories of the proposed algorithm are analyzed. In particular, theoretical guarantees of constraint enforcement, convergence of the proposed learning algorithm, feasibility of the learning algorithm, and asymptotic convergence of the modified reference (output of the ERG) to the original constant commands are provided.

3. This chapter demonstrates the effectiveness and broad applicability of the learning algorithm by considering applications to control of delivery robots, power management of electrical vehicles, and ground vehicle rollover avoidance.

3.2 Problem Formulation

In this chapter, we consider systems that are represented as

$$\dot{x}(t) = f\big(x(t), \nu(t)\big), \tag{3.1}$$

where $x(t) \in \mathbb{R}^n$ denotes the state at time $t \in [0, \infty)$, $\nu(t)$ denotes the reference input at t, taking values in a compact and convex set $V \subset \mathbb{R}^{n_\nu}$, and $f : \mathbb{R}^n \times \mathbb{R}^{n_\nu} \to \mathbb{R}^n$ is a nonlinear function. In applications, (3.1) typically represents a closed-loop system consisting of the plant being controlled and its nominal controller, while $\nu(t)$ defines the set-point.

Given an original command, $r(t) \in V$, provided by an operator or by a higher-level control algorithm (e.g., the interaction-aware control strategy introduced in Chapter II), our objective is to modify $r(t)$ to the reference input $\nu(t) \in V$ to enforce the pointwise-in-time constraints

$$x(t) \in X, \quad \forall\, t \in [0, \infty), \tag{3.2}$$

where $X \subset \mathbb{R}^n$ is closed with nonempty interior. We assume that the command signal r is piecewise continuous in t.

To solve this problem, we exploit the *explicit reference governor* (ERG) approach [126], which leads to a dynamic feedback law for $\nu(t)$ in the form of

$$\dot{\nu}(t) = \rho\big(\nu(t), r(t), x(t)\big), \tag{3.3}$$

where $\rho : \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n \to \mathbb{R}^{n_\nu}$ is to be designed.

The following assumptions are made:

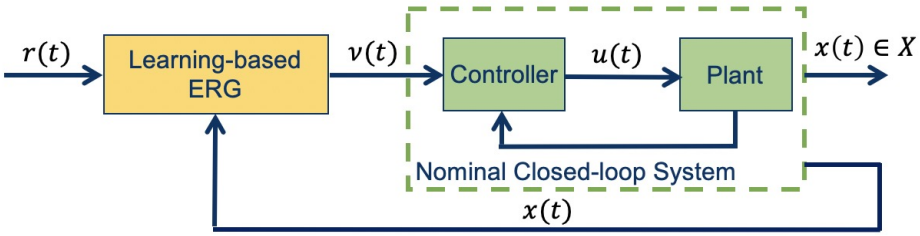

Figure 3.1: Diagram of a nominal closed-loop system augmented with a learning-based ERG to enforce constraints.

Assumption 3.1. The solutions of (3.1), $x : [0, \infty) \to \mathbb{R}^n$, to any initial conditions $x(0) = x_0 \in \mathbb{R}^n$ and any continuous signals $\nu : [0, \infty) \to V$ exist and are unique.

Assumption 3.1 is reasonable for practical systems.

Assumption 3.2. For any constant $\nu \in V$, the autonomous system

$$\dot{x}(t) = f\big(x(t), \nu\big), \quad x(0) = x_0, \tag{3.4}$$

has a unique equilibrium point $x_\nu = x_\nu(\nu)$, which is asymptotically stable globally in $x_0 \in \mathbb{R}^n$ and uniformly in $\nu \in V$. Specifically, 1) for any $\nu \in V$ and $\epsilon > 0$, there exists $\delta = \delta(\epsilon) > 0$ (independent of ν) such that $\|x_0 - x_\nu\| \leq \delta \implies \|x(t) - x_\nu\| \leq \epsilon$ for all $t \in [0, \infty)$, and 2) for any $\nu \in V$, $x_0 \in \mathbb{R}^n$, and $\epsilon > 0$, there exists $t_1 = t_1(\nu, x_0, \epsilon) \in [0, \infty)$ such that $\|x(t) - x_\nu\| \leq \epsilon$ for all $t \in [t_1, \infty)$, i.e., $\lim_{t\to\infty} x(t) = x_\nu$, where $x : [0, \infty) \to \mathbb{R}^n$ is the solution of (3.4).

Assumption 3.2 is reasonable when (3.1) represents a closed-loop system that has been pre-stabilized. The assumptions on uniqueness and global stability of x_ν can be relaxed by introducing additional constraints to let the feasible region X be contained in the domains of attraction of user-desired equilibria x_ν. To accomplish this, X may need to be replaced by $X(\nu)$ in (3.2).

Assumption 3.3. The equilibrium-point mapping $x_\nu : V \to \mathbb{R}^n$ is a continuous function of ν. Furthermore, $x_\nu(V) \subset \text{int}(X)$.

Assumption 3.3 implies that constraints are strictly satisfied in steady state. Thus, ERG is used to avoid transient constraint violations.

Assumption 3.4. There exists a pair of continuous functions (ε, Γ), $\varepsilon : \mathbb{R}^n \times V \to \mathbb{R}$ and $\Gamma : V \to \mathbb{R}$, such that 1)

$$\varepsilon(x_0, \nu) \leq \Gamma(\nu) \implies x(t) \in X, \quad \forall\, t \in [0, \infty), \tag{3.5}$$

where $x : [0, \infty) \to \mathbb{R}^n$ is the solution of (3.4) (with a constant ν), and 2) for any $\nu \in V$, there exists $\sigma = \sigma(\nu) > 0$,

$$\varepsilon(x_\nu, \nu) \leq \Gamma(\nu) - \sigma(\nu). \tag{3.6}$$

Assumption 3.4 is reasonable. For example, if $\varepsilon(\cdot, \nu)$ is a Lyapunov function for the system (3.4) and the equilibrium x_ν, and $\{x : \varepsilon(x, \nu) \leq \Gamma(\nu)\}$ represents the largest sublevel set of $\varepsilon(\cdot, \nu)$ that is contained in X, then, (3.5) is satisfied. Some other candidates for the pair (ε, Γ) are discussed in [126] and in Proposition 3 of this chapter. Indeed, the pair (ε, Γ) represents a way to define the so-called *dynamic safety margin* (DSM) defined in [126][1].

Furthermore, the following assumptions are made:

Assumption 3.5. The function $f : \mathbb{R}^n \times \mathbb{R}^{n_\nu} \to \mathbb{R}^n$ and the set X are unknown.

Assumption 3.6. The value of $\varepsilon\big(x(t), \nu(t)\big)$ and the value of $\mathbb{I}\big(x(t) \in X\big)$[2] are measured for all $t \in [0, \infty)$.

The assumption that X is unknown but $\mathbb{I}\big(x(t) \in X\big)$ can be measured is reasonable in many systems, such as systems that operate in changing environments where constraint boundaries can shift and sensors are available to detect constraint violations. For instance, in a gasoline engine, borderline spark values may change depending on fuel type (and other variables), while knock sensors can indicate constraint violations. Similarly, the onset of compressor surge can be detected from pressure oscillations, but the surge boundary may shift depending on the ambient conditions (such as icing or high angle of attack in gas turbine engines).

3.3 Learning Explicit Reference Governor

In this section, we will introduce the Explicit Reference Governor (ERG) and the proposed learning algorithm that evolves the design of the ERG through observed constraint violations. Theoretical properties of the ERG and the learning algorithm are also analyzed and provided in this section.

[1]Note that the extra requirement that $\varepsilon(x_0, \nu) = \Gamma(\nu) \implies \varepsilon(x(t), \nu) \leq \Gamma(\nu)$ for all $t \in [0, \infty)$ was imposed in the definition of DSM in [126]. However, as also shown by various examples of DSM given in the same chapter, this requirement is not strictly needed and can be dropped.

[2]indicator function, equal to 1 if the argument is true and 0 otherwise.

3.3.1 Enforcing constraints using ERG

Suppose that a pair of functions (ε, Γ) satisfying Assumption 3.4 is known. Then, we choose a dynamic feedback law for $\nu(t)$ as

$$\dot{\nu}(t) = \begin{cases} \frac{K_{\mathrm{p}}(r(t)-\nu(t))}{\max\{\|r(t)-\nu(t)\|,\eta\}}, & \text{if } \varepsilon\big(x(t),\nu(t)\big) < \Gamma\big(\nu(t)\big), \\ 0, & \text{otherwise}, \end{cases} \tag{3.7}$$

where $K_{\mathrm{p}} \in \mathbb{R}^{n_\nu \times n_\nu}$ is a diagonal matrix with (typically large) positive elements on the diagonal, and $\eta > 0$ is a (typically small) positive scalar.

We now discuss the properties of constraint enforcement and asymptotic convergence of $\nu(t)$ to steady-state constraint-admissible constant commands r_s for the dynamic feedback law (3.7). These properties are similar to the corresponding properties in [125] when Lyapunov functions are used and, with slight differences, are in line with the general theory on definition of DSM functions presented in [126].

Proposition 3.1. Under Assumptions 3.1, 3.4, and the feedback law (3.7), if $\varepsilon\big(x(0),\nu(0)\big) \leq \Gamma\big(\nu(0)\big)$, then $x(t) \in X$ for all $t \in [0,\infty)$.

Proof. Under the system dynamics (3.1) and the feedback law (3.7), x and ν are continuous in t. As $\varepsilon - \Gamma$ is continuous in x and ν, $\varphi(t) = \varepsilon(x(t),\nu(t)) - \Gamma(\nu(t))$ is continuous in t. Since $\varphi(0) \leq 0$, for any $t_1 \in [0,\infty)$, if $\varphi(t_1) > 0$, then by the intermediate value theorem, there exists $t_0 \in [0,t_1)$ such that $\varphi(t_0) = 0$ and $\varphi(t) > 0$ for $t \in (t_0,t_1]$. Then, by (3.7), $\nu(t) = \nu(t_0)$ for all $t \in [t_0,t_1]$. By (3.5), $x(t) \in X$ for all $t \in [t_0,t_1]$. Because $t_1 \in [0,\infty)$ is arbitrary, $x(t) \in X$ for all $t \in [0,\infty)$. ■

Proposition 3.2. Under Assumptions 3.1, 3.2, 3.4, and the feedback law (3.7), if there exists $t_s \in [0,\infty)$ such that $r(t) = r_s \in V$ for all $t \in [t_s,\infty)$, then $\lim_{t\to\infty} \nu(t) = r_s$.

Proof. Consider the function $\mathcal{V}(\nu) = \frac{1}{2}(\nu - r_s)^\top(\nu - r_s) \geq 0$. It holds that $\dot{\mathcal{V}}(\nu(t)) = (\nu(t) - r_s)^\top \dot{\nu}(t) \leq 0$ under the feedback law (3.7). Let $\Sigma = \{\nu \in V : \dot{\mathcal{V}}(\nu) = 0\}$ and Ω be the largest invariant set in Σ. By LaSalle's invariance principle, for arbitrary $\nu(t_s)$, $\lim_{t\to\infty} \mathrm{dist}(\nu(t),\Omega) = 0$. It remains to show that $\Omega = \{r_s\}$. Suppose that $\nu(t)$ is a trajectory contained completely in Ω. Then, at any t_0, either $\nu(t_0) = r_s$, or $\nu(t_0) \neq r_s$ and $\varepsilon\big(x(t_0),\nu(t_0)\big) \geq \Gamma\big(\nu(t_0)\big)$. For the former case, under (3.7), $\nu(t) = r_s$ for all $t \in [t_0,\infty)$. For the latter case, there must exist a time instant $t_1 \in (t_0,\infty)$ such that $\dot{\nu}(t_1) \neq 0$. This can be proven by contradiction. Indeed, suppose that $\dot{\nu}(t) = 0$ for all $t \in [t_0,\infty)$. Then, by Assumption 3.2, $\lim_{t\to\infty} x(t) = x_\nu(\nu(t_0))$, i.e., for any

$\delta > 0$, there exists $t_1 = t_1(\delta) \in [t_0, \infty)$ such that $\|x(t_1) - x_\nu(\nu(t_0))\| < \delta$. Since ε is continuous in x, given the $\sigma = \sigma(\nu(t_0)) > 0$ defined in (3.6), there exists $\delta = \delta(\sigma)$ such that $\|x(t_1) - x_\nu(\nu(t_0))\| < \delta \to \|\varepsilon\big(x(t_1), \nu(t_0)\big) - \varepsilon\big(x_\nu(\nu(t_0)), \nu(t_0)\big)\| < \sigma$. Then, by (3.6), $\varepsilon\big(x(t_1), \nu(t_0)\big) < \varepsilon\big(x_\nu(\nu(t_0)), \nu(t_0)\big) + \sigma \leq \Gamma\big(\nu(t_0)\big)$. Then, by the feedback law (3.7), if $\nu(t_0) \neq r_s$, then $\dot{\nu}(t_1) \neq 0$. This contradicts the assumption that $\nu(t)$ is a trajectory contained completely in Ω. Therefore, $\nu(t) = r_s$ is the only trajectory contained in Ω, i.e., $\Omega = \{r_s\}$. Please see [126] for more details. ■

Propositions 3.1 and 3.2 rely on Assumption 3.4. We now show that under Assumptions 3.1 to 3.3, there exist many candidates for the pair (ε, Γ) to satisfy Assumption 3.4.

Proposition 3.3. Suppose that Assumptions 3.1 to 3.3 hold and that 1) $y = g(x)$ is a measured output signal, where g is a continuous function; 2) for each $\nu \in V$, the steady-state output $y_\nu = g(x_\nu)$ is known; and 3) there exists a class-$\mathcal{K}$ function κ such that for any x and ν, $\kappa\big(\|x - x_\nu\|\big) \leq \|y - y_\nu\|$. Then, if we let $\varepsilon(x, \nu) = \|y - y_\nu\|$, there exists $\Gamma(\nu) = h(\nu)$, where h is a positive continuous function, such that (ε, Γ) is a pair of functions satisfying Assumption 3.4. Here, $\|\cdot\|$ can be an arbitrary norm. In the case of $y = x$, $\varepsilon(x, \nu)$ can be chosen as $\varepsilon(x, \nu) = \|x - x_\nu\|$.

Proof. First note that $\varepsilon(x, \nu) = \|y - y_\nu\| = \|g(x) - g\big(x_\nu(\nu)\big)\|$, where $g(x)$ and $x_\nu(\nu)$ are continuous functions $\implies$ $\varepsilon(x, \nu)$ is continuous in x and ν.

By Assumption 3.3, for any $\nu \in V$, $x_\nu \in \text{int}(X)$, where X is a closed set. Then, there exists $\epsilon = \epsilon(\nu) > 0$ such that $\|x - x_\nu\| \leq \epsilon \implies x \in X$. By Assumption 3.2, there exists $\delta = \delta\big(\epsilon(\nu)\big) > 0$ such that $\|x_0 - x_\nu\| \leq \delta \implies \|x(t) - x_\nu\| \leq \epsilon \implies x(t) \in X$ for all $t \in [0, \infty)$. Let $h(\nu)$ satisfy $0 < h(\nu) \leq \kappa(\delta)$. Firstly, κ is a class-$\mathcal{K}$ function and $\delta > 0 \implies \kappa(\delta) > 0$. Secondly, if $\varepsilon(x_0, \nu) = \|g(x_0) - y_\nu\| \leq h(\nu) \leq \kappa(\delta)$, then $\kappa\big(\|x_0 - x_\nu\|\big) \leq \|g(x_0) - y_\nu\| \leq \kappa(\delta) \implies \|x_0 - x_\nu\| \leq \delta \implies x(t) \in X$ for all $t \in [0, \infty)$.

It remains to show that there exists a continuous function $h(\nu)$ satisfying $0 < h(\nu) \leq \kappa\big(\delta(\nu)\big)$ for all $\nu \in V$. Since V is compact and $x_\nu(\nu)$ is continuous, $x_\nu(V)$ is compact, which is contained completely in $\text{int}(X)$. Then, there exists $\underline{\epsilon} > 0$ such that $x_\nu(V) \oplus \mathcal{B}(0, \underline{\epsilon}) \subset X$. Let $\epsilon(\nu) = \underline{\epsilon}$ and $h(\nu) = \kappa\big(\delta(\underline{\epsilon})\big)$ for all $\nu \in V$. Then, such a constant $h(\nu)$ is an example satisfying the needed properties, thus, verifies the existence of such functions.

Note that (3.6) holds since $\varepsilon(x_\nu, \nu) = \|y_\nu - y_\nu\| = 0$ and $\Gamma(\nu) = h(\nu) > 0$. ■

Proposition 3.3 shows that for a proper choice of ε, there exists a corresponding Γ such that (ε, Γ) satisfies Assumption 3.4. Such a Γ may not be unique. We denote

by $\Pi(\varepsilon)$ the set of functions Γ satisfying Assumption 3.4. Suppose that a $\Gamma \in \Pi(\varepsilon)$ is known, then the dynamic feedback law (3.7) can be used to adjust the reference input $\nu(t)$ to track the command $r(t)$ during which satisfaction of the constraints (3.2) is guaranteed and any steady-state constraint-admissible constant commands r_s can be asymptotically approached. We next describe an algorithm to learn a Γ function corresponding to a specified ε function under Assumptions 3.5 and 3.6.

3.3.2 Learning algorithm for enforcing constraints

The learning algorithm to find a Γ function corresponding to a specified ε function to eliminate constraint violations is conceptually described as follows: The algorithm starts with a sufficiently large guess of Γ so that ERG drives $\nu(t)$ to track $r(t)$ through a first-order filter without intervention (see (3.7)). Whenever a constraint violation is observed at a time instant τ, the algorithm modifies the corresponding $\Gamma(\nu(\tau))$ value, more specifically, reduces it by a proper amount, while keeping Γ satisfying certain functional properties. As a result, when a state-input pair in a neighbourhood of the present state-input pair $(x(\tau), \nu(\tau))$ occurs in future operation, ERG modifies the evolution of $\nu(t)$, more specifically, slows down the response of $\nu(t)$, which in turn slows down the response of $x(t)$, so that a constraint violation may be avoided. Such a procedure continues until the performance in terms of constraint enforcement is satisfactory. The learning algorithm described above is formally presented as Algorithm 3.1.

In Step 7 of Algorithm 3.1, the $\lambda^k > 0$ is a tunable parameter. Firstly, the choice of λ^k must ensure $\overline{\Gamma}^k\big(\nu(\tau_1)\big) > 0$. Secondly, its choice affects the learning speed and the resulting conservativeness in the response (the larger λ^k is, the faster the learning is to achieve constraint enforcement and the more conservative the resulting response may be).

Note that in Algorithm 3.1 the following properties hold: 1) $[\tau_3, \tau_4] \subset [\tau_1, \tau_2]$ under the feedback law (3.8); and 2) any $\Gamma \in \Pi(\varepsilon)$ satisfies

$$\Gamma\big(\nu(\tau_1)\big) < \min_{t\in[\tau_1,\tau_4]} \varepsilon\big(x(t), \nu(t)\big). \tag{3.10}$$

Proposition 3.4. In Algorithm 3.1, for any $t \in [0, \infty)$, there exists $t' \in [t, \infty)$ such that $\dot{\nu}(t') \neq 0$.

Proof. It suffices to consider the cases where $\dot{\nu}(t) = 0$. Suppose that $\dot{\nu}(t') = 0$ for all $t' \in [t, \infty)$, then $\nu(t') = \nu(t) \in V$ for all $t' \in [t, \infty)$. By Assumptions 3.2 and 3.3, $\lim_{t'\to\infty} x(t') = x_\nu\big(\nu(t)\big) \in \mathrm{int}(X)$. Then, there exists $t' \in [t, \infty)$ such that $x(t'') \in X$

Algorithm 3.1 Learning algorithm for enforcing constraints

1: Select ε in the form of $\varepsilon(x,\nu) = \|y - y_\nu\|$;
2: Initialize $\overline{\Gamma}^0(\nu) = \overline{\gamma}$ for all $\nu \in V$ with $\overline{\gamma} > 0$ sufficiently large, and select the parameters K_p and η;
3: Set $t_0 = 0$ and initialize the system (3.1) with an arbitrary initial condition $x(0)$;
4: **for** $k = 1 : k_{\max}$ **do**
5: Randomly generate $r_k \in V \setminus \{\nu(t_{k-1})\}$, e.g., based on a uniform distribution;
6: Use the dynamic feedback law

$$\dot{\nu}(t) = \begin{cases} 0, & \text{if } \varepsilon\big(x(t),\nu(t)\big) \geq \overline{\Gamma}^{k-1}\big(\nu(t)\big) \\ & \text{or } \mathbb{I}(x(t) \in X) = 0, \\ \frac{K_\mathrm{p}(r_k - \nu(t))}{\max\{\|r_k - \nu(t)\|, \eta\}}, & \text{otherwise,} \end{cases} \tag{3.8}$$

to adjust the reference input $\nu(t)$ over the time interval $[t_{k-1}, t_k]$ where $t_k \in (t_{k-1}, \infty)$ is sufficiently large;
7: If $\nu(t) = \nu(\tau_1)$ on $[\tau_1, \tau_2]$ and $\mathbb{I}(x(t) \in X) = 0$ on $[\tau_3, \tau_4] \subset [\tau_1, \tau_2]$, then update $\overline{\Gamma}^{k-1} \to \overline{\Gamma}^k$ so that

$$\overline{\Gamma}^k\big(\nu(\tau_1)\big) = \min\Big(\overline{\Gamma}^{k-1}\big(\nu(\tau_1)\big), \min_{t\in[\tau_1,\tau_4]} \varepsilon\big(x(t),\nu(t)\big)\Big) - \lambda^k, \tag{3.9}$$

and $\overline{\Gamma}^k$ is continuous in ν and satisfies $0 < \overline{\Gamma}^k(\nu) \leq \overline{\Gamma}^{k-1}(\nu)$ for all $\nu \in V$; otherwise let $\overline{\Gamma}^k = \overline{\Gamma}^{k-1}$.
8: **end for**

for all $t'' \in [t', \infty)$. As $y = g(x)$ where g is continuous, $\lim_{t'\to\infty} \|y(t') - y_\nu(\nu(t))\| = \lim_{t'\to\infty} \big\|g\big(x(t')\big) - g\big(x_\nu(\nu(t))\big)\big\| = 0$. Note that here $x_\nu\big(\nu(t)\big)$ is a constant. As $\overline{\Gamma}^{k-1}\big(\nu(t)\big) > 0$, there exists $t' \in [t, \infty)$ such that $\varepsilon\big(x(t''),\nu(t)\big) = \|y(t'') - y_\nu(\nu(t))\| < \overline{\Gamma}^{k-1}\big(\nu(t)\big)$ for all $t'' \in [t', \infty)$. Thus, there exists $t' \in [t, \infty)$ such that

$$\dot{\nu}(t') = \frac{K_\mathrm{p}(r_k - \nu(t'))}{\max\{\|r_k - \nu(t')\|, \eta\}} \neq 0. \tag{3.11}$$

Note that (3.11) follows from $r_k \neq \nu(t')$, which holds for all $t' \in [t, \infty)$ since 1) $r_k \neq \nu(t_{k-1})$ and 2) the finite-time convergence of ν to r_k cannot occur due to the $\eta > 0$ in the denominator of (3.11). Since (3.11) contradicts the assumption $\dot{\nu}(t') = 0$ for all $t' \in [t, \infty)$, the proof is completed. ■

Proposition 3.4 says that the reference input $\nu(t)$ getting stuck at a constant value for an infinitely long period of time so that learning cannot proceed will not occur.

Proposition 3.5. In Algorithm 3.1, as $k \to \infty$, $\overline{\Gamma}^k$ converges pointwise, i.e., for every $\nu \in V$, $\lim_{k\to\infty} \overline{\Gamma}^k(\nu)$ exists.

Proof. For every $\nu \in V$, the sequence $\left\{\overline{\Gamma}^k(\nu)\right\}_{k=0}^{\infty}$ is monotone non-increasing and is bounded by 0 from below. Thus, by the monotone convergence theorem, $\left\{\overline{\Gamma}^k(\nu)\right\}_{k=0}^{\infty}$ converges, i.e., $\overline{\Gamma}^k$ converges pointwise. ■

In practical implementations of Algorithm 3.1, a function approximator, such as a lookup table + interpolation or a neural network, can be used to represent the function $\overline{\Gamma}^k$ and get updated as k increases; a sufficiently small constant $\lambda^k \equiv \lambda > 0$ can be used; and the sequence $\{r_k\}_{k=1}^{k_{\max}}$ can also be generated using quasi-random, low-discrepancy sequences [128] to cover V more quickly and evenly. The algorithm is run for a sufficiently large but finite number of iterations until the constraints can be enforced satisfactorily.

3.4 Applications and Results

In this section, we consider several applications of the proposed learning algorithm. The first example is the constrained control of a delivery robot. This example satisfies the required assumptions and is able to demonstrate the desired properties of the learning algorithm. Subsequently, we consider more practical and complicated applications, which consist of power management of electric vehicles and vehicle rollover avoidance.

3.4.1 Constrained control of a delivery robot

As the first example, we consider a delivery robot as shown in Figure 3.2(a) [129]. The blue platform moves on a rail; the green robotic arm holds a mass – the arm adjusts the angle so that it can put the mass in a particular position after the platform arrives at a particular location. A motor is used to provide torque to the arm. The equations of motion of the system are given by:

$$\begin{aligned}
&\dot{s} = v, \\
&\dot{\theta} = \omega, \\
&(m_1 + m_2 + m_3)\dot{v} + (\frac{1}{2}m_2 + m_3)l\dot{\omega}\cos(\theta) - (\frac{1}{2}m_2 + m_3)l\omega^2\sin(\theta) = F, \\
&(\frac{1}{3}m_2 + m_3)l^2\dot{\omega} + (\frac{1}{2}m_2 + m_3)l\dot{v}\cos(\theta) + (\frac{1}{2}m_2 + m_3)gl\sin(\theta) = \tau,
\end{aligned} \tag{3.12}$$

where s and v are the horizontal position and velocity of the platform, θ and ω are the angle and the angular velocity of the delivery robot arm, m_1 is the mass of the platform, m_2 is the mass of the robot arm, m_3 is the mass of the load, l is the length

of the robot arm, F is the force applied to the platform along the position s-axis, and τ is the torque generated to move the robot arm. All these variables are shown in Figure 3.2(a).

Since the objectives of the system are to reach commanded position-and-angle pairs $(\overline{s}, \overline{\theta})$, we first design the following proportional-derivative controls:

$$\begin{aligned} F &= -k_{\rm p}^F(s-\bar{s}) - k_{\rm d}^F v, \\ \tau &= (\frac{1}{2}m_2 + m_3)gl\sin(\bar{\theta}) - k_{\rm p}^\tau(\theta-\bar{\theta}) - k_{\rm d}^\tau\omega, \end{aligned} \tag{3.13}$$

to stabilize the system.

A motor is used to provide the required torque. The following model is used to describe the motor temperature:

$$\begin{aligned} \tau &= IK_\tau, \\ \dot{T} &= -\alpha(T - T_{\rm amb}) + \beta I^2 R, \end{aligned} \tag{3.14}$$

where T is the temperature of the motor, K_τ is the motor torque constant, I is the current to generate the torque, $T_{\rm amb}$ is the ambient temperature, and I^2R is the heat generated by the current I and the motor resistance R.

We consider the following constraint on the motor temperature:

$$T \le T_{\max} = 65\,(^o\mathrm{C}). \tag{3.15}$$

We apply Algorithm 3.1 to evolve an ERG to adjust reference commands $r = (\overline{s}, \overline{\theta})$ to modified reference inputs $\nu = (s_{\rm rg}, \theta_{\rm rg})$ and pass modified reference inputs ν to the nominal controller (3.13) for enforcing the constraint (3.15).

In the implementation of Algorithm 3.1, we use the following ε function:

$$\varepsilon(x,\nu) = a_1|s-s_\nu| + a_2|v-v_\nu| + a_3|\theta-\theta_\nu| + a_4|\omega-\omega_\nu| + a_5|T-T_\nu|, \tag{3.16}$$

where $(s, v, \theta, \omega, T)$ are the system states, $(s_\nu, v_\nu, \theta_\nu, \omega_\nu, T_\nu)$ are their steady-state values corresponding to the current reference input ν, and $a_i > 0$, $i = 1, \cdots, 5$, are weights that can are tuned. In principle, the choices of a_i influence the learning speed and the eventual system performance. Their optimization is left to future research. We note that (3.16) satisfies the conditions of Proposition 3.3, and hence, a Γ satisfying Assumption 3.4 is guaranteed to exist.

The set for reference command $(\bar{s}, \bar{\theta})$ is $V = [0, 100]\,(\mathrm{m}) \times [-\pi/3, \pi/3]\,(\mathrm{rad})$, and

the duration of each reference command is randomly generated in the interval $[40, 60]$ (s).

We note that in the implementation of our algorithm, the system model, including (3.12), (3.13), and (3.14), is treated as a black-box.

The simulation results are reported in Figure 3.2. In Figure 3.2(b), the violation rate is defined as the proportion of time over a 1000 (s) time window during which the system is in a constraint-violation state. At the beginning of the learning, constraint violation occurs with a relatively high rate. The constraint violation rate gradually decreases as the learning progresses and converges to 0. Figure 3.2(c) shows the temperature response over time – the temperature is maintained below the constraint 65 (^{o}C) at the end of the learning. Figure 3.2(d) shows how the ERG modifies commands to constraint-admissible references for enforcing the constraints at the end of the learning. When there is no danger of constraint violation, the ERG passes the original command (red-dashed) to the nominal controller as the reference input (blue-solid) with a negligible delay. When there is a danger of constraint violation if the original command is directly passed to the nominal controller, the ERG slows down the change in the controller inputs and thus reduces the maneuver aggressiveness by a necessary but minimum amount to avoid a constraint violation. Furthermore, the modified reference input converges to the command, which implies a convergence of the response of the system augmented with the ERG to the desired response of the nominal system.

3.4.2 Electric vehicle velocity control and battery management

The second example we consider deals with the constrained velocity control and battery management for an electric vehicle (EV). The model of the EV including its battery power system is given as follows [130]:

$$\begin{aligned} \dot{T} &= \frac{I_{\text{bat}}^2 R_{\text{bat}} + \dot{Q}}{m_{\text{bat}} C_{\text{th,bat}}}, \\ \dot{SOC} &= -\frac{I_{\text{bat}}}{C_{\text{nom}}}, \\ \dot{v} &= a, \end{aligned} \tag{3.17}$$

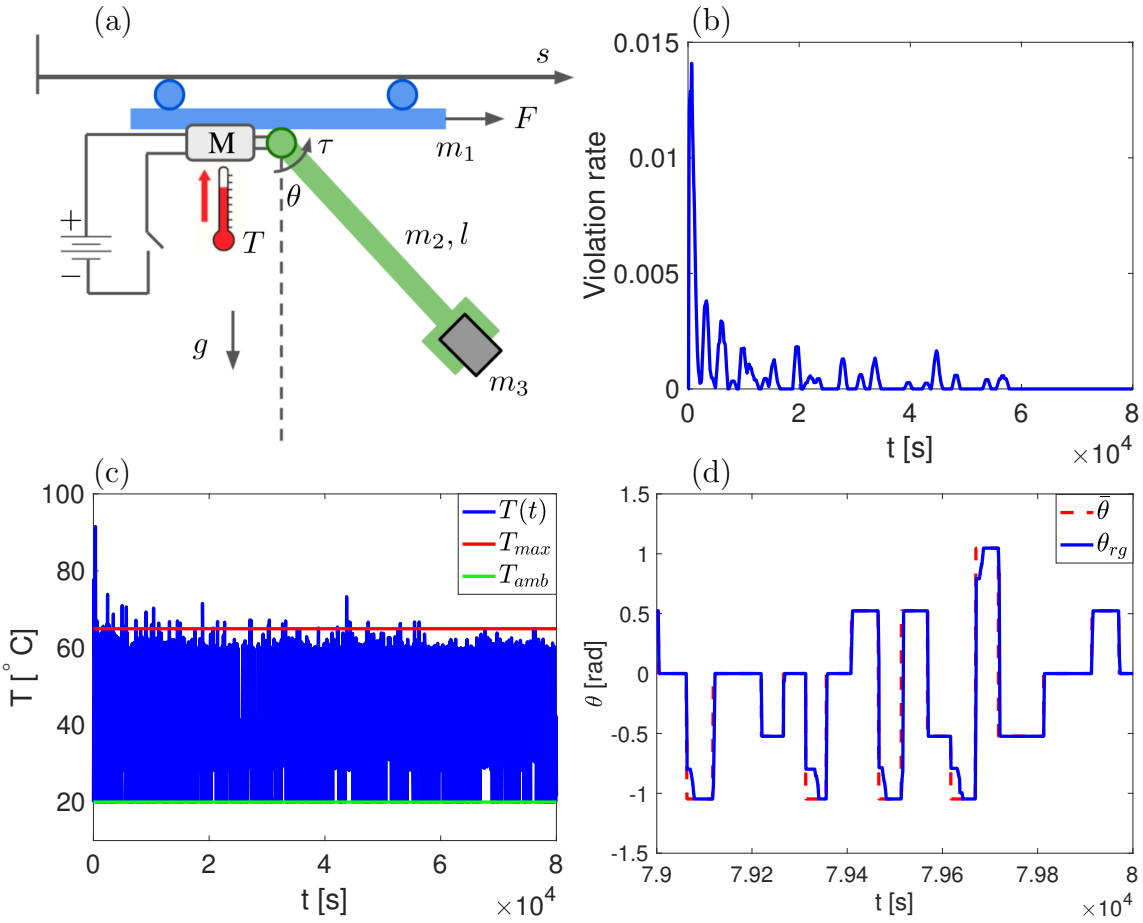

Figure 3.2: (a) Delivery robot diagram. (b) Violation rate profile during learning. (c) Temperature profile during learning. (d) Original command profile versus modified reference profile at the end of learning.

and

$$
\begin{aligned}
P_{\text{traction}} &= v\left(\frac{1}{2}\rho v^2 A_{\text{f}} C_{\text{d}} + C_{\text{f}}\, mg + ma\right), \\
P_{\text{temp}} &= \begin{cases} a_{\text{h}}\,|\dot{Q}|, & \text{if } \dot{Q} \geq 0, \\ a_{\text{c}}\,|\dot{Q}|, & \text{if } \dot{Q} < 0, \end{cases} \\
I_{\text{bat}} &= \frac{U_{\text{oc}} - \sqrt{U_{\text{oc}}^2 - 4R_{\text{bat}}(P_{\text{traction}} + P_{\text{temp}})}}{2R_{\text{bat}}},
\end{aligned}
\tag{3.18}
$$

where T is the battery temperature, SOC is the battery state-of-charge, v is the vehicle velocity, and the vehicle acceleration a and the heat flow rate $\dot{Q}$ are the control inputs. The P_{traction} is the required traction power to overcome the aerodynamic drag, the rolling resistance, and to provide the desired acceleration; P_{temp} is the power

employed to generate the heating ($\dot{Q} \geq 0$) or cooling ($\dot{Q} < 0$) for stabilizing the temperature of the battery and is assumed to be proportional to $|\dot{Q}|$; and I_{bat} is the current, which depends on the open circuit voltage U_{OC} and the internal resistance of the battery R_{bat}. Both $U_{\text{OC}} = U_{\text{OC}}(T, SOC)$ and $R_{\text{bat}} = R_{\text{bat}}(T, SOC)$ depend on the battery temperature T and state-of-charge SOC based on lookup tables.

Nominal controllers are first designed to control the vehicle acceleration a to track velocity commands, $\bar{v}$, and to control the heat flow $\dot{Q}$ to stabilize the temperature of the battery T to a reference temperature, T_{ref}. The nominal controllers are designed as follows:

$$\begin{aligned} a &= k_v(\bar{v} - v), \\ \dot{Q} &= k_{T,1}(T_{\text{ref}} - T) - k_{T,2}\bar{v}^2. \end{aligned} \tag{3.19}$$

We consider the following tight constraints on the battery temperature and on the time rate of change of SOC:

$$\begin{aligned} T &\in [24, 26]\,(^o\text{C}), \\ \dot{SOC} &\in [-1, 1] \times 10^{-4}. \end{aligned} \tag{3.20}$$

We apply Algorithm 3.1 to evolve an ERG to adjust velocity commands $\bar{v}$ to ν for enforcing the constraints (3.20). Some results as the learning progresses are shown in Figure 3.3. The constraints (3.20) are enforced after the learning is completed.

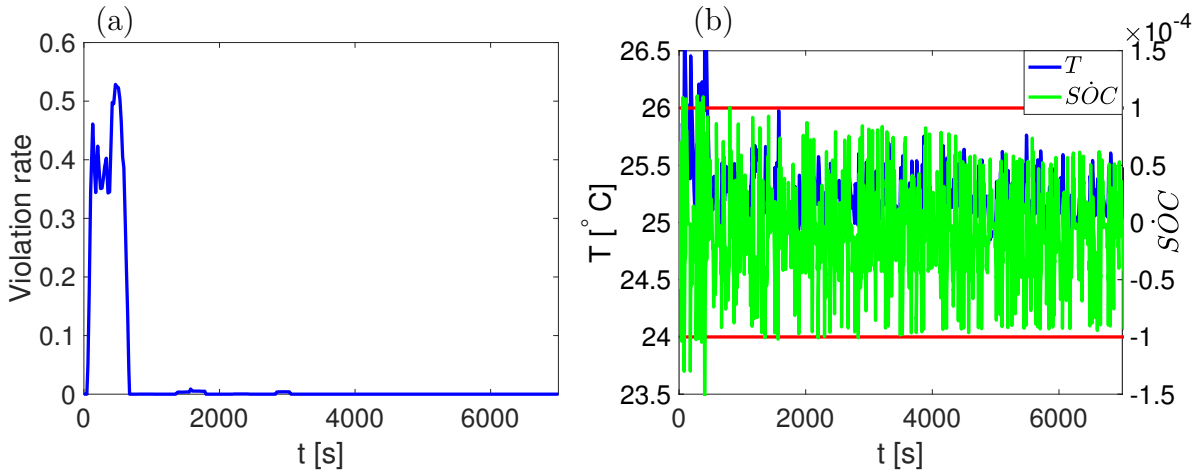

Figure 3.3: Learning algorithm application to electric vehicle velocity control and battery management. (a) Violation rate profile during learning. (b) Temperature and time rate of change of SOC profiles during learning.

The way the ERG modifies the velocity command and the performance of the

vehicle in tracking the velocity command at the beginning of the learning and at the end of the learning are shown in Figure 3.4. The red dashed lines represent the velocity commands. The green dash-dotted lines represent the ERG outputs, i.e., the reference velocities the nominal controller actually tracks. At the beginning of the learning, the ERG does not modify the command and, as a consequence, the constraints (3.20) are violated, which can be seen from the early phase of the violation rate, T, and $S\dot{O}C$ profiles during the learning in Figure 3.3. At the end of the learning, the ERG modifies the command profile to a constraint-admissible reference profile for the EV to track. The blue solid lines represent the velocity responses of the EV. Comparing the blue lines in Figure 3.4(a) and (b), the velocity response has been slightly slowed down after the learning – the average 2% settling time of tracking 1000 randomly generated velocity commands is 10.7 (s) at the beginning of the learning, and is 12.7 (s) at the end of the learning. With the small sacrifice in response speed, the constraints are enforced, which can be seen from the late phase of the violation rate, T, and $S\dot{O}C$ profiles during the learning in Figure 3.3.

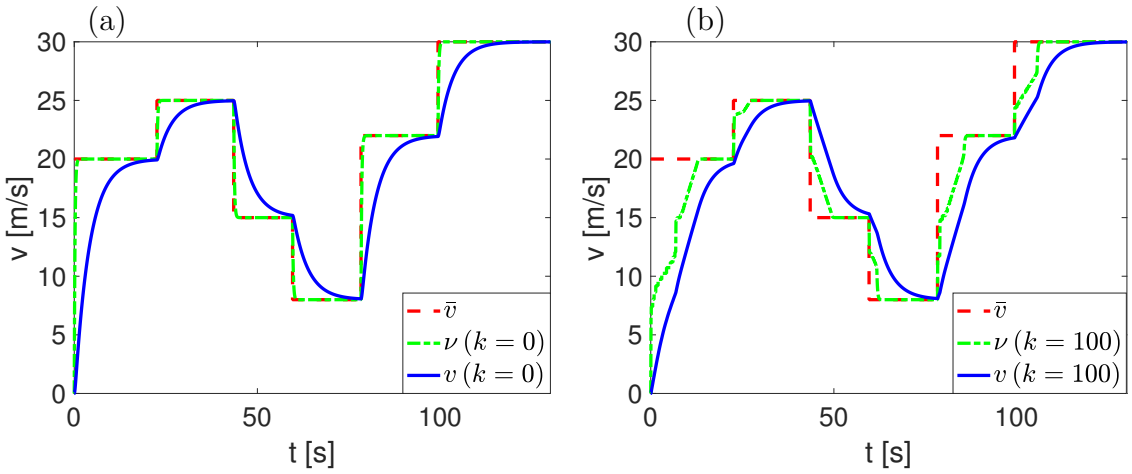

Figure 3.4: Reference and velocity responses of the vehicle, (a) at the beginning, and (b) at the end, of learning.

3.4.3 Vehicle rollover avoidance

At last, we consider the application of the scheme to ground vehicle rollover avoidance. A model-based design for rollover avoidance based on a reference governor scheme is presented in [131]. Differently from [131], the application of our scheme does not rely on an explicit model of the system. In this chapter, we use a model

in *CarSim* (Figure 3.5) to generate vehicle responses, and we treat this model as a black-box.

Figure 3.5: Utility truck *CarSim* model.

Specifically, we use the utility truck model in the standard *CarSim* vehicle configuration for this example. The scenario to be considered is that the truck is driving on a track at a constant speed of 80 (km/h). The steering wheel angle (SW) is the input command.

We apply Algorithm 3.1 to evolve an ERG to adjust SW commands for preventing the vehicle from rollover. In the implementation of Algorithm 3.1, we use the following ε function:

$$\varepsilon(x,\nu) = a_1|\phi - \phi_\nu| + a_2|p - p_\nu| + a_3|v - v_\nu| + a_4|\gamma - \gamma_\nu|, \tag{3.21}$$

where ϕ is the vehicle roll angle, p is the vehicle roll rate, v is the vehicle lateral velocity, γ is the vehicle yaw rate, and $(\phi_\nu, p_\nu, v_\nu, \gamma_\nu)$ are their steady-state values corresponding to the current reference input $\nu = \text{SW}$. The choice of this ε function is based on the reduced-order vehicle model in [131].

The rollover constraints are defined through the load transfer ratio (LTR):

$$\text{LTR} := \frac{F_{z,R} - F_{z,L}}{mg}, \tag{3.22}$$

where $F_{z,R}$ and $F_{z,L}$ are, respectively, the total vertical force on the right-side tires and that on the left-side tires, mg is the vehicle weight. The LTR measures how much of the vehicle vertical load is concentrated on one side of the vehicle. Based on the

LTR, the rollover constraints are imposed as

$$-\text{LTR}_{\text{lim}} \leq \text{LTR} \leq \text{LTR}_{\text{lim}}. \tag{3.23}$$

Remark 3.1. The absolute value of LTR can be larger than 1 even when the wheels do not lift off, which is due to the suspension roll moment and aerodynamic forces. Also, even if wheels lifting off occurs, it does not imply rollover, because vehicle rollover requires additional work to move the gravity center of the vehicle up. Thus, in this application, the constraint LTR_{lim} is tuned to a value that best describes the situation of rollover. Specially, we use $\text{LTR}_{\text{lim}} = 1.35$.

For the learning phase, the SW command is randomly generated from the set $V = [-300, 300]$ (deg), and the duration of each reference command is randomly distributed over the interval $[10, 20]$ (s).

Some results as the learning progresses are shown in Figure 3.6. The violation rate is defined similarly to that in Figure 3.2(b). The rate decreases and converges to 0. The LTR is maintained in the range $[-\text{LTR}_{\text{lim}}, \text{LTR}_{\text{lim}}]$ after the learning is completed.

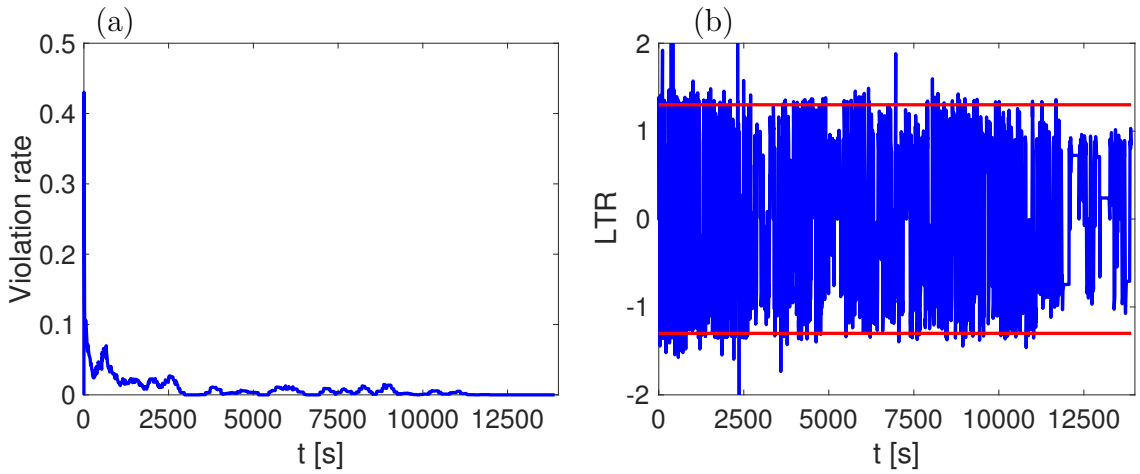

Figure 3.6: Learning algorithm application to vehicle rollover avoidance. (a) Violation rate profile during learning. (b) LTR profile during learning.

We test the vehicle using a standard sine-and-dwell test profile [132] at different phases of the learning. The corresponding reference and vehicle responses are shown in Figure 3.7 and Figure 3.8. The simulations terminate if the vehicle rolls over, which is also reflected from the curves for $k = 0, 300, 600$.

The original sine-and-dwell test profile causes the vehicle to roll over if directly

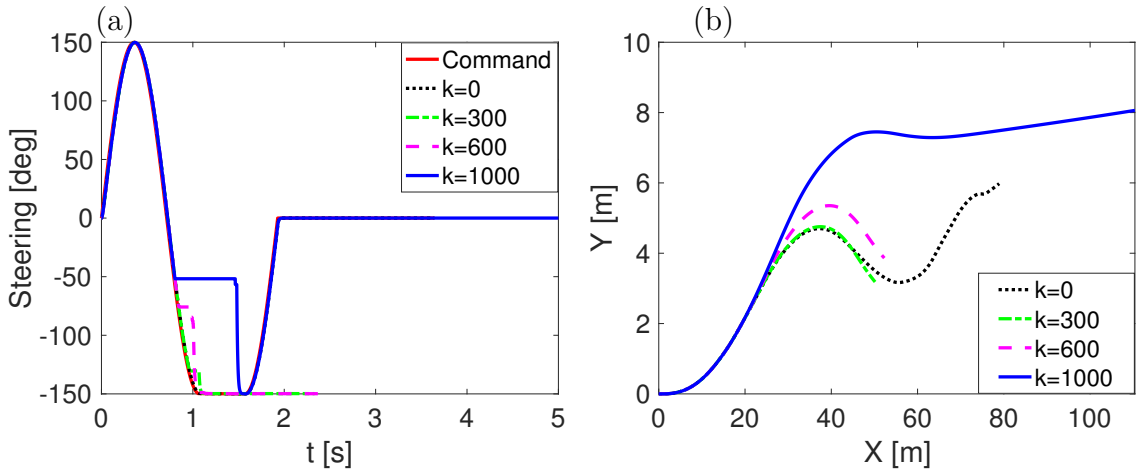

Figure 3.7: (a) The SW trajectories and (b) the vehicle trajectories on the (x, y)-plane for the sine-and-dwell test corresponding to different phases of learning.

passed to the vehicle as steering wheel command. However, after the learning is completed, the ERG modifies the steering wheel command and prevents the vehicle from rolling over.

3.5 Summary

In this chapter, we proposed an algorithm to avoid constraint violations based on an explicit reference governor scheme and learning. The operation of the algorithm does not rely on an explicit model of the system. Several properties, including constraint enforcement, asymptotic convergence of modified reference to steady-state constraint-admissible constant command, and convergence of the learning were discussed. Three examples including constrained control of a delivery robot, velocity control and battery management of an electric vehicle, and application to vehicle rollover avoidance were reported to illustrate the algorithm.

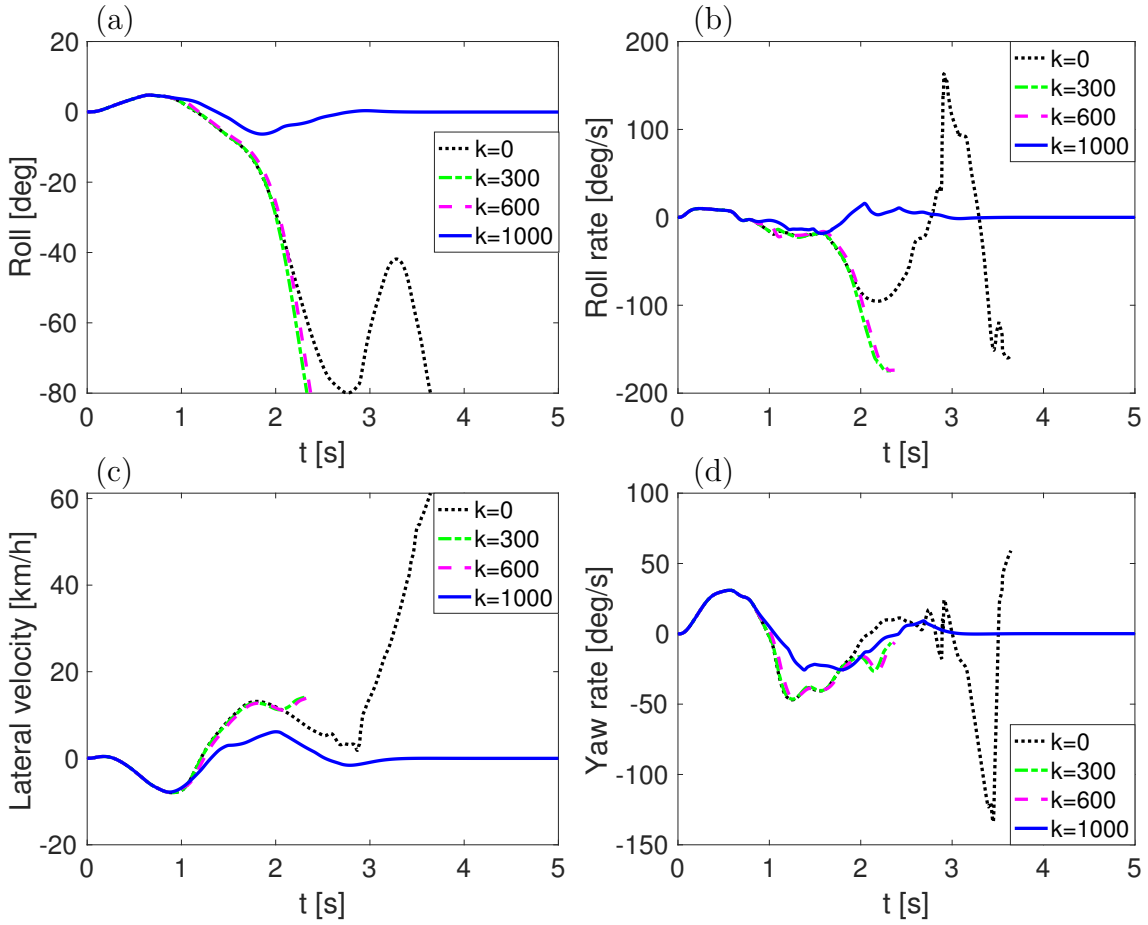

Figure 3.8: (a) The roll angle, (b) the roll rate, (c) the lateral velocity, and (d) the yaw rate responses for the sine-and-dwell test corresponding to different phases of learning.

CHAPTER IV

Mode-free Learning to Avoid Constraint Violations for Safety Critical Systems

In Chapter III, we presented a model-free learning algorithm to design a safety supervisor for non-safety critical systems, where constraint violations are allowed during the learning phase. However, in safety-critical control systems, constraint violations can have catastrophic consequences, and the method developed in Chapter III may not be applicable.

This chapter presents a learning-based approach to design a safety supervisor for the safety-critical control systems, which utilizes standard reference governor instead of the Explicit Reference Governor as in Chapter III. A reference governor is an add-on scheme used to guard the nominal system against violation of pre-specified constraints by modifying set-point commands. A learning algorithm is developed in this chapter to evolve the reference governor parametrization to gradually improve its performance in terms of response speed. In particular, the learning algorithm does not rely on an explicit model of the control system, i.e., it is model-free, and guarantees constraint satisfaction for all time, both during and after learning. To illustrate its functionality and characteristics, the approach is applied to case studies of ground vehicle rollover avoidance and fuel truck (tank truck) rollover avoidance under sloshing effects.

4.1 Introduction

Safety-critical systems are common in various areas, including transportation control, aerospace applications, nuclear plants, medical applications, etc. The safety conditions can often be expressed as pointwise-in-time state and control constraints, such as actuator range and rate limits, thermal and power limits, safety and comfort limits,

as well as obstacle avoidance requirements. In these applications, the controller needs to enforce constraints during the system operation, as constraint violation can lead to catastrophic consequences.

The Reference Governor (RG) is an add-on scheme to a nominal closed-loop system, acting as a pre-filter to modify set-point commands to guard the system against potential constraint violations [133]. Such an add-on scheme is suitable for integration with legacy systems as it does not need to change the existing/legacy controllers. Typical RG designs are model-based. For the case when the model is unknown, a model-free learning-based approach to enforcing state/output constraints based on an RG scheme has been proposed in Chapter III for non-safety-critical control systems, where constraint violations are undesirable but do not lead to catastrophic consequences. The learning algorithm proposed in Chapter III evolves an Explicit Reference Governor (one variant of RG) based on observed constraint violations during the learning phase and gradually eliminates these violations.

In this chapter, we focus on the learning reference governor (LRG) design for safety-critical systems, where constraints need to be enforced both during and after learning. As an application of LRG, we focus on the steering control of ground vehicles and tank trucks that are used in transporting chemical and petroleum products. Accidents associated with these ground vehicles and tank trucks may lead to severe injury and property damage. According to Federal Motor Carrier Safety Administration, rollover involves as the first harmful event in 4% of all fatal crashes associated with large trucks [134]. Since these trucks are partially filled most of the time, liquid sloshing is the main cause of the rollover accidents [135], [136] as the free space in the partially-filled tank allows liquid sloshing to happen when the vehicle state changes, and the truck's driving stability is severely affected due to the sloshing force. Although many techniques have been pursued to reduce the effect of sloshing, such as placing baffles inside the tank, these methods cannot cancel the sloshing effects. Consequently, we focus on solutions that minimally modify vehicle steering for vehicles with active front steering (including automated vehicles) if it becomes necessary to avoid vehicle rollover. Model-based reference governor solutions for rollover protection have been proposed in [131]. As accurate and suitable for online use models of fuel sloshing effects are presently unavailable, we consider the application of LRG, which does not need an accurate model.

In spacecraft applications, a pendulum-mass analogy [137] or a spring-mass analogy [138] have been considered to capture sloshing modes. A trammel pendulum model is proposed in [139] to study the driving stability of tank trucks under sloshing

effects. We choose the latter approach to establish a model for simulations of LRG, which itself does not require an accurate model.

The contributions of this chapter and the proposed safe LRG are as follows:

1. We present a novel safe LRG algorithm that integrates safe learning with the RG framework to achieve constraint management of safety-critical systems. This safe LRG algorithm is different from the non-safety-critical algorithm proposed in Chapter III. The proposed safe LRG algorithm relies on minimal prior knowledge of the system, where the required prior knowledge can be obtained by running a series of experiments before learning. Then, such LRG can be used to perform learning on systems' hardware or their black box models.

2. We conduct a thorough analysis of the properties of the proposed algorithm and clearly outline the required assumptions. At the same time, we provide theoretical guarantees of constraint enforcement during learning and after learning is completed. We also provide theoretical guarantees of convergence of the proposed learning algorithm, and finite time convergence of the modified reference (output of LRG) to the original constant commands.

3. We demonstrate the effectiveness of the algorithm through a case study of ground vehicle rollover avoidance. We first apply the algorithm to a linearized vehicle model where all of the assumptions (and hence propositions) hold. Then, the LRG is integrated with *CarSim* and is demonstrated to be effective on protecting the ground vehicles from rollover accidents while assuming minimal prior knowledge of the vehicle.

4. We demonstrate an application of LRG to fuel truck rollover avoidance under sloshing effects. An equivalent mechanical model for lateral fuel sloshing based on [140] is first described, and a vehicle dynamics model is integrated with the lateral fuel sloshing model for the simulation study. Simulation results are reported which illustrate the learning process and vehicle responses after learning for step commands, sine-and-dwell tests and when driving conditions change.

The notations used in this chapter are standard. In particular, for a right-continuous signal $\theta : [0, \infty) \to \Theta$, we use $\theta(t^-)$ to denote the left-sided limit $\lim_{t' \nearrow t} \theta(t')$, and for a better distinction, we also use $\theta(t^+)$ to denote $\theta(t)$.

4.2 Problem Formulation

In this chapter, we consider a stable (or a pre-stabilized) system, the dynamics of which can be represented by the following equations:

$$\dot{x}(t) = f\big(x(t), \nu(t)\big), \tag{4.1a}$$
$$y(t) = g\big(x(t), \nu(t)\big), \tag{4.1b}$$

where $x(t) \in \mathbb{R}^n$ denotes the system state at time $t \in [0, \infty)$, $y(t) \in \mathbb{R}^m$ denotes the system output, $\nu(t)$ denotes the reference input, taking values in a compact and convex set $V \subset \mathbb{R}^{n_\nu}$ and determining the set-point of the system, and $f : \mathbb{R}^n \times \mathbb{R}^{n_\nu} \to \mathbb{R}^n$ and $g : \mathbb{R}^n \times \mathbb{R}^{n_\nu} \to \mathbb{R}^m$ are nonlinear functions. We note that such a pre-stabilized system typically consists of a plant to be controlled and a nominal controller, as illustrated in Figure 4.1. Also note that we assume the system dynamics (4.1) (i.e., f and g) do not change over time.

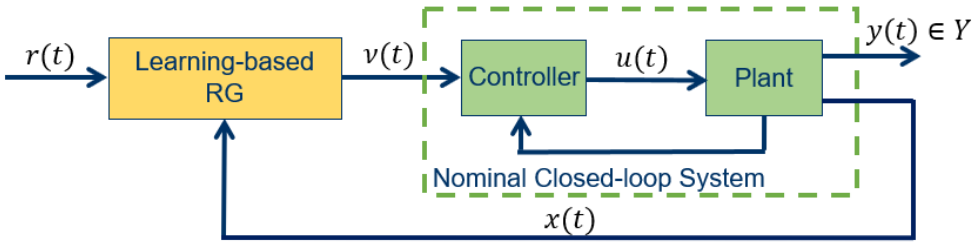

Figure 4.1: Diagram of a nominal closed-loop system augmented with a learning-based reference governor for handling constraints.

We assume that the system must operate without violating a prescribed set of specifications, which are represented as pointwise-in-time constraints on the outputs of the form:

$$y(t) \in Y, \quad \forall t \in [0, \infty), \tag{4.2}$$

where $Y \subset \mathbb{R}^m$ is a closed set with a nonempty interior.

We make the following assumptions:

Assumption 4.1. For any initial condition $x(0) = x_0 \in \mathbb{R}^n$ and piecewise continuous reference input signal $\nu : [0, \infty) \to V$, the solution to the differential equation (4.1a), $x : [0, \infty) \to \mathbb{R}^n$, exists and is unique.

Assumption 4.2. For any constant reference input $\nu \in V$, the autonomous system,

$$\dot{x}(t) = f\big(x(t), \nu\big), \tag{4.3}$$

has a unique equilibrium point, $x_\nu = x_\nu(\nu)$, which is globally asymptotically stable (GAS).

We note that since (4.1) represents a system that has been stabilized by a nominal/legacy controller (see Figure 4.1), Assumptions 4.1 and 4.2 are reasonable.

Assumption 4.3. The steady-state mapping in *(A2)*, $x_\nu : V \to \mathbb{R}^n$, and the output function in (4.1b), $g : \mathbb{R}^n \times \mathbb{R}^{n_\nu} \to \mathbb{R}^m$, are continuous functions.

The above Assumptions 4.1 to 4.3 characterize the class of systems to be treated. Although the system (4.1) is stable (or pre-stabilized), the nominal controller may not have the ability to enforce the imposed constraints (4.2). In addition, for many practical systems, the functions f and g in their corresponding models (4.1) can be highly complex or not given explicitly (e.g., when the model is given as a black-box simulation code). Moreover, in some circumstances, the system may not even have an accurate model, as is the case for a vehicle that has undergone in-field modifications. Therefore, the proposed LRG scheme relies on learning rather than explicit knowledge of f and g to achieve constraint enforcement. For situations where the system does not have a model, the learning process may be performed directly on the hardware. In such a case, it can be important to ensure constraint satisfaction even during the learning process, especially for safety-critical systems, in which constraint violations may cause catastrophic consequences such as damage to the hardware or to the human operator. The proposed control scheme achieves such a safe learning based on Assumptions 4.4 to 4.6 stated below.

Firstly, let $\psi(\cdot, x_0, \nu) : [0, \infty) \to \mathbb{R}^n$ denote the solution to (4.3) corresponding to the initial condition $x(0) = x_0$ and constant reference input $\nu \in V$, and let $\phi(\cdot, x_0, \nu) = g\big(\psi(\cdot, x_0, \nu), \nu\big)$ denote the corresponding output trajectory y. Then, let $D : \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n \to \mathbb{R}$ be defined as

$$D(\nu, \delta\nu, \delta x) := \sup_{t \in [0,\infty)} \big\|\phi\big(t, x_\nu(\nu) + \delta x, \nu + \delta\nu\big) - y_\nu(\nu)\big\|, \tag{4.4}$$

where $y_\nu(\nu) = g\big(x_\nu(\nu), \nu\big)$ denotes the steady-state output corresponding to the reference ν, and $\|\cdot\| = \|\cdot\|_{\mathbb{R}^m}$ denotes an arbitrary vector norm on $\mathbb{R}^m$.

The above function D represents the maximum deviation of the output trajectory y from the steady-state output $y_\nu(\nu)$ when the initial condition $x(0)$ is deviated from

the equilibrium $x_\nu(\nu)$ by δx and the reference input $\nu(t)$ is deviated from ν by a constant $\delta\nu$. Note that according to its definition, $D(\nu, 0, 0) = 0$ for all $\nu \in \mathbb{R}^{n_\nu}$.

This function D will be exploited in the proposed control scheme for constraint enforcement. However, in general, there does not exist an explicit formula for D, even when the functions f and g in the model (4.1) are explicitly known. Therefore, we employ a data-driven approach to learn/estimate D. In particular, as discussed above, constraint satisfaction should be ensured during the process of data collection and learning. For this, we rely on the following assumptions on the function D:

Assumption 4.4. There is a known pair of constants $L > 0, \beta \geq 1$ such that for any $z_1, z_2 \in \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$, it holds that

$$\big|D(z_1) - D(z_2)\big| \leq L \, \|z_1 - z_2\|^{\frac{1}{\beta}}, \tag{4.5}$$

where $\|\cdot\| = \|\cdot\|_{\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n}$ denotes an arbitrary vector norm on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$. Note that this Hölder continuity assumption is a weaker and more general assumption than the Lipschitz continuity assumption relied upon in our preliminary works [86] and [88], because when $\beta = 1$, (4.5) reduces to the Lipschitz continuity condition.

The above Hölder continuity assumption for D is reasonable. For instance, in Lemmas A.1 and A.2 in the Appendix A, we show that this assumption holds true for all asymptotically stable linear time-invariant (LTI) systems, where we also provide formulas to compute estimates of the constants L and β in (4.5) when the LTI system has an explicit model. For nonlinear systems with explicit models, the technique in [141], which derives an explicit bound on the system responses to deviations in the initial condition and reference input using the logarithmic norms, can be used to check Assumption 4.4 and estimate L and β. In practice, the constants L and β may also be estimated based on engineering insight or by a data-driven approach using sampled trajectories, in both cases avoiding the need for explicit knowledge of the functions f and g in (4.1). On the one hand, estimating constants L and β can be much easier than identifying f and g globally from data. On the other hand, in principle, for a fixed β, the estimate of L can be arbitrarily conservative as long as it is a finite number. Note, however, that a more conservative estimate of L may result in a slower learning rate; this will be further shown in our simulation case study in Section 4.6.

We next make the following assumption:

Assumption 4.5. At each time instant $t \in [0, \infty)$, the state $x(t)$, the output $y(t)$, and the distance from the steady-state output, $y_\nu\big(\nu(t)\big)$, associated with the current

reference input, $\nu(t)$, to the constraint boundary,

$$d\big(\nu(t)\big) := \operatorname{dist}\big(y_\nu\big(\nu(t)\big), Y^C\big) = \inf_{y\in Y^C} \big\|y_\nu\big(\nu(t)\big) - y\big\|, \tag{4.6}$$

can all be measured. In (4.6), Y^C denotes the complement of the constraint set Y, i.e., $Y^C = \mathbb{R}^m \backslash Y$.

We note that measuring $d\big(\nu(t)\big)$ requires knowledge of the steady-state output mapping $y_\nu(\cdot) = g\big(x_\nu(\cdot), \cdot\big) : V \to \mathbb{R}^m$. When f and g are not explicitly known, the mapping y_ν, or the distance mapping $d(\cdot) = \operatorname{dist}\big(y_\nu(\cdot), Y^C\big) : V \to \mathbb{R}$, can be estimated using data of preliminary steady-state experiments with the system, and be function-fitted or stored as a look-up table for online use, as done in [142].

Assumption 4.6. There is a known pair (T, ε), with $\varepsilon > 0$ being sufficiently small, such that for any $(\nu, \delta\nu, \delta x) \in \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$ satisfying $\nu, \nu + \delta\nu \in V$, it holds that

$$\tilde{D}(\nu, \delta\nu, \delta x) := \max_{t\in[0,T]} \big\|\phi\big(t, x_\nu(\nu) + \delta x, \nu + \delta\nu\big) - y_\nu(\nu)\big\| + \varepsilon \geq D(\nu, \delta\nu, \delta x). \tag{4.7}$$

Assumption 4.6 is reasonable and ensures that the value of $D(\nu, \delta\nu, \delta x)$, which is defined in (4.4) as a supremum over an infinite interval, can be estimated using trajectory data of finite length (of length T) with high accuracy (with error bounded by ε). Note that according to their definitions in (4.4) and (4.7), $D(\nu, \delta\nu, \delta x)$ and $\tilde{D}(\nu, \delta\nu, \delta x)$ also satisfy $D(\nu, \delta\nu, \delta x) + \varepsilon \geq \tilde{D}(\nu, \delta\nu, \delta x)$. Therefore, we have $D \leq \tilde{D} \leq D + \varepsilon$ pointwise on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$.

4.3 Learning Reference Governor

We adopt a reference governor approach to enforce the constraints. The RG is an add-on scheme to the nominal closed-loop system, illustrated by Figure 4.1. It acts as a pre-filter, which monitors the commanded reference input, $r(t)$, and adjusts it to a modified reference input, $\nu(t)$, to ensure constraint satisfaction. Differently from conventional RG schemes, the design of which requires explicit knowledge of the model (4.1) [133], the proposed RG relies on learning and is thereby referred to as *learning reference governor (LRG)*.

4.3.1 Reference Governor

The LRG updates the reference input at sample time instants $\{t_k\}_{k=0}^{\infty} \subset [0,\infty)$ (with $t_k \to \infty$ as $k \to \infty$) based on the following reference update law,

$$\nu(t^+) = \nu(t^-) + \kappa\big(x(t), r(t), \nu(t^-)\big)\big(r(t) - \nu(t^-)\big), \tag{4.8}$$

where $\nu(t^-)$ and $\nu(t^+)$ denote the reference input values before and after the update, respectively, and $\kappa : \mathbb{R}^n \times \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \to [0,1]$ is a scalar function. The LRG maintains $\nu(t)$ as constant over each interval $[t_k, t_{k+1})$. This way, the resulting reference input signal $\nu : [0,\infty) \to \mathbb{R}^{n_\nu}$ is piecewise constant and right continuous, and by Assumption 4.1, the system (4.1) has a unique solution.

The following lemma is exploited by the LRG to enforce the constraints (4.2).

Lemma 4.1. Given a pair $(x_0, \nu) \in \mathbb{R}^n \times V$, suppose that $\phi(t, x_0, \nu) \in Y$ for all $t \in [0,\infty)$. Then, any adjustment $\delta\nu$ satisfying $\nu + \delta\nu \in V$ and

$$D\big(\nu, \delta\nu, x_0 - x_\nu(\nu)\big) \le d(\nu), \tag{4.9}$$

guarantees $\phi(t, x_0, \nu + \delta\nu) \in Y$ for all $t \in [0,\infty)$.

Proof. Firstly, $\phi(t, x_0, \nu) \in Y$ for all $t \in [0,\infty)$ implies $y_\nu(\nu) \in Y$, since $x_\nu(\nu)$ is GAS by Assumption 4.2, g is continuous by Assumption 4.3, and Y is closed.

Then, we consider two cases separately: 1) $d(\nu) = 0$, and 2) $d(\nu) > 0$. For the former case, i.e., if $d(\nu) = 0$, we have

$$0 \le D\big(\nu, \delta\nu, x_0 - x_\nu(\nu)\big) = \sup_{t\in[0,\infty)} \big\|\phi(t, x_0, \nu + \delta\nu) - y_\nu(\nu)\big\| \le d(\nu) = 0, \tag{4.10}$$

which implies $\phi(t, x_0, \nu + \delta\nu) \equiv y_\nu(\nu) \in Y$.

For the latter case $d(\nu) > 0$, according to the definition of $d(\nu)$ in (4.6), it holds that the closed ball centered at $y_\nu(\nu)$ with radius $d(\nu)$ is contained entirely in the constraint set Y, i.e., $\overline{B}\big(y_\nu(\nu), d(\nu)\big) := \big\{y \in \mathbb{R}^m \big| \|y_\nu(\nu) - y\| \le d(\nu)\big\} \subseteq Y$. This is because for any $y' \in Y^C$, we have $\|y_\nu(\nu) - y'\| \ge \inf_{y\in Y^C} \|y_\nu(\nu) - y\| = d(\nu)$, which implies the open ball $B\big(y_\nu(\nu), d(\nu)\big) := \big\{y \in \mathbb{R}^m \big| \|y_\nu(\nu) - y\| < d(\nu)\big\} \subseteq Y$. Since $\overline{B}\big(y_\nu(\nu), d(\nu)\big)$ is the closure of $B\big(y_\nu(\nu), d(\nu)\big)$ and Y is closed, it holds that $\overline{B}\big(y_\nu(\nu), d(\nu)\big) \subseteq Y$.

Then, according to the definition of D in (4.4), $D\big(\nu, \delta\nu, x_0 - x_\nu(\nu)\big) \le d(\nu)$ implies $\phi(t, x_0, \nu + \delta\nu) \in \overline{B}\big(y_\nu(\nu), d(\nu)\big) \subseteq Y$ for all $t \in [0,\infty)$. ■

In principle, while guaranteeing the satisfaction of (4.2), minimizing the deviation of the modified reference $\nu(t^+)$ from the commanded value $r(t)$ is desired, which, according to the reference update law (4.8), corresponds to maximizing the value of $\kappa\big(x(t), r(t), \nu(t^-)\big)$ in $[0, 1]$. Note also that when $\kappa\big(x(t), r(t), \nu(t^-)\big) = 1$, we have $\nu(t^+) = r(t)$, i.e., the commanded reference input is reached.

On the basis of Lemma 4.1, if the function D, which is defined in (4.4), is known, then one can design the function κ in the reference update law (4.8) (or equivalently, determine the reference adjustment $\delta\nu$ at each sample time instant) such that the condition (4.9) is satisfied, which in turn guarantees the satisfaction of (4.2) after the reference update. However, as discussed in Section 4.2, the function D is typically not known a priori. Therefore, in what follows we introduce a safe learning algorithm to learn/estimate D from data, and at the same time evolve the design of κ. Here, "safe" means that the algorithm guarantees the constraints (4.2) to be satisfied over the entire process of data collection and learning.

4.3.2 Safe learning algorithm

The proposed learning algorithm is formally presented as Algorithm 4.1 with the `Kappa` function in line 5 determined according to Algorithm 4.2. This learning algorithm continually improves the estimate of D and at the same time evolves κ as more data are collected.

In Algorithm 4.1, the sampling period T in lines 3, 5 and 7 is the T in Assumption 4.6. The $\|\cdot\|$'s in (4.14) and (4.15) are the norm $\|\cdot\| = \|\cdot\|_{\mathbb{R}^{n_\nu}\times\mathbb{R}^{n_\nu}\times\mathbb{R}^n}$ used in (4.5) restricted to the subspaces $\{0\}\times\mathbb{R}^{n_\nu}\times\{0\}$, $\mathbb{R}^{n_\nu}\times\{0\}\times\mathbb{R}^n$, and $\{0\}\times\{0\}\times\mathbb{R}^n$, respectively, i.e., $\|\delta\nu\| = \|(0, \delta\nu, 0)\|$, $\|(\nu, \delta x)\| = \|(\nu, 0, \delta x)\|$, and $\|\delta x\| = \|(0, 0, \delta x)\|$. We note also that (4.15) is the solution to the following optimization problem,

$$\begin{aligned} &\max\ \kappa \in [0, 1], \quad \text{subject to} \\ &\big\|\kappa(r-\nu) - 0\big\| \le \left(\frac{d-0}{L}\right)^{\beta} - \left\| \begin{bmatrix} \nu \\ x - x_\nu(\nu) \end{bmatrix} - \begin{bmatrix} \nu \\ 0 \end{bmatrix} \right\|, \end{aligned} \tag{4.16}$$

which has the same form as (4.14), but with $(\nu_i, \delta\nu_i, \delta x_i, \tilde{D}_i)$ replaced by $(\nu, 0, 0, 0)$.

We note that Algorithm 4.1 continually improves the following estimate of D,

$$\overline{D}(\nu, \delta\nu, \delta x) = \min\left(\min_{i\in\mathcal{D}} \left(\tilde{D}_i + L \left\| \begin{bmatrix} \nu \\ \delta\nu \\ \delta x \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta\nu_i \\ \delta x_i \end{bmatrix} \right\|^{\frac{1}{\beta}} \right),\ L \left\| \begin{bmatrix} \delta\nu \\ \delta x \end{bmatrix} \right\|^{\frac{1}{\beta}} \right). \tag{4.17}$$

Algorithm 4.1 Safe learning algorithm

1: Initialize the system (4.1) with a strictly constraint-admissible steady state $x_\nu\big(\nu(0^-)\big)$ as the initial condition, i.e., $x(0) = x_\nu\big(\nu(0^-)\big)$ with $\nu(0^-) \in V$ and satisfying $y(0^-) = g\big(x(0), \nu(0^-)\big) \in \mathrm{int}(Y)$, and initialize the dataset $\mathcal{D} \leftarrow \emptyset$;

2: **for** $n = 0 : n_{\max} - 1$ **do**

3: Generate $r_n \in V \setminus \big\{\nu\big(nk_{\max}T^-\big)\big\}$ either randomly (e.g., based on a uniform distribution) or according to a training profile $\{r_i\}_{i=0}^{\infty}$ that covers the operating range of the system;

4: **for** $k = 0 : k_{\max} - 1$ **do**

5: At the sample time instant $t = (nk_{\max} + k)T$, compute

$$\kappa(t) = \texttt{Kappa}\big(x(t), r_n, \nu(t^-), d\big(\nu(t^-)\big), \mathcal{D}\big); \tag{4.11}$$

6: Adjust the reference input according to

$$\nu(t^+) = \nu(t^-) + \kappa(t)\big(r_n - \nu(t^-)\big); \tag{4.12}$$

7: At the next sample time instant $t' = t + T$, measure $\tilde{D}(t) = \tilde{D}\big(\nu(t^-), \delta\nu(t), \delta x(t)\big)$, where $\delta\nu(t) = \kappa(t)\big(r_n - \nu(t^-)\big)$ and $\delta x(t) = x(t) - x_\nu\big(\nu(t^-)\big)$;

8: Add the new data point $\big(\nu(t^-), \delta\nu(t), \delta x(t), \tilde{D}(t)\big)$ to the dataset $\mathcal{D}$, i.e.,

$$\mathcal{D} \leftarrow \mathcal{D} \cup \big(\nu(t^-), \delta\nu(t), \delta x(t), \tilde{D}(t)\big). \tag{4.13}$$

9: **end for**

10: **end for**

Algorithm 4.2 `Kappa`$(x, r, \nu, d, \mathcal{D})$

1: **for** $(\nu_i, \delta\nu_i, \delta x_i, \tilde{D}_i) \in \mathcal{D}$ **do**

2: Compute κ_i as the solution to the following optimization problem,

$$\begin{aligned} &\max\ \kappa \in [0,1], \quad \text{subject to} \\ &\left\|\kappa(r-\nu) - \delta\nu_i\right\| \leq \left(\frac{d-\tilde{D}_i}{L}\right)^{\beta} - \left\| \begin{bmatrix} \nu \\ x - x_\nu(\nu) \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta x_i \end{bmatrix} \right\|, \end{aligned} \tag{4.14}$$

if a solution exists, and $\kappa_i = 0$ otherwise;

3: **end for**

4:

$$\kappa' = \operatorname*{sat}_{[0,1]} \left(\frac{d}{L}\right)^{\beta} \frac{1}{\|r-\nu\|} - \frac{\|x - x_\nu(\nu)\|}{\|r-\nu\|}, \tag{4.15}$$

5: **return** $\kappa = \max\left(\max_i \kappa_i, \kappa'\right)$.

In particular, under Assumptions 4.4 and 4.6, (4.17) is an upper bound for D. This is because for any $(\nu, \delta\nu, \delta x) \in \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$, on the one hand, Assumptions 4.4 and 4.6 imply

$$\begin{aligned} D(\nu, \delta\nu, \delta x) &\leq D(\nu', \delta\nu', \delta x') + L \left\| \begin{bmatrix} \nu \\ \delta\nu \\ \delta x \end{bmatrix} - \begin{bmatrix} \nu' \\ \delta\nu' \\ \delta x' \end{bmatrix} \right\|^{\frac{1}{\beta}} \\ &\leq \tilde{D}(\nu', \delta\nu', \delta x') + L \left\| \begin{bmatrix} \nu \\ \delta\nu \\ \delta x \end{bmatrix} - \begin{bmatrix} \nu' \\ \delta\nu' \\ \delta x' \end{bmatrix} \right\|^{\frac{1}{\beta}}, \end{aligned} \tag{4.18}$$

for all measured data $\left(\nu', \delta\nu', \delta x', \tilde{D}(\nu', \delta\nu', \delta x')\right)$; and, on the other hand, Assumption 4.4 and the fact that $D(\nu, 0, 0) = 0$ for all $\nu \in \mathbb{R}^{n_\nu}$ imply

$$D(\nu, \delta\nu, \delta x) \leq D(\nu, 0, 0) + L \left\| \begin{bmatrix} \nu \\ \delta\nu \\ \delta x \end{bmatrix} - \begin{bmatrix} \nu \\ 0 \\ 0 \end{bmatrix} \right\|^{\frac{1}{\beta}} = L \left\| \begin{bmatrix} \delta\nu \\ \delta x \end{bmatrix} \right\|^{\frac{1}{\beta}}. \tag{4.19}$$

Then, (4.14) and (4.15) maximize $\kappa \in [0,1]$ such that $\overline{D}(\nu, \delta\nu, \delta x) \leq d(\nu)$.

The optimization problem (4.14) is a convex program with a scalar decision variable and can generally be solved with a bisection method. For the following two special cases, which are commonly encountered in practical situations, closed-form solutions to (4.14) exist:

Firstly, if the norm $\|\cdot\| = \|\cdot\|_{\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n}$ in (4.5) restricted to the subspace $\{0\} \times \mathbb{R}^{n_\nu} \times \{0\}$ is a quadratic norm, i.e., $\left\|\kappa(r-\nu) - \delta\nu_i\right\| = \left\|\kappa(r-\nu) - \delta\nu_i\right\|_Q =$

$\sqrt{(\kappa(r-\nu)-\delta\nu_i)^\top Q(\kappa(r-\nu)-\delta\nu_i)}$ for some positive-definite matrix $Q \in \mathbb{R}^{n_\nu \times n_\nu}$, then (4.14) admits the following closed-form solution,

$$\kappa_i = \operatorname*{sat}_{[0,1]} \frac{1}{(r-\nu)^\top Q(r-\nu)} \Bigg((r-\nu)^\top Q\,\delta\nu_i + \sqrt{\big((r-\nu)^\top Q\,\delta\nu_i\big)^2 - (r-\nu)^\top Q(r-\nu)\big(\delta\nu_i^\top Q\,\delta\nu_i - \Theta_i\big)} \Bigg), \tag{4.20}$$

where

$$\Theta_i = \left(\Big(\frac{d-\tilde{D}_i}{L}\Big)^\beta - \left\| \begin{bmatrix} \nu \\ x - x_\nu(\nu) \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta x_i \end{bmatrix} \right\| \right)^2. \tag{4.21}$$

Secondly, if the reference input is a scalar, i.e., $n_\nu = 1$, and $\big\|\kappa(r-\nu)-\delta\nu_i\big\| = \big|\kappa(r-\nu)-\delta\nu_i\big|$, then the closed-form solution (4.20) further reduces to

$$\kappa_i = \operatorname*{sat}_{[0,1]} \max \frac{1}{r-\nu} \left(\delta\nu_i \pm \left(\Big(\frac{d-\tilde{D}_i}{L}\Big)^\beta - \left\| \begin{bmatrix} \nu \\ x - x_\nu(\nu) \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta x_i \end{bmatrix} \right\| \right) \right). \tag{4.22}$$

The learning algorithm terminates either after the maximum number of training commands, $n_{\max}$, has been reached or when the moving average of the tracking error between the commanded reference input $r(t)$ and the modified reference input $\nu(t)$, defined as $\frac{1}{T_{\text{win}}} \int_{t-T_{\text{win}}}^{t} \|r(s)-\nu(s)\|\,\mathrm{d}s$ with $T_{\text{win}} > 0$ denoting the window size, converges lower than a specified threshold value. After termination of the learning phase, the system is ready for operation. In principle, learning can be stopped at any time and the resulting reference governor will enforce the constraints, however, its performance in terms of moving average of the tracking error can be more conservative than after training is completed.

During the operating phase, the LRG uses (4.8) to adjust the reference input $\nu(t)$ at each sample time instant $t \in \{t_k\}_{k=0}^{\infty}$, where $\kappa\big(x(t), r(t), \nu(t^-)\big)$ is determined according to (4.11) and Algorithm 4.2 with the dataset $\mathcal{D}$ obtained from the learning phase. We note that the LRG can use a different sampling period for the operating phase than for the learning phase and smaller values in the operating phase can improve performance.

4.3.3 Theoretical properties

The LRG algorithm introduced in Sections 4.3.1 and 4.3.2 guarantees constraint satisfaction during both the learning and the operating phases, pointwise convergence of the estimate of D, and finite-time convergence of the modified reference input $\nu(t)$

to constant, strictly steady-state constraint-admissible reference command $r(t)$ during the operating phase. These properties are presented as the following propositions.

Proposition 4.2. During the learning phase, constraints $y(t) \in Y$ are guaranteed to be satisfied for all $t \in [0, \infty)$.

Proof. Let $\tau = \sigma T$, $\sigma = 0, 1, 2, \ldots$, be an arbitrary sample time instant during the learning phase. Suppose $\phi(t', x(\tau), \nu(\tau^-)) \in Y$ for all $t' \in [0, \infty)$, where $\phi(\cdot, x(\tau), \nu(\tau^-)) = g\big(\psi(\cdot, x(\tau), \nu(\tau^-)), \nu(\tau^-)\big)$ denotes the output trajectory of (4.3) with the initial condition $x_0 = x(\tau)$ and constant reference input $\nu = \nu(\tau^-)$. Then, for any $\delta\nu$ satisfying

$$\big\|\delta\nu - \delta\nu_i\big\| \leq \left(\frac{d\big(\nu(\tau^-)\big) - \tilde{D}_i}{L}\right)^{\beta} - \left\|\begin{bmatrix} \nu(\tau^-) \\ x(\tau) - x_\nu\big(\nu(\tau^-)\big) \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta x_i \end{bmatrix}\right\|, \tag{4.23}$$

for some $i \in \mathcal{D}$, it holds that

$$\begin{aligned} &D\big(\nu(\tau^-), \delta\nu, x(\tau) - x_\nu(\nu(\tau^-))\big) \\ &\leq \tilde{D}_i + L \left\| \begin{bmatrix} \nu(\tau^-) \\ \delta\nu \\ x(\tau) - x_\nu(\nu(\tau^-)) \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta\nu_i \\ \delta x_i \end{bmatrix} \right\|^{\frac{1}{\beta}} \\ &\leq \tilde{D}_i + L \left(\left\| \begin{bmatrix} \nu(\tau^-) \\ x(\tau) - x_\nu(\nu(\tau^-)) \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta x_i \end{bmatrix} \right\| + \|\delta\nu - \delta\nu_i\| \right)^{\frac{1}{\beta}} \\ &\leq d\big(\nu(\tau^-)\big), \end{aligned} \tag{4.24}$$

where we have used the triangle inequality to derive the second inequality. Then, according to Lemma 4.1, (4.24) implies $\phi(t', x(\tau), \nu(\tau^-) + \delta\nu) \in Y$ for all $t' \in [0, \infty)$. Similarly, for any $\delta\nu$ satisfying

$$\|\delta\nu\| \leq \left(\frac{d\big(\nu(\tau^-)\big)}{L}\right)^{\beta} - \big\|x(\tau) - x_\nu\big(\nu(\tau^-)\big)\big\|, \tag{4.25}$$

it holds that

$$\begin{aligned} D\big(\nu(\tau^-), \delta\nu, x(\tau) - x_\nu(\nu(\tau^-))\big) &\leq L \left\| \begin{bmatrix} \delta\nu \\ x(\tau) - x_\nu(\nu(\tau^-)) \end{bmatrix} \right\|^{\frac{1}{\beta}} \\ &\leq L \left(\big\|x(\tau) - x_\nu\big(\nu(\tau^-)\big)\big\| + \|\delta\nu\| \right)^{\frac{1}{\beta}} \leq d\big(\nu(\tau^-)\big), \end{aligned} \tag{4.26}$$

which implies $\phi(t', x(\tau), \nu(\tau^-) + \delta\nu) \in Y$ for all $t' \in [0, \infty)$. Therefore, for $\nu(\tau^+)$

determined according to (4.11), (4.12), and Algorithm 4.2, it must hold that $\phi(t', x(\tau), \nu(\tau^+)) \in Y$ for all $t' \in [0, \infty)$. Note that if there exists no $\delta\nu$ satisfying (4.23) or (4.25), then Algorithm 2 returns $\kappa(t) = 0$. In this case, we have $\nu(\tau^+) = \nu(\tau^-)$, and thus, $\phi(t', x(\tau), \nu(\tau^+)) = \phi(t', x(\tau), \nu(\tau^-)) \in Y$.

Since the learning algorithm is initialized with $x(0) = x_\nu\big(\nu(0^-)\big)$ such that $\phi(t', x(0), \nu(0^-)) \equiv g\big(x(0), \nu(0^-)\big) \in Y$ for all $t' \in [0, \infty)$, the above result says $\phi(t', x(0), \nu(0^+)) \in Y$ for all $t' \in [0, \infty)$, which also implies $\phi(t', x(T), \nu(T^-)) = \phi(t' + T, x(0), \nu(0^+)) \in Y$ for all $t' \in [0, \infty)$. Then, by induction on $\tau = 0, T, 2T, \ldots$, we obtain that $\phi(t', x(\tau), \nu(\tau^+)) \in Y$ for all $t' \in [0, \infty)$ and all sample time instants τ. Now let $t \in [0, \infty)$ be arbitrary and consider the sample time instant τ immediately before t (i.e., $\tau \leq t < \tau + T$). It holds that $y(t) = \phi(t - \tau, x(\tau), \nu(\tau^+)) \in Y$. This proves $y(t) \in Y$ for all $t \in [0, \infty)$. ■

Proposition 4.2 certifies that our algorithm is a safe learning algorithm, i.e., ensures constraint satisfaction over the entire learning process.

Corollary 4.3. During the operating phase, constraints $y(t) \in Y$ are guaranteed to be satisfied for all $t \in [0, \infty)$. In particular, this guarantee does not depend on the length of the learning phase.

Proof. The constraint satisfaction result established in the proof of Proposition 4.2 depends only on the reference update law used by LRG, neither on the sampling period T nor on the number of data points in the dataset $\mathcal{D}$. Therefore, the same proof can be used to prove Corollary 1, with T now corresponding to the sampling period for the operating phase. ■

Corollary 4.3 establishes the constraint enforcement property of our LRG after training. Although the guaranteed constraint satisfaction does not depend on the length of learning, the performance, in terms of reference tracking, will be improved as learning proceeds. The following proposition formalizes such a result.

Proposition 4.4. Let $\overline{D}^\sigma$ denote the estimate of D, (4.17), after the sample time instant $\tau = \sigma T$ during the learning phase. The following properties hold: (i) $D \leq \overline{D}^{\sigma+1} \leq \overline{D}^\sigma$ pointwise on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$ for all $\sigma = 0, 1, \ldots$; (ii) $\overline{D}^\sigma$ converges pointwise on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$ as $\sigma \to \infty$; and (iii) the pointwise limit $\overline{D}^\infty := \lim_{\sigma \to \infty} \overline{D}^\sigma$ satisfies $\overline{D}^\infty \geq D$ and is Hölder continuous with constants L and β on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$.

Proof. Firstly, $D \leq \overline{D}^\sigma$ for all $\sigma = 0, 1, \ldots$ follows from the expression (4.17) and the inequalities (4.18) and (4.19). Then, $\overline{D}^{\sigma+1} \leq \overline{D}^\sigma$ follows from the fact that the

dataset corresponding to $\overline{D}^{\sigma+1}$, denoted as $\mathcal{D}^{\sigma+1}$, over which the second minimum in the expression (4.17) is taken, is a superset of that corresponding to $\overline{D}^{\sigma}$, $\mathcal{D}^{\sigma}$. We have $\mathcal{D}^{\sigma+1} \supseteq \mathcal{D}^{\sigma}$ because as learning proceeds, more data points are collected into $\mathcal{D}$. This proves (i).

For each $(\nu, \delta\nu, \delta x) \in \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^{n}$, since $0 \leq D(\nu, \delta\nu, \delta x) \leq \overline{D}^{\sigma+1}(\nu, \delta\nu, \delta x) \leq \overline{D}^{\sigma}(\nu, \delta\nu, \delta x)$ for all $\sigma = 0, 1, \ldots$, by the monotone convergence theorem, the sequence $\left\{\overline{D}^{\sigma}(\nu, \delta\nu, \delta x)\right\}_{\sigma=0}^{\infty}$ must converge. This proves the pointwise convergence of $\overline{D}^{\sigma}$ on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^{n}$ as $\sigma \to \infty$.

For (iii), $\overline{D}^{\infty} \geq D$ follows from the fact that $\overline{D}^{\sigma} \geq D$ for all $\sigma = 0, 1, \ldots$ It remains to show that $\overline{D}^{\infty}$ is Hölder continuous with constants L and β.

Let $\sigma = 0, 1, \ldots$ be arbitrary and denote the corresponding dataset as $\mathcal{D}^{\sigma}$. It is easy to see that for each $i \in \mathcal{D}^{\sigma}$, the following function,

$$\overline{D}^{\sigma,i}(\nu, \delta\nu, \delta x) := \tilde{D}_i + L \left\| \begin{bmatrix} \nu \\ \delta\nu \\ \delta x \end{bmatrix} - \begin{bmatrix} \nu_i \\ \delta\nu_i \\ \delta x_i \end{bmatrix} \right\|^{\frac{1}{\beta}}, \tag{4.27}$$

is Hölder continuous with constants L and β. Because $\overline{D}^{\sigma,i}$ for all $i \in \mathcal{D}^{\sigma}$ and $\overline{D}'(\nu, \delta\nu, \delta x) := L \left\| \begin{bmatrix} \delta\nu \\ \delta x \end{bmatrix} \right\|^{\frac{1}{\beta}}$ share the same Hölder constants L and β, we have that $\overline{D}^{\sigma} = \min\left(\min_{i \in \mathcal{D}^{\sigma}} \overline{D}^{\sigma,i}, \overline{D}'\right)$ is also Hölder continuous with constants L and β. Note that this result holds for all $\sigma = 0, 1, \ldots$

Since $\overline{D}^{\infty}$ is the pointwise limit of the non-increasing sequence of functions $\{\overline{D}^{\sigma}\}_{\sigma=0}^{\infty}$, it can also be expressed as $\overline{D}^{\infty} = \inf_{\sigma=0,1,\ldots} \overline{D}^{\sigma}$. In this case, because $\overline{D}^{\sigma}$ for all $\sigma = 0, 1, \ldots$ share the same Hölder continuous with constants L and β, we can conclude that $\overline{D}^{\infty} = \inf_{\sigma=0,1,\ldots} \overline{D}^{\sigma}$ is Hölder continuous with constants L and β. This completes the proof of (iii). ■

Recall that Algorithm 4.2 maximizes $\kappa \in [0, 1]$ such that $\overline{D}(\nu, \delta\nu, \delta x) \leq d(\nu)$. According to Proposition 4.4(i), as learning proceeds, the estimate $\overline{D}(\nu, \delta\nu, \delta x)$ becomes less conservative (i.e., smaller), and correspondingly, the feasible region for κ is enlarged. In turn, κ can take more aggressive values for $\nu(t)$ to track $r(t)$, which transfers to improved reference tracking performance.

After the learning process, the dataset $\mathcal{D}$ may contain a large number of data points. For online implementation of our LRG, Algorithm 4.2 needs to run fast enough to produce $\kappa(t)$ for updating the reference input $\nu(t)$ in real time. This is possible because, on the one hand, the optimization problem (4.14) may admit closed-

form solutions, such as (4.20) and (4.22); and on the other hand, the computational tasks of (4.14) for different data points $i \in \mathcal{D}$ are ready to be performed in parallel. Furthermore, one may partition the space $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$ by bounded subsets $\{U_i\}_{i=1}^{\infty}$, such as cubes, with diameter $\operatorname{diam}(U_i) = \sup\{\|z_1 - z_2\| \mid z_1, z_2 \in U_i\} \leq m$ for all i, and post-process the dataset $\mathcal{D}$ such that if there are multiple data points in the same U_i, then only one of them is kept and the others are dropped. This way, the number of data points in $\mathcal{D}$ will be reduced, and in turn, the computational cost of Algorithm 4.2 will become lower.

Denote the estimate $\overline{D}$ corresponding to the dataset before post-processing as $\overline{D}^{\text{pre}}$ and that corresponding to the dataset after post-processing as $\overline{D}^{\text{post}}$. The following result can be used to guide the design of the parameter m to balance the tradeoff between estimation performance and dataset complexity.

Proposition 4.5. After post-processing, the estimate, $\overline{D}^{\text{post}}$, satisfies $\overline{D}^{\text{pre}} \leq \overline{D}^{\text{post}} \leq \overline{D}^{\text{pre}} + 2Lm^{\frac{1}{\beta}} + \varepsilon$ pointwise on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$.

Proof. Firstly, the dataset before post-processing, $\mathcal{D}^{\text{pre}}$, is a superset of that after post-processing, $\mathcal{D}^{\text{post}}$. Thus, according to the expression (4.17), we have $\overline{D}^{\text{pre}} \leq \overline{D}^{\text{post}}$.

Secondly, for each $(\nu, \delta\nu, \delta x) \in \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$, according to (4.17), we have

$$\overline{D}^{\text{pre}}(\nu, \delta\nu, \delta x) = \min\left(\tilde{D}_j + L\left\|\begin{bmatrix}\nu\\ \delta\nu\\ \delta x\end{bmatrix} - \begin{bmatrix}\nu_j\\ \delta\nu_j\\ \delta x_j\end{bmatrix}\right\|^{\frac{1}{\beta}}, \quad L\left\|\begin{bmatrix}\delta\nu\\ \delta x\end{bmatrix}\right\|^{\frac{1}{\beta}}\right), \tag{4.28}$$

for some data point $j \in \mathcal{D}^{\text{pre}}$. The post-processing procedure introduced above ensures that there exists some data point $k \in \mathcal{D}^{\text{post}}$ such that $\|(\nu_k, \delta\nu_k, \delta x_k) - (\nu_j, \delta\nu_j, \delta x_j)\| \leq m$. Therefore, we have

$$
\begin{aligned}
\overline{D}^{\mathrm{post}}(\nu,\delta\nu,\delta x) &= \min\left(\min_{i\in\mathcal{D}^{\mathrm{post}}}\left(\tilde{D}_i + L\left\|\begin{bmatrix}\nu\\ \delta\nu\\ \delta x\end{bmatrix} - \begin{bmatrix}\nu_i\\ \delta\nu_i\\ \delta x_i\end{bmatrix}\right\|^{\frac{1}{\beta}}\right), L\left\|\begin{bmatrix}\delta\nu\\ \delta x\end{bmatrix}\right\|^{\frac{1}{\beta}}\right)\\
&\le \min\left(\tilde{D}_k + L\left\|\begin{bmatrix}\nu\\ \delta\nu\\ \delta x\end{bmatrix} - \begin{bmatrix}\nu_k\\ \delta\nu_k\\ \delta x_k\end{bmatrix}\right\|^{\frac{1}{\beta}}, L\left\|\begin{bmatrix}\delta\nu\\ \delta x\end{bmatrix}\right\|^{\frac{1}{\beta}}\right)\\
&\le \min\left(\tilde{D}_k + (\tilde{D}_j - D_j) + L\left(\left\|\begin{bmatrix}\nu\\ \delta\nu\\ \delta x\end{bmatrix} - \begin{bmatrix}\nu_j\\ \delta\nu_j\\ \delta x_j\end{bmatrix}\right\| + \left\|\begin{bmatrix}\nu_k\\ \delta\nu_k\\ \delta x_k\end{bmatrix} - \begin{bmatrix}\nu_j\\ \delta\nu_j\\ \delta x_j\end{bmatrix}\right\|\right)^{\frac{1}{\beta}}, L\left\|\begin{bmatrix}\delta\nu\\ \delta x\end{bmatrix}\right\|^{\frac{1}{\beta}}\right)\\
&\le \min\left(\tilde{D}_k + (\tilde{D}_j - D_j) + L\left\|\begin{bmatrix}\nu\\ \delta\nu\\ \delta x\end{bmatrix} - \begin{bmatrix}\nu_j\\ \delta\nu_j\\ \delta x_j\end{bmatrix}\right\|^{\frac{1}{\beta}} + L\left\|\begin{bmatrix}\nu_k\\ \delta\nu_k\\ \delta x_k\end{bmatrix} - \begin{bmatrix}\nu_j\\ \delta\nu_j\\ \delta x_j\end{bmatrix}\right\|^{\frac{1}{\beta}}, L\left\|\begin{bmatrix}\delta\nu\\ \delta x\end{bmatrix}\right\|^{\frac{1}{\beta}}\right)\\
&\le \min\left(\tilde{D}_j + L\left\|\begin{bmatrix}\nu\\ \delta\nu\\ \delta x\end{bmatrix} - \begin{bmatrix}\nu_j\\ \delta\nu_j\\ \delta x_j\end{bmatrix}\right\|^{\frac{1}{\beta}}, L\left\|\begin{bmatrix}\delta\nu\\ \delta x\end{bmatrix}\right\|^{\frac{1}{\beta}}\right)\\
&\quad + (\tilde{D}_k - D_j) + L\left\|(\nu_k,\delta\nu_k,\delta x_k) - (\nu_j,\delta\nu_j,\delta x_j)\right\|^{\frac{1}{\beta}}\\
&\le \overline{D}^{\mathrm{pre}}(\nu,\delta\nu,\delta x) + 2Lm^{\frac{1}{\beta}} + \varepsilon,
\end{aligned}
\tag{4.29}
$$

where we have used $\tilde{D}_j - D_j \ge 0$, by Assumption 4.6, and the triangle inequality to derive the second inequality, and the triangle inequality variant with exponents $1/\beta \in (0,1]$ to derive the third inequality, and have used the following result to derive the last inequality,

$$
\begin{aligned}
\tilde{D}_k - D_j &\le D_k + \varepsilon - D_j\\
&\le \left|D(\nu_k,\delta\nu_k,\delta x_k) - D(\nu_j,\delta\nu_j,\delta x_j)\right| + \varepsilon\\
&\le L\left\|(\nu_k,\delta\nu_k,\delta x_k) - (\nu_j,\delta\nu_j,\delta x_j)\right\|^{\frac{1}{\beta}} + \varepsilon\\
&\le Lm^{\frac{1}{\beta}} + \varepsilon,
\end{aligned}
\tag{4.30}
$$

in which the third inequality is due to Assumption 4.4. This proves $\overline{D}^{\mathrm{post}} \le \overline{D}^{\mathrm{pre}} + 2Lm^{\frac{1}{\beta}} + \varepsilon$. ■

We next study the convergence of the modified reference $\nu(t)$ to the commanded value $r(t)$. To achieve an enhanced convergence property, following the approach in

[141], we exploit the following two sets,

$$\begin{aligned} V_1(r) &:= \big\{\nu \in V \,\big|\, d(\nu) \geq \min\big(d(r), \delta\big)\big\}, \\ V_2(\overline{\nu}, r) &:= \big\{\nu \in V \,\big|\, \|\nu - r\| \leq \max\big(\|\overline{\nu} - r\| - \lambda, 0\big)\big\}, \end{aligned} \tag{4.31}$$

where $\delta > 0$ is arbitrarily small and $\lambda = \lambda(r) \in \Big(0, \big(\frac{\min(d(r),\delta)}{L}\big)^{\beta}\Big)$, and we slightly modify the reference update law (4.8) during the operating phase to

$$\begin{aligned} \nu(t^+) &= \tilde{\nu}(t^+)\,\chi(t) + \nu(t^-)\big(1 - \chi(t)\big), \\ \tilde{\nu}(t^+) &= \nu(t^-) + \kappa\big(x(t), r(t), \nu(t^-)\big)\big(r(t) - \nu(t^-)\big), \end{aligned} \tag{4.32}$$

where $\chi(t) = 1$ if $\tilde{\nu}(t^+) \in V_1\big(r(t)\big) \cap V_2\big(\nu(t^-), r(t)\big)$, and $\chi(t) = 0$ otherwise.

Proposition 4.6. Consider the operation of the reference governor based on the update law (4.32). Suppose that there exists $t_s \in [0, \infty)$ such that $r(t) = r_s$ for all $t \in [t_s, \infty)$, with $r_s \in V$ satisfying $d(r_s) > 0$. Suppose also that for any ν on the line segment connecting $\nu(t_s^-)$ and r_s, we have $\nu \in V_1(r_s)$. Then, with the reference update law (4.32), there exists $t_f \in [t_s, \infty)$ such that $\nu(t) = r(t)$ for all $t \in [t_f, \infty)$. In particular, this result does not depend on the length of the learning phase.

Proof. Let $J_k := \|\nu(t_k^+) - r_s\|$. Because in the reference update law (4.32), κ, determined by Algorithm 2, takes values in the interval $[0, 1]$, the sequence $\{J_k\}_{k=k_s}^{\infty}$, with t_{k_s} denoting the first sample time instant after t_s, is non-increasing. Since $\{J_k\}_{k=k_s}^{\infty}$ is also bounded from below by 0, by the monotone convergence theorem, J_k converges as $k \to \infty$. Let $\hat{J} := \lim_{k\to\infty} J_k$.

In particular, due to the requirement $\tilde{\nu}(t^+) \in V_2\big(\nu(t^-), r_s\big) = \big\{\nu \in V \,\big|\, \|\nu - r_s\| \leq \max\big(\|\nu(t^-) - r_s\| - \lambda, 0\big)\big\}$ in (4.32) for the reference input $\nu(t^+)$ to be updated from $\nu(t^-)$ to $\tilde{\nu}(t^+)$, it must hold that $J_{k+1} \leq \max(J_k - \lambda, 0)$ whenever $J_{k+1} \neq J_k$. Note that the λ here is a positive constant. In this case, the sequence $\{J_k\}_{k=k_s}^{\infty}$ converges to $\hat{J}$ through at most a finite number of jumps. In other words, there exists $k_f \in \mathbb{N}_0$ such that $J_k = \hat{J}$ for all $k \geq k_f$. Under (4.32), this also implies $\nu(t) = \nu(t_{k_f}^+) := \hat{\nu}$ for all $t \in [t_{k_f}, \infty)$, i.e., the modified reference $\nu(t)$ converges to $\hat{\nu}$ in finite time.

We now show that $\hat{J} = 0$ and $\hat{\nu} = r_s$ by contradiction.

Suppose $\hat{\nu} \neq r_s$. First note that the requirement $\tilde{\nu}(t^+) \in V_1(r_s)$ in (4.32) for $\nu(t^+)$ to be updated from $\nu(t^-)$ to $\tilde{\nu}(t^+)$ ensures $\hat{\nu} = \nu(t_{k_f}^+) \in V_1(r_s)$, which implies $\big(\frac{d(\hat{\nu})}{L}\big)^{\beta} - \lambda \geq \big(\frac{\min(d(r_s),\delta)}{L}\big)^{\beta} - \lambda > 0$. Then, since $\nu(t) = \hat{\nu}$ for all $t \in [t_{k_f}, \infty)$ and $x_\nu(\hat{\nu})$ is GAS by Assumption 4.2, there exists $\tau \in \{t_k\}_{k=k_f}^{\infty}$ such that $\|x(\tau) - x_\nu(\hat{\nu})\| \leq \big(\frac{d(\hat{\nu})}{L}\big)^{\beta} - \lambda$, which can also be expressed as $\big(\frac{d(\hat{\nu})}{L}\big)^{\beta} - \|x(\tau) - x_\nu(\hat{\nu})\| \geq \lambda > 0$.

Then, according to (4.15), the κ produced by Algorithm 4.2 at the sample time instant τ satisfies either $\kappa \geq \frac{(d(\hat{\nu})/L)^\beta - \|x(\tau) - x_\nu(\hat{\nu})\|}{\|r_s - \hat{\nu}\|} > 0$ or $\kappa = 1$. In this case, $\tilde{\nu} := \hat{\nu} + \kappa(r_s - \hat{\nu})$ satisfies either

$$\begin{aligned} &\|\hat{\nu} - r_s\| - \|\tilde{\nu} - r_s\| = \|\hat{\nu} - r_s\| - \|\hat{\nu} + \kappa(r_s - \hat{\nu}) - r_s\| \\ &= \kappa \|\hat{\nu} - r_s\| \geq \left(\frac{d(\hat{\nu})}{L}\right)^\beta - \|x(\tau) - x_\nu(\hat{\nu})\| \geq \lambda, \end{aligned} \tag{4.33}$$

or $\tilde{\nu} = \hat{\nu} + \kappa(r_s - \hat{\nu}) = r_s$, leading to $\|\tilde{\nu} - r_s\| = 0$. In either case, $\tilde{\nu} = \hat{\nu} + \kappa(r_s - \hat{\nu}) \in V_2(\hat{\nu}, r_s)$. Note also that with the reference update law (4.32), all of $\{\nu(t_k^+)\}_{k=k_s}^\infty$ lie on the line segment connecting $\nu(t_s^-)$ and r_s, including $\hat{\nu} = \nu(t_{k_f}^+)$. In this case, $\tilde{\nu} = \hat{\nu} + \kappa(r_s - \hat{\nu})$ also lies on the line segment connecting $\nu(t_s^-)$ and r_s. Then, by the second assumption in the proposition statement, we have $\tilde{\nu} \in V_1(r_s)$.

Since $\kappa > 0$ and $\tilde{\nu} = \hat{\nu} + \kappa(r_s - \hat{\nu}) \in V_1(r_s) \cap V_2(\hat{\nu}, r_s)$, under (4.32), the reference input should be updated from $\hat{\nu}$ to $\tilde{\nu}$ at the sample time instant $\tau \in \{t_k\}_{k=k_f}^\infty$. This contradicts our assumption that $\nu(t)$ converges to some $\hat{\nu} \neq r_s$ at $t = t_{k_f}$. Since in the above we have shown that $\nu(t)$ indeed converges to some $\hat{\nu}$ at $t = t_{k_f}$, we can conclude that $\hat{\nu} = r_s$. This completes the proof.

It is clear that the above proof does not depend on the number of data points in the dataset $\mathcal{D}$. Therefore, this finite-time convergence result does not depend on the length of learning. ■

4.4 Ground Vehicle Rollover Avoidance

Rollover is a type of vehicle accident in which a vehicle tips over onto its side or roof. Rollovers have a higher fatality rate than other types of vehicle crashes. The rollover propensities of a vehicle may change if the vehicle carries a load or is driven on a different road surface [131], and as a result, a model that can accurately represent the vehicle dynamics under the current operating condition is often not a priori at hand. In such a case, a learning-based approach may be desired, with experiments performed either on the real vehicle or on a high fidelity vehicle model.

In this section, we apply our learning-based reference governor algorithm to guarding a vehicle against rollover. We first use a simplified model to represent the vehicle dynamics. This model satisfies our Assumptions 4.1 to 4.6, and thus Propositions 4.2 to 4.6 are guaranteed to hold. We then use a high-fidelity *CarSim* model to represent the vehicle dynamics, to illustrate the effectiveness of the algorithm in a more complex application scenario. Note that both the simplified model and the high-fidelity

CarSim model are treated as black-box and only used to simulate vehicle responses.

4.4.1 Application to a reduced-order linear model

We consider a scenario where the vehicle is driven with a constant longitudinal speed and the command is the steering wheel angle (SW), generated by a human operator or a higher-level planning algorithm. The reference governor is used to modify the SW command to avoid potential rollover.

Following [131], we use a linear model with the four states: vehicle roll angle q, roll rate p, lateral velocity v, and yaw rate γ, to represent the vehicle dynamics. The rollover constraints are defined through the load transfer ratio (LTR):

$$\text{LTR} := \frac{F_{z,R} - F_{z,L}}{mg}, \tag{4.34}$$

where $F_{z,R}$ and $F_{z,L}$ represent, respectively, the total vertical force on the right-side tires and that on the left-side tires, and mg is the vehicle weight. Based on LTR, the rollover constraints are imposed as

$$-\text{LTR}_{\text{lim}} \leq \text{LTR} \leq \text{LTR}_{\text{lim}}. \tag{4.35}$$

Note that $|\text{LTR}| > 1$ means wheels lifting off, so we set $\text{LTR}_{\text{lim}} = 1$ in this example.

The linear model is represented as

$$\begin{aligned} \dot{x} &= Ax + B\nu, \\ y &= Cx, \end{aligned} \tag{4.36}$$

where $x = [q, p, v, \gamma]^\top$, $\nu = \text{SW}$, and $y = \text{LTR}$. The parameters are identified based on a *CarSim* model of a standard utility truck driving at a constant speed of 80

(km/h)[1], and are as follows:

$$A = \begin{bmatrix} 0.00499 & 0.997 & 0.0154 & -6.81 \times 10^{-5} \\ -78.3 & -12.2 & -65.3 & -3.89 \\ -0.932 & -0.799 & -6.20 & -1.57 \\ 1.52 & 3.32 & 8.27 & -1.49 \end{bmatrix},$$
$$B = \begin{bmatrix} -5.76 \times 10^{-5} & 2.80 & 0.278 & 0.655 \end{bmatrix}^\top, \tag{4.37}$$
$$C = \begin{bmatrix} 0.120 & 0.0124 & -0.0108 & 0.0109 \end{bmatrix}.$$

We use the 1-norm for every $\|\cdot\|$ involved in our learning algorithm. Suppose (4.36) and (4.37) are known, then it can be shown by analytical calculations that for $\beta = 1$, any $L \geq 0.132$ satisfies (4.5). In this example, we choose $L = 0.3$ to illustrate that a conservative estimate of L is sufficient for our learning algorithm. During the learning phase, the commands r_n can be randomly generated. However, to better visualize how the learning algorithm gradually pushes the system to its mobility limit, in this example we use a repeated profile for r_n, switching between ± 100 (deg) with a duration of 20 (s).

The learning progress is illustrated in Figure 4.2. In Figure 4.2(a), the tracking error is defined as the average of $|r-\nu|$ over a time window of the most recent 1000 (s). At the beginning of learning, the tracking error is relatively high, indicating that the reference governor operates conservatively and significantly modifies the command. The tracking error gradually decreases as the learning progresses and converges to a low value. Figure 4.2(b) shows the LTR response during learning. As the learning progresses, the vehicle can operate with maneuvers that cause the LTR response to reach the constraint boundary but without violating it. Note that constraints are satisfied during the entire learning process.

Figure 4.3 shows the SW and LTR responses of tracking step SW commands (plotted in solid black) when without the reference governor (dashed red) and when with the reference governor before (dotted green) and after learning (solid blue). Without the reference governor, the input to the system tracks the command perfectly, but this results in occasional constraint violations. Before learning, the reference governor operates conservatively and is not able to satisfactorily track the command. After learning with $n = 750$ commands, the reference governor passes the command unchanged to the system when there is no danger of constraint violation, and modifies

[1]To be more precise, the vehicle is tracking the constant reference speed of 80 (km/h) based on a feedback control on the gas pedal.

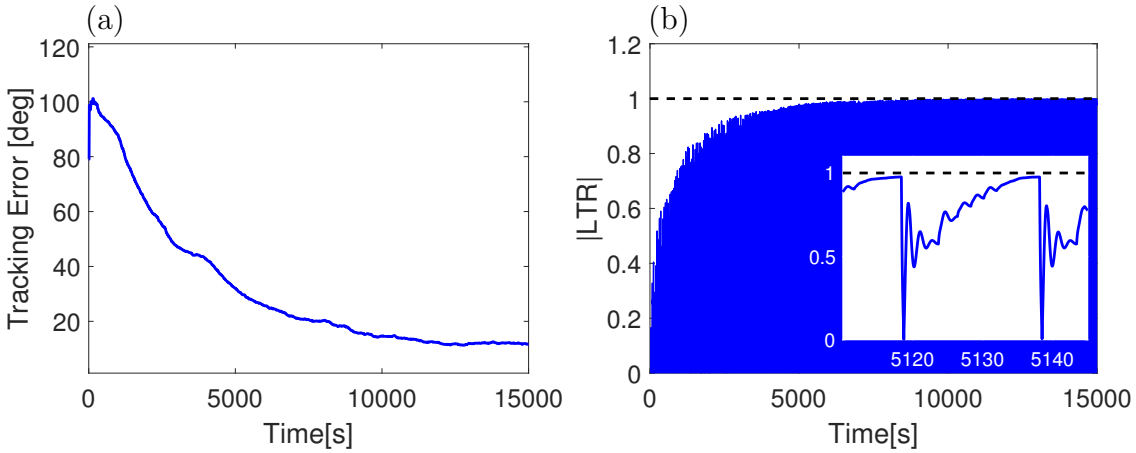

Figure 4.2: Learning algorithm application to vehicle rollover avoidance based on a linear model. (a) Tracking error profile during learning. (b) LTR profile during learning.

the command by a minimal amount when necessary to ensure constraint satisfaction. Furthermore, the modified reference input converges to the original command in a small amount of time.

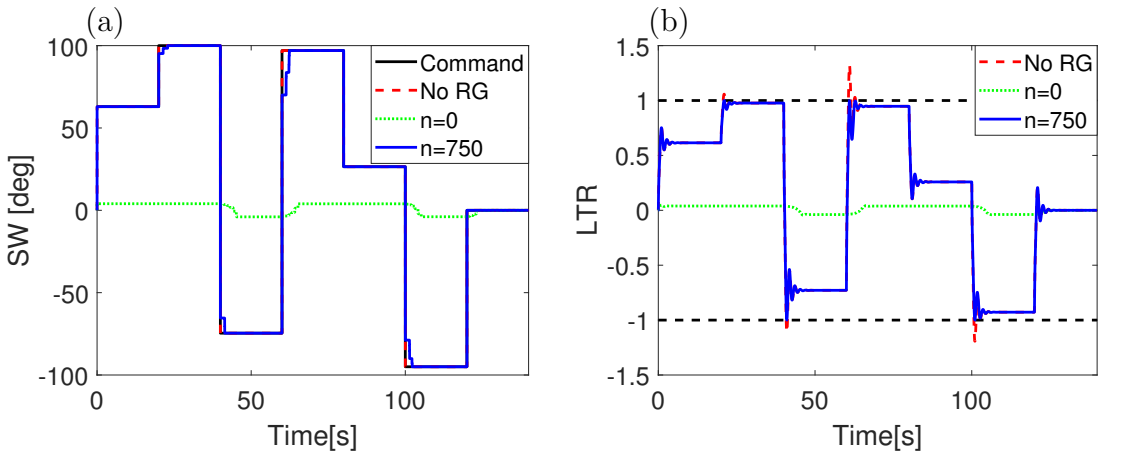

Figure 4.3: (a) Tracking results for step SW commands and (b) corresponding LTR responses without reference governor and with reference governor before and after learning.

4.4.2 Application to a high-fidelity CarSim model

We consider the same scenario as before, where the vehicle is driving at the constant speed of 80 (km/h) and the maneuver command is the steering wheel angle (SW), but we now apply the learning algorithm directly to the high-fidelity *CarSim*

model of standard utility truck.

In the implementation, we still treat $[q, p, v, \gamma]^\top$ as x in defining the function D (see (4.4)), although the high-fidelity *CarSim* model has a much larger number of states. The steady-state maps x_ν and y_ν are now estimated by applying samples of constant $\nu = \text{SW}$ until the $x = [q, p, v, \gamma]^\top$ and $y = \text{LTR}$ converge. In this example, we choose again $\beta = 1$ and $L = 0.3$. The rollover limit is again defined through (4.35). However, as noted in Remark 3.1 in Section 3.4, LTR going above 1 does not necessarily lead to rollover due to suspension roll moment and aerodynamic forces and also due to the fact that additional work is needed to move the gravity center of the vehicle up for rollover. Therefore, following Remark 3.1, LTR_{lim} is tuned to a value that best describes the situation of rollover. In this example, LTR_{lim} is chosen as 1.3.

In this example, we use a repeated profile for r_n, switching between ± 200 (deg) with a duration of 10 (s). In practice, the training profile can be randomly generated.

The learning progress is illustrated in Figure 4.4. The tracking error is defined in the same way as that in Figure 4.2(a). The tracking error gradually decreases and converges to a much smaller value compared to that at the beginning of learning. Note that the final tracking error of this example being larger than that in Figure 4.2 is due to the fact that we now consider a wider range of SW commands, $[-200, 200]$ (deg), compared to the $[-100, 100]$ (deg) for Figure 4.2. The LTR is always maintained in the range of $[-\text{LTR}_{\text{lim}}, \text{LTR}_{\text{lim}}]$ during the entire learning process.

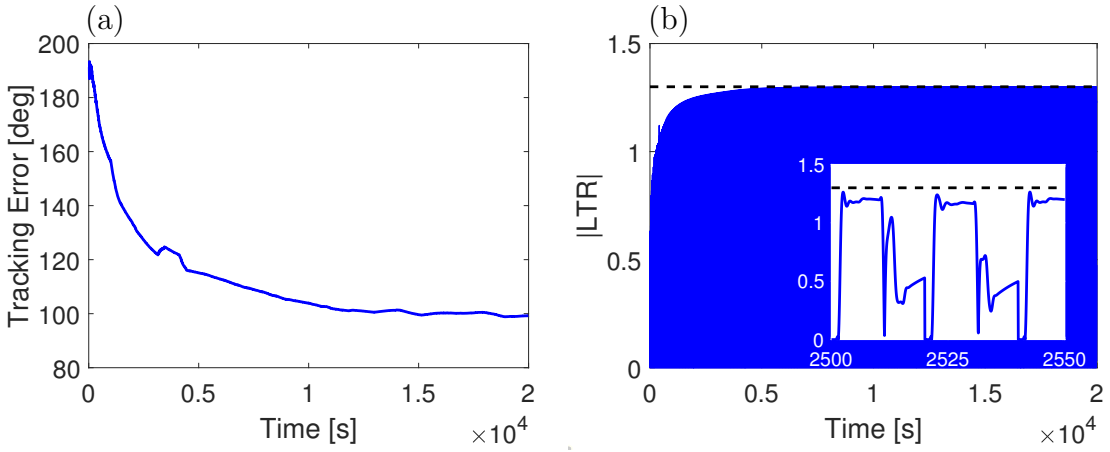

Figure 4.4: Learning algorithm application to vehicle rollover avoidance based on high-fidelity *CarSim* model. (a) Tracking error profile during learning. (b) LTR profile during learning.

We test the vehicle system augmented with the reference governor using the stan-

dard sine-and-dwell profile [132] at different phases of learning. The corresponding modified SW inputs and vehicle responses are shown in Figures 4.5 and 4.6.

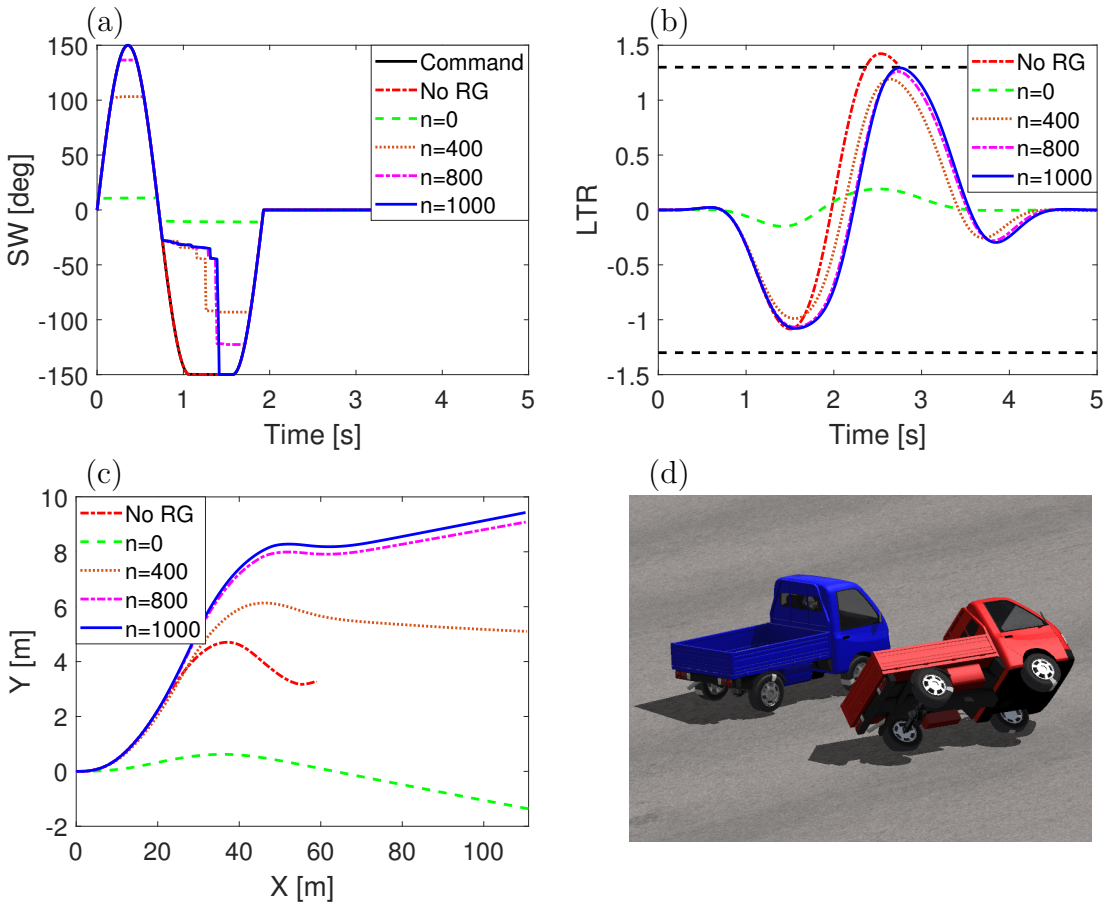

Figure 4.5: (a) SW trajectories, (b) LTR responses, and (c) vehicle trajectories on the (x, y)-plane for the sine-and-dwell test at different phases of learning. (d) CarSim visualizations of the sine-and-dwell test for no RG case (red) and $n = 1000$ case (blue) at $t = 2.2$ (s).

Without the reference governor, the original sine-and-dwell SW profile causes the vehicle to roll over. The augmentation with the reference governor is able to protect the vehicle from rollover accidents. At early phases of learning, the reference governor makes relatively large modifications to the SW command and results in conservative responses. As learning progresses, the reference governor is able to track the command with decreased error while maintaining constraint satisfaction, as shown in Figure 4.5(b).

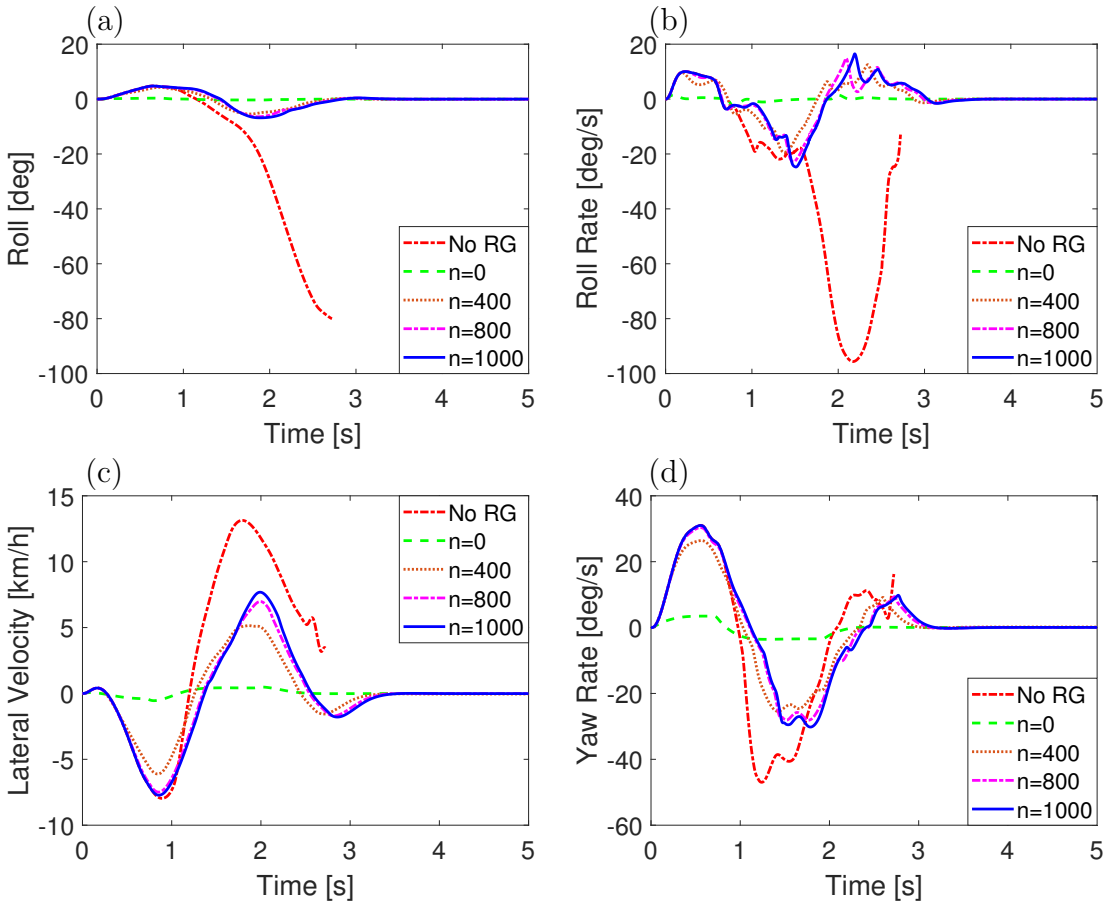

Figure 4.6: The (a) roll angle, (b) roll rate, (c) lateral velocity, and (d) yaw rate responses for the sine-and-dwell test corresponding to different phases of learning.

4.5 Fuel Truck Dynamic Model under Sloshing Effects

We now consider an application of the proposed learning reference governor (LRG) to fuel truck rollover avoidance. In what follows, we introduce models to represent the roll dynamics of a fuel truck with a partially filled tank. These models were originally proposed in [139, 140], where sloshing dynamics of liquid fuel in the tank are accounted for and modeled by an equivalent trammel pendulum. We summarize them here for the sake of completeness.

4.5.1 Equivalent trammel pendulum model of liquid sloshing

A trammel pendulum model, shown in Figure 4.7, is used to describe the sloshing dynamics of liquid fuel in the tank. In particular, the model divides the liquid mass m_l into two parts: the fixed mass m_f, which is fixed relative to the tank, and the pendulum mass m_p, which accounts for the dynamic motion of liquid. The pivot A of

the pendulum can move along the z_p axis, and the pivot B can move along the y_p axis. The rod linking pivots A, B and the pendulum mass m_p is assumed to be massless and rigid, i.e., the lengths a_p and b_p are constants. Consequently, this pendulum system has only one degree of freedom, which can be defined by the pendulum angle θ in Figure 4.7. This angle θ is related to the tilt angle of liquid free surface and is assumed to be measured (e.g., using a combination of wave gauges, and level and optical sensors [143]) for our LRG implementation. The following equation of motion for θ in the tank-fixed inertia frame can be derived [140],

$$\ddot{\theta}(a_p^2 \sin^2\theta + b_p^2 \cos^2\theta) + \frac{1}{2}\dot{\theta}^2(a_p^2 - b_p^2)\sin 2\theta + g b_p \cos\theta = 0, \tag{4.38}$$

where $g = 9.81$ (m/s) is the acceleration due to gravity.

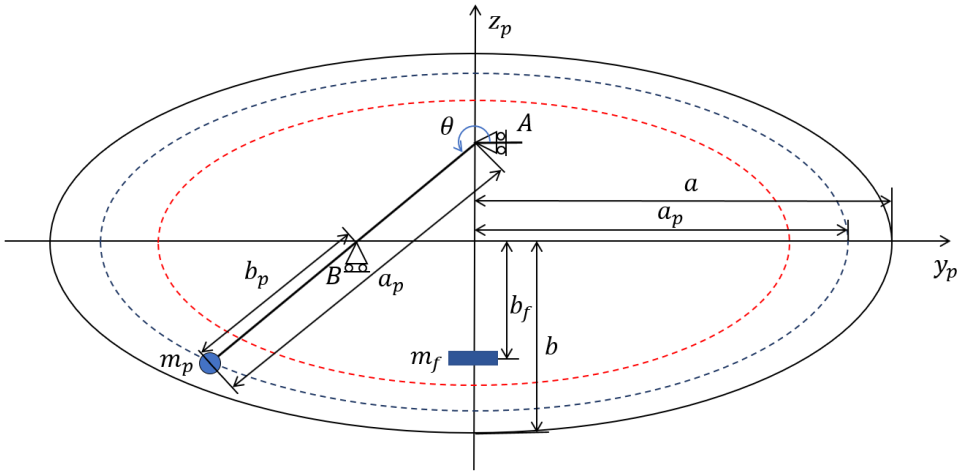

Figure 4.7: Diagram of the trammel pendulum model.

For a given fuel tank with a given liquid fill ratio, the trammel pendulum parameters m_f, m_p, a_p and b_p are estimated according to,

$$m_f = m_l - m_p, \tag{4.39}$$

$$m_p = (m_1 + m_2\Delta + m_3\Lambda + m_4\Delta^2 + m_5\Delta\Lambda + m_6\Lambda^2 + m_7\Delta^3 + m_8\Delta^2\Lambda + m_9\Delta\Lambda^2)\, m_l, \tag{4.40}$$

$$a_p = \Lambda\, b_p, \tag{4.41}$$

$$b_p = (b_1 + b_2\Delta + b_3\Lambda + b_4\Delta^2 + b_5\Delta\Lambda + b_6\Lambda^2 + b_7\Delta^3 + b_8\Delta^2\Lambda + b_9\Delta\Lambda^2)\, b, \tag{4.42}$$

where Δ is the liquid fill ratio, which is defined as the ratio between the height of

liquid's free surface and the height of the tank, Λ is the ratio between the tank's width and its height, i.e., $\Lambda = \frac{b}{a}$, m_l is the liquid mass, and b is half of the height of the tank. The coefficients m_1 to m_9 and b_1 to b_9 can be determined by fitting the dynamics of this pendulum model to liquid sloshing dynamics simulated by high-fidelity fluid simulation software (such as ANSYS Fluent).

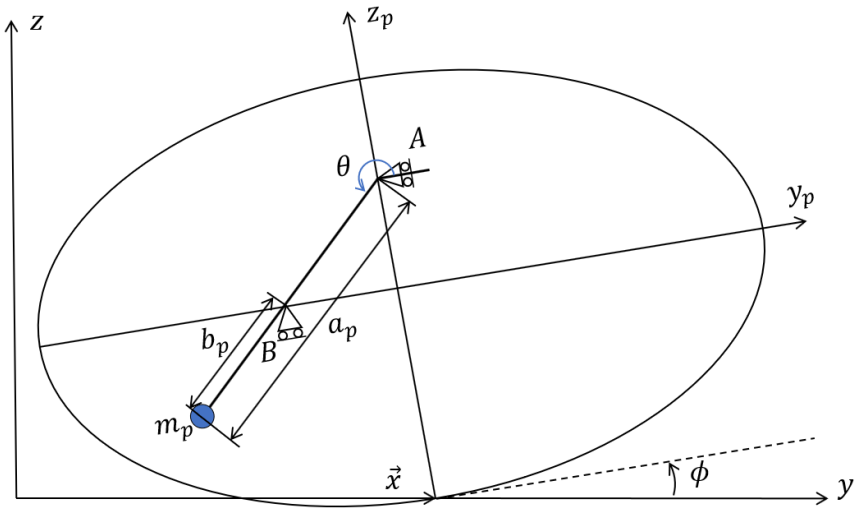

Figure 4.8: Illustration of motion of the trammel pendulum system.

Now taking into account the motion of the tank when the truck is making a turn (illustrated in Figure 4.8), the following equations of motion of the trammel pendulum system in the tank-fixed non-inertia frame can be derived [140],

$$\begin{aligned}
&\ddot{\theta}(a_p^2 \sin^2\theta + b_p^2 \cos^2\theta) + \ddot{\phi}(a_p b_p + a_p b \sin\theta) \\
&+ \dot{\phi}^2\Big[\frac{1}{2}(a_p^2 - b_p^2)\sin 2\theta - b_p b \cos\theta\Big] + \frac{1}{2}\dot{\theta}^2(a_p^2 - b_p^2)\sin 2\theta \\
&- \ddot{x}(a_p \sin\theta\cos\phi + b_p \cos\theta \sin\phi) + g(b_p \cos\theta\cos\phi - a_p \sin\theta \sin\phi) = 0,
\end{aligned} \tag{4.43}$$

$$\begin{aligned}
&\ddot{\phi}\Big[(b + b_p\sin\theta)^2 + a_p^2\cos^2\theta\Big] + 2\dot{\theta}\dot{\phi}\Big[\frac{1}{2}(b_p^2 - a_p^2)\sin 2\theta + b_p b\cos\theta\Big] \\
&+ \ddot{\theta}(a_p b_p + a_p b\sin\theta) - \ddot{x}(b\cos\phi + a_p\cos\theta\sin\phi + b_p\sin\theta\cos\phi) \\
&+ \dot{\theta}^2 a_p b\cos\theta - g\Big[(b + b_p\sin\theta)\sin\phi - a_p\cos\theta\cos\phi\Big] = 0,
\end{aligned} \tag{4.44}$$

where ϕ denotes the roll angle of the tank, which is also the roll angle of the truck, and x represents the translation of the bottom point of the tank in the horizontal direction.

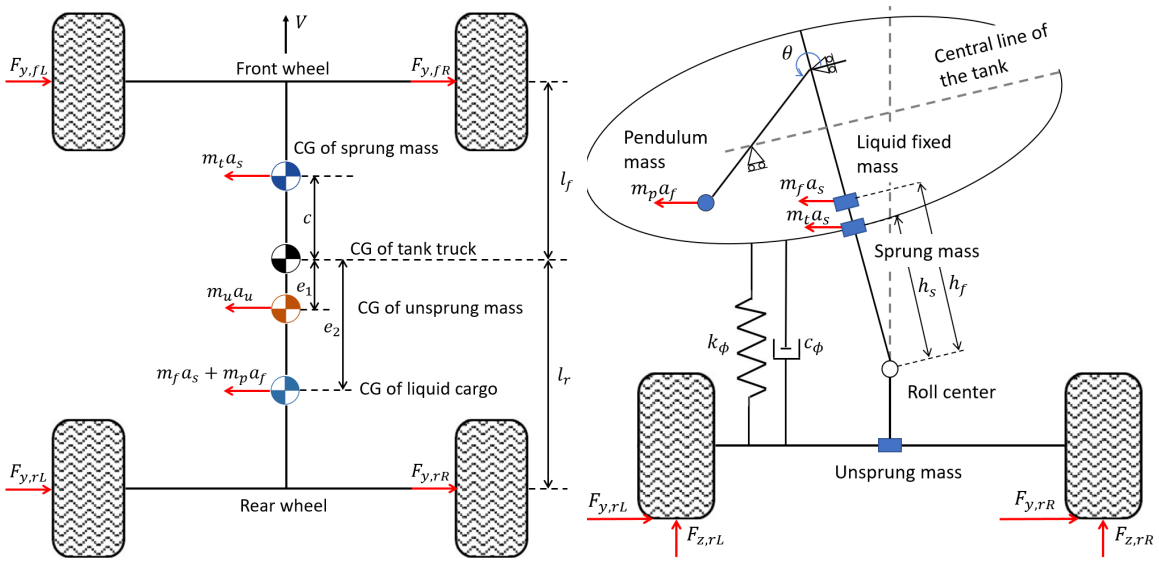

Figure 4.9: Top view and back view of the tank truck dynamic model.

4.5.2 Tank truck dynamic model

The model representing the tank truck dynamics is illustrated in Figure 4.9. Firstly, considering forces acting on the truck in the lateral direction and according to Newton's second law, in the earth-fixed inertia frame we have,

$$(m_t + m_f)a_s + m_u a_u + m_p a_f = F_{y,f} + F_{y,r}, \tag{4.45}$$

where m_t is the sprung mass of the tank truck without load, m_u is the unsprung mass of the truck, m_f and m_p are defined as above. a_s, a_u, and a_f are the lateral acceleration of the sprung mass, the unsprung mass, and the pendulum mass m_p, respectively. $F_{y,f} = F_{y,fL} + F_{y,fR}$ and $F_{y,r} = F_{y,rL} + F_{y,rR}$ are the front tire cornering force and rear tire cornering force, respectively.

The lateral accelerations of the sprung mass and the unsprung mass can be expressed as [140],

$$\begin{aligned} a_s &= V(\dot{\beta} + r) - h_s\ddot{\phi} + c\dot{r}, \\ a_u &= V(\dot{\beta} + r) - e_1\dot{r}, \end{aligned} \tag{4.46}$$

where V is the vehicle's driving speed and is treated as a parameter, β is the vehicle's slip angle, r is the vehicle's yaw rate, h_s (shown in Figure 4.9) is the distance between the roll center and center of gravity (CG) of the sprung mass, c (shown in Figure 4.9) is the longitudinal distance between the CG of the sprung mass and that of the tank truck, and e_1 (shown in Figure 4.9) is the longitudinal distance between the CG of the

unsprung mass and that of the tank truck. The lateral acceleration of the pendulum mass can be expressed as [140],

$$a_f = \ddot{x} - \mathcal{D}_1\ddot{\phi} - \mathcal{D}_2\ddot{\theta} + 2\mathcal{D}_3\dot{\phi}\dot{\theta} + \mathcal{D}_4\dot{\phi}^2 + \mathcal{D}_5\dot{\theta}^2, \tag{4.47}$$

where $\ddot{x}$ is defined as in (4.43) and satisfies $\ddot{x} = V\dot{\beta}$, and

$$\begin{aligned}
\mathcal{D}_1 &= b\cos\phi + a_p\cos\theta\sin\phi + b_p\sin\theta\cos\phi,\\
\mathcal{D}_2 &= a_p\sin\theta\cos\phi + b_p\cos\theta\sin\phi,\\
\mathcal{D}_3 &= a_p\sin\theta\sin\phi - b_p\cos\theta\cos\phi,\\
\mathcal{D}_4 &= b\sin\phi - a_p\cos\theta\cos\phi + b_p\sin\theta\sin\phi,\\
\mathcal{D}_5 &= -a_p\cos\theta\cos\phi + b_p\sin\theta\sin\phi.
\end{aligned} \tag{4.48}$$

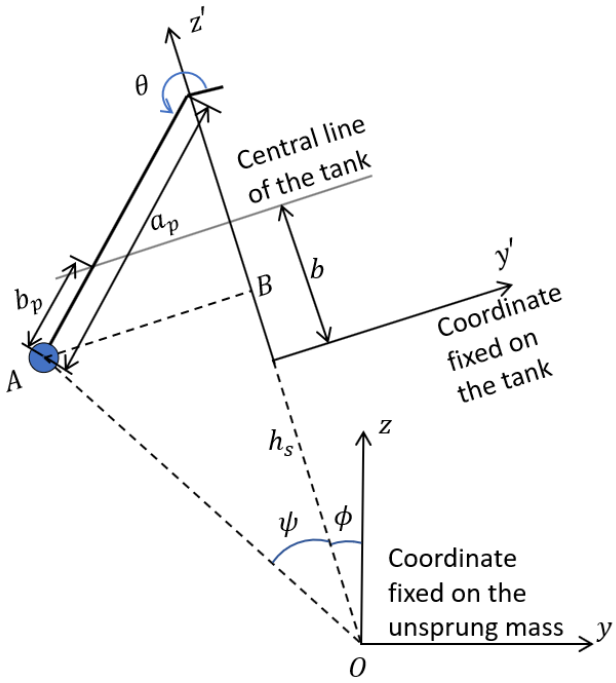

Figure 4.10: Illustration of the rotational dynamics of the trammel pendulum system about the roll center.

The roll angle of the pendulum mass about the roll center, $\angle$, can be expressed as (see Figure 4.10),

$$\angle = \phi + \psi, \tag{4.49}$$

where ϕ is defined as above and ψ satisfies

$$\psi = \arctan\frac{\mathrm{AB}}{\mathrm{BO}} = \arctan\frac{-a_p\cos\theta}{h_s + b + b_p\sin\theta}. \tag{4.50}$$

Then, the roll rate and roll angular acceleration in the earth-fixed inertia frame can be expressed as follows [140],

$$\dot{\angle} = \dot{\phi} + \dot{\psi}, \qquad \ddot{\angle} = \ddot{\phi} + \ddot{\psi}, \tag{4.51}$$

where

$$\dot{\psi} = \frac{\mathcal{E}_2 \dot{\theta}}{\mathcal{E}_1}, \qquad \ddot{\psi} = \frac{\mathcal{E}_4 \ddot{\theta} - \mathcal{E}_5 \dot{\theta}^2}{\mathcal{E}_3}, \tag{4.52}$$

in which

$$\begin{aligned}
\mathcal{E}_1 &= (h_s + b + b_p \sin\theta)^2 + (a_p \cos\theta)^2, \\
\mathcal{E}_2 &= a_p \Big[(h_s + b) \sin\theta + b_p \Big], \\
\mathcal{E}_3 &= \mathcal{E}_1^2, \quad \mathcal{E}_4 = \mathcal{E}_1 a_p \Big[(h_s + b) \sin\theta + b_p \Big], \\
\mathcal{E}_5 &= 2 a_p \cos\theta \Big[(h_s + b) \sin\theta + b_p \Big] \\
&\quad \times \Big[b_p (h_s + b + b_p \sin\theta) - a_p^2 \sin\theta \Big] - \mathcal{E}_1 a_p (h_s + b) \cos\theta.
\end{aligned} \tag{4.53}$$

Furthermore, according to Newton's second law for rotation and using the expressions in (4.51), the vehicle's roll moment balance and yaw moment balance can be expressed as follows [140],

$$\begin{aligned}
&I_z \dot{r} - I_{xz} \ddot{\phi} - m_f a_s e_2 - m_p a_f e_2 - I_{xzp} (\ddot{\phi} + \ddot{\psi}) + I_{zzp} \dot{r} \\
&- I_{xyp} (\dot{\phi} + \dot{\psi})^2 - I_{yzp} r (\dot{\phi} + \dot{\psi}) = F_{y,f} l_f - F_{y,r} l_r,
\end{aligned} \tag{4.54}$$

$$\begin{aligned}
&I_x \ddot{\phi} - I_{xz} \dot{r} + m_t h_s V (\dot{\beta} + r) + m_f h_f a_s + (h_s + b) m_p a_f \\
&+ I_{xxp} (\ddot{\phi} + \ddot{\psi}) - I_{xzp} \dot{r} + I_{xyp} r (\dot{\phi} + \dot{\psi}) + I_{yzp} r^2 \\
&= -k_\phi \phi - c_\phi \dot{\phi} + \phi \Big(m_t h_s g + m_f h_f g + (h_s + b) m_p g \Big) - m_p g a_p \cos\theta,
\end{aligned} \tag{4.55}$$

where l_f (resp. l_r) is the longitudinal distance between the CG of the tank truck and the front wheel axle (resp. the rear wheel axle), e_2 (shown in Figure 4.9) is the longitudinal distance between the CG of the tank truck and that of the liquid tank, h_f (shown in Figure 4.9) is the vertical distance between the CG of the liquid fixed mass m_f and the roll center, k_ϕ is the suspension roll stiffness, and c_ϕ is the suspension roll damping. Moreover, I_{xxp} and I_{zzp} are the moments of inertia of the pendulum mass m_p about the x-axis and z-axis, respectively; I_{xyp}, I_{xzp}, and I_{yzp} are the products of inertia of the pendulum mass about the x- and y-axes, the x- and z-axes, and the y- and z-axes, respectively; and I_x, I_z, and I_{xz} are determined according to the parallel

axis theorem of the moment of inertia as follows [140],

$$\begin{aligned} I_x &= I_{xxs} + I_{xxf} + m_t h_s^2 \\ I_z &= I_{zzs} + I_{zzu} + I_{zzf} + m_t c^2 + m_u e_1^2, \\ I_{xz} &= I_{xzs} + I_{xzf} + m_t h_s c, \end{aligned} \tag{4.56}$$

where I_{xxs} and I_{zzs} (resp. I_{xxf} and I_{zzf}) are the moments of inertia of the sprung mass m_t (resp. the liquid fixed mass m_f) about the x-axis and z-axis, respectively; I_{zzu} is the moment of inertia of the unsprung mass m_u about the z-axis; and I_{xzs} and I_{xzf} are the product of inertia of the sprung mass and that of the liquid fixed mass about the x- and z-axes.

The tire cornering forces $F_{y,f}$ and $F_{y,r}$ are determined according to the magic formula [109],

$$F_{y,i} = \mathcal{D} \sin\Big(\mathcal{C} \arctan\Big(\mathcal{B}\alpha_i - \mathcal{E}\big(\mathcal{B}\alpha_i - \arctan \mathcal{B}\alpha_i\big)\Big)\Big), \tag{4.57}$$

with $i \in \{f, r\}$, where $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ are constant parameters to be fitted, and α_f and α_r are the front and rear tire sideslip angles, computed as

$$\begin{aligned} \alpha_f &= \delta_f - \arctan\left(\frac{V\beta + r l_f}{V}\right), \\ \alpha_r &= \delta_r - \arctan\left(\frac{V\beta - r l_r}{V}\right), \end{aligned} \tag{4.58}$$

where δ_f is the front wheel steering angle, and δ_r is the rear wheel steering angle and is assumed to be 0 in this chapter (corresponding to a truck with only front wheel steering).

To sum up, equations (4.43)-(4.58) define a 6-order system with the vehicle's roll angle ϕ, roll rate $\dot{\phi}$, slip angle β, yaw rate r, the pendulum angle θ, and angular velocity $\dot{\theta}$ as its states and with the front wheel steering angle δ_f as its scalar input.

We again choose to use the load transfer ratio (LTR) to represent rollover constraints [131], which are defined as:

$$\text{LTR} := \frac{F_{z,R} - F_{z,L}}{mg}, \tag{4.59}$$

where $F_{z,R}$ and $F_{z,L}$ are the total vertical forces on the right-side tires and on the left-side tires, respectively, and $mg = (m_t + m_u + m_l)g$ is the total weight of the vehicle. The LTR measures how much of the vehicle vertical load is concentrated

on one side of the vehicle. In particular, the absolute value of LTR being greater than 1 implies that the tires on one side of the vehicle may have been off the ground. Following [144], the LTR is estimated as,

$$\text{LTR} \approx -\frac{2}{mgW}(k_\phi \phi + c_\phi \dot{\phi}), \tag{4.60}$$

where W is the width of the vehicle.

4.6 Fuel Truck Rollover Avoidance

In this section, we present the results of the simulation study. Firstly, the trammel pendulum model is verified against the FLUENT simulation results in [139]. Then, we present and compare the simulation results of the tank truck dynamic model under no load, solid load, and liquid load scenarios. Finally, we apply learning reference governor (LRG) to protect the tank truck from rollover accidents, and the results during learning process and after learning with different rollover tests are reported.

4.6.1 Validation of modeling liquid sloshing using the trammel pendulum

The fuel sloshing dynamics are represented by an equivalent trammel pendulum model. The parameter values of the trammel pendulum model for generating our simulation results are taken from [139], which correspond to a typical fuel tank in the market.

To verify the fuel sloshing model, we focus on the natural frequency of the fuel sloshing dynamics and the force exerted by the fuel on the tank. The natural frequency of the trammel pendulum is shown in Figure 4.11 (a). Higher tank fill ratio results in higher frequency of the trammel pendulum, and this reasonably accurately matches the FLUENT simulation results in [139]. Next, we consider the maximum force exerted on the tank due to fuel sloshing. In the trammel pendulum model, this force is calculated as [139],

$$F_l = \max_t \; m_p a_p(-\dot{\theta}\sin\theta + \ddot{\theta}\cos\theta). \tag{4.61}$$

Shown in Figure 4.11 (b) is the maximum force exerted by the trammel pendulum on the tank per unit liquid mass as we vary the tank fill ratio. This also matches the FLUENT simulation results in [139].

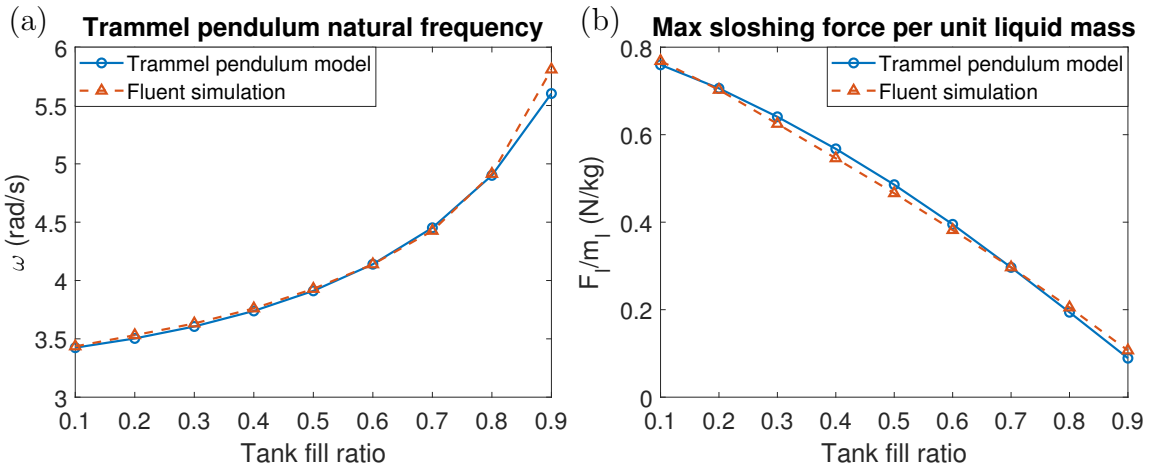

Figure 4.11: (a) Natural frequency of the trammel pendulum and (b) forces exerted by the pendulum on the tank as functions of tank fill ratio compared with FLUENT simulation results presented in [139].

4.6.2 Simulation results of tank truck dynamics

A series of simulations has been performed based on the tank truck dynamic model described in Section 4.5 under different load conditions. Some parameters used in the simulations are shown in Table 4.1.

Parameters	Values
V	25 m/s
m_u	300 kg
h_s	0.8580 m
l_f	1.160 m
l_r	1.750 m
I_{xxs}	1280 kg/m^2
I_{zzs}	2800 kg/m^2
I_{xzs}	0
c_ϕ	7471 N·m·s/rad
k_ϕ	95707 N·m

Table 4.1: Tank truck simulation parameters.

The first scenario we considered is the truck running without any load, i.e., $m_t = 1700$ (kg), $m_l = 0$ (kg). We apply constant steering inputs of 0.02 (rad) and 0.05 (rad) to the system (these steering angles correspond to the actual steering of the front wheels).

From Figure 4.12, we can observe convergence of states to an equilibrium under different steering wheel angles. Note also that larger steering wheel angle results in

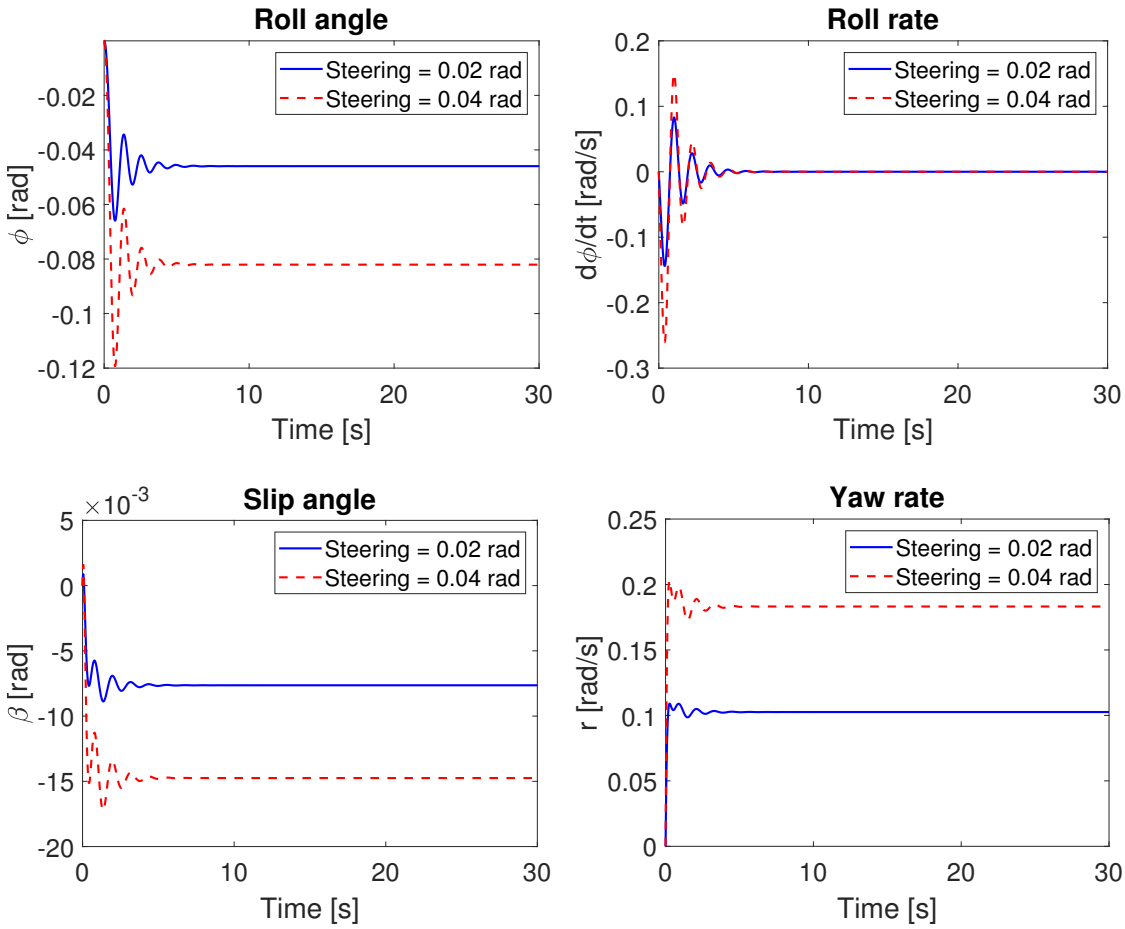

Figure 4.12: The roll angle, roll rate, slip angle, and yaw rate responses for step commands in the no load scenario ($m_t = 1700$ (kg) and $m_l = 0$ (kg)).

larger roll angle, slip angle and yaw rate at steady state.

The second scenario we consider is the case where there is 2000 (kg) solid load only, and the load weight is added to the sprung mass, i.e., $m_t = 3700$ (kg), $m_l = 0$ (kg). Constant steering commands are applied and the results are shown in Figure 4.13.

Comparing Figure 4.12 and Figure 4.13, the solid load case exhibits larger overshoot in roll rate and higher propensity for rolling over. Note that in both the no load and the solid load cases, the response of the pendulum angle θ and the pendulum angular velocity $\dot{\theta}$ are not shown, because in both cases the pendulum mass m_p is 0 (kg).

Next, we consider another scenario where the truck carries a circular liquid tank with radius 1 (m), and the tank is 50% filled. The mass of the liquid is $m_l = 2000$ (kg). The response is shown in Figure 4.14. Due to the effect of liquid fuel sloshing, it takes longer for states to converge to the equilibrium.

The resulting load transfer ratio (LTR) responses of all three scenarios are shown in

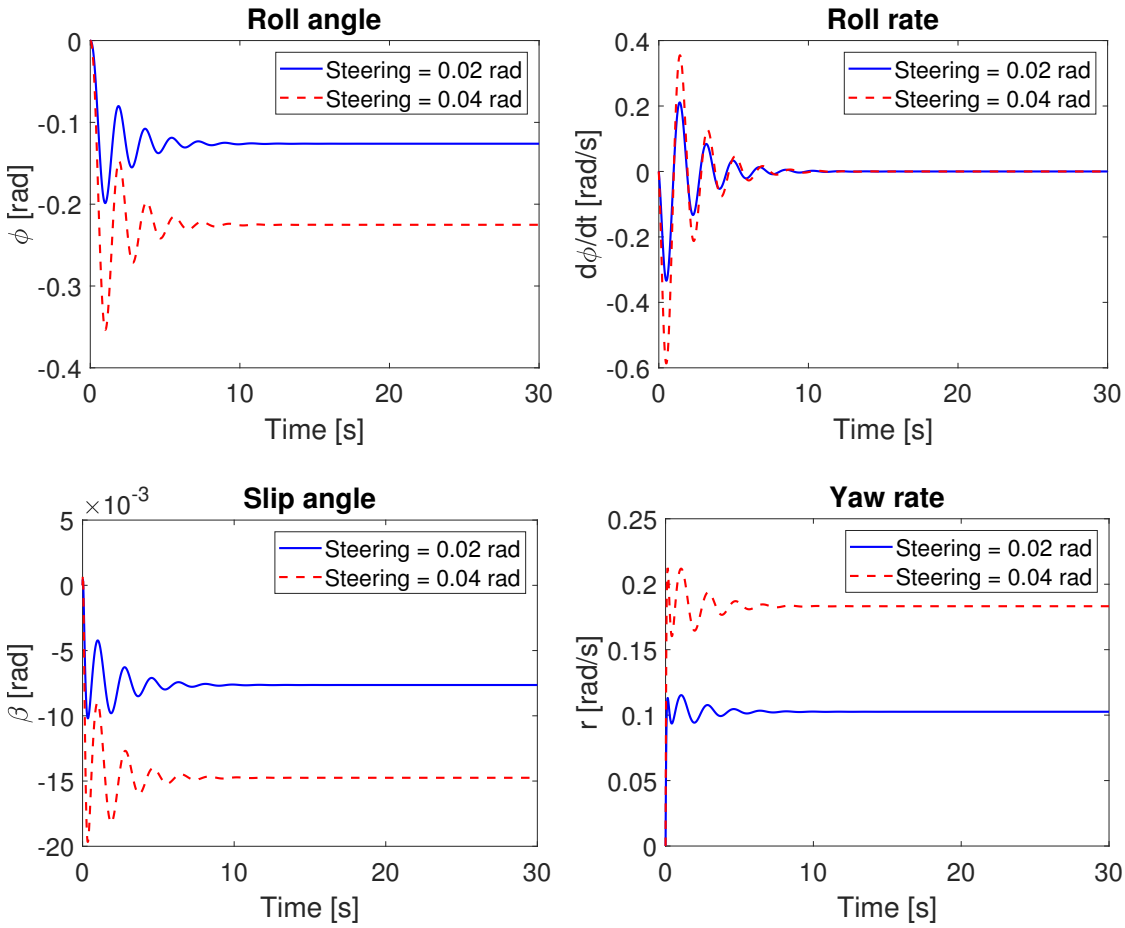

Figure 4.13: The roll angle, roll rate, slip angle, and yaw rate responses for step commands in the solid load scenario ($m_t = 3700$ (kg) and $m_l = 0$ (kg)).

Figure 4.15, which indicates that the vehicle with the liquid load has higher propensity for rolling over under the same steering command.

4.6.3 Applying learning reference governor (LRG) to the tank truck

The liquid load scenario is chosen to demonstrate the effectiveness of our learning algorithm described in Section 4.3. The tank is a circular tank ($a = b = 1$ (m)); the sprung mass is $m_t = 1700$ (kg); the liquid load is $m_l = 2000$ (kg); and the tank fill ratio is $\Delta = 0.5$. Other parameters are as listed in Table 4.1.

To implement LRG, the Hölder constants L and β in (4.5) are required. With $\beta > 1$, the estimated $\overline{D}$ by the LRG in (4.17) can be sensitive to small changes in $(\nu, \delta\nu, \delta x)$ due to the nature of the exponent $1/\beta$. This sensitivity to small changes can lead to longer learning time for LRG to achieve its maximal aggressiveness. As a result, in the application of tanker truck, we assume $\beta = 1$ and estimate L in (4.5), which corresponds to estimating the Lipschitz constant of D. The estimation

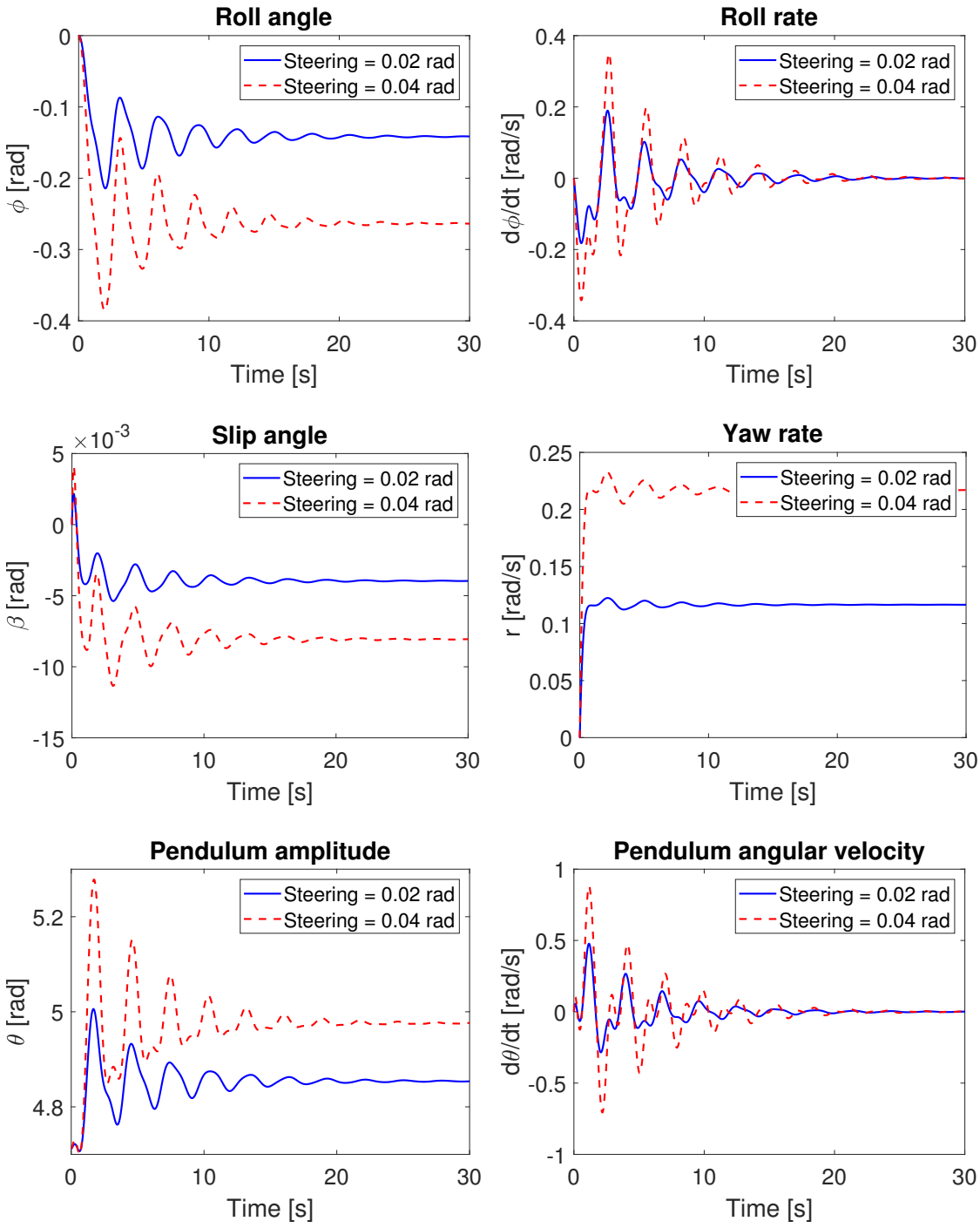

Figure 4.14: The roll angle, roll rate, slip angle, yaw rate, pendulum angle, and pendulum angular velocity responses for step commands in the liquid load scenario ($m_t = 1700$ (kg) and $m_l = 2000$ (kg)).

of a Lipschitz constant is a common problem in optimization. For example, [145] and [146] use sampled or ordered evaluation points to construct an under-estimate of a Lipschitz constant, and a Lipschitz constant is obtained by multiplying the under-estimate value by a factor greater than 1. References [147] and [148] use order statistics to estimate a Lipschitz constant for univariate functions. Reference [149]

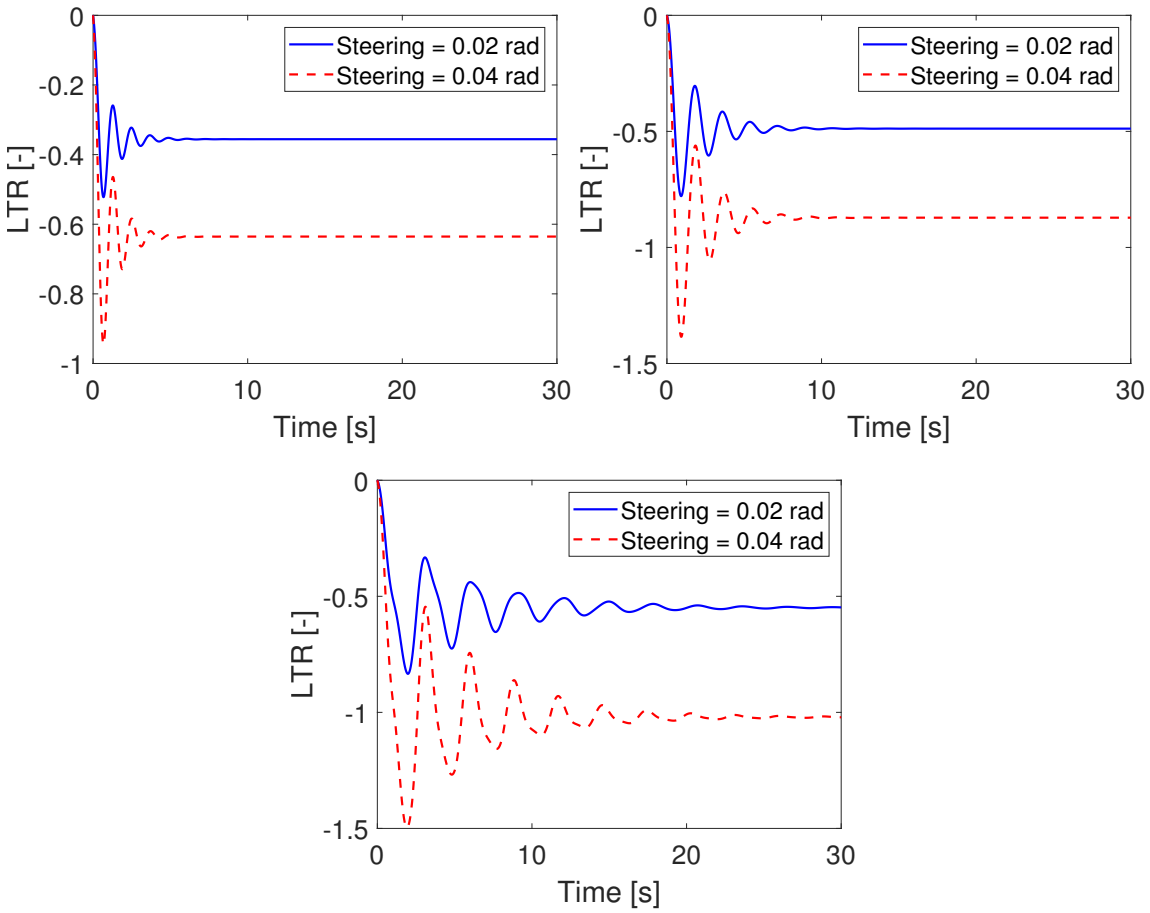

Figure 4.15: The LTR responses of the no load scenario (top left), solid load scenario (top right), and the liquid load scenario (bottom).

proposes an interval-based global optimization algorithm to numerically estimate a Lipschitz constant for arbitrary nonlinear functions. In this chapter, points in the space of $(\nu, \delta\nu, \delta x)$ are sampled and the corresponding derivatives of D are numerically estimated. Specifically, 80 points were sampled and from them, an estimate of L is inferred, which gives $L \geq 0.28$. We therefore consider choices of $L = 0.3$ and $L = 0.5$ to illustrate that a conservative estimate of L is sufficient for our learning algorithm.

Note that in the simulations, a constant ratio k_{δ_f} between the steering wheel angle and the forward tires steering angles is assumed. The control input is the steering wheel angle SW and

$$\delta_f = k_{\delta_f} SW \tag{4.62}$$

where $k_{\delta_f} = 1/20$.

Based on the LTR, the rollover constraints are imposed as

$$-\text{LTR}_{\text{lim}} \leq \text{LTR} \leq \text{LTR}_{\text{lim}}. \tag{4.63}$$

According to Remark 3.1, $|\text{LRT}| > 1$ may lead to wheels lifting off, so we set $\text{LTR}_{\text{lim}} = 1$ in this example. Note that $|\text{LRT}| > 1$ does not indicate rollover happening because additional work is required to move the gravity center of the truck up. However, for heavy-duty trucks carrying hazard liquid, wheels lifting off is already very dangerous, so we set the constraint to be the case where the truck wheels may start lifting off.

During the learning phase, the commands r_n can be randomly generated. However, to better visualize how the learning algorithm gradually pushes the system to its mobility limits, in this example we use a repeated profile for r_n, switching between ± 50 (deg) with a duration of 20 (sec).

The learning progress is illustrated in Figure 4.16. In Figure 4.16(a), the tracking error is defined as the average of $|r(t) - \nu(t)|$ over a past time window of 1000 (sec). At the beginning of learning, the tracking error is relatively high, and LRG operates conservatively with significant modifications of the command. The tracking error gradually decreases as the learning progresses and converges to a low value. Figure 4.16(b) shows the LTR response during learning. As the learning progresses, the vehicle gains the ability to operate with maneuvers that cause the LTR response to reach the constraint boundary but without violating it. Note that constraints are satisfied during the entire learning process.

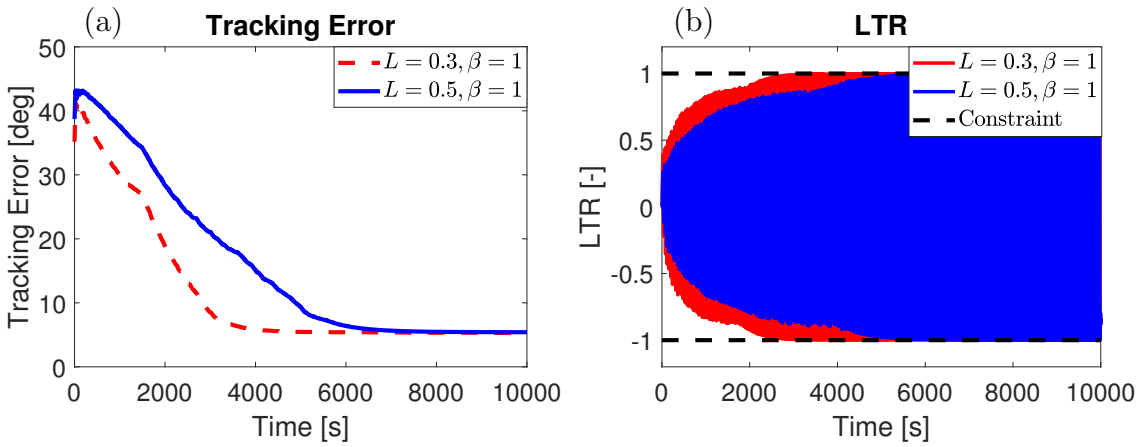

Figure 4.16: Learning algorithm application to fuel truck rollover avoidance based on the tank truck dynamic model developed: (a) Tracking error profile during learning; (b) LTR profile during learning.

After the learning is completed, the vehicle response to step commands is examined. Figure 4.17 shows the response of the system without LRG, with LRG before learning and after learning respectively. Without LRG, the steering wheel command is directly applied to the fuel truck system as shown in Figure 4.17(a), and constraint

violations occur in the resulting LTR response in Figure 4.17(b). Before learning, LRG operates conservatively permitting almost no changes in response to the command. After learning is completed, LRG is significantly less conservative and allows the modified reference input to converge to the original command within a relatively short time period for the same step commands.

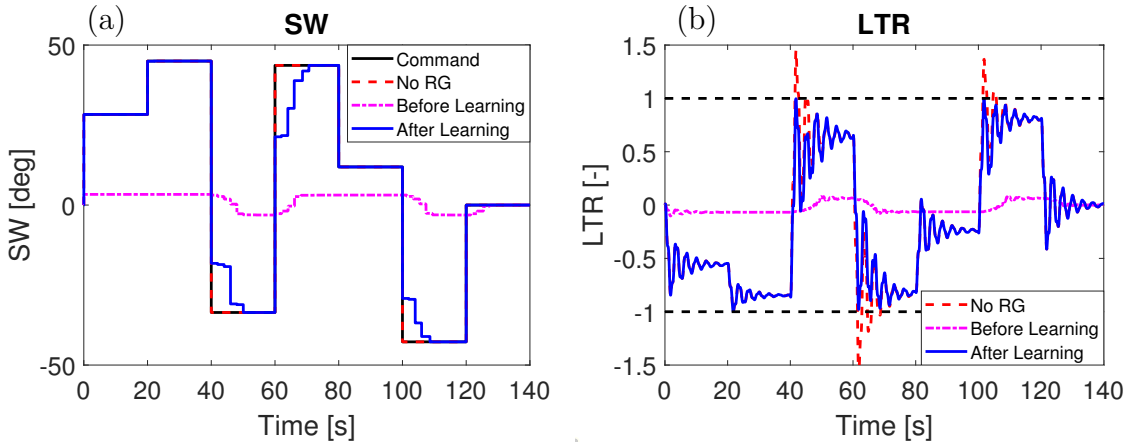

Figure 4.17: (a) Steering angle and (b) LTR responses for the step command test.

We next consider vehicle response to the standard sine-and-dwell steering angle profile. The corresponding steering angle, LTR, and states responses are shown in Figure 4.18 and Figure 4.19.

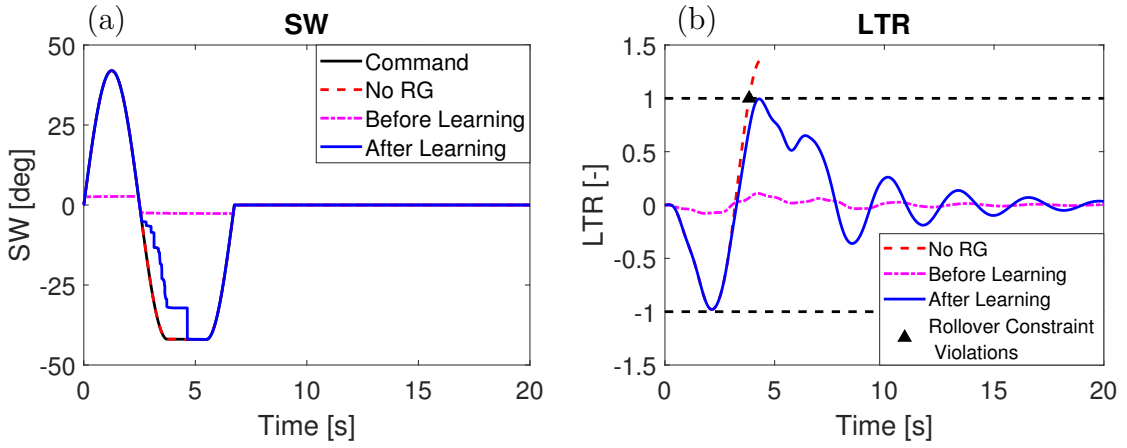

Figure 4.18: (a) Steering angle and (b) LTR responses for the sine-and-dwell test.

Without LRG, the sine-and-dwell steering angle profile causes constraint violations, which may lead to the fuel truck rolling over. By augmenting LRG, rollover

accidents can be avoided. Before learning, LRG makes relatively large modifications to the steering commands and results in conservative responses. After learning is completed, LRG is able to enforce the constraints with significantly less command modifications. Note that our simulations are performed in MATLAB R2019a on an PC with Intel Xeon E3-1246 v3 @ 3.50 GHz CPU and 16 GB RAM. After learning is completed, the dataset $\mathcal{D}$ contains 5600 data points, and the computation of Algorithm 4.2, which generates the reference output given the current state and command, takes on average 0.57 (ms).

4.6.4 Applying LRG to variable load and speed scenarios

In the previous section, LRG is applied to the fuel truck that has specified constant fill ratio and velocity. When the fuel truck's fill ratio and velocity change, the LRG needs to be re-trained to estimate D, or it can be trained at different load conditions and speeds from the beginning, and $\overline{D}$ in (4.17) can be made a function of these additional parameters.

With the latter approach followed, two cross-sections of the resulting $\overline{D}$ are shown in Figure 4.20. In Figure 4.20(a), for the same $\delta\nu$, higher vehicle speed and larger tank fill ratio will result in higher value of $\overline{D}$, which reflects the fact that the truck is easier to rollover. In Figure 4.20(b), $\overline{D}$ increases significantly when the tank fill ratio approaches 1 and the vehicle speed approaches 30 (m/s). Note that $\overline{D}$ is constructed using (4.17) based on collected measurements and L, β. This means $\overline{D}(\nu, \delta\nu, \delta x)$ will be close to $D(\nu, \delta\nu, \delta x)$ when there are measurements near $(\nu, \delta\nu, \delta x)$. On the other hand, $\overline{D}(\nu, \delta\nu, \delta x)$ relies on L and β to extrapolate values and can be conservative when all measurements are far from $(\nu, \delta\nu, \delta x)$. The significantly higher value of $\overline{D}$ shown in Figure 4.20(b) is due to the fact that when the tank fill ratio and vehicle speed are large (e.g., $V = 30$ (m/s) and $\Delta = 0.9$), $\delta\nu = -25$ (deg) results in constraint violations. As no measurements were collected around the point $(\nu = 0, \delta\nu = -25, \delta x = 0)$, the LRG relies on a conservative estimate of $\overline{D}$ at these loads and speeds.

After $\overline{D}$ is made dependent on tank fill ratio and vehicle speed, the sine-and-dwell test is performed again with the truck accelerating or decelerating. Following [150] and guidelines in [151], the acceleration rate is selected as 1 (m/s^2), and the deceleration rate is chosen as -3 (m/s^2). The tank fill ratio is 0.5 and is not changing as we perform the sine-and-dwell test.

In Figure 4.21(a), the sine-and-dwell test is performed while the truck is decelerating. Compared with Figure 4.18, where the truck is driving at a constant speed of

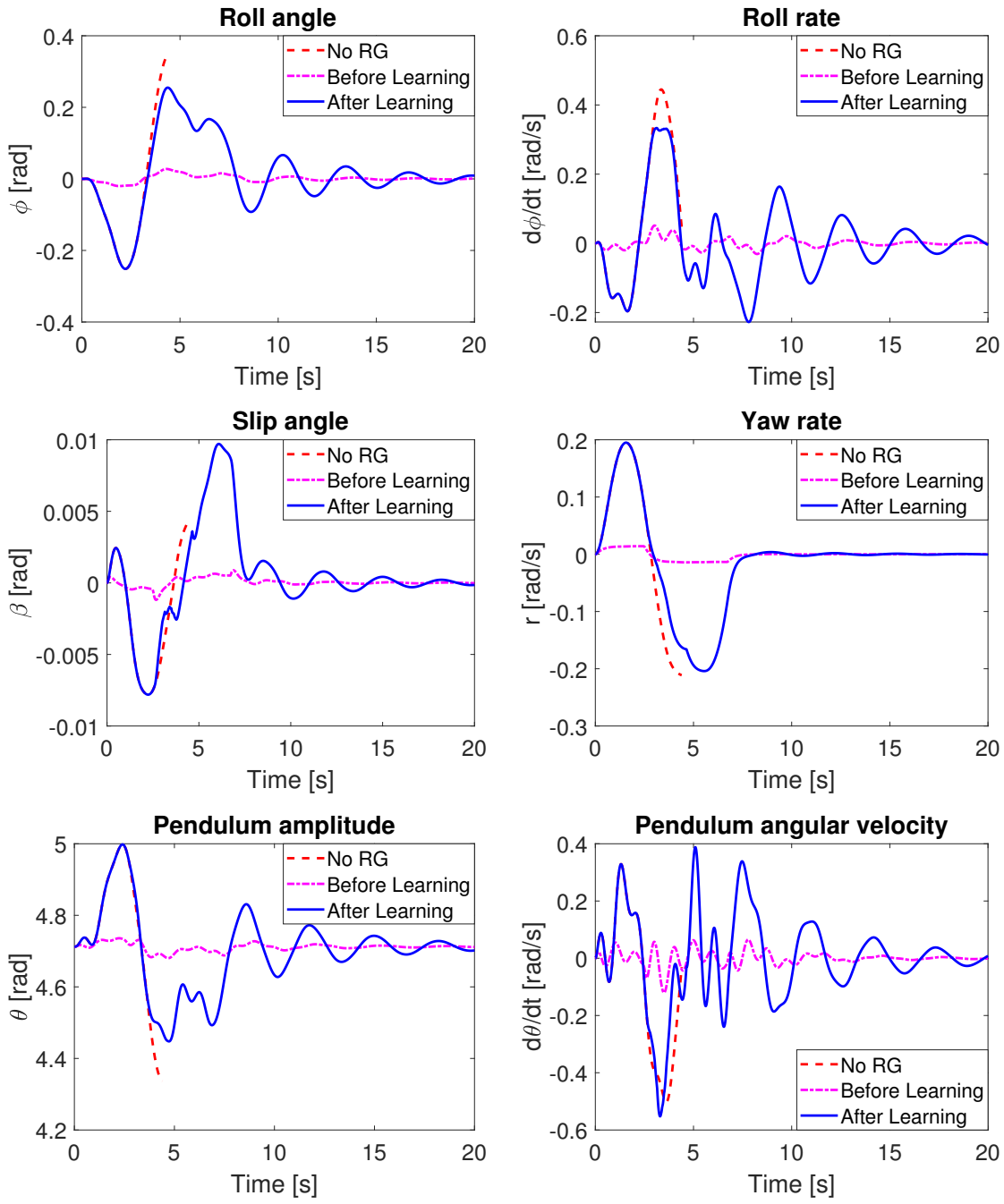

Figure 4.19: The roll angle, roll rate, slip angle, yaw rate, pendulum angle, and pendulum angular velocity responses for the sine-and-dwell test.

25 (m/s), rollover constraint violations occur when the truck is performing positive steering, which is attributed to the larger speed at the beginning. As a consequence, LRG modifies the positive steering command so that the constraints are enforced. Figure 4.21(b) shows the sine-and-dwell test results when the truck is accelerating. LRG makes smaller modifications when responding to the negative steering com-

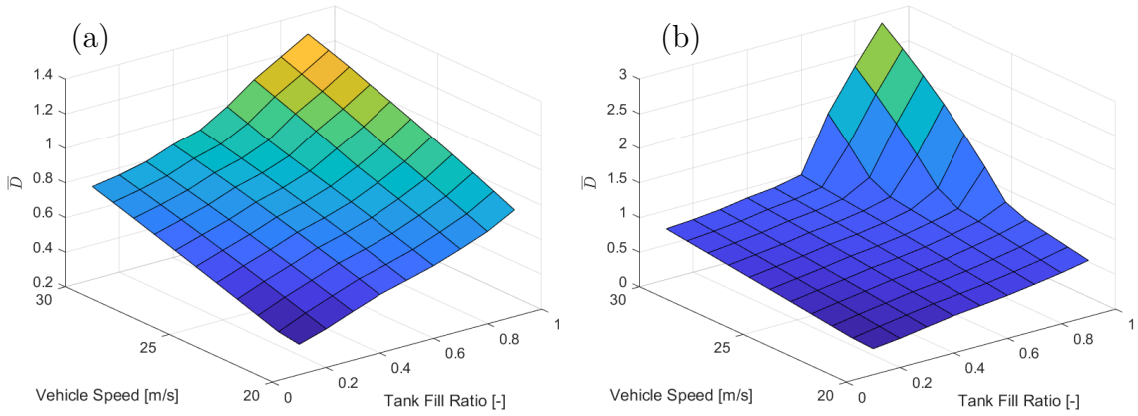

Figure 4.20: Estimated $\overline{D}$ values at different speeds and fill ratios with (a) $\nu = -25$ (deg), $\delta\nu = 25$ (deg), $\delta x = 0$ and (b) $\nu = 0$ (deg), $\delta\nu = -25$ (deg), $\delta x = 0$.

mand compared with Figure 4.18 because of the lower speed before 5 (sec). In either Figure 4.21(a) or (b), LRG is able to guard the fuel truck from violating rollover constraints in presence of speed changes.

4.7 Summary

In this chapter, we presented a learning reference governor (LRG) approach for safety-critical systems to enforce state and control constraints through the modification of the reference command in systems where an accurate model is unavailable. The LRG uses learning to improve the command tracking performance without causing constraint violations. Theoretical guarantees of safety, convergence of the learning algorithm and finite-time convergence of the modified reference command to the original constant reference command have been given. Finally, the applications of the proposed approach to ground vehicle rollover avoidance and fuel truck rollover avoidance under fuel sloshing effects has been considered. Simulation results have been presented that demonstrate LRG can protect the ground vehicle and the fuel truck from rollover and the conservatism can be reduced through learning, pushing this vehicle to its mobility limits. Furthermore, LRG can be configured to support vehicle operation at different tank fill ratios and varying vehicle speeds. A similar approach could be employed to handle variability due to road surface conditions, and bank and incline angles.

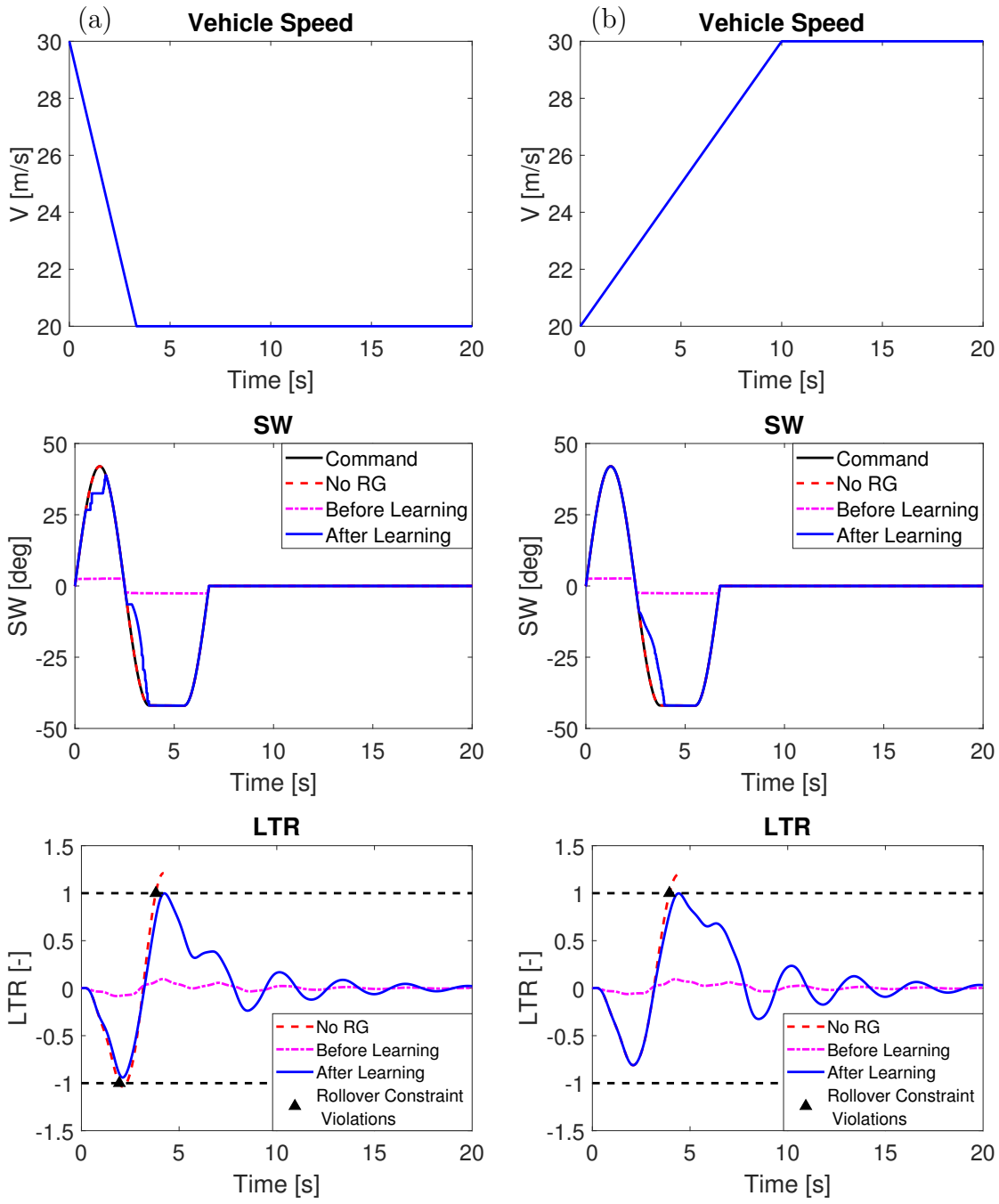

Figure 4.21: Vehicle speed, steering angle, and LTR responses for the sine-and-dwell test when the truck is (a) decelerating from 30 (m/s) to 20 (m/s) at a rate of -3 (m/s^2) and (b) accelerating from 20 (m/s) to 30 (m/s) at a rate of 1 (m/s^2).

CHAPTER V

Conclusion and Future Work

This chapter provides a conclusion to the thesis by summarizing the key contributions and outlining potential areas for future research. In particular, this thesis proposes an interaction-aware behavior planner based on game theory and demonstrates its effectiveness in the forced merge scenarios. Two learning-based safety supervisor schemes, which are able to adapt to different systems/conditions with minimal system information, are proposed with applications to non-safety critical systems and safety critical systems.

5.1 Conclusion

As outlined in Figure 1.4, the main contributions of this thesis are in two areas: behavior planner and safety supervisor.

In order to appropriately account for interactions when sharing the road with human drivers, Chapter II introduces an interaction-aware control strategy, where the interactions between autonomous vehicle and other drivers are formulated based on game theory (more specifically, Leader-Follower Game). A specific application of forced merging is considered where the autonomous vehicle needs to negotiate with other drivers to facilitate its merging. The proposed Leader-Follower Game Controller (LFGC) is able to handle uncertainties in other drivers' intentions by modeling them as a latent state and estimating it online based on observed trajectories. The LFGC integrates the game-theoretic model with an model predictive control based strategies which online-optimizes the performance (e.g., reaching the goal faster) and the liveness (e.g., avoiding collisions). An extensive set of simulation-based evaluations is performed on the LFGC, which includes vehicles controlled by various driver models and following the naturalistic driving dataset. The proposed strategy demonstrates a high success rate in all of these evaluations.

Chapters III and IV then introduce the design of the safety supervisor to further ensure safety for autonomous systems. Model-free designs of such safety supervisors are pursued in these chapters due to unknown/varying operating environments, unavailability of accurate system models, uncertain constraint boundaries, etc. Relying on the integration of prediction and learning/adaptation, these chapters are able to design specific safety supervisory schemes that can guard the system from violating critical limits without sacrificing performances. More specifically, Chapter III introduces a model-free learning safety supervisor design for non-safety critical systems (or constraints), where the violation of the constraints is not desirable but does not lead to severe consequences. To handle non-safety critical systems (or constraints), we leverage Explicit Reference Governor (ERG) theory and design a learning algorithm to gradually evolve the design of the ERG through observed constraint violations. Such a learning algorithm may have constraint violations during learning but will eliminate them after learning is completed. Theoretical properties of the designed learning algorithm have been analyzed and provided, and several examples are provided to illustrate its effectiveness.

Chapter IV focuses on safety-critical systems (or constraints), where constraint violations will lead to catastrophic consequences (e.g., personal injury, property loss, etc.). The design of the safety supervisor is based on the standard reference governor to enforce state and control constraints through modification of the command to the control systems. Under the assumption that there is no accurate model available, a learning algorithm is designed to gradually improve the command tracking performance of the reference governor without causing any constraint violations. Theoretical guarantees are provided in this chapter, which include convergence of the algorithm, guaranteed safety both during learning and after learning is accomplished, as well as finite-convergence of the modified reference command to the original constant reference command. The effectiveness of the algorithm is then demonstrated in case studies of ground vehicle rollover avoidance and fuel (tank) truck rollover avoidance under sloshing effects.

5.2 Future Work

Along the line of work presented in this dissertation, there are still many open research questions that can further advance the state-of-the-art of the current design of the behavior planner and the safety supervisor.

In light of the behavior planner designed based on game theory, there can be mul-

tiple future directions to further explore and improve the performance and adaptivity.

1. The current reward function design is based on certain indicators (such as collision, distance to goal) with associated weights. Such a reward function formulation is a common design in the literature. However, this reward function design may require significant weight tuning to correctly represent drivers' behaviors, and the function may need to be re-tuned when the driving condition changes (e.g., different merge lane lengths, different speed limits, different traffic density, etc.). Bear in mind that each driver may have a different and potentially unique reward, the reward function design can be further advanced by leveraging machine learning techniques (e.g., inverse reinforcement learning), game-theoretic models, and geographical or temporal information.

2. In some situations, there might be a lot of traffic participants that can influence each other's behaviors. The application of game-theoretic models to these scenarios can be limited due to the significant computational demand when solving for the Nash equilibrium with multiple players. This can potentially be improved by considering different game-theoretic frameworks and leveraging neural networks to improve online computational speed.

3. The robustness to perception errors/failures can be further studied. The development of current behavior planner assumes an accurate state measurements/perception. There might be cases where the states of surrounding agents are corrupted by errors, or some agents are missing due to perception errors or in-feasibility. In adversarial cases, it is hard to develop specific safety guarantees. Further studies can be made to leverage game-theoretic models to help detect missing agents/objects (e.g., cross-walk pedestrians blocked by other cars/trucks) or help correct other agents' states.

For the model-free learning-based safety supervisor design, there are various future directions that are applicable to both non-safety critical and safety critical systems (constraints).

1. For both algorithms introduced in Chapters III and IV, it may take a significant amount of time to learn or train a specific safety supervisor. This might be time-consuming and expensive if training is performed directly on hardware. Further investigation on how to improve the learning speed can be performed. Potential approaches can include leveraging machine learning techniques to predict un-

trained state/command regions, designing specific training profiles, analyzing and obtaining a specific form of the target functions before training, etc.

2. The current algorithms will need retraining when systems dynamics change. This can significantly limit the applicability of the proposed algorithm. Further studies can be made to realize the adaptability of the algorithm after learning is accomplished. For example, when only certain parameters of the system have changed, we can transfer the learning results to the newly changed system and quickly adapt to the new dynamics instead of starting the training from scratch.

3. Another limitation of the current work is that it assumes accurate state measurements. Such assumption is not feasible in real world applications. Further studies can be performed to enable the algorithms to handle stochastic systems. Potential directions may include leveraging stochastic optimization techniques and achieving probabilistic guarantees of constraint satisfactions.

4. Additionally, the model-free learning natures of the proposed algorithms do not exclude the model-based methods or model information. Additionally, the proposed methods can be further explored to combine with model-based methods, where certain model information or a simplified model is available, so that the learning speed can be improved. Such a combination can be extremely useful since many models have already been developed for control applications, and using these models can greatly accelerate the learning speed while enabling adaptivity to the actual systems through the proposed learning algorithms.

APPENDIX

APPENDIX A

Derivation of Hölder Continuity on Linear Systems

Here we show that the Hölder continuity assumption, Assumption 4.4, for the function D defined in (4.4) holds true for all stable linear time-invariant systems. In particular, D for stable linear time-invariant systems are actually Lipschitz continuous (Hölder continuous with $\beta = 1$).

Lemma A.1. For linear time-invariant systems in the following form:

$$\dot{x}(t) = Ax(t) + B\nu(t), \tag{A.1a}$$

$$y(t) = Cx(t) + F\nu(t), \tag{A.1b}$$

where A is a stable matrix, i.e., every eigenvalue of A has strictly negative real part. The function D defined in (4.4) for the system (A.1) satisfies the Hölder continuity condition (4.5) with $\beta = 1$ and some finite $L > 0$.

Proof. For any $(\nu_1, \delta\nu_1, \delta x_1), (\nu_2, \delta\nu_2, \delta x_2) \in \mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$, we have

$$
\begin{aligned}
&\big|D(\nu_1, \delta\nu_1, \delta x_1) - D(\nu_2, \delta\nu_2, \delta x_2)\big| \\
=&\big| \sup_{t\in[0,\infty)} \big\|\phi\big(t, x_\nu(\nu_1) + \delta x_1, \nu_1 + \delta\nu_1\big) - y_\nu(\nu_1)\big\| - \\
&\sup_{t\in[0,\infty)} \big\|\phi(t, x_\nu(\nu_2) + \delta x_2, \nu_2 + \delta\nu_2) - y_\nu(\nu_2)\big\| \big| \\
\leq& \sup_{t\in[0,\infty)} \big\|\phi(t, x_\nu(\nu_1) + \delta x_1, \nu_1 + \delta\nu_1) - \\
&\phi(t, x_\nu(\nu_2) + \delta x_2, \nu_2 + \delta\nu_2) - \big(y_\nu(\nu_1) - y_\nu(\nu_2)\big)\big\| \\
=& \sup_{t\in[0,\infty)} \big\|C\psi\big(t, x_\nu(\nu_1 - \nu_2) + (\delta x_1 - \delta x_2), (\nu_1 - \nu_2) + \\
&(\delta\nu_1 - \delta\nu_2)\big) + F\big((\nu_1 - \nu_2) + (\delta\nu_1 - \delta\nu_2)\big) - y_\nu(\nu_1 - \nu_2)\big\| \\
\leq& \sup_{t\in[0,\infty)} \big\|C\psi\big(t, x_\nu(\nu_1 - \nu_2) + (\delta x_1 - \delta x_2), (\nu_1 - \nu_2) + \\
&(\delta\nu_1 - \delta\nu_2)\big)\big\| + \big\|F\big((\nu_1 - \nu_2) + (\delta\nu_1 - \delta\nu_2)\big) - y_\nu(\nu_1 - \nu_2)\big\|,
\end{aligned}
$$

where we have used the superposition property for linear systems and all suprema are finite as A is stable.

Using $x_\nu(\nu) = -A^{-1}B\nu$, $y_\nu(\nu) = (-CA^{-1}B + F)\nu$, $\psi(t, x_0, \nu) = e^{At}x_0 + A^{-1}(e^{At} - I)B\nu$, and the fact that A is a stable matrix, it is easily seen from the above inequality that there exists some finite $L' > 0$ that is independent of $(\nu_1, \delta\nu_1, \delta x_1)$ and $(\nu_2, \delta\nu_2, \delta x_2)$ such that

$$
\big|D(\nu_1, \delta\nu_1, \delta x_1) - D(\nu_2, \delta\nu_2, \delta x_2)\big| \leq L'\big(\|\nu_1 - \nu_2\| + \|\delta\nu_1 - \delta\nu_2\| + \|\delta x_1 - \delta x_2\|\big).
$$

Since $\|\nu\| + \|\delta\nu\| + \|\delta x\|$ defines a norm on $\mathbb{R}^{n_\nu} \times \mathbb{R}^{n_\nu} \times \mathbb{R}^n$, using the equivalence of norms for finite-dimensional vector spaces, we obtain (4.5) with some finite $L > 0$ and exponent $\beta = 1$. Under Euclidean norm, L' can be expressed as $L' = \max\Big(\eta\|C\|, (\eta + 1)\|CA^{-1}\|\|B\| + \|F\|\Big)$, where $\eta = \sup_{t\in[0,\infty)} \|e^{At}\|$. ■

In the next lemma, we show an example of a nonlinear system, to which the corresponding function D in (4.4) satisfies our Hölder continuity Assumption 4.4 and is not necessarily Lipschitz continuous. This example also illustrates the fact that the Hölder continuity assumption made in this chapter, Assumption 4.4, is a weaker and more general assumption than the Lipschitz continuity assumption relied upon in our previous work [86] and [88].

Lemma A.2. For systems in the following form:

$$\dot{x}(t) = Ax(t) + B\nu(t)^{\frac{1}{\beta}}, \tag{A.2a}$$

$$y(t) = Cx(t) + F\nu(t)^{\frac{1}{\beta}}, \tag{A.2b}$$

where $\beta \geq 1$, and A is a stable matrix, and the state space $X \subset \mathbb{R}^n$ is bounded, and $\nu(t) \in \mathbb{R}$ is a scalar, the function D defined in (4.4) for the system (A.1) satisfies the Hölder continuity condition (4.5) with exponent $1/\beta$ and some finite $L > 0$.

Proof. Let $u = \nu^{\frac{1}{\beta}}$. Then we can regard u as the input to the (A.2) and apply the results obtained in Lemma A.1. For any $(\nu_1, \delta\nu_1, \delta x_1), (\nu_2, \delta\nu_2, \delta x_2) \in \mathbb{R} \times \mathbb{R} \times X$, we obtain

$$\big|D(\nu_1, \delta\nu_1, \delta x_1) - D(\nu_2, \delta\nu_2, \delta x_2)\big| \leq L'\big(\|u_1 - u_2\| + \|\delta u_1 - \delta u_2\| + \|\delta x_1 - \delta x_2\|\big),$$

where $u = \nu^{\frac{1}{\beta}}$ and $\delta u = (\nu + \delta\nu)^{\frac{1}{\beta}} - \nu^{\frac{1}{\beta}}$. Substitute u and δu with ν and $\delta\nu$,

$$\begin{aligned}
&\big|D(\nu_1, \delta\nu_1, \delta x_1) - D(\nu_2, \delta\nu_2, \delta x_2)\big| \\
\leq\, & L'\big(\|\nu_1^{\frac{1}{\beta}} - \nu_2^{\frac{1}{\beta}}\| + \|\delta x_1 - \delta x_2\| + \|(\nu_1 + \delta\nu_1)^{\frac{1}{\beta}} - \nu_1^{\frac{1}{\beta}} - (\nu_2 + \delta\nu_2)^{\frac{1}{\beta}} + \nu_2^{\frac{1}{\beta}}\|\big) \\
\leq\, & L'\big(2\|\nu_1^{\frac{1}{\beta}} - \nu_2^{\frac{1}{\beta}}\| + \|\delta x_1 - \delta x_2\| + \|(\nu_1 + \delta\nu_1)^{\frac{1}{\beta}} - (\nu_2 + \delta\nu_2)^{\frac{1}{\beta}}\|\big) \\
\leq\, & L'\big(2\|\nu_1 - \nu_2\|^{\frac{1}{\beta}} + \|\delta x_1 - \delta x_2\| + \|\nu_1 + \delta\nu_1 - \nu_2 - \delta\nu_2\|^{\frac{1}{\beta}}\big) \\
\leq\, & L'\big(3\|\nu_1 - \nu_2\|^{\frac{1}{\beta}} + \|\delta x_1 - \delta x_2\| + \|\delta\nu_1 - \delta\nu_2\|^{\frac{1}{\beta}}\big) \\
\leq\, & L'\big(3\|\nu_1 - \nu_2\|^{\frac{1}{\beta}} + K\|\delta x_1 - \delta x_2\|^{\frac{1}{\beta}} + \|\delta\nu_1 - \delta\nu_2\|^{\frac{1}{\beta}}\big),
\end{aligned}$$

where the second and fourth inequalities are derived from the triangle inequality, and the third inequality is obtained from the inverse triangle inequality with exponent $1/\beta$, and $K = \max_{\delta x_{1,2} \in X} \|\delta x_1 - \delta x_2\|^{1-\frac{1}{\beta}}$.

Note that functions $f(x) = x^{\frac{1}{\beta}}$ with $\beta \geq 1$ is concave on $x \in [0, \infty)$. Then,

$$\begin{aligned}
&\big|D(\nu_1, \delta\nu_1, \delta x_1) - D(\nu_2, \delta\nu_2, \delta x_2)\big| \\
\leq\, & L'\big(a_1\|\nu_1 - \nu_2\| + a_2\|\delta x_1 - \delta x_2\| + a_3\|\delta\nu_1 - \delta\nu_2\|\big)^{\frac{1}{\beta}},
\end{aligned}$$

where $a_1 = \frac{12^\beta}{4}$, $a_2 = \frac{4^\beta K^\beta}{4}$, and $a_3 = 2^{\beta-1}$. Since $a_1\|\nu\| + a_2\|\delta x\| + a_3\|\delta\nu\|$ defines a norm on $\mathbb{R} \times \mathbb{R} \times X$, using the equivalence of norms for finite-dimensional vector spaces, we obtain (4.5) with some finite $L > 0$ and exponent $1/\beta$. L' can be expressed as $L' = \max\Big(\eta\|C\|, (\eta+1)\|CA^{-1}\|\|B\| + \|F\|\Big)$, where $\eta = \sup_{t\in[0,\infty)} \|e^{At}\|$. ∎

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Jenn U. The road to driverless cars: 1925 - 2025, 2016. https://www.engineering.com/story/the-road-to-driverless-cars-1925---2025, Last accessed on 2023-05-21.

[2] The Auto Editors of Consumer Guide. 1957-1959 imperial, 1950s. https://web.archive.org/web/20110612145659/http://auto.howstuffworks.com/1957-1959-imperial9.htm, Last accessed on 2023-05-21.

[3] Ernst Dieter Dickmanns. *Dynamic vision for perception and control of motion*. Springer Science & Business Media, 2007.

[4] 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.

[5] Bernard Marr. Key milestones of waymo - google's self-driving cars, 2018. https://www.forbes.com/sites/bernardmarr/2018/09/21/key-milestones-of-waymo-googles-self-driving-cars/, Last accessed on 2023-05-21.

[6] Keith Barry. Big bets and broken promises: A timeline of tesla's self-driving aspirations, 2021. https://www.consumerreports.org/autonomous-driving/timeline-of-tesla-self-driving-aspirations-a9686689375/, Last accessed on 2023-05-21.

[7] Peter C Baker. Collision course: why are cars killing more and more pedestrians?, 2019. https://www.theguardian.com/technology/2019/oct/03/collision-course-pedestrian-deaths-rising-driverless-cars, Last accessed on 2023-05-21.

[8] Melissa Kirschner. Dude, where's my autonomous car?, 2019. https://semiengineering.com/dude-wheres-my-autonomous-car/, Last accessed on 2023-05-21.

[9] On-Road Automated Driving (ORAD) Committee. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, 2021.

[10] Andrew Tarantola. Mercedes is the first certified level-3-autonomy car company in the US, 2023. https://www.engadget.com/mercedes-first-certified-level-3-autonomy-car-company-us-201021118.html, Last accessed on 2023-05-21.

[11] Eric D. Lawrence. Breaking down the language barrier between autonomous cars and pedestrians, 2018. https://www.freep.com/story/money/cars/2018/10/04/self-driving-robot-car-guidelines/1524894002/, Last accessed on 2023-05-21.

[12] Johannes Deichmann, Eike Ebel, Kersten Heineke, Ruth Heuss, Martin Kellner, and Fabian Steiner. Autonomous driving's future: Convenient and connected. Technical report, McKinsey Center for Future Mobility, 2023.

[13] Tirthankar Bandyopadhyay, Kok Sung Won, Emilio Frazzoli, David Hsu, Wee Sun Lee, and Daniela Rus. Intention-aware motion planning. In *Algorithmic Foundations of Robotics X*, pages 475–491. Springer, 2013.

[14] Chiyu Dong, John M Dolan, and Bakhtiar Litkouhi. Interactive ramp merging planning in autonomous driving: Multi-merging leading PGM (MML-PGM). In *2017 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1–6. IEEE, 2017.

[15] Constantin Hubmann, Jens Schulz, Marvin Becker, Daniel Althoff, and Christoph Stiller. Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction. *IEEE Transactions on Intelligent Vehicles*, 3(1):5–17, 2018.

[16] Nan Li, Anouck Girard, and Ilya Kolmanovsky. Stochastic predictive control for partially observable Markov decision processes with time-joint chance constraints and application to autonomous vehicle control. *Journal of Dynamic Systems, Measurement, and Control*, 141(7), 2019.

[17] 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 Conference on Decision and Control (CDC)*, pages 2181–2187. IEEE, 2019.

[18] Vasileios Lefkopoulos, Marcel Menner, Alexander Domahidi, and Melanie N Zeilinger. Interaction-aware motion prediction for autonomous driving: A multiple model Kalman filtering scheme. *IEEE Robotics and Automation Letters*, 6(1):80–87, 2021.

[19] Alexander G Cunningham, Enric Galceran, Ryan M Eustice, and Edwin Olson. MPDM: Multipolicy decision-making in dynamic, uncertain environments for autonomous driving. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1670–1677. IEEE, 2015.

[20] George E Monahan. State of the art–a survey of partially observable Markov decision processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

[21] Vishnu S Chipade, Qiang Shen, Lixing Huang, Necmiye Ozay, Sze Zheng Yong, and Dimitra Panagou. Safe autonomous overtaking with intention estimation. In *2019 European Control Conference (ECC)*, pages 2050–2057. IEEE, 2019.

[22] Necmiye Ozay, Yunus E Sahin, Zexiang Liu, Kwesi Rutledge, Sze Zheng Yong, and Dimitra Panagou. Intention-aware supervisory control with driving safety applications, 2022. US Patent 11,242,059.

[23] Szilárd Aradi. Survey of deep reinforcement learning for motion planning of autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):740–759, 2020.

[24] David Isele, Reza Rahimi, Akansel Cosgun, Kaushik Subramanian, and Kikuo Fujimura. Navigating occluded intersections with autonomous vehicles using deep reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2034–2039. IEEE, 2018.

[25] Changxi You, Jianbo Lu, Dimitar Filev, and Panagiotis Tsiotras. Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning. *Robotics and Autonomous Systems*, 114:1–18, 2019.

[26] Baiyu Peng, Qi Sun, Shengbo Eben Li, Dongsuk Kum, Yuming Yin, Junqing Wei, and Tianyu Gu. End-to-end autonomous driving through dueling double deep Q-network. *Automotive Innovation*, 4(3):328–337, 2021.

[27] Pin Wang, Ching-Yao Chan, and Arnaud de La Fortelle. A reinforcement learning based approach for automated lane change maneuvers. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1379–1384. IEEE, 2018.

[28] Carl-Johan Hoel, Krister Wolff, and Leo Laine. Automated speed and lane change decision making using deep reinforcement learning. In *2018 Intelligent Transportation Systems Conference (ITSC)*, pages 2148–2155. IEEE, 2018.

[29] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[30] Ali Baheri, Subramanya Nageshrao, H Eric Tseng, Ilya Kolmanovsky, Anouck Girard, and Dimitar Filev. Deep reinforcement learning with enhanced safety for autonomous highway driving. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1550–1555. IEEE, 2020.

[31] Huayi Li, Nan Li, Ilya Kolmanovsky, and Anouck Girard. Energy-efficient autonomous vehicle control using reinforcement learning and interactive traffic simulations. In *2020 American Control Conference (ACC)*, pages 3029–3034. IEEE, 2020.

[32] Dhruv Mauria Saxena, Sangjae Bae, Alireza Nakhaei, Kikuo Fujimura, and Maxim Likhachev. Driving in dense traffic with model-free reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5385–5392. IEEE, 2020.

[33] Guofa Li, Shenglong Li, Shen Li, Yechen Qin, Dongpu Cao, Xingda Qu, and Bo Cheng. Deep reinforcement learning enabled decision-making for autonomous driving at intersections. *Automotive Innovation*, 3(4):374–385, 2020.

[34] Zheng Wu, Liting Sun, Wei Zhan, Chenyu Yang, and Masayoshi Tomizuka. Efficient sampling-based maximum entropy inverse reinforcement learning with application to autonomous driving. *IEEE Robotics and Automation Letters*, 5(4):5355–5362, 2020.

[35] Zhiyu Huang, Jingda Wu, and Chen Lv. Driving behavior modeling using naturalistic human driving data with inverse reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):10239–10251, 2021.

[36] Sascha Rosbach, Xing Li, Simon Großjohann, Silviu Homoceanu, and Stefan Roth. Planning on the fast lane: Learning to interact using attention mechanisms in path integral inverse reinforcement learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5187–5193. IEEE, 2020.

[37] Marcel Menner, Karl Berntorp, Melanie N Zeilinger, and Stefano Di Cairano. Inverse learning for data-driven calibration of model-based statistical path planning. *IEEE Transactions on Intelligent Vehicles*, 6(1):131–145, 2020.

[38] Maxime Bouton, Alireza Nakhaei, Kikuo Fujimura, and Mykel J Kochenderfer. Cooperation-aware reinforcement learning for merging in dense traffic. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3441–3447. IEEE, 2019.

[39] Yeping Hu, Alireza Nakhaei, Masayoshi Tomizuka, and Kikuo Fujimura. Interaction-aware decision making with adaptive strategies under merging scenarios. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 151–158. IEEE, 2019.

[40] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *2016 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971, 2016.

[41] 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.

[42] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *2019 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2090–2096. IEEE, 2019.

[43] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *2020 Conference on Robot Learning (CoRL)*, pages 86–99. PMLR, 2020.

[44] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *2020 European Conference on Computer Vision (ECCV)*, pages 541–556. Springer, 2020.

[45] Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *2022 Conference on Robot Learning (CoRL)*, pages 203–212. PMLR, 2022.

[46] Sangjae Bae, Dhruv Saxena, Alireza Nakhaei, Chiho Choi, Kikuo Fujimura, and Scott Moura. Cooperation-aware lane change maneuver in dense traffic based on model predictive control with recurrent neural network. In *2020 American Control Conference (ACC)*, pages 1209–1216. IEEE, 2020.

[47] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2255–2264, 2018.

[48] Wenshuo Wang, Chang Liu, and Ding Zhao. How much data are enough? a statistical approach with case study on longitudinal driving behavior. *IEEE Transactions on Intelligent Vehicles*, 2(2):85–98, 2017.

[49] Dorsa Sadigh, Shankar Sastry, Sanjit A Seshia, and Anca D Dragan. Planning for autonomous cars that leverage effects on human actions. In *2016 Robotics: Science and Systems (RSS)*, volume 2, 2016.

[50] Nan Li, Dave W Oyler, Mengxuan Zhang, Yildiray Yildiz, Ilya Kolmanovsky, and Anouck R Girard. Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems. *IEEE Transactions on Control Systems Technology*, 26(5):1782–1797, 2017.

[51] 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 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9590–9596. IEEE, 2019.

[52] Qi Dai, Xunnong Xu, Wen Guo, Suzhou Huang, and Dimitar Filev. Towards a systematic computational framework for modeling multi-agent decision-making at micro level for smart vehicles in a smart world. *Robotics and Autonomous Systems*, 144:103859, 2021.

[53] 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.

[54] 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.

[55] 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.

[56] Benjamin Recht. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2018.

[57] Pingan He and Sarangapani Jagannathan. Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(2):425–436, 2007.

[58] Hamidreza Modares, Frank L. Lewis, and Mohammad-Bagher Naghibi-Sistani. Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica*, 50(1):193 – 202, 2014.

[59] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[60] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *2017 Advances in Neural Information Processing Systems (NIPS)*, pages 908–918, 2017.

[61] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. *2018 Advances in Neural Information Processing Systems (NIPS)*, 31, 2018.

[62] Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *2019 AAAI Conference on Artificial Intelligence*, volume 33(01), pages 3387–3395, 2019.

[63] Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:411–444, 2022.

[64] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[65] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[66] James Blake Rawlings, David Q Mayne, and Moritz Diehl. *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI, 2017.

[67] Lukas Hewing, Kim P Wabersich, Marcel Menner, and Melanie N Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:269–296, 2020.

[68] Veronica Adetola, Darryl DeHaan, and Martin Guay. Adaptive model predictive control for constrained nonlinear systems. *Systems & Control Letters*, 58(5):320–326, 2009.

[69] Chris J. Ostafew, Angela P. Schoellig, Timothy D. Barfoot, and Jack Collier. Learning-based nonlinear model predictive control to improve vision-based mobile robot path tracking. *Journal of Field Robotics*, 33(1):133–152, 2016.

[70] Stéphanie Lefevre, Ashwin Carvalho, and Francesco Borrelli. A learning-based framework for velocity control in autonomous driving. *IEEE Transactions on Automation Science and Engineering*, 13(1):32–42, 2016.

[71] Anil Aswani, Humberto Gonzalez, S. Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216 – 1226, 2013.

[72] Chris J. Ostafew, Angela P. Schoellig, and Timothy D. Barfoot. Robust constrained learning-based NMPC enabling reliable mobile robot path tracking. *The International Journal of Robotics Research*, 35(13):1547–1563, 2016.

[73] Juš Kocijan, Roderick Murray-Smith, Carl Edward Rasmussen, and Agathe Girard. Gaussian process model based predictive control. In *2004 American Control Conference (ACC)*, volume 3, pages 2214–2219. IEEE, 2004.

[74] Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. Learning-based model predictive control for safe exploration. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6059–6066. IEEE, 2018.

[75] Martin Guay, Veronica Adetola, and Darryl DeHaan. *Robust and adaptive model predictive control of nonlinear systems*. Institution of Engineering and Technology, 2015.

[76] Tam W Nguyen, Syed Aseem Ul Islam, Adam L Bruce, Ankit Goel, Dennis S Bernstein, and Ilya V Kolmanovsky. Output-feedback RLS-based model predictive control. In *2020 American Control Conference (ACC)*, pages 2395–2400. IEEE, 2020.

[77] Andrew J Taylor and Aaron D Ames. Adaptive safety with control barrier functions. In *2020 American Control Conference (ACC)*, pages 1399–1405. IEEE, 2020.

[78] Andrew J Taylor, Victor D Dorobantu, Hoang M Le, Yisong Yue, and Aaron D Ames. Episodic learning with control lyapunov functions for uncertain robotic systems. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6878–6884. IEEE, 2019.

[79] Douglas A Bristow, Marina Tharayil, and Andrew G Alleyne. A survey of iterative learning control. *IEEE Control Systems Magazine*, 26(3):96–114, 2006.

[80] Mikael Norrlof. An adaptive iterative learning control algorithm with experiments on an industrial robot. *IEEE Transactions on Robotics and Automation*, 18(2):245–251, 2002.

[81] Sadao Kawamura and Norimitsu Sakagami. Analysis on dynamics of underwater robot manipulators based on iterative learning control and time-scale transformation. In *2002 IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1088–1094, 2002.

[82] James Colyar and John Halkias. US highway 101 dataset. *U.S. Federal Highway Administration*, 2005.

[83] Kaiwen Liu, Nan Li, H Eric Tseng, Ilya Kolmanovsky, Anouck Girard, and Dimitar Filev. Cooperation-aware decision making for autonomous vehicles in merge scenarios. In *2021 IEEE Conference on Decision and Control (CDC)*, pages 5006–5012. IEEE, 2021.

[84] Kaiwen Liu, Nan Li, H Eric Tseng, Ilya Kolmanovsky, and Anouck Girard. Interaction-aware trajectory prediction and planning for autonomous vehicles in forced merge scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[85] Kaiwen Liu, Nan Li, Denise Rizzo, Emanuele Garone, Ilya Kolmanovsky, and Anouck Girard. Model-free learning to avoid constraint violations: An explicit reference governor approach. In *2019 American Control Conference (ACC)*, pages 934–940. IEEE, 2019.

[86] Kaiwen Liu, Nan Li, Ilya Kolmanovsky, Denise Rizzo, and Anouck Girard. Model-free learning for safety-critical control systems: A reference governor approach. In *2020 American Control Conference (ACC)*, pages 943–949. IEEE, 2020.

[87] Kaiwen Liu, Nan Li, Ilya Kolmanovsky, Denise Rizzo, and Anouck Girard. Safe learning reference governor: theory and application to fuel truck rollover avoidance. *Journal of Autonomous Vehicles and Systems*, 1(4):041003, 2021.

[88] Kaiwen Liu, Nan Li, Ilya Kolmanovsky, Denise Rizzo, and Anouck Girard. Tanker truck rollover avoidance using learning reference governor. *SAE International Journal of Advances and Current Practices in Mobility*, 3(2021-01-0256):1385–1394, 2021.

[89] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using SUMO. In *2018 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2575–2582. IEEE, 2018.

[90] Chun-Wei Kong, Kaiwen Liu, H Eric Tseng, Ilya Kolmanovsky, and Anouck Girard. Simulation based methodology for assessing forced merging strategies for autonomous vehicles. In *2023 American Control Conference (ACC)*. IEEE, 2023.

[91] Kosuke Ikeya, Kaiwen Liu, Anouck Girard, and Ilya V Kolmanovsky. Learning to satisfy constraints in spacecraft rendezvous and proximity maneuvering: A learning reference governor approach. In *2022 AIAA SCITECH Forum*, page 2514, 2022.

[92] Kosuke Ikeya, Kaiwen Liu, Anouck Girard, and Ilya Kolmanovsky. Learning reference governor for constrained spacecraft rendezvous and proximity maneuvering. *Journal of Spacecraft and Rockets*, pages 1–15, 2023.

[93] Kichun Jo, Junsoo Kim, Dongchul Kim, Chulhoon Jang, and Myoungho Sunwoo. Development of autonomous car—part II: A case study on the implementation of an autonomous driving system based on distributed architecture. *IEEE Transactions on Industrial Electronics*, 62(8):5119–5132, 2015.

[94] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE access*, 8:58443–58469, 2020.

[95] Luca Lambertini. *Differential games in industrial economics*. Cambridge University Press, 2018.

[96] James M Anderson, Kalra Nidhi, Karlyn D Stanley, Paul Sorensen, Constantine Samaras, and Oluwatobi A Oluwatola. *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, 2014.

[97] Jonas Meyer, Henrik Becker, Patrick M Bösch, and Kay W Axhausen. Autonomous vehicles: The next jump in accessibilities? *Research in Transportation Economics*, 62:80–91, 2017.

[98] Tulga Ersal, Ilya Kolmanovsky, Neda Masoud, Necmiye Ozay, Jeffrey Scruggs, Ram Vasudevan, and Gábor Orosz. Connected and automated road vehicles: state of the art and future challenges. *Vehicle System Dynamics*, 58(5):672–704, 2020.

[99] Nan Li, Yu Yao, Ilya Kolmanovsky, Ella Atkins, and Anouck R Girard. Game-theoretic modeling of multi-vehicle interactions at uncontrolled intersections. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):1428–1442, 2020.

[100] Nan Li. *Game-theoretic and set-based methods for safe autonomous vehicles on shared roads*. PhD thesis, University of Michigan, 2021.

[101] Wei Liu, Seong-Woo Kim, Scott Pendleton, and Marcelo H Ang. Situation-aware decision making for autonomous driving on urban road using online pomdp. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1126–1133. IEEE, 2015.

[102] Haoyu Bai, Shaojun Cai, Nan Ye, David Hsu, and Wee Sun Lee. Intention-aware online POMDP planning for autonomous driving in a crowd. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 454–460. IEEE, 2015.

[103] Nan Li, Ilya Kolmanovsky, Anouck Girard, and Yildiray Yildiz. Game theoretic modeling of vehicle interactions at unsignalized intersections and application to autonomous vehicle control. In *2018 American Control Conference (ACC)*, pages 3215–3220. IEEE, 2018.

[104] Constantin Hubmann, Nils Quetschlich, Jens Schulz, Julian Bernhard, Daniel Althoff, and Christoph Stiller. A POMDP maneuver planner for occlusions in urban scenarios. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2172–2179. IEEE, 2019.

[105] I Ge Jin, Bastian Schürmann, Richard M Murray, and Matthias Althoff. Risk-aware motion planning for automated vehicle among human-driven cars. In *2019 American Control Conference (ACC)*, pages 3987–3993. IEEE, 2019.

[106] Karl Berntorp, Tru Hoang, Rien Quirynen, and Stefano Di Cairano. Control architecture design for autonomous vehicles. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 404–411. IEEE, 2018.

[107] Omveer Sharma, Nirod C Sahoo, and Niladri B Puhan. Recent advances in motion and behavior planning techniques for software architecture of autonomous vehicles: A state-of-the-art survey. *Engineering applications of artificial intelligence*, 101:104211, 2021.

[108] Subramanya Nageshrao, Yousaf Rahman, Vladimir Ivanovic, Mrdjan Jankovic, Eric Tseng, Michael Hafner, and Dimitar Filev. Robust AI driving strategy for autonomous vehicles. In *AI-enabled Technologies for Autonomous and Connected Vehicles*, pages 161–212. Springer, 2022.

[109] Rajesh Rajamani. *Vehicle Dynamics and Control*. Springer Science & Business Media, 2011.

[110] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1094–1099. IEEE, 2015.

[111] Haoran Song, Wenchao Ding, Yuxuan Chen, Shaojie Shen, Michael Yu Wang, and Qifeng Chen. Pip: Planning-informed trajectory prediction for autonomous driving. In *2020 European Conference on Computer Vision (ECCV)*, pages 598–614. Springer, 2020.

[112] Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. Covernet: Multimodal behavior prediction using trajectory sets. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14074–14083, 2020.

[113] Iakovos Papadimitriou and Masayoshi Tomizuka. Fast lane changing computations using polynomials. In *2003 American Control Conference (ACC)*, pages 48–53. IEEE, 2003.

[114] Tamer Başar and Geert Jan Olsder. *Dynamic noncooperative game theory*. SIAM, 1998.

[115] Ran Tian, Nan Li, Ilya Kolmanovsky, Yildiray Yildiz, and Anouck R Girard. Game-theoretic modeling of traffic in unsignalized intersection network for autonomous vehicle control verification and validation. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[116] Inseok Hwang, Hamsa Balakrishnan, and Claire Tomlin. State estimation for hybrid systems: Applications to aircraft tracking. *IEE Proceedings-Control Theory and Applications*, 153(5):556–566, 2006.

[117] Jehong Yoo and Reza Langari. A predictive perception model and control strategy for collision-free autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 20(11):4078–4091, 2018.

[118] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805, 2000.

[119] Xiao-Yun Lu and Alexander Skabardonis. Freeway traffic shockwave analysis: exploring the NGSIM trajectory data. In *2007 Annual Meeting of the Transportation Research Board*. Citeseer, 2007.

[120] Florent Altché and Arnaud de La Fortelle. An LSTM network for highway trajectory prediction. In *2017 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 353–359. IEEE, 2017.

[121] Nachiket Deo and Mohan M Trivedi. Multi-modal trajectory prediction of surrounding vehicles with maneuver based LSTMs. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1179–1184. IEEE, 2018.

[122] Marcello Montanino and Vincenzo Punzo. Making NGSIM data usable for studies on traffic flow theory: Multistep method for vehicle trajectory reconstruction. *Transportation Research Record*, 2390(1):99–111, 2013.

[123] Abraham Savitzky and Marcel JE Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964.

[124] Tung Phan-Minh, Forbes Howington, Ting-Sheng Chu, Sang Uk Lee, Momchil S Tomov, Nanxiang Li, Caglayan Dicle, Samuel Findler, Francisco Suarez-Ruiz, Robert Beaudoin, et al. Driving in real life with inverse reinforcement learning. *arXiv preprint arXiv:2206.03004*, 2022.

[125] Emanuele Garone and Marco M Nicotra. Explicit reference governor for constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 61(5):1379–1384, 2016.

[126] Marco M Nicotra and Emanuele Garone. The explicit reference governor: A general framework for the closed-form control of constrained nonlinear systems. *IEEE Control Systems*, 38(4), 2018.

[127] Emanuele Garone, Marco Nicotra, and Lorenzo Ntogramatzidis. Explicit reference governor for linear systems. *International Journal of Control*, 91(6):1415–1430, 2018.

[128] Josef Dick and Friedrich Pillichshammer. Discrepancy theory and quasi-monte carlo integration. *A Panorama of Discrepancy Theory*, pages 539–619, 2014.

[129] Ilya Kolmanovsky and Jing Sun. A multi-mode switching-based command tracking in network controlled systems with pointwise-in-time constraints and disturbance inputs. In *2006 World Congress on Intelligent Control and Automation (WCICA)*, pages 199–204. IEEE, 2006.

[130] Sebastian Bauer, Andre Suchaneck, and Fernando Puente León. Thermal and energy battery management optimization in electric vehicles using Pontryagin's maximum principle. *Journal of Power Sources*, 246:808–818, 2014.

[131] Ricardo Bencatel, Ran Tian, Anouck R Girard, and Ilya Kolmanovsky. Reference governor strategies for vehicle rollover avoidance. *IEEE Transactions on Control Systems Technology*, 26(6):1954–1969, 2017.

[132] National Highway Traffic Safety Administration. FMVSS No. 126 electronic stability control systems. Technical report, U.S. Department of Transportation, 2007.

[133] Emanuele Garone, Stefano Di Cairano, and Ilya Kolmanovsky. Reference and command governors for systems with constraints: A survey on theory and applications. *Automatica*, 75:306 – 328, 2017.

[134] Federal Motor Carrier Safety Administration. Large truck and bus crash facts 2020. Technical report, U.S. Department of Transportation, 2022.

[135] Mohamed I Salem, Victor H Mucino, Mridul Gautam, and Matthew Aquaro. Review of parameters affecting stability of partially filled heavy-duty tankers. *SAE Transactions*, pages 461–476, 1999.

[136] Xinhui Kang, Subhash Rakheja, and Ion Stiharu. Cargo load shift and its influence on tank vehicle dynamics under braking and turning. *International Journal of Heavy Vehicle Systems*, 9(3):173–203, 2002.

[137] H. Norman Abramson, Wen-hwa Chu, and Guido E. Ransleben. Representation of fuel sloshing in cylindrical tanks by an equivalent mechanical model. *American Rocket Society Journal*, 31(12):1697–1705, 1961.

[138] Marcel J Sidi. *Spacecraft dynamics and control: a practical engineering approach*, volume 7. Cambridge University Press, 1997.

[139] Zheng Xue-lian, Li Xian-sheng, and Ren Yuan-yuan. Equivalent mechanical model for lateral liquid sloshing in partially filled tank vehicles. *Mathematical Problems in Engineering*, 2012, 2012.

[140] Xian-sheng Li, Xue-lian Zheng, Yuan-yuan Ren, Yu-ning Wang, and Zhu-qing Cheng. Study on driving stability of tank trucks based on equivalent trammel pendulum for liquid sloshing. *Discrete Dynamics in Nature and Society*, 2013:1–15, 2013.

[141] Nan Li, Ilya V Kolmanovsky, and Anouck Girard. A reference governor for nonlinear systems with disturbance inputs based on logarithmic norms and quadratic programming. *IEEE Transactions on Automatic Control*, 65(7):3207–3214, 2019.

[142] Ardalan Vahidi, Ilya Kolmanovsky, and Anna Stefanopoulou. Constraint handling in a fuel cell system: A fast reference governor approach. *IEEE Transactions on Control Systems Technology*, 15(1):86–98, 2006.

[143] Roman Gabl, Jeffrey Steynor, David IM Forehand, Thomas Davey, Tom Bruce, and David M Ingram. Capturing the motion of the free surface of a fluid stored within a floating structure. *Water*, 11(1):50, 2019.

[144] Selim Solmaz, Martin Corless, and Robert Shorten. A methodology for the design of robust rollover prevention controllers for automotive vehicles: Part 2-active steering. In *2007 American Control Conference (ACC)*, pages 1606–1611. IEEE, 2007.

[145] Roman G Strongin and Yaroslav D Sergeyev. *Global optimization with non-convex constraints: Sequential and parallel algorithms*, volume 45. Springer Science & Business Media, 2013.

[146] János D Pintér. *Global optimization in action: continuous and Lipschitz optimization: algorithms, implementations and applications*, volume 6. Springer Science & Business Media, 2013.

[147] Laurens De Haan. Estimation of the minimum of a function using order statistics. *Journal of the American Statistical Association*, 76(374):467–469, 1981.

[148] GR Wood and BP Zhang. Estimation of the Lipschitz constant of a function. *Journal of Global Optimization*, 8(1):91–103, 1996.

[149] Sebastian A Nugroho, Ahmad F Taha, and Vu Hoang. Nonlinear dynamic systems parameterization using interval-based global optimization: Computing lipschitz constants and beyond. *IEEE Transactions on Automatic Control*, 67(8):3836–3850, 2021.

[150] Guangchuan Yang, Hao Xu, Zhongren Wang, and Zong Tian. Truck acceleration behavior study and acceleration lane length recommendations for metered on-ramps. *International Journal of Transportation Science and Technology*, 5(2):93–102, 2016.

[151] Douglas W Harwood. *Review of truck characteristics as factors in roadway design*, volume 505. Transportation Research Board, 2003.