Intelligent Traffic Control with Connected and Automated Vehicles

Yang Shi
University of Tennessee, Knoxville, yshi38@vols.utk.edu

Follow this and additional works at: https://trace.tennessee.edu/utk_graddiss

Part of the Acoustics, Dynamics, and Controls Commons

Recommended Citation
https://trace.tennessee.edu/utk_graddiss/8143

This Dissertation is brought to you for free and open access by the Graduate School at TRACE: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Doctoral Dissertations by an authorized administrator of TRACE: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.
To the Graduate Council:

I am submitting herewith a dissertation written by Yang Shi entitled "Intelligent Traffic Control with Connected and Automated Vehicles." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Mechanical Engineering.

Zhenbo Wang, Major Professor

We have read this dissertation and recommend its acceptance:

Subhadeep Chakraborty, Jindong Tan, Yunli Shao

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)
Intelligent Traffic Control with Connected and Automated Vehicles

A Dissertation Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Yang Shi
May 2023
Acknowledgements

I would like to express my sincere gratitude to my advisor, Dr. Zhenbo Wang, for his unwavering support, patience, guidance, and mentorship throughout my PhD journey. His expertise and insights have been invaluable, and I am truly grateful for his constant encouragement and belief in me.

I would also like to extend my gratitude to my PhD committee members, Dr. Subhadeep Chakraborty, Dr. Jindong Tan, and Dr. Yunli Shao, for their time and expertise in reviewing my research and offering constructive feedback. Their insights have helped shape my work, and I am honored to have had the opportunity to work with such knowledgeable and supportive individuals.

I would also like to express my appreciation to our collaborators at Oak Ridge National Laboratory for providing me with the resources and opportunities necessary to complete my research. Their support has been crucial to my success, and I am grateful for the opportunity to conduct my work under their auspices.

I would like to acknowledge the support of my family and friends, who have been my source of strength and motivation throughout this process. Their love and support have been invaluable.

Finally, I would like to thank all of the individuals who have contributed to my research, either through direct collaboration or through their support and encouragement. I am grateful for the privilege of having had the opportunity to work with so many talented and dedicated individuals.

Thank you all for your support and for being a part of my journey.
Abstract

The recent advancements in communication technology, transportation infrastructure, computational techniques, and artificial intelligence are driving a revolution in future transportation systems. Connected and Automated Vehicles (CAVs) are attracting a lot of attention due to their potential to reduce traffic accidents, ease congestion, and improve traffic efficiency. This study focuses on addressing the challenges in controlling future CAV-enabled transportation systems. The aim is to develop a framework for the control of CAV-based traffic systems to improve roadway safety, travel efficiency, and energy efficiency. The study proposes new methods for vehicle speed control and traffic signal control at signalized intersections and corridors as well as merging roadways, to increase the understanding of how traffic elements interact and are impacted by individual actors. The vehicle speed control method is based on sequential convex programming (SCP) algorithms, combining the pseudospectral collocation method with line-search and trust-region techniques for optimal solutions with real-time performance and efficient handling of multiple constraints. In terms of on-ramp merging control, the study develops a new merging control approach that balances computational efficiency, solution optimality, and real-time performance for safe merging operations. The traffic signal control framework uses deep reinforcement learning (DRL) with a novel convolutional autoencoder network for a concise representation of traffic information to improve the learning efficiency of the DRL algorithm. The proposed method extends the action space by
including both phase duration and cycle length, allowing for more adaptability to dynamic traffic flow.

This study presents a comprehensive framework for the control of CAV-based traffic system that enhances the positive attributes of CAV technology while minimizing negative effects. The framework will contribute to improving road safety, travel, and energy efficiency while synchronizing CAV motion planning with traffic signal optimization to reduce traffic congestion and idling as well as fuel consumption with guaranteed collision avoidance. This study explores the interface of multiple disciplines including control theory, optimization, machine learning, data analytics, and real-time computation. The results of this study will inform future research in the area of intelligent control of data-rich, interactive systems and will benefit the development of intelligent transportation systems with CAV technologies.
# Table of Contents

1 Introduction 1

1.1 Background ................................................................. 2

1.2 Motivation ................................................................. 3

1.2.1 Study I: Real-Time Control of Connected Vehicles in Signalized Corridors using Pseudospectral Convex Optimization ............. 4

1.2.2 Study II: Pseudospectral Convex Optimization for On-Ramp Merging Control of Connected Vehicles ................................. 5

1.2.3 Study III: A Novel Deep Reinforcement Learning Approach to Traffic Signal Control with Connected Vehicles ......................... 6

1.2.4 Study IV: Data-Driven Optimization Framework for On-Ramp Merging Control with Connected and Automated Vehicles 7

1.3 Research Contributions .................................................. 8

2 Literature Review 12

2.1 Optimal Control of Connected and Automated Vehicles .............. 13

2.1.1 Trajectory Optimization for Vehicles Passing Through Signalized Intersections ................................................................. 14

2.1.2 On-ramp Merging Control .............................................. 17

2.2 Traffic Signal Control ...................................................... 20

2.2.1 Optimization-based Approaches ....................................... 20

2.2.2 Data-driven Approaches ............................................... 24
## 3 Study I: Real-Time Control of Connected Vehicles in Signalized Corridors using Pseudospectral Convex Optimization

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Abstract</td>
<td>31</td>
</tr>
<tr>
<td>3.2 Introduction</td>
<td>31</td>
</tr>
<tr>
<td>3.3 Problem Formulation and System Dynamics</td>
<td>33</td>
</tr>
<tr>
<td>3.3.1 Vehicle Dynamics</td>
<td>34</td>
</tr>
<tr>
<td>3.3.2 Problem Formulation</td>
<td>36</td>
</tr>
<tr>
<td>3.4 Optimal Control Method</td>
<td>43</td>
</tr>
<tr>
<td>3.4.1 Pseudo-spectral Discretization</td>
<td>44</td>
</tr>
<tr>
<td>3.4.2 Convex Optimization</td>
<td>47</td>
</tr>
<tr>
<td>3.4.3 Model Predictive Control</td>
<td>51</td>
</tr>
<tr>
<td>3.5 Simulation Results</td>
<td>56</td>
</tr>
<tr>
<td>3.5.1 Comparison of Solutions</td>
<td>56</td>
</tr>
<tr>
<td>3.5.2 Case 1: Uniform SPaT with 10 Vehicles</td>
<td>60</td>
</tr>
<tr>
<td>3.5.3 Case 2: Uniform SPaT with 20 Vehicles</td>
<td>62</td>
</tr>
<tr>
<td>3.5.4 Case 3: Random SPaT</td>
<td>64</td>
</tr>
<tr>
<td>3.5.5 Comparison with Baseline Simulation in SUMO</td>
<td>64</td>
</tr>
<tr>
<td>3.6 Conclusion</td>
<td>66</td>
</tr>
</tbody>
</table>

## 4 Study II: Pseudospectral Convex Optimization for On-Ramp Merging Control of Connected Vehicles

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Abstract</td>
<td>70</td>
</tr>
<tr>
<td>4.2 Introduction</td>
<td>71</td>
</tr>
<tr>
<td>4.3 Problem Formulation</td>
<td>72</td>
</tr>
<tr>
<td>4.3.1 Vehicle Dynamics</td>
<td>74</td>
</tr>
<tr>
<td>4.4 Merging Sequence Determination</td>
<td>75</td>
</tr>
<tr>
<td>4.4.1 Optimization-based Approach</td>
<td>77</td>
</tr>
<tr>
<td>4.4.2 Rule-based Approach</td>
<td>78</td>
</tr>
</tbody>
</table>
4.5 Optimal Control Problem ........................................... 83
4.6 Merging Preparation ................................................. 85
4.7 Optimal Control Method ............................................. 86
   4.7.1 Pseudospectral Discretization ................................. 87
   4.7.2 Convex Optimization ........................................... 90
   4.7.3 Sequential Convex Programming Algorithms ............... 91
   4.7.4 Model Predictive Control ..................................... 95
4.8 Simulation Results .................................................. 96
   4.8.1 Convergence Analysis ......................................... 98
   4.8.2 Case Study: Coordination of 40 Vehicles ...................... 100
   4.8.3 Case Study: Continuous Simulation ........................... 102
4.9 Conclusion .......................................................... 108

5 Study III: A Novel Deep Reinforcement Learning Approach to
Traffic Signal Control with Connected Vehicles ................. 109
5.1 Abstract ............................................................ 110
5.2 Introduction ........................................................ 110
5.3 Methodology ......................................................... 112
   5.3.1 Scenario and Simulation Environment ....................... 112
   5.3.2 Compressed Representation of Traffic States .............. 117
   5.3.3 Deep Reinforcement Learning Structure .................... 120
5.4 Simulation Results .................................................. 123
   5.4.1 Simulation Parameters ....................................... 123
   5.4.2 Hyper-parameter of Deep Reinforcement Learning Network .. 125
   5.4.3 Convergence of DRL-based Signal Controller Training .... 125
   5.4.4 Comparison with Baselines .................................. 125
   5.4.5 Robustness Analysis .......................................... 135
5.5 Conclusion .......................................................... 137
6 Study IV: Data-Driven Optimization Framework for On-Ramp Merging Control with Connected and Automated Vehicles

6.1 Abstract .................................................. 140

6.2 Introduction ............................................... 140

6.3 Methodology ............................................... 142
  6.3.1 Deep Reinforcement Learning-based Merging Control .... 142
  6.3.2 Optimal Merging Speed Control of CAVs using DRL Results . 150

6.4 Preliminary Simulation Results ............................ 152
  6.4.1 Convergence of DRL-based Signal Controller Training .... 152
  6.4.2 Case Study: Merging with CAVs ........................ 154
  6.4.3 Case Study: Continuous Simulation ...................... 156

6.5 Conclusion .............................................. 156

7 Conclusion and Future Work ............................... 159

Bibliography .................................................. 161

Vita ............................................................. 181
List of Tables

3.1 Parameters of Vehicle Dynamics. ........................................ 37
3.2 Comparison of average computational times. ....................... 59
3.3 Average velocity of traffic flow. ..................................... 67

4.1 Values of vehicle model parameters. ............................... 76
4.2 Cases for vehicles in control zone to recalculate arrival time when new vehicles join. .................................................. 81

5.1 Adopted parameters of simulation environment. .................. 124
5.2 Hyper-parameters of deep reinforcement learning network. .... 126

6.1 Adopted parameters of simulation environment. .................. 153
# List of Figures

3.1 Schematic of CAV-based optimal speed control. ............................................. 35
3.2 Convex optimization-based real-time speed control framework. ........ 54
3.3 Comparison of position, velocity, and control profiles. .................. 57
3.4 Convergence of trajectories and objective function. ......................... 59
3.5 Vehicle trajectories of Case 1. ............................................................... 61
3.6 Vehicle trajectories of Case 2. ............................................................... 63
3.7 Vehicle trajectories of Case 3. ............................................................... 65
3.8 Vehicle trajectories simulated by SUMO. ............................................. 67
3.9 Comparison of Fuel Consumption. ............................................................ 67
4.1 Roadway on-ramp merging scenario in this study. .............................. 73
4.2 A rule-based cooperative merging strategy. ........................................... 80
4.3 Flowchart of convex optimization-based real-time speed control. ....... 97
4.4 Comparison of optimal control profiles. .................................................... 99
4.5 Convergence of vehicle states and objective functions. .................... 99
4.6 Vehicles trajectories of case study with balanced traffic in control zone. 101
4.7 Vehicles trajectories of case study with balanced traffic in both control and merging zone. .............................................................. 103
4.8 Vehicles trajectories of case study with unbalanced traffic in control zone. .............................................................. 104
4.9 Vehicles trajectories of case study with unbalanced traffic in both control and merging zone .............................................................. 105
4.10 Performance comparison with baselines. ........................................ 107

5.1 Proposed data-driven optimization framework for traffic signal control. 113
5.2 The simulated intersection. .............................................................. 115
5.3 An example of traffic state matrix. ................................................. 115
5.4 A typical signal cycle with four phases. ........................................ 116
5.5 The structure of proposed Convolutional AutoEncoder (CAE) for traffic state representation. ......................................................... 119
5.6 Training history of the proposed convolutional autoencoder. .......... 119
5.7 Proposed deep reinforcement learning model. ................................. 121
5.8 Convergence of the proposed DRL network. ................................ 126
5.9 Training history of a reference DRL-based traffic signal controller. .. 128
5.10 Performance comparisons with baselines. .................................... 130
5.11 Simulation comparisons with baselines. ....................................... 131
5.12 Traffic flow simulated by proposed method in each lane. ............... 133
5.13 Traffic flow simulated by referenced method in each lane. ............. 133
5.14 Simulation comparisons with the DRL-based controller trained using symmetric SPaT. ............................................................. 134
5.15 Robustness analysis by comparing the performance of DRL-based controller. ................................................................. 136

6.1 An example of traffic state matrix for on-ramp merging. ................. 144
6.2 The structure of the proposed Convolutional AutoEncoder (CAE) for traffic state representation. ......................................................... 146
6.3 Training history of the proposed convolutional autoencoder. .......... 146
6.4 Convergence of the proposed DRL network. ................................ 153
6.5 Vehicles trajectories of case study with balanced traffic in control zone. 155
6.6 Performance comparison with SUMO simulation. .......................... 157
Chapter 1

Introduction
1.1 Background

The latest advances in communication technology, transportation infrastructure, computational techniques, and artificial intelligence are breeding a revolution in future transportation systems. There has been an acceleration in the research and development efforts toward this transition in many countries Aziz et al. (2017). Among the new technologies under development, Connected and Automated Vehicles (CAVs) are the most frequently studied due to the great promise that they hold to decrease traffic accidents, reduce congestion and improve vehicle efficiency. The term “CAV technology” refers to the vehicle capable of navigating and self-driving without intervention from a human driver, as well as communicating with other vehicles and/or infrastructure and other devices Anderson et al. (2014). The U.S. Department of Transportation is also actively preparing to lead the advance in CAV technologies, and four main potential benefits of introducing CAVs to transportation systems have been spotlighted US Department of Transportation (2020): road safety, economic and societal benefits, efficiency and convenience, and public mobility.

With speculation of a substantial transformation toward an automated transportation system, a number of studies have been conducted to investigate the challenges and opportunities involving CAV applications and implications Fagnant and Kockelman (2015); Bagloee et al. (2016); Li et al. (2017); Taiebat et al. (2018); Guanetti et al. (2018). Among the objectives of these studies, driving safety is a particular concern for guiding the continuing evolution of automotive technology. According to the World Health Organization, the number of deaths in traffic accidents has been increasing since 2000 and reached 1.35 million in 2016 Organization (2018). Accidents also cause traffic congestion, loss of productivity, medical costs, and property damage, and human error is one of the critical reasons that cause accidents. By optimizing the vehicle operations, CAV technology can not only improve safety but also enhances the traffic efficiency. These benefits can be realized by solving vehicle trajectory optimization problems with different goals, such as safety improvement Lee and
Park (2012); Zohdy and Rakha (2016), travel efficiency Wei et al. (2016); Roncoli et al. (2016); Chen et al. (2017), fuel economy Ozatay et al. (2012); Wan et al. (2016); Jin et al. (2016); Jiang et al. (2017); Turri et al. (2017), and other objectives Asadi and Vahidi (2010); Bai et al. (2017). As a legitimate speculation, in a fully automated environment, CAV technologies can eliminate traffic accidents, relieve traffic congestion, and reduce energy consumption through the optimized vehicle operations. Moreover, human productivity can be improved by allowing vehicle occupants to participate in more productive activities when they are relieved of the driving task Taiebat et al. (2018).

In particular, CAV technologies create a new environment for drivers/vehicles and traffic infrastructure to interact in real world. In this environment, connectivity plays a critical role that wireless communication enables the vehicles to communicate with each other (V2V) and with the infrastructure (V2I) about the real-time vehicle location, speed, acceleration, and other data. On one hand, with the availability of the real-time states of surrounding vehicles, it is possible for CAVs to coordinate inter-vehicle interactions to minimize congestion, maximize fuel efficiency, and reduce collisions Guanetti et al. (2018). On the other hand, the availability of these real-time traffic data provides opportunities for traffic controllers to make better signal phase and timing (SPaT) decisions to improve the mobility and reduce energy consumption. For instance, the Oak Ridge National Laboratory Laclair et al. (2019) is developing the Real-Time Mobility Control System (RTMCS) for CAVs applications that include traffic data management, route planning, centralized communications and visualization.

1.2 Motivation

The overall goal of this study is to establish a framework for the control of CAV-based traffic system that accommodates the complex, data-rich traffic network and aims to improve roadway safety, travel and energy efficiency. It is clear that CAV technology
has a positive effect in road safety, fuel economy and emissions reduction on vehicle level. However, on traffic network level, CAVs could create further burden to an already congested network due to eco-driving operations. Since traffic congestion and idling are the main causes of energy waste, an cooperative and interactive control method is needed to minimize negative effects, enhance positive attributes and achieve substantial net improvements. Therefore, the ultimate objective of this study is to develop a cooperative control strategy for the CAVs at merging roadways and the signalized intersections that synchronize CAV motion planning with traffic signal optimization to reduce the traffic congestion and idling as well as the fuel consumption with guaranteed collision avoidance.

1.2.1 Study I: Real-Time Control of Connected Vehicles in Signalized Corridors using Pseudospectral Convex Optimization

The use of traffic signal phase and timing (SPaT) information to coordinate vehicle operations has been shown to improve vehicle fuel efficiency Misener et al. (2010). The optimal speed scheme for a vehicle can be determined through the solution of the corresponding optimal control problem Guanetti et al. (2018). Despite numerous studies demonstrating the potential of using SPaT information to optimize fuel economy, most have concentrated on enhancing the performance of individual vehicles and signal timing control Wang et al. (2020a,b). Furthermore, previous works primarily focus on creating viable trajectories for CAVs, overlooking real-time execution considerations such as computational efficiency and guaranteed convergence.

The onboard algorithms for the motion control system of CAVs play a crucial role in ensuring safety and require real-time updates to respond to dynamic surroundings. While previous efforts have aimed to optimize vehicle trajectories, these methods are often computationally expensive, cannot guarantee optimal solutions, and struggle to
handle nonconvex motion constraints and dynamic environments Asadi and Vahidi (2010); De Nunzio et al. (2016). This study seeks to overcome these limitations by introducing a novel, convex optimization-based method for producing speed profiles using SPaT information. With its advantages of efficient, polynomial solution time and globally optimal convergence, convex optimization approaches have strong potential for practical onboard implementation.

1.2.2 Study II: Pseudospectral Convex Optimization for On-Ramp Merging Control of Connected Vehicles

Ensuring driving safety remains the paramount concern in the ongoing advancements in automotive technology. On-ramp merging control, a critical aspect of highway transportation and a complex traffic negotiation process, has garnered extensive attention from researchers. With complex interactions and a limited time and distance for assessment and decision-making, the merging process presents a high-risk scenario where any misstep could result in a crash. The objective of on-ramp merging control is to facilitate the safe and seamless passage of vehicles through the merging area, by coordinating the vehicles on both roads.

A wealth of research has shown the benefits of coordinating and optimizing the motion of CAVs for improved road safety, reduced travel time and energy consumption, and increased capacity during on-ramp merging scenarios Rios-Torres and Malikopoulos (2016b); Rios-Torres et al. (2021); Letter and Elefteriadou (2017); Shi et al. (2022a). Despite this progress, current methods often neglect the real-time execution of the generated merging trajectories, which is crucial in a safety-critical system that requires continuous updates to react to dynamic and uncertain road conditions.

Moreover, existing optimal control methods face significant challenges, such as high computational cost, difficulty in handling nonlinear vehicle dynamics, and nonconvex constraints, which limit their real-world applicability. This study aims
to overcome these challenges by presenting a new merging control approach that strikes a balance between computational efficiency, solution optimality, and real-time performance for safe and smooth on-ramp merging operations.

1.2.3 Study III: A Novel Deep Reinforcement Learning Approach to Traffic Signal Control with Connected Vehicles

With the availability of the real-time traffic data through V2I communication, traffic controllers can potentially generate better signal phase and timing (SPaT) plans for safer and more sustainable ground mobility. However, SPaT optimization is an NP-Complete problem Wünsch (2008); Al Islam and Hajbabaie (2017). The complexity of the associated optimization problem increases with the number of vehicles and traffic lights in the traffic network, especially when realistic driving/car-following behavior and multiple optimization objectives are considered. As a consequence, designing SPaT optimization and control methods for the traffic network is subject to the “curse of dimensionality” and remains an open challenge.

Although many research and development efforts have focused on enhancing the performance of individual vehicles and signal timing control Wang et al. (2020a,b), relatively few studies address systematic, real-time, optimal vehicle control strategies at signalized intersections. Nevertheless, the possibility of reducing red light idling and fuel consumption through the intelligent use of upcoming traffic signal information is very attractive.

By taking advantage of the large volumes of traffic data from CVs, it is possible to characterize the inherent interacting relationships among vehicles and infrastructure components, which can be used to develop data-driven traffic control schemes. Due to the stochasticity and nonlinearity of traffic flow, non-parametric learning approaches are particularly suitable for the signal controller to learn policies through observing the transition of the traffic states.
However, it is a challenging task to build and train a learning-based controller directly from raw data. Without well-designed learning models and training algorithms, the learning-based controller cannot acquire an effective control strategy. Therefore, to address the “curse of dimensionality”, many researchers had to simplify the training model by limiting the action and state space, which diminishes the authenticity and optimality of the simulated controllers. This study aims to overcome the limitation of the existing methods by redesigning the state space for better learning performance, and defining the action space in a way that enables more practical and flexible signal timings.

1.2.4 Study IV: Data-Driven Optimization Framework for On-Ramp Merging Control with Connected and Automated Vehicles

Data-driven control approaches are ideal for learning policies to control merging traffic, given the stochastic and nonlinear nature of the traffic system. These approaches are different from model-based optimization methods, as they do not require prior knowledge of the traffic system and are less computationally intensive for generating merging sequences. Furthermore, data-driven methods can overcome the complexity associated with rule-based methods by eliminating the need to build complex decision-making models, and can offer better optimality and adaptability. These benefits make data-driven approaches highly promising for addressing various traffic control challenges associated with the merging scenarios.

By taking advantage of the learning ability and adaptability of deep reinforcement learning (DRL), DRL-based merging control methods can potentially achieve more efficient and safer merging operations in complex traffic environments, resulting in improved overall traffic flow and reduced fuel consumption and emissions. DRL-based methods can handle complex and uncertain traffic scenarios that rule-based methods may struggle with, and adapt to changing traffic conditions in real-time. Furthermore,
they can learn from experience and improve their performance over time, whereas
rule-based methods are typically static and require manual adjustments to improve. Ultimately, DRL-based merging control methods have the potential to enhance the
efficiency and safety of the merging areas compared to rule-based methods.

Assisted by the optimal merging sequence generated by the DRL-based controller, CAVs could merge more efficiently and safely, reducing traffic congestion and improving the overall traffic flow. Incorporating optimal control into traffic control systems can also reduce fuel consumption and emissions, making transportation more sustainable. The primary objective of the research on DRL-based merging control methods with CAVs is to develop a robust and efficient control framework that enables CAVs to merge cooperatively and safely, resulting in improved traffic flow and reduced fuel consumption and emissions in a real-world traffic environment.

1.3 Research Contributions

This study aims at addressing some open challenging topics in the control of future CAV-enabled traffic systems. These topics are of key importance to advance the knowledge that increases understanding of how the traffic elements interact and are impacted by individual actors. In terms of vehicle control, this study aims to develop a new method that is different from conventional approaches in three different key ways: 1) achieves optimal solutions using optimal control theory, 2) guarantees real-time performance via convex optimization, and 3) efficiently handles multiple constraints in dynamic traffic environments. These advancements in vehicle control are essential for the successful integration of CAVs in future intelligent transportation systems.

To achieve these objectives, this study makes novel contributions to the vehicle control problem primarily in the following three aspects. First, the proposed sequential convex programming (SCP) algorithms address the nonlinear and nonconvex optimal speed control problem with guaranteed convergence and polynomial solution time in solving a convexified problem in each step. Second, this study leverages the
pseudospectral collocation method in combination with the line-search and trust-region techniques to fundamentally improve the proposed SCP algorithms for higher accuracy and better real-time and convergence performance. Third, thanks to the advanced computational efficiency, the proposed method enables a real-time model predictive control (MPC) framework with instant response to the dynamic traffic environment for collision avoidance and vehicle coordination.

Considering the challenging on-ramp merging scenario, this study develops a new merging control approach that balances the computational efficiency and solution optimality while maintaining real-time performance and safe merging operations. In particular, different methods of merging sequence determination are explored. A pseudospectral convex optimization formulation with hard collision-avoidance constraints is devised for the merging of CVs at roadway on-ramps. With advantages of globally optimal solutions and polynomial solution time, the proposed sequential convex programming (SCP) algorithms address the nonlinear vehicle dynamics and nonconvex motion constraints with guaranteed real-time performance. Furthermore, the performance of the merging control algorithm is enhanced by a line-search technique and a trust-region method, thus leading to two improved SCP algorithms. Moreover, the proposed algorithms are implemented under a model predictive control (MPC) framework to deal with errors and uncertainties for better inter-vehicle coordination.

In terms of traffic signal control, this study proposes a new traffic signal control framework using deep reinforcement learning (DRL) by introducing a novel convolutional autoencoder network to compress the dimensionality of the input traffic states. This results in a concise representation of comprehensive traffic information that is used to facilitate the learning of effective SPaT plans. Furthermore, the action space is extended by including both phase duration and cycle length, allowing for more adaptability to dynamic traffic flow. With a combinatorial action space of different phase durations and cycle lengths, the proposed method can handle unbalanced traffic flow with varying traffic volumes. Additionally, several state-of-the-art techniques
such as target network Van Hasselt et al. (2016), dueling network Wang et al. (2015), and experience replay Schaul et al. (2015) are employed to fundamentally improve the learning efficiency of the DRL algorithm. Finally, the effectiveness and performance of the proposed method are demonstrated through cross-comparisons with several existing traffic signal control methods based on simulations on the widely used Simulation of Urban MOBility (SUMO) traffic simulator. Both the signalized traffic intersections and non-signalized merging areas are addressed by the proposed DRL method.

The research on DRL-based merging control with optimal control CAVs offers a novel approach that combines the learning and adaptability of DRL with the optimal control method, which strikes a balance between computational efficiency and solution optimality while ensuring real-time performance and safe merging operations. This method holds potential to achieve efficient and safe merging operations in complex traffic environments, leading to improved overall traffic flow and reduced fuel consumption and emissions than the rule-based methods. By developing and validating the proposed DRL-based merging control method in a real-world traffic environment could also provide valuable insights into the effectiveness and feasibility of the method, and inform the development of future traffic control systems. This research could also serve as a foundation for further investigations into the applications of DRL and optimal control in traffic control systems, and could potentially lead to the development of more advanced and sophisticated control methods that can handle even more complex traffic scenarios.

In summary, this study makes novel use and explores the interface of multiple disciplines including control theory, optimization, machine learning, data analytics, and real-time computation. This study aims to contribute a new DRL-based traffic signal control framework. The resulting contribution is significant as it is one of the first key steps towards revolutionizing intelligent transportation systems with CAV technologies. The insights gained from this research will provide a foundation for further evaluation and deployment of these new techniques in real-world traffic systems. Additionally, the results from this study will inform other problems that
require learning-based control methods and will benefit future research in the area of intelligent control of data-rich, interactive systems.
Chapter 2

Literature Review
As technology advances in connectivity and sensing, CAVs have attracted more and more attention from both academia and industry. There is a clear consensus that introducing CAVs to transportation systems could offer substantial benefits in terms of road safety, traffic mobility, and energy efficiency. This chapter will explore previous work done in the area of optimal control of CAVs in the scenarios of signalized intersections and on-ramp merging, as well as CAV-based traffic signal control. The discussion will focus on control and planning architectures, specifically on control objectives, problem formulation, optimization algorithms, real-time control techniques, and coordination strategies for multiple vehicles.

2.1 Optimal Control of Connected and Automated Vehicles

The primary goal of developing CAV technology is to improve transportation safety, mobility, and efficiency US Department of Transportation (2017). An successful CAV control method should be able to provide optimal control under a variety of scenarios using interactive, real-time, robust algorithms in a safety-first paradigm Laclair et al. (2019); Xin et al. (2022). Due to the complex interaction, nonlinear dynamics, and large number of vehicles, achieving optimal speed control of CAVs in dynamic scenarios requiring real-time response to traffic environments, where uncertainty is always present, is a challenging task.

Specifically, the onboard control system of an automated vehicle typically has a hierarchical structure Paden et al. (2016): the route planning module decides the continuous path on a large scale map, the path planning module takes route information and computes a reference trajectory towards the next waypoint while considering local traffic conditions, the real-time motion planning module handles the interaction with the traffic environment and ensures that the reference trajectory is executed in a robust manner, and the powertrain control system executes the
acceleration/deceleration and steering operations. The boundaries of these modules are variable with different variations of the driving scenario. An optimal control approach needs to be developed based on the specific application, including the type of vehicle and the expected traffic environment. As a consequence, the variety of related optimization problems and computational techniques is vast.

To improve energy efficiency, recent studies have formulated the fuel economy optimization problem as optimal control problems and obtained the most economical speed profiles by minimizing accelerating and braking events Katrakazas et al. (2015). In the literature, this is also known as eco-driving Mensing et al. (2013), green light optimal speed advisory Eckhoff et al. (2013); Wan et al. (2016), or speed trajectory planning He et al. (2015); Huang and Peng (2017). For example, cooperative adaptive cruise control takes advantage of V2V communications to enable improved energy efficiency of traffic flow by reducing aerodynamic resistance via reduced inter-vehicle spacing Wang et al. (2018). Considering the route cruising scenario, (Ozatay et al., 2014) developed a dynamic programming-based optimization method to generate an optimal speed trajectory by collecting road geographical and traffic information from cloud platform. Similarly, (Sciarretta et al., 2015) formulated an eco-driving optimal control problem and investigated three solution methods: Pontryagin’s minimum principle, dynamic programming, and analytical solutions. In these studies, for which the interactions with other vehicles and the infrastructure were not considered, the concept of CAVs is more like an advanced cruise control system, where the uncertainty and real-time performance were neglected.

2.1.1 Trajectory Optimization for Vehicles Passing Through Signalized Intersections

When signalized intersections are taken into account for fuel economy optimization problems, the vehicle’s trajectory can be planned to avoid frequent stops at traffic lights to improve fuel efficiency and reduce travel time. To obtain the optimal
speed profile, many optimization algorithms have been proposed. For example, analytical solutions were derived in Ozatay et al. (2012) and Wan et al. (2016) for fuel minimization at signalized intersections. The analytical methods are computationally less expensive; however, the resulting two-point boundary value problems are difficult to solve, and complicated mathematical derivations are needed. The lack of generality and robustness hinders the application of analytical solutions for onboard implementation.

Recently, numerical optimization methods received much attention for speed control of vehicles at signalized intersections. For example, (Jiang et al., 2017) formulated an optimal control problem for an isolated signalized intersection with mixed traffic scenario and applied an iterative algorithm using Pontryagin’s minimum principle to obtain the optimal speed profiles of the CAVs. (He et al., 2015) considered each signalized intersection as one stage of a multistage optimal control problem solved using the General Purpose OPtimal Control Software (GPOPS), which is a pseudospectral method for multiple-stage optimal control problems. Dynamic programming has been employed as a numerical approach in Mahler and Vahidi (2014); Ozatay et al. (2014); Kamalanathsharma et al. (2015); Sun et al. (2020); Hao et al. (2019) to obtain globally optimal trajectories. Unfortunately, since the dynamic programming algorithm searches exhaustively in the solution space, it suffers from the curse of dimensionality. Therefore, this method is computationally intensive and usually cannot be executed in real time.

To address the curse of dimensionality, (Huang and Peng, 2017) applied an SCP method to obtain the sub-optimal speed trajectory for multiple-vehicle and multiple-intersection cases. However, only simplified dynamics and no V2V information were considered in this work. For online implementation purposes, (De Nunzio et al., 2016) used a pruning algorithm to reduce the optimization domain into a weighted directed graph, where each node represents a feasible crossing time of the intersection’s green phases. The number of nodes were determined manually as the fineness of the graph approximation. The graph path with most efficient energy was found by
a shortest path algorithm, and the sub-optimal speed trajectory was obtained by
convex optimization using the selected path. For real-time applicability, (Canosa and
HomChaudhuri, 2019) modeled the solution of red-light idling avoidance problem as a
Gaussian distribution, however the feasible solution is not guaranteed. The sampled
control input was used as an initial guess of an MPC-based nonlinear controller.

In recent years, pseudospectral optimal control has emerged as a popular direct
numerical optimal control method that transforms the original continuous-time
optimal control problem into a finite-dimensional parameter optimization problem
by approximating the continuous trajectory through interpolating polynomials at a
set of collocation points characterized by state and control variables as parameters
Fahroo and Ross (2002). It has been proven to have many advantages such as fast
convergence, near-optimality, and lower sensitivity to the initial value in the aerospace
fields Ross and Karpenko (2012). In the area of ground transportation, only a few
studies involving energy management Sotoudeh and HomChaudhuri (2022a,b) have
adopted this method. Meanwhile, it has shown that the computational cost of the
direct method is unpredictable and the convergence of the algorithms cannot be
guaranteed if nonlinear programming (NLP) solvers are employed with or without
pseudospectral discretization.

Another promising approach to CAV speed control is the optimization-based MPC
framework Kamal et al. (2010); Asadi and Vahidi (2010); Kamal et al. (2012); Kim
and Kumar (2014); Cao et al. (2015); Yu et al. (2015); HomChaudhuri et al. (2016);
Zhou et al. (2019); Karimi et al. (2020); Shao and Sun (2021b,a, 2020), which is
suitable for potential real-time implementation. MPC is a general control design
methodology for dynamical systems. With MPC, the trajectory optimization problem
is solved iteratively over a short rolling time horizon. Specifically, only the first step
of optimal control command is applied to the vehicle, then the states and constraints
of the vehicle are sampled again and the optimization is repeated for the new time
horizon. Conceptually, MPC enables real-time vehicle control with instant response
to the maneuvering of preceding vehicles and traffic signal switching. However,
when the interactions between vehicles are considered, a large amount of constraints render a nonconvex solution space. Solving such a nonconvex optimization problem is computationally expensive and may not converge to an optimal or even feasible solution Asadi and Vahidi (2010). To enable fast yet optimal solutions for real-world CAV applications, advances in optimization algorithms need to be made to MPC-based approaches.

To summarize, despite the significant potential of reducing red light idling and fuel consumption through intelligent use of upcoming traffic signal information, most existing speed control approaches use either simplified models that fail to capture the characteristics of vehicle dynamics or high-fidelity models that render complex optimal control problems that cannot be solved in real time. Additionally, most work didn’t consider the interaction among vehicles, which is inevitable in the real-world driving scenarios. In the presence of multiple vehicles, speed coordination is of key importance to the collision avoidance, platoon fuel efficiency, and passenger comfort. Nevertheless, the imposed constraints result in a nonconvex solution space to the optimization problem. Solving such complex and nonconvex problems is computationally intensive and sensitive. To address these methodological challenges, this study proposes a pseudospectral convex optimization formulation to balance the solution optimality and computational efficiency. The combination of pseudospectral discretization and convex optimization has recently been introduced to address trajectory optimization problems in aerospace engineering Sagliano (2018); Wang et al. (2019); Sagliano and Mooij (2021); Song et al. (2021), and shows great performance in solution accuracy, convergence rate, and computational efficiency.

### 2.1.2 On-ramp Merging Control

In the merging scenario, vehicle coordination is of key importance to the collision avoidance, fuel efficiency, and passenger comfort. In particular, the essence of the on-ramp merging problem is to coordinate the speed of the vehicles on either or
both mainstream or ramp roads to pass the merging area safely. In the CAV environment, the vehicles are able to communicate with each other (V2V) and with the infrastructure (V2I) about the real-time vehicle location, speed, acceleration, trajectory, and other relevant information. With the help of these real-time data, traffic controllers should be able to coordinate the vehicles to make smooth merging maneuvers for safety, mobility, and fuel efficiency purposes. However, the imposed constraints by vehicle interactions would result in a non-convex solution space to the optimization problem.

Conventional methods for such problems are computationally intensive and sensitive. Most existing optimization-based methods adopted the analytical solutions, which are computationally less expensive. However, in consideration of nonconvex constraints, such as nonlinear fuel consumption model and aero-resistance, the resulting two-point boundary value problems (TPBVPs) are very challenging to solve analytically. By comparison, numerical methods are often considered as a practical tool of onboard implementation for trajectory optimization problems. For example, dynamic programming (DP) has been employed as a numerical approach in Wu et al. (2014); Pei et al. (2019) to obtain globally optimal trajectories. Detailed implementation methodology of DP can be found in Sundström et al. (2010). Unfortunately, since the DP algorithm searches exhaustively the solution space, it suffers from the curse of dimensionality. Therefore, DP is computationally intensive and usually not suitable for real-time implementation of large-scale, highly nonlinear OCPs.

Many recent studies have formulated the on-ramp merging problem as optimal control problems (OCPs) and obtained the speed profiles for individual vehicles by minimizing specific performance measures Katrakazas et al. (2015). In Ntousakis et al. (2016) and Ding et al. (2019), the optimal speed trajectory was obtained by minimizing the acceleration or deceleration rates of vehicles with analytic solutions. To improve road capacity, Rios-Torres and Malikopoulos (2016a) introduced the gap between vehicles as a target to minimize into the objective function of the optimization
problem. The optimization objective in Letter and Elefteriadou (2017) and Hu and Sun (2019) was set to be maximizing the average speed of each vehicle travelling in the control zone. The optimal control formulation in Zhou et al. (2018) minimized both the control effort and travel time, and the solution was obtained in a recursive manner using Pontryagin’s minimum principle. This work was then extended to more congested scenarios by adding state constraints in Zhou et al. (2019). To consider conventional human-driven vehicles, Karimi et al. (2020) divided the merging process into three phases and investigated six scenarios of vehicle interaction. For each scenario and phase, different control targets of CVs were defined, and the respective OCP was formulated as a target speed tracking problem. Similar models with analytical solutions were proposed in Jing et al. (2019) and Liu et al. (2021) to minimize fuel consumption, travel delay, and passenger discomfort with a multi-objective optimization functional of acceleration and its derivatives. In all these studies, for which the interactions with other vehicles were not fully considered, the motion planning methods did not find obvious advantages over an cooperative adaptive cruise control strategy Wang et al. (2018), where the uncertainty and real-time performance were neglected.

In summary, while many researchers have demonstrated the potential of using optimal control to improve the performance of CAVs, most existing approaches depend on either simplified formulations that fail to capture the characteristics of vehicle dynamics and interactions or on complex optimization models that cannot be solved in real time. For the merging scenario, vehicle coordination is of key importance to collision avoidance, fuel efficiency, and passenger comfort. Nevertheless, the constraints imposed by vehicle coordination result in a nonconvex solution space to the optimization problem. Conventional methods for such problems are computationally intensive and lack efficient means to achieve fast yet stable merging operations. This study aims to address these challenges by developing and numerically demonstrating a novel pseudospectral convex optimization framework and two enhanced SCP
algorithms for on-ramp merging compromising solution optimality and computational efficiency for potential real-world implementation.

2.2 Traffic Signal Control

For the past 20 years, numerous studies has been conducted to tackle traffic signal control issues in the presence of CVs. These studies primarily focus on improving the performance of isolated intersections, with the goal of scaling the solutions to larger networks and corridors. According to the mathematical models used, these methods can be broadly categorized into two groups: optimization-based and machine learning-based approaches Guo et al. (2019).

2.2.1 Optimization-based Approaches

Optimization-based approaches assume that the traffic model is known and the future traffic flow states could be predicted accordingly. Next, certain optimization problems are formulated and solved for the optimal SPaT control plans. The objectives of these optimization problems are usually to minimize traffic performance measures, such as traffic delay and queue length, which are estimated on the basis of predicted vehicle arrivals Guo et al. (2019). However, this approach requires accurate predictions of future traffic states, which can be challenging due to the complexity of the optimization problem that involves traffic flow models and coupled with SPaT data. Therefore, the key challenges in CV-based traffic controls are to predict the future traffic states accurately, coordinate multiple intersections effectively by accounting for the conflicts of traffic flows, and efficiently solve the underlying large-scale optimization problem Guo et al. (2019). These challenges have led to the development of three different groups of optimization methods: centralized, decentralized, and hierarchical approaches.
Centralized Methods

In comparison to conventional signal control methods, such as adaptive control and coordinated control, the biggest challenge in implementing optimization-based methods is the high complexity of optimization models. To overcome this, centralized approaches reformulate the optimization problem by reducing the number of variables. For instance, in He et al. (2012), individual vehicles were grouped into pseudo-platoons based on the headways between them and a mixed-integer linear program (MILP) was utilized to determine the optimal signal phase sequence and phase initialization in real-time using platoon request data and traffic controller status. This work also introduced a dynamic arterial coordination strategy to promote traffic progression by taking into account platoon queue delay, signal delay in current intersection, and possible delay at downstream intersections.

In Feng et al. (2015), a real-time adaptive phase allocation algorithm was proposed that utilizes dynamic programming and optimization techniques to allocate signal phase sequences and duration based on predicted vehicle arrivals. Zhao et al. (2015) adopted an interactive grid search method to solve an optimization problem, considering accumulated fuel consumption and travel time as the cost function, to determine the optimal traffic light timing of for each cycle at an intersection.

Mohebifard and Hajbabaie (2019) used a cell transmission model Daganzo (1994) to categorize the traffic network into cells and groups for higher-level representation and then formulated an MILP to maximize network throughput, which was solved using Benders decomposition technique BnnoBRs (1962). Bin Al Islam et al. (2021) formulated an optimization problem to minimize network-level traffic delay, considering the energy consumption as a constraint, and solved the resulting non-convex problem using a stochastic gradient approximation algorithm. In Hong et al. (2022), a linear dynamic traffic system model was built for a large-scale traffic network and a linear-quadratic regulator was applied to minimize both traffic delay and
control-input changes, allowing for an online update of the traffic model to be adaptive to signal control outcomes.

**Decentralized Methods**

Decentralized approaches aim to simplify the model and lower the computational demands of the traffic control problem by utilizing distributed control and optimization techniques. These methods optimize objective functions for each intersection individually and disregard coordination among neighboring intersections, leading to sub-optimal, local solutions instead of globally optimal ones. These approaches typically predict only traffic states, often just the arrivals, of the current intersection over a specified time horizon. To address this challenge, **Li and Ban (2017)** transformed the problem into a dynamic programming model by dividing the timing decisions into stages with one stage for each phase, and minimizing the accumulated fuel consumption and travel time by calculating the objective function for each phase. **Goodall et al. (2013)** proposed a predictive microscopic simulation algorithm to estimate future traffic conditions and objectives over a rolling horizon of 15 seconds, assuming vehicles maintain heading and speed during this time. To account for the impact of queue spillbacks, **Noaeen et al. (2021)** presented a decentralized method to maximize global network throughput by maximizing the effective outflow rate of each intersection locally and independently. This approach determines the minimum saturated green time of all possible phases based on queue lengths, arrival flows, and downstream queue lengths at each intersection to facilitate vehicle discharge at full capacity.

In **Al Islam and Hajbabaie (2017)**, a distributed, coordinated approach was developed to tackle the network control problem through dividing it into a series of local controllers that can exchange traffic data with each other. At each decision time step, each controller collects data on queue lengths and incoming vehicle numbers from neighboring intersections, and decides to whether to end or maintain the existing signal phase for local signal timing till the next step. Moreover, **Islam et al. (2020)**
expanded upon this work by taking into account unconnected vehicles. Specifically, they developed two algorithms to estimate the traffic states of unconnected vehicles relying on the traffic information from fuse loop detectors and CVs using car-following concepts. In Liang et al. (2020), both connected and identified non-connected vehicles were grouped into platoons, resulting in the generation of all possible platoon departure sequences. Rather than solving for optimal signal timing directly, the platoon departure sequence that minimizes total vehicle delay was found by enumerating all possible departure sequences. The optimal SPaT was then calculated as the time needed to discharge all the vehicles in a platoon.

Hierarchical Methods

Hierarchical approaches address the complexities of traffic network optimization problem by breaking it down into multi-level optimization problems with different objectives for each level. The defining aspect of these approaches is to establish the macroscopic and microscopic models for each level of control problem. For instance, Beak et al. (2017) proposed a two-level adaptive signal control method for corridor coordination. Two optimization problems with distinct objectives were formulated at the intersection and corridor levels. At the corridor level, an MILP was developed to optimize the offsets along the corridor while minimizing the platoon delay based on the movement of vehicle platoons. The optimized offsets were then sent to the intersection level as the coordination constraints. At the intersection level, individual vehicle movements were computed using a dynamic programming method to minimize individual vehicle delay and handle phase allocation for both coordinated and non-coordinated phases.

In Qiao et al. (2019), a three-level multi-agent signal control system was proposed for an urban traffic network, including an intersection agent, a regional agent, and a central agent. Three corresponding objective functions were designed to minimize total delay time, reduce the total green ratio-related delay, and find the optimal signal cycle. The fireworks algorithm was employed in Tan et al. (2013) to solved the
optimization problems, resulting in the optimal cycle length, offset, and green ratio that minimizes the total delay time of all intersections.

### 2.2.2 Data-driven Approaches

The increasing availability of data and computation power has made machine learning-based approaches more and more popular in traffic control. These approaches offer the advantage of being model-free, eliminating the need to build complex mathematical models to describe the traffic states and solve nonlinear optimization problem. Without a need for prior knowledge of the traffic system, machine learning-based approaches also reduce the likelihood of introducing errors to the estimation of traffic states. Additionally, machine learning approaches are less computationally intensive and have great potential for real-time applications, making them more practical than traditional model-based optimization methods. Moreover, machine learning-based controllers have the ability to continuously learn and adapt to changes in the traffic pattern, leading to improved optimality. There have been efforts to use machine learning techniques to model the complicated relationship between the signal timing plans and traffic delays, as reported in Bala Subramaniyan et al. (2022). Overall, machine learning-based approaches hold great potential for addressing the various challenges in the field of traffic signal control.

The basic components to designing a machine learning-based approach include 1) capturing the traffic state efficiently, 2) selecting an appropriate learning algorithm, 3) defining the learning objectives, and 4) designing the action space. For instance, in Liang et al. (2019), the traffic state of a single intersection was captured as image-like grids, represented using an \( n \times n \times 2 \) matrix which conveyed the position and speed information of vehicles in the grids. This matrix served as the input to a convolutional neural network (CNN) that calculated expected future rewards for selecting certain actions from the action space, specifically the adjustment of the current signal phase duration. The reward was defined as the reduction in cumulative waiting time between
two consecutive signal cycles. By maximizing the expected reward with the double Q-learning method, the neural network learned how to reduce the average waiting time of vehicles at an intersection.

In Al Islam et al. (2018), the traffic state of an intersection was represented using a set of normalized queue lengths in each lane, which were discretized through the application of the $k$-means clustering algorithm. To optimize energy consumption and mobility simultaneously, they utilized a RL algorithm with three different reward functions. At each decision time step, the signal controller agent made a decision to either end or continue the current signal phase. Similarly, Du et al. (2019) proposed a reward function with respect to a fixed-time controller. The agent receives positive rewards for better performance than the fixed-time controller, and negative rewards for underperformance.

In Chen et al. (2019), the traffic state was characterized by a 2-D matrix consisting of the number of stopped vehicles in each direction and the average speed measured in each section. The action space comprised of two options: selecting signal phases and adjusting phase offsets. The reward function was a combination of the total volume that passed through the arterial network and the difference in queue lengths between the two different directions. To increase the adaptability of the signal control model, Yoon et al. (2021) proposed a graph-based method that depicts the traffic state as graph-structured data, which was then input into a graph neural network to train the signal control policy. The study focused on an isolated intersection with only straight traffic flows and thus the SPaT was comprised of two green-red phases and two yellow-red phases, and the action was defined as the ratio of green time over a fixed signal cycle.

To improve the generalization capability of the RL-based algorithm, Zeng et al. (2019) introduced the prior traffic knowledge. Specifically, a fully-connected network was used to classify the demand pattern of the intersection states, and the results were combined with outputs of a convolutional network to jointly generate Q-value approximations. The intersection state was described by a discrete encoding matrix.
consisting of vehicle position, vehicle speed, and signal phase. The reward structure combined the numbers of stopped vehicles and passed vehicles, phase change, and the total waiting time of vehicles.

In the scenario of network-wide traffic control, many challenges arise when applying a centralized RL method. For example, as the number of traffic lights increases, the action space increases exponentially and it becomes difficult to find an effective joint control policy. To overcome this issue, some previous studies, such as Aslani et al. (2017) and Chu et al. (2019), trained each intersection as an individual agent based on the observed local traffic states and the information received from neighboring intersections, while the state and performance of the whole traffic system were determined by the joint control actions of all the intersections. Meanwhile, the reward distribution and environment dynamics are required to be stationary in a Markov decision process. If the rewards of an intersection are also affected by the actions of its neighboring intersections, it is difficult for the agent to converge to a stationary policy. To mitigate this issue, Li et al. (2021b) designed a knowledge sharing mechanism to improve cooperation and collaboration among traffic signals. Specifically, the “knowledge” was a collective representation of the traffic environment collected by all agents and used for learning individual policies of each agent. Similarly, Wang et al. (2021) adopted a $k$-nearest-neighbor-based joint state representation and combined a group of traffic signals into a single agent to improve the learning convergence of the multi-agent RL algorithm.

In summary, the above reviewed works have demonstrated the potential of data-driven methods to learn a signal control strategy that outperforms some conventional controllers such as fix-timing and actuated controllers. However, the considered scenarios were significantly simplified. Some studies limited the number of lanes and traffic directions to have a smaller state space. For the action space, some studies selected continuous space by fixing the sequence of signal phases and defining the actions as phase splits. Other studies that chose discrete action space either limited the options of both phases and duration or fixed the cycle/phase length with
phase switching as the actions. All these simplifications were employed to reduce the complexity of the data-driven model, which implies that the existing data-driven methods have difficulties in learning a practical and truly optimal signal control strategy.

2.3 Cooperative Control of Signals and Vehicles

As can be seen from the above discussion, extensive efforts have been devoted by researchers to traffic signal optimization or vehicle control separately. Most traffic signal control approaches rely on the estimation or prediction of the arrival information of vehicles to make best SPaT plans. However, such predictions are coupled with SPaT, which makes the optimization model more complex and intractable. This complexity increases exponentially for corridor level or network level optimization. Fortunately, with the cooperative control of CAVs and traffic signal, the future traffic state could be more predictable. In the 100% CAVs environment, the whole traffic system may be completely under control. Intuitively, the efficiency of traffic network could be maximized in an active manner.

Research on signal-vehicle cooperative control has just received attention. For example, Li et al. (2014) developed an optimization algorithm for optimizing the CAVs’ trajectories and the traffic signal timing simultaneously. Since they only considered a single intersection with single-lane, a simple enumeration method was used for determining the optimal signal timing plan. Given a signal timing plan, they first determined the trajectory of the first vehicle based on its speed and distance to the intersection, then calculated the trajectories of the following vehicles one by one, and finally assigned the vehicles that cannot pass the intersection to the next iteration. Using a rolling horizon scheme, the overall planning horizon was divided into overlapped stages. In the beginning of each stage, only signal timing was optimized. At the overlapped period, the optimization of vehicle trajectories were performed. As such, the overall optimization was implemented over the time
horizon continuously. Compared to traditional actuated signal control, the approach proposed in Li et al. (2014) was able to reduce the average travel time delay and increase the throughput under different demand scenarios.

In addition, a CAV-based cooperative control method was proposed in Xu et al. (2017) to concurrently optimize traffic signal and vehicles’ trajectories to improve traffic efficiency and fuel economy. With the speed and position information from CAVs as input, the SPaT information was optimized by minimizing the total travel time of all vehicles. Then, the calculated arriving time was used as constraints to minimize the fuel consumption of each vehicle. The traffic signal optimization was conducted at the end of every traffic signal cycle, while the vehicle optimal control is implemented onboard with the rolling horizon procedure. The main contribution of this work was the investigation of how to achieve cooperation between traffic signal and CAVs with consideration of multiple important objectives.

These studies could serve as a good starting point of intelligent control of signals and vehicles, while a lot of issues need to be further investigated. For example, how to handle non-connected vehicles in the network, how to improve the energy efficiency on network level, how to extend the cooperative methods to control traffic corridors and networks, and how to efficiently combine signal and vehicle control for real-time implementation. The stochastic and non-linear nature of traffic flow makes it difficult to build an optimal control model for signalized traffic systems. Additionally, the dimensions of traffic state and action space increase exponentially with the increasing number of participants in the traffic system. It makes the control problem impractical to be solved analytically. With the machine learning methods, it is possible to learn non-parametric models and control polices through observing the transition of traffic states. However, as can be seen from the previous discussion, the existing DRL-based methods suffer various issues in performing learning and control. In particular, the reward structure and action space need to be modified to handle dynamic traffic conditions and perform more efficient learning. As a summary, to
develop a generic, scalable, data-driven optimization framework for network signal control, much work need to be done.
Chapter 3

Study I: Real-Time Control of Connected Vehicles in Signalized Corridors using Pseudospectral Convex Optimization
3.1 Abstract

Recent advances in Connected and Automated Vehicle (CAV) technologies have opened up new opportunities to enable safe, efficient, and sustainable transportation systems. However, developing reliable and rapid speed control algorithms in highly dynamic environments with complex inter-vehicle interactions and nonlinear vehicle dynamics is still a daunting task. In this chapter, a novel speed control method is developed for CAVs to produce optimal speed profiles that minimize the fuel consumption and avoid idling at signalized intersections. To this end, an optimal control problem is formulated using the information of the upcoming traffic signal to adapt vehicles’ speeds to avoid frequent stop-and-go driving patterns. By applying the pseudospectral discretization method and the sequential convex programming method, the computational efficiency is greatly improved, enabling potential real-time on-vehicle applications. In addition, the algorithm is implemented under a model predictive control framework to ensure online control with instant response for collision avoidance and robust vehicle coordination. The proposed algorithm is verified through numerical simulations of three different traffic scenarios. The convergence and accuracy of the proposed approach are demonstrated by comparing with a popular nonlinear solver. Furthermore, the benefit of the proposed method in both traffic mobility and fuel efficiency is validated using the speed profile determined from a traffic following model in a simulation software as the baseline.

3.2 Introduction

The recent advancements in communication technology, transportation infrastructure, computational techniques, and artificial intelligence are driving a revolution in future transportation systems. There has been an acceleration in the research and development efforts toward this transition in many countries Aziz et al. (2017). Among the new technologies under development, Connected and Automated Vehicles
(CAVs) are the most frequently studied due to their potential to reduce traffic accidents, reduce congestion and improve vehicle efficiency. The term “CAV technology” refers to the vehicle capable of navigating and self-driving without intervention from a human driver, as well as communicating with other vehicles and/or infrastructure and other devices Anderson et al. (2014). The U.S. Department of Transportation lists four primary potential benefits of introducing CAV technology to transportation systems US Department of Transportation (2020): road safety, economic and societal benefits, energy efficiency, and public mobility.

CAV technologies create a new environment for drivers/vehicles and traffic infrastructure to interact in the real world. In this environment, connectivity plays a critical role in which wireless communication enables the vehicles to communicate with each other (V2V) and with the infrastructure (V2I) about real-time vehicle location, speed, acceleration, and other data. The availability of these real-time data provides CAVs with the opportunity to coordinate traffic interactions to minimize congestion, maximize fuel efficiency, and reduce collisions Guanetti et al. (2018). With speculation of a substantial transformation toward an automated transportation system, a number of studies have been conducted to investigate the challenges and opportunities involving CAV applications and implications Fagnant and Kockelman (2015); Bagloee et al. (2016); Li et al. (2017); Taiebat et al. (2018). For example, the Oak Ridge National Laboratory Laclair et al. (2019) is developing the Real-Time Mobility Control System (RTMCS) for CAVs applications that include traffic data management, route planning, centralized communications and visualization.

It has been proven that vehicle fuel efficiency can be improved using traffic signal phase and timing (SPaT) information to coordinate vehicle operations Misener et al. (2010). It has also been established that the optimal speed scheme for a vehicle can be determined by solving the associated optimal control problem Guanetti et al. (2018). However, while many researchers have demonstrated the potential of using SPaT information to optimize fuel economy, most efforts have focused on enhancing the performance of individual vehicles and signal timing control Wang et al. (2020a,b).
Moreover, the related works mainly focus on generating feasible trajectories for CAVs, while ignoring the real-time execution of the generated trajectory in terms of computational efficiency and guaranteed convergence.

The motion control system of CAVs is safety-critical and relies heavily on the onboard algorithms. It requires real-time update of maneuvers to react to the dynamic surrounding environment. Despite the fact that many methods have been proposed to optimize vehicles’ trajectories, their optimization methods are not suitable for real-world implementation due to high computational cost, no guarantee of optimal solutions, and inability to cope with nonconvex motion constraints and dynamic environments Asadi and Vahidi (2010); De Nunzio et al. (2016). This study will address this need by developing a novel convex optimization-based method that produces the speed profile using SPaT information. With advantages of polynomial solution time and globally optimal convergence, the convex optimization approaches are very promising for on-board application.

The contribution of this study is threefold. First, the proposed sequential convex programming (SCP) algorithms address the nonlinear and nonconvex optimal speed control problem with guaranteed convergence and polynomial solution time in solving a convexified problem in each step. Second, we leverage the pseudospectral collocation method in combination with the line-search and trust-region techniques to fundamentally improve the proposed SCP algorithms for higher accuracy and better real-time and convergence performance. Third, thanks to the advanced computational efficiency, the proposed method enables a real-time model predictive control (MPC) framework with instant response to the dynamic traffic environment for collision avoidance and vehicle coordination.

3.3 Problem Formulation and System Dynamics

The objective of this study is to develop an optimal speed control model that minimizes the fuel consumption and avoid idling at intersections for vehicles.
traveling in signalized arterial corridors. As shown in Fig. 3.1, after accurate SPaT information is determined and broadcast to CAVs, speed profiles need to be computed in real-time by each incoming vehicle to regulate its movement with minimum acceleration/deceleration maneuvers. This study assumes that the SPaT information is always available to the CAVs through V2I communication. In addition, speed trajectories of the preceding and following vehicles computed in the previous time step are also shared to the vehicles through V2V communications. This section details the vehicle dynamics model and the formulation of the speed profile optimization problem.

3.3.1 Vehicle Dynamics

In this study, the aim is to determine the optimal speed profile. Longitudinal dynamics of the vehicle are considered as Vahidi et al. (2005):

$$m\ddot{v} = \frac{T_e}{r_g} - F_b - F_{aero} - F_{\text{grade}}$$

(3.1)

where $m$ is the total mass of the vehicle, $v$ is the longitudinal velocity, $T_e$ represents the engine torque, and $r_g$ is the wheel radius divided by total gear ratio. $F_b$ denotes the braking force at the wheels, and $F_{aero}$ is the force due to aerodynamic resistance and given by:

$$F_{aero} = \frac{1}{2} \rho AC_D v^2,$$

(3.2)

where $C_D$ denotes the aerodynamic drag coefficient, $A$ is the frontal area of the vehicle, and $\rho$ is air density. $F_{\text{other}}$ represents the resistant force due to frictions and road grades, which is defined as:

$$F_{\text{other}} = mg(\mu \cos \theta + \sin \theta),$$

(3.3)

where $g$ is the gravity constant, $\mu$ denotes the rolling friction coefficient, and $\theta$ is the road gradient, which is assumed to be constant during the control cycle. In addition,
Figure 3.1: Schematic of CAV-based optimal speed control.
\( r_g \) is also assumed to be constant for the optimization problem considered in this study.

The fuel consumption model is adopted from Kamal et al. (2011), where the fuel consumption rate \( \dot{m}_f \) is approximated by:

\[
\dot{m}_f = \begin{cases} 
\alpha_0 + \alpha_1 v + \alpha_2 v^2 + \alpha_3 v^3 + a(\beta_0 + \beta_1 v + \beta_2 v^2), & a \geq 0 \\
\alpha_0, & a < 0 
\end{cases}
\]

where \( a \) is the vehicle's acceleration, \( \alpha_i \) and \( \beta_i \) are constants of the third order polynomial function determined by fitting the experiment data. When vehicles are braking or idling, the fuel consumption is a constant \( \alpha_0 \). The selected values of above parameters are summarized in Table 3.1.

### 3.3.2 Problem Formulation

The optimal speed control problem can be formulated as a nonlinear optimal control problem that minimizes fuel consumption while fulfilling multiple driving requirements over the time horizon of length \( T \), such as safe distance with the preceding vehicle, passenger comfort, and desired speed. As such, the cost functional \( J \) is defined as Kamal et al. (2010):
Table 3.1: Parameters of Vehicle Dynamics.

<table>
<thead>
<tr>
<th>Parameter (Unit)</th>
<th>Value</th>
<th>Parameter (Unit)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$ (kg)</td>
<td>1200</td>
<td>$u_e^{\text{max}}$ (m/s$^2$)</td>
<td>3</td>
</tr>
<tr>
<td>$\rho$ (kg/m$^3$)</td>
<td>1.184</td>
<td>$u_b^{\text{max}}$ (m/s$^2$)</td>
<td>3</td>
</tr>
<tr>
<td>$A$ (m$^2$)</td>
<td>2.5</td>
<td>$v^{\text{max}}$ (m/s)</td>
<td>20</td>
</tr>
<tr>
<td>$C_D$</td>
<td>0.32</td>
<td>$v^{\text{min}}$ (m/s)</td>
<td>0</td>
</tr>
<tr>
<td>$g$ (m/s$^2$)</td>
<td>9.8</td>
<td>$\mu$</td>
<td>0.015</td>
</tr>
<tr>
<td>$\alpha_0$ (mL/s)</td>
<td>0.1569</td>
<td>$\beta_0$ (mLs/m)</td>
<td>0.07224</td>
</tr>
<tr>
<td>$\alpha_1$ (mL/m)</td>
<td>2.450e-2</td>
<td>$\beta_1$ (mLs$^2$/m$^2$)</td>
<td>9.681e-2</td>
</tr>
<tr>
<td>$\alpha_2$ (mLs/m$^2$)</td>
<td>-7.415e-4</td>
<td>$\beta_2$ (mLs$^3$/m$^3$)</td>
<td>1.075e-3</td>
</tr>
<tr>
<td>$\alpha_3$ (mLs$^2$/m$^3$)</td>
<td>5.975e-5</td>
<td>$R_0$ (m)</td>
<td>2.5</td>
</tr>
</tbody>
</table>
Problem 1.

\[
\text{minimize } J = \int_t^{t+T} \omega_1 \frac{\dot{m}_f}{v} + \omega_2 R(t)^2 + \omega_3 (v - v^{max})^2 + \omega_4 u(t)^2 \, dt \quad (3.5)
\]

subject to

\[
\dot{x} = v \quad (3.6)
\]

\[
u = u_e - u_b \quad (3.7)
\]

\[
\dot{v} = u - \frac{\rho AC_D v^2}{2m} - (\mu \cos \theta + \sin \theta) g \quad (3.8)
\]

\[
R(t) = R_0 + v t_{hd} + x - x^p \quad (3.9)
\]

\[
x^p - x \geq R_0 \quad (3.10)
\]

\[
0 \leq u_e \leq u_e^{max} \quad (3.11)
\]

\[
0 \leq u_b \leq u_b^{max} \quad (3.12)
\]

\[
- u_b^{max} \leq u \leq u_e^{max} \quad (3.13)
\]

\[
v^{min} \leq v \leq v^{max} \quad (3.14)
\]

where \( x \) denotes the position of the vehicle, \( u_b = \frac{F_b}{m} \) is the vehicle deceleration generated by the braking power, while \( u_e = \frac{T_e}{m r g} \) is the vehicle acceleration due to the engine power. Constraints Eqs. (3.11) to (3.14) define the lower and upper bounds for the control variables and the vehicle speed.

In this problem, the control power is the vehicle acceleration and deceleration. The first term of the cost functional Eq. (3.5) minimizes the fuel consumption per unit distance over the time horizon. Choosing to minimize the fuel consumption per unit distance instead of pure fuel consumption because the simulations revealed that the former case results in smoother control profiles and more stable convergence. One possible reason is that the speed \( v \) incorporated in the term of fuel consumption per unit distance acts as an adaptive weight that may benefit the convergence of the algorithm. The second term \( R \) is the deviation from the safety distance to the preceding vehicle and given by Eq. (3.9), where \( R_0 \) is the minimum safe distance between the vehicles, \( t_{hd} \) is the headway requirement measured in time, and \( x^p \) is the...
position of the preceding vehicle. The third term is to force the vehicle to maintain a desired traveling speed, such as maximum allowable speed or the most economic speed. The last term is a penalty for avoiding control jerks. Each term is multiplied by an adjustable weight to aid behavioral decision-making.

The penalty weight for maintaining inter-vehicle distance $\omega_2$ is calculated as Kamal et al. (2010):

$$\omega_2 = \gamma e^{\alpha(x-x_p)}$$ \hspace{1cm} (3.15)

where $\gamma$ and $\alpha$ are adjustable parameters. Intuitively, when the vehicle is close to the preceding vehicle, $\omega_2$ grows to very large values to avoid collision. While the relative distance exceeds the desired value, the whole penalty term will still increase, but if the preceding vehicle is far away, $\omega_2$ becomes negligible small.

The problem described in Problem 1 is a highly nonconvex optimal control problem due to the nonlinear dynamics and nonconvex terms in the cost functional. This results in a nonconvex solution space, which leads computational complexity to the problem Asadi and Vahidi (2010); HomChaudhuri et al. (2016). To the best of our knowledge, no existing solvers are able to directly handle this problem with stable convergence and real-time performance. Therefore, it is important to reformulate the problem in a manner that will expedite the computation of solutions with guaranteed stability in real-world signalized traffic environments. Specifically, a set of preprocessing rules are developed and detailed below to reconfigure the search space (e.g., speed range) that allows the vehicle to pass the traffic signal during a green phase. This set of rules helps break down the original nonconvex problem to simpler ones by refining the search space.

First, despite the nonconvex fuel consumption model, an optimal cruise speed can always be found for the vehicle moving at the cruise stage (i.e., maintaining a constant speed) with the lowest fuel consumption rate per unit distance. Meanwhile, the speed of the vehicle is limited between an upper bound and a lower bound when the vehicle is passing the intersection during a green window. Therefore, if the optimal cruise
speed is within the range of reference speeds, it can then be selected as the most fuel-efficient speed, and fuel-efficient driving strategies can be enabled by tracking the reference speed under the constraints from the upcoming red lights and the preceding vehicle. As a result, the fuel consumption term can be removed from the objective, and Problem 1 can be relaxed into an optimal cruise speed tracking problem to find a sub-optimal solution. A new optimal speed control problem is formulated as follows:

**Problem 2.**

\[
\begin{align*}
\text{minimize} & \quad J = \int_t^{t+T} \left( \omega_2 R(t)^2 + \omega_3 (v - v_d)^2 + \omega_4 (u(t) - u_d(t))^2 \right) dt \\
\text{subject to} & \quad u_d(t) = \frac{\rho AC_D v_d^2}{2m} + (\mu \cos \theta + \sin \theta) g \\
& \text{and } \text{Eq. (3.6), Eq. (3.7), Eq. (3.8), Eq. (3.9), Eq. (3.10), Eq. (3.11), Eq. (3.12), Eq. (3.13), Eq. (3.14).}
\end{align*}
\]

As discussed above, the purpose of Problem 2 is to track a reference velocity while keeping a safe distance with the preceding vehicle and comfort driving behavior. Note that a compensation term \( u_d(t) \) is added to the cost functional with the aim to reduce the deviation of the control from its desired value calculated at the cruising speed, which is expected to improve the fuel economy and speed tracking performance. In fact, the simulations reveal some unwanted jerks around the reference velocity in the solutions of Problem 1 if \( u_d(t) \) is not included. The reference velocity \( v_d \) is computed based on the upcoming traffic signal and the states of surrounding vehicles similar as in Asadi and Vahidi (2010); HomChaudhuri et al. (2016). At each time step, the controller calculates the minimum time cost \( t_{\text{min}} \) for the vehicle to reach the next intersection. \( t_{\text{min}} \) is estimated as the time taken by vehicle to accelerate from its current speed to the maximum speed with the maximum acceleration and then maintain the maximum speed till arriving at the intersection. Thereafter, considering the upcoming traffic signals, the upper bound of the reference velocity is determined as follows:
if current signal phase is red
and \( t_{\text{min}} < t_{r} \)

if current signal phase is red
and \( t_{\text{min}} \geq t_{r} + t_{\text{green}} \);
and \( t_{\text{min}} \mod (t_{\text{green}} + t_{\text{red}}) < t_{\text{red}} \);
\[ (3.18) \]

if current signal phase is green
and \( t_{\text{min}} \geq t_{r} \)
and \( t_{\text{min}} \mod (t_{\text{green}} + t_{\text{red}}) < t_{\text{red}} \);

otherwise.

where \( d \) is the distance between the vehicle and the next intersection, \( t_{r} \) is the remaining time of current signal phase, \( t_{\text{green}} \) and \( t_{\text{red}} \) are the duration of green phase and red phase, respectively. If the current signal is red and the vehicle is able to reach the intersection before the end of this red phase, then the upper bound of the reference velocity is calculated based on the beginning time of next green phase. If the current signal is red and the minimum time for the vehicle to reach the intersection is after the end of next green phase and during a red phase, or if the current signal is green and the minimum time for the vehicle to reach the intersection is after this green phase but during a red phase, then the upper bound of the reference velocity is calculated as the vehicle arrives at the intersection at the beginning of the first green phase it can reach. For other cases, the upper bound of vehicle’s velocity is not limited by the traffic signal, as such, it is set to the maximum allowable speed \( v_{\text{max}} \).

Meanwhile, the end point of the selected green phase can serve as the lower bound of the desired speed \( v_{\text{lb}} \), which is computed as follows:
The next step is to incorporate the fuel efficiency requirement to the reference velocity. Based on the fuel consumption model in Eq. (3.4), when the road slope $\theta$ is 0, it can be derived that the optimal cruising speed occurs at the minimum fuel consumption rate per unit distance. That is, for $\min \frac{\dot{m}}{v}$, the necessary and sufficient conditions are:

$$\frac{d}{dv} \left( \frac{\dot{m}}{v} \right) = -\frac{\alpha_0}{v^2} + 2\alpha_3 v + \alpha_2 = 0 \quad (3.20)$$

The optimal cruising speed $v^*$ can be obtained as the solution of Eq. (3.20). Combined with the bounds of the desired speed, the reference velocity can be modified as

$$v_d = \begin{cases} v^* & \text{if } v_{lb} \leq v^* \leq v_{ub}, \\ \min_{v' \in [v_{lb}, v_{ub}]} |v^* - v'| & \text{otherwise.} \end{cases} \quad (3.21)$$

where $\min_{v' \in [v_{lb}, v_{ub}]} |v^* - v'|$ is the function to find a value in the set $[v_{lb}, v_{ub}]$ with smallest deviation from $v^*$. 
To improve the traffic mobility and group fuel efficiency, the reference velocity of the following vehicle $v_d^f$ also needs to be taken into consideration as:

$$
v_d = \begin{cases} 
  v_d + (v_d^f - v_d)e^{\alpha(R_0 + v_d^f t_{hd} - d')} & \text{if } d_f < d_0 \text{ and } v_d < v_d^f \\
  v_d & \text{otherwise}
\end{cases}
$$

(3.22)

where $d_f$ is the distance between ego-vehicle and the following vehicle, $d_0$ is a predefined constant working as the operation range, and the exponential weight is used for collision avoidance.

In summary, Problem 2 is an optimal cruise speed tracking problem, in which the optimal cruise speed is obtained by analyzing the fuel consumption model and considering to avoid stops at the traffic lights. In addition, the optimal cruise speed can be modified to make space for the following vehicles to pass through the intersection under the current green phase.

### 3.4 Optimal Control Method

While the Problem 2 proposed above would better enable the design of optimization algorithms without nonconvex terms, solving nonlinear optimal control problems is still necessary. It can be solved with the well-known NLP packages, such as SNOPT Gill et al. (2005) or IPOPT Biegler and Zavala (2009). However, these NLP methods cannot solve the nonlinear problems in polynomial time and the convergence is not guaranteed. Moreover, they often require a good initial guess, which might need to be supplied by users. Consequently, it is difficult to use them onboard directly. Recently, pseudospectral optimal control Ross and Karpenko (2012); Fahroo and Ross (2008) and convex optimization Boyd et al. (2004) have emerged as two new technologies that allow optimal solutions in real time. Combination of these techniques allows development of a pseudospectral convex optimization method for the proposed problem. Specifically, the pseudospectral method is applied to
discretize Problem 2, and the nonlinear structures in the problem are converted into tractable formulations to find approximate optimal solutions using sequential convex programming.

### 3.4.1 Pseudo-spectral Discretization

Pseudospectral optimal control is a particular area of direct methods for solving optimal control problems. It discretizes the original continuous-time optimal control problem and transforms it into a parameter optimization problem. In recent years, pseudospectral methods have been successfully applied to aerospace trajectory optimization problems and integrated into general purpose commercial optimal control software packages because of their advantages over traditional direct methods Ross and Karpenko (2012). These advantages include the higher accuracy, lower sensitivity to the initial value, and faster convergence Fahroo and Ross (2008). In this study, different from many works that directly utilize the GPOPS software package Patterson and Rao (2014) to solve nonlinear problems, this study implements the pseudospectral collocation method to the convex optimization algorithms from scratch.

The basic idea of pseudospectral discretization is to approximate the continuous trajectory through an $N$th-order weighted interpolating polynomials at orthogonal collocation points Gong et al. (2008). Generally, a pseudospectral method is determined by three design elements: domain transformation $\Gamma$, $N$th-order collocation points $y^N$, and weight function $W$ Ross and Karpenko (2012). In this study, the Lagrange polynomials are used to approximate both state and control variables evaluated at the Legendre-Gauss-Lobatto (LGL) points, which are defined in the domain $[-1, +1]$. The distribution of the LGL points is non-uniform and dense near the end points. This property helps avoid the Runge phenomenon in Lagrange polynomial approximation Garg (2011). For finite-horizon optimal control problems, the time domain $[t_0, t_f] \ni t$ is mapped to the computational domain $[-1, +1] \ni \tau$ by
the following affine transformation:

\[ t = \Gamma(\tau) = \frac{t_f - t_0}{2} \tau + \frac{t_f + t_0}{2} \iff \tau = \Gamma^{-1}(t) = \frac{2}{t_f - t_0} t - \frac{t_f + t_0}{t_f - t_0} \quad (3.23) \]

Let \( \pi^N := \{\tau_i, i = 0, ..., N\} \) be the discretized LGL points, then the state trajectory \( X(\tau) \) is approximated on these LGL points using the Lagrange polynomials as

\[ X(\tau) \approx X^N(\tau) = \sum_{i=0}^{N} X_i L_i(\tau), \quad -1 \leq \tau \leq 1, \quad (3.24) \]

where \( L_i(\tau) \) represents the \( N \)th-order Lagrange interpolating polynomial:

\[ L_i(\tau) = \prod_{j=0, j \neq i}^{N} \frac{\tau - \tau_j}{\tau_i - \tau_j} \quad (3.25) \]

After the time domain is discretized and the discrete-to-continuous mappings of states and control are defined, the next step is to transcribe dynamics function Eqs. (3.6) and (3.8) to approximation equations by differentiating the interpolating polynomial as:

\[ \dot{X}(\tau_k) \approx \dot{X}^N(\tau_k) = \sum_{i=0}^{N} X_i \dot{L}_i(\tau_k) = \sum_{i=0}^{N} D_{ki} X_i \quad (3.26) \]

where \( k = 0, 1, ..., N \), and \( D \) is the Gauss pseudospectral differentiation matrix with constant elements for particular polynomials Fahroo and Ross (2008):

\[ D_{ki} = \begin{cases} -\frac{N(N+1)}{4} & k = i = 0 \\ \frac{L_N(\tau_k)}{L_N(\tau_i)(\tau_k - \tau_i)} & k \neq i, 0 \leq k, i \leq N \\ \frac{N(N+1)}{4} & k = i = N \\ 0 & 1 \leq k = i \leq N \end{cases} \quad (3.27) \]
Combining Eqs. (3.6), (3.8) and (3.26), the state-space constraints for the collocated control problem can be expressed as:

$$\sum_{i=0}^{N} D_{ki} X_i = \frac{t_f - t_0}{2} f(X_k, u_k, \tau_k)$$

(3.28)

where \(f(X_k, u_k, \tau_k)\) denotes the dynamics function in the state-space representation of Eqs. (3.6) and (3.8).

The last step is to approximate the cost functional \(J\) using Gauss–Lobatto quadrature rule:

$$J = \int_{t_0}^{t_f} g(X(t), u(t)) dt \approx \frac{t_f - t_0}{2} \sum_{i=0}^{N} g(X_i, u_i) w_i$$

(3.29)

where \(g(X(t), u(t))\) denotes the same function as the terms in Eq. (3.16), and \(w_i\) are the LGL integration weights given by:

$$w_i = \frac{2}{N(N+1)L_N^2(\tau_i)}, \quad 0 \leq i \leq N$$

(3.30)

In addition, to implement the pseudospectral method, it is crucial to compute the collocation points accurately. The explicit expressions for the LGL points do not exist, but they can be computed numerically. An efficient algorithm is to use Newton method to find the roots \(L_N'(\tau)\) recursively. The iteration rule and initial approximation are given as Shen et al. (2011):

$$\tau_{i+1}^{k+1} = \tau_i^k - \frac{1 - \tau_i^2}{2 \tau_i L_N'(\tau_i) - \frac{N(N+1)}{2} L_N(\tau_i)}, \quad k \geq 0, \quad i = 1, 2, \ldots, N - 1$$

(3.31)

$$\tau_i^0 = \frac{\sigma_i + \sigma_{i+1}}{2}, \quad i = 1, 2, \ldots, N - 1$$

(3.32)

$$\sigma_i^0 = \left[1 - \frac{N - 1}{8N^3} - \frac{1}{384N^4} \left(39 \cdot \frac{28}{\sin^2 \theta_i} \right) \right] \cos \theta_i + O(N^{-5}) \quad i = 1, 2, \ldots, N$$

(3.33)

$$\theta_i = \frac{4i - 1}{4N + 2} \pi \quad i = 1, 2, \ldots, N$$

(3.34)
After the above root-finding iteration converges to a criteria, such as max $[|\tau^{k+1}_i - \tau^k_i|] < \epsilon$, the corresponding differential matrix $D$ and weights $W$ can be calculated by Eqs. (3.27) and (3.30), respectively.

Through the above steps, the continuous-time optimal control problem defined in Problem 2 is transformed into a discrete-time numerical optimization problem as follows:

**Problem 3.**

$$
\text{minimize } J = \frac{T}{2} \sum_{i=0}^{N} w_i [\omega_2 R_i^2 + \omega_3 (v_i - v_i^d)^2 + \omega_4 (u_i - u_i^d)^2] \\
\text{subject to } R_i = R_0 + v_i t_{bd} + x_i - x_i^p \\
\quad u_i^d = \frac{\rho AC v_i^2}{2m} + (\mu \cos \theta + \sin \theta) g \\
\quad x_i^p - x_i \geq R_0 \\
\quad -u_{b}^{\max} \leq u_i \leq u_{e}^{\max} \\
\quad v_{\min} \leq v_i \leq v_{\max} \\
\quad \text{and Eq. (3.28)}. 
$$

where $i = 0, 1, ..., N$; and $T$ is the length of time horizon.

### 3.4.2 Convex Optimization

With advancements in computing and optimization theory, convex optimization has become increasingly popular in optimal control. Compared to nonlinear programming (NLP) algorithms, convex optimization offers many advantages. It allows for a wide range of optimization problems to be reformulated as a convex optimization problem, which is computationally efficient and can be solved in polynomial time Boyd et al. (2004). Additionally, as long as the feasible set is not empty, a globally optimal solution can be obtained without requiring a user-supplied initial guess, making it suitable for onboard applications.
To formulate a convex optimization problem, the equality constraint functions in Problem 2 and Problem 3 must be affine. As such, the nonlinear terms in Eq. (3.8) need to be linearized by a first-order Taylor series expansion method with respect to a reference speed trajectory, \( v^{k-1}(t) \), from the previous iteration of the SCP. As such, the nonlinear system dynamics in Eqs. (3.6) and (3.8) are linearized for the \( k \)th iteration, and their discrete forms in combination with the pseudospectral discretization are as follows:

\[
\sum_{i=0}^{N} D_{ki} X_i \approx \frac{T}{2} \left[ u_i - \frac{\rho A C_D}{2m} (2v_i^{k-1}v_i - (v_i^{k-1})^2) - (\mu \cos \theta + \sin \theta)g \right]
\]  

which will be enforced at the discretized segments and incorporated with Problem 3.

Similarly, the nonlinear term \( u_d \) in Eq. (3.37) is also linearized about the previous solution as follows:

\[
u_i^d \approx \frac{\rho A C_D}{2m} (2v_i^{k-1}v_i - (v_i^{k-1})^2) + (\mu \cos \theta + \sin \theta)g
\]  

which will be applied at each discretized node and then merged into Problem 3.

The exponential weight \( \omega_2 \) in the cost function is also a nonconvex function of state variable. Thanks to the sequential convex programming algorithm, it can be calculated as deterministic values using the solution from previous iteration as follows:

\[
\omega_2^k \approx \gamma e^{\alpha (x^{k-1} - x^p)}
\]  

With the approximations of the nonlinear dynamics and the nonlinear term in the cost functional, the discrete, nonconvex problem (Problem 3) is transformed into the following convex optimization problem with respect to the solution obtained from the previous iteration:
Problem 4.

\[
\begin{align*}
\text{minimize} & \quad \text{Eq. (3.35)} \\
\text{subject to} & \quad \text{Eq. (3.36), Eq. (3.38), Eq. (3.39), Eq. (3.40), Eq. (3.41), Eq. (3.42).}
\end{align*}
\]

By solving a sequence of Problem 4 with discretized state and control variables, the approximate optimal solution can be found for Problem 2. Based on the SCP algorithms investigated in the previous work Wang and Grant (2017, 2018); Wang and Lu (2020); Wang and McDonald (2020); Shi et al. (2022b), two improved SCP algorithms are developed and tailored to the CAV speed control problem in this study. The first algorithm is an SCP method with a line-search strategy as illustrated in Algorithm 3.1.

Step 4 of Algorithm 3.1 is the line-search algorithm used to find a feasible search step that facilitates stable convergence of the SCP. In addition, the inequality condition in Eq. (3.45) is called the Goldstein conditions, which ensures that the step length, \(\alpha^k\), makes sufficient but not too small decrease in the cost function Nocedal and Wright (2006). A rigorous proof of the convergence of the SCP method is still challenging; however, numerical simulations show a strong evidence of convergence by comparing with results with NLP method. Moreover, the accuracy and feasibility of the converged solution can be verified numerically by simply applying the converged optimal control to the original dynamic system, then comparing the propagated trajectory and the optimal solution obtained by the SCP method.

An alternative approach proposed in this study is based on trust-region methods, which have the same purpose of improving the convergence of SCP as line-search methods. Line-search methods use the previous solution to generate a search direction first, then focus on finding a suitable step length along that direction for the next iteration. Trust-region methods first define a trustworthy region around the initial guess then compute the step as an approximate solution in this region. If the step does not meet the criteria, the size of trust-region is reduced and another iteration
**Algorithm 3.1 Line-search SCP**

1) Set $k = 0$. Generate an initial vehicle trajectory $\hat{z}^0$ for the solution procedure by propagating the equation of motion in Eqs. (3.6) and (3.8) with a specific constant control from the current state of the vehicle, $x(t_0) = x_0$ and $v(t_0) = v_0$. Set $k = k + 1$.

2) For $k \geq 1$, parameterize Problem 4 using $\hat{z}^{k-1}$, solve Problem 4 given the dynamic constraint Eq. (3.28) and cost functional integration rule Eq. (3.29) for a solution pair $z^k = \{x^k, v^k, u^k\}$.

3) Check the convergence criteria:

$$\begin{align*}
\sup_{t_0 \leq t \leq t_f} |x^k(t) - x^{k-1}(t)| &\leq \epsilon_1, \\
\sup_{t_0 \leq t \leq t_f} |v^k(t) - v^{k-1}(t)| &\leq \epsilon_2,
\end{align*}$$

$k > 1$ (3.44)

where $\epsilon_1$ and $\epsilon_2$ are prescribed tolerances for the convergence criteria. If the above criteria are satisfied, the algorithm goes to step 5; otherwise to step 4.

4) Compute the search direction $p^k = z^k - \hat{z}^{k-1}$ for next iteration. Find a suitable step length $\alpha^k$ by starting from $\alpha^0 = 1$ and decreasing it with a contraction factor $c_1$, such that $\alpha^k = c_1 \alpha^{k-1}$, until sufficient decrease in the objective functional $J$ is achieved with a specified constant $c_2$, as described by the following inequality:

$$J(z^k) + (1 - c_2)\alpha^k \nabla T J_k^T p^k \leq J(z^k + \alpha^k p^k) \leq J(z^k) + c_2 \alpha^k \nabla T J_k^T p^k, \quad 0 < c_2 < 1/2$$

(3.45)

Then, update the reference trajectory $\hat{z}^k = \hat{z}^{k-1} + \alpha^k p^k$, set $k = k + 1$, and go back to step 2.

5) The iteration is terminated and solution is found as $z^* = \{x^*, v^*, u^*\} = \{x^k, v^k, u^k\}$. 

50
is performed. If the step makes sufficient progress, the trust-region radius can be
enlarged or remains the same. In this research, a trust-region SCP method is designed
for the considered speed control problem as illustrated in Algorithm 3.2.

For each $k \geq 1$, the trust-region constraint Eq. (3.46) is introduced in step 3
of Algorithm 3.2 to enhance the convergence of the SCP method. Step 4 is the
process of finding a suitable trust-region radius for the next iteration. The size of
the trust-region $\delta$ is critical to the speed of convergence. When $\delta$ is too large, it is
unlikely to find a step that is close to the optimal solution. If $\delta$ is too small, it may
takes many extra iterations to reach the optimal solution. Detailed discussion of the
trust-region methods can be found in Nocedal and Wright (2006). Compared to the
line-search method in Algorithm 3.1, trust-region method does not need to calculate
the derivative of objective functional $J$, such that the computation cost could be
reduced. However, in some cases, it may take more iterations to converge due to the
large model errors or strong artificial infeasibility.

### 3.4.3 Model Predictive Control

Model Predictive Control is a general control methodology that is suitable for real-
world driving conditions. Advances in computing hardware as well as efficient
numerical algorithms have made the MPC framework capable of handling nonlinear
dynamics and complex constraints in real time. MPC has been significantly advanced
and detailed descriptions can be found in Camacho and Alba (2013). Conceptually,
this approach is to solve the control problem over a short time horizon and take a short
interval of the control solutions to apply to the vehicle. After the vehicle performs
the control commands, the control problem is re-solved based on new system state
for the next time horizon. The algorithms developed in the previous subsections are
implemented under the MPC framework. Since the system in Problem 4 is discretized
by pesudo-spectral method, the MPC control law is given by: $u_t = u_0^*$.
Algorithm 3.2 Trust-region SCP

1) Set $k = 0$. Generate an initial vehicle trajectory $\hat{z}^0$ for the solution procedure by propagating the equation of motion in Eqs. (3.6) and (3.8) with a specific constant control from the current state of the vehicle, $x(t_0) = x_0$ and $v(t_0) = v_0$. Select appropriate values for trust-region radius $\delta = \delta_0$ and constants $0 < \eta < \beta_1 < 1 < \beta_2$. Set $k = k + 1$.

2) For $k \geq 1$, parameterize Problem 4 using $\hat{z}^{k-1}$, solve Problem 4 given the dynamic constraint Eq. (3.28), cost functional integration rule Eq. (3.29), and the trust-region constraint Eq. (3.46) for a solution pair $z^k = \{x^k, v^k, u^k\}$.

$$|z^k - \hat{z}^{k-1}| \leq \delta$$  \hspace{1cm} (3.46)

3) Check the convergence criteria:

$$\begin{align*}
\sup_{t_0 \leq t \leq t_f} |x^k(t) - x^{k-1}(t)| &\leq \epsilon_1, \\
\sup_{t_0 \leq t \leq t_f} |v^k(t) - v^{k-1}(t)| &\leq \epsilon_2, \\
&k > 1
\end{align*}$$  \hspace{1cm} (3.47)

where $\epsilon_1$ and $\epsilon_2$ are prescribed tolerances for the convergence criteria. If the above criteria are satisfied, the algorithm goes to step 5; otherwise to step 4.

4) Use the obtained solution $z^k$ to compute the model ratio $\nu^k$ as:

$$\nu^k = \frac{J(\hat{z}^{k-1}) - J(z^k)}{J(\hat{z}^{k-1}) - J'(z^k)}$$  \hspace{1cm} (3.48)

5) The iteration is terminated and solution is found as $z^* = \{x^*, v^*, u^*\} = \{x^k, v^k, u^k\}$.

where $J(z^k)$ is the original objective functional defined in Eq. (3.16) with nonlinear dynamics, and $J'(z^k)$ is the parameterized objective functional with linearized dynamics. The model ratio is a quality measurement of the approximate solution, in which the numerator of Eq. (3.48) is the actual reduction of the objective functional, and the denominator is the predicted reduction of the objective functional. If $\nu^k < 0.25$, then the trust-region radius $\delta$ is reduced such that $\delta = \beta_1 \delta$; if $\nu^k > 0.75$, $\delta$ is enlarged by $\delta = \beta_2 \delta$; otherwise, $\delta$ remains the same. Next, check if the model ratio is good enough such that $\nu^k > \eta$, in which case update the reference trajectory $\hat{z}^k = z^k$, set $k = k + 1$, and go back to step 2; otherwise, discard the solution $z^k$, and go back to step 2 to resolve the Problem 4.
future state of the vehicle is simulated by integrating the original dynamics with the Runge–Kutta method Dormand and Prince (1980).

Based on the previous development, a convex optimization-based real-time speed control framework is designed and displayed in Fig. 3.2. As introduced above, the control framework is formed by three major components: MPC framework, pseudospectral discretization, and sequential convex optimization. MPC enables the real-time control with instant response to the disturbance and uncertainties from environments in a robust manner. Pseudospectral discretization improves the accuracy and convergence of the numerical optimization algorithms. Convex optimization is an efficient optimization method that is suitable for real-time, onboard application. At each time step, the control system generates the control commands and the predicted trajectory for each vehicle using the SPaT information obtained by V2I communications and states and predicted trajectories of the surrounding vehicles received via V2V communications. After the vehicle executes the control commands, it broadcasts the new states and predicted trajectory to other vehicles, and then starts a new loop.

Remark 1: In this section, the signalized speed control problem has been formulated as a convex optimization problem solved in each step in Algorithm 3.1 and Algorithm 3.2. First, the solutions obtained by these two algorithms are locally optimal solutions to the original nonconvex problem (Problem 2), although the solution to Problem 4 solved in each step is a globally optimal solution to that convex subproblem. Second, the proposed approach is expected to have great potential for real-world automotive control because the convexified problem (Problem 4) has much lower computational complexity than its original nonconvex form (Problem 2) and can then be solved very reliably and efficiently using interior-point methods. Interior-point methods work very well in practice, and in some cases can be proved to solve the problem to a specified accuracy with a number of operations that does not exceed a polynomial of the problem dimensions Boyd et al. (2004). Specifically,
Figure 3.2: Convex optimization-based real-time speed control framework.
interior-point methods can solve a general convex optimization problem in a number of steps or iterations that is almost always in the range between 10 and 100. Ignoring any structure in the problem (e.g., sparsity), each step requires on the order of \( \max\{n^3, n^2m, F\} \) operations, where \( n \) is the problem size/dimension, \( m \) is the number of constraint functions, and \( F \) is the cost of evaluating the first and second derivatives of the objective and constraint functions. Advanced interior-point methods have been implemented in many popular software, including MOSEK (Andersen and Andersen 2000), SeDuMi (Sturm 1999), SDPT3 (Toh et al. 1999), and Gurobi (Gurobi Optimization 2021), and have proven to have a polynomial complexity bound of \( O(\sqrt{n}\log(1/\epsilon)) \) Yang (2011) for convex quadratic programming problems such as the ones solved in this work. Third, as will be seen in the following section that the computational efficiency of the proposed method is demonstrated in a MATLAB simulation environment, and it is expected to be much faster if implemented in compiled languages. These great advantages make the developed convex approach reliable and efficient to be embedded in actual automotive electronic control units such as the speed control units for embedded real-time automotive control and optimization. In the future, the developed algorithms will be integrated with the existing hardware-in-the-loop environment to test the performance of the proposed algorithms in actual automotive hardware platforms.

Remark 2: The proposed method also brings some benefit in finite precision arithmetic. For example, the pseudospectral discretization method utilized in the developed convex algorithms requires fewer discrete nodes than other discretization methods, such as the commonly used Euler and trapezoidal rules, to achieve the same solution precision. That is to say, the pseudospectral method can achieve higher accuracy (at least machine precision) with the same number of discrete nodes than some other discretization methods (Garg, 2011). Additionally, normalizing the problem formulation with nondimensional variables is planned to further reduce the potential ill-condition of the problem, which is believed to reduce rounding errors and improve numerical stability of the solution approach.
3.5 Simulation Results

To demonstrate the effectiveness and performance of the proposed speed optimization method, a series of simulations were conducted considering a single lane road with traffic lights at every 500 m. The optimization methodology described in this study has been implemented in YALMIP Lofberg (2004), a MATLAB-based optimization framework. The solver for convex optimization is Gurobi Gurobi Optimization (2021) and the nonlinear optimization solver used for comparison is IPOPT Biegler and Zavala (2009).

In the simulations, all vehicles are considered to be identical with the parameters listed in Table 3.1. The road grade is assumed to be 3%. The optimal cruising speed is calculated as $v^* = 13.46$ m/s by Eq. (3.20). The maximum speed limit of the road is considered to be $v^{\text{max}} = 20$ m/s, while the minimum allowable speed is $v^{\text{min}} = 0$ m/s. All the vehicles start from position 0 with a random initial speed range from 10 m/s to 20 m/s. Except the first vehicle, every other vehicle joins the simulation at the time when there is a safe distance from the preceding vehicle.

3.5.1 Comparison of Solutions

To verify the effectiveness and demonstrate the accuracy of the proposed algorithms, a simple example is first considered, in which a single vehicle travels 500 m to pass an intersection at $t = 30$ s. The solutions obtained by solving Problem 2 with nonlinear programming algorithm (IPOPT) and Problem 4 with the proposed algorithms are compared. Serving as a baseline, the trajectory obtained by IPOPT is discretized uniformly in very small time step, i.e., 1 ms. In Fig. 3.3, the blue and green lines with markers represent the trajectories obtained by proposed PS-convex optimization Algorithm 3.1 and Algorithm 3.2, respectively, which are discretized into 45 intervals. The red lines are results from IPOPT. It can be seen from the comparison that the profiles matches extremely well. This comparison demonstrates the optimality and accuracy of the solutions of the proposed SCP algorithms. The small difference in the
Figure 3.3: Comparison of position, velocity, and control profiles.
control profile may be attributed to several potential causes, including the different methods for linear approximation of nonlinear dynamics, different discretization strategies, and different convergence criteria used in the problems and algorithms.

The convergence of the SCP algorithms is displayed in Fig. 3.4. The convergence tolerances, as specified in Eq. (3.44), are set to $\epsilon_1 = 1e^{-6}, \epsilon_2 = 1e^{-8}$. As shown in the figure, both algorithms converge in 3 iterations for this example and their converged results are very close. The initial trajectory is propagated by constant zero control, causing large changes in the values of the state variables and the cost functional during the first iteration. It is worth noting that, by using the MPC framework, the initial trajectory can be given by the solution from the previous time step. Such that, the convergence can be improved and the number of iterations can be reduced.

In addition, the CPU time costs are compared as well. In general, solving Problem 4 by convex optimization takes considerably less computational time than solving Problem 2 with IPOPT due to the fewer number of variables and less complicated dynamics in Problem 4. The performance of NLP solvers highly rely on the initial guess. The simulations show that the IPOPT solver has an unstable or slow convergence if a good initial guess is not supplied. As listed in Table 3.2, for the accurate baseline solutions with 1 ms step size, it takes IPOPT about 5 to 15 s to converge for solving Problem 2 without a user-supplied initial guess. While using the previous solutions as initial guesses under the MPC framework, the CPU time of IPOPT reduces to 1 to 2 s. For rough discretization with the same number of nodes (46 nodes in this subsection) as used for the convex algorithms, the computational time of IPOPT ranges from 30 to 100 ms at the expense of accuracy. A group of simulation cases with 46 nodes are examined, and the results show that IPOPT takes an average of 70.69 ms to solve Problem 2 without user-supplied initial guesses and 37.24 ms on average to solve Problem 2 with user-supplied initial guesses, while Gurobi takes an average of 41.56 ms to solve Problem 4 in every iteration and 2 to 3 iterations to converge for each solution. More importantly, as long as the problem is feasible, convex optimization solvers have guaranteed convergence within
Table 3.2: Comparison of average computational times.

<table>
<thead>
<tr>
<th>Problem</th>
<th>46 nodes</th>
<th>1001 nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 2 (IPOPT without user-supplied initial guess)</td>
<td>37.24 ms</td>
<td>1.22 s</td>
</tr>
<tr>
<td>Problem 2 (IPOPT with user-supplied initial guess)</td>
<td>70.69 ms</td>
<td>9.85 s</td>
</tr>
<tr>
<td>Problem 4 (Gurobi)</td>
<td>41.56 ms</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 3.4: Convergence of trajectories and objective function.
finite number of iterations, without the need of initial guesses. The simulations are performed in Matlab on a MacBook Pro with a 64-bit Mac OS and an Intel Core i7 2.2 GHz processor. Moreover, if the number of discretization intervals is reduced or the tolerances of convergence criteria is increased, the CPU time cost will be less. The computational efficiency could be further improved if the solver is transferred to compiled programming environment or more powerful processors are used.

3.5.2 Case 1: Uniform SPaT with 10 Vehicles

The previous subsection has demonstrated effectiveness and accuracy of the developed SCP algorithm through comparison with IPOPT. In this subsection, a complicated case with multiple vehicles is considered, and the MPC framework is applied along with Algorithm 3.1. Different from the previous example, the time horizon of the optimization problem is selected as 10 s and the entire simulation duration is 300 s. To balance the computational cost and solution accuracy, the number of discretization nodes is set to be 26 for each time horizon. Since only the first control command from the optimal solutions is used, and \( \tau_1 = -0.9887 \) is obtained by the pseudospectral method with 26 nodes utilized in this case, the sampling time is calculated by \( \frac{T}{2}(\tau_1+1) \) and equal to 56.36 ms. The duration of signal phases is consider to be constant in this case. Specifically, the red phase and green phase are 30 s and 15 s, respectively.

Fig. 3.5 displays ten trajectories obtained by the proposed approach. It can be seen that all the vehicles successfully pass the traffic signals during green phases with a safe distance between each other. Profiles of velocity indicate that, when a new vehicle joins the simulation, the preceding vehicles need to accommodate their speed such that the lastly joined vehicle can pass the intersection during the green phase. After all the vehicles have joined and run for a while, the whole group reaches a stable state that all of them are able to maintain the optimal cruising speed. This means, although the vehicles are controlled independently, they can cooperatively utilize the green phase to achieve a group-level, efficient energy consumption and
Figure 3.5: Vehicle trajectories of Case 1.
traffic throughput. Meanwhile, it also means that the considered duration of green phase has enough capacity for the simulated number of vehicles. Therefore, to further investigate the effectiveness of the proposed algorithms, the examples with varying signal duration and more vehicles are considered in the following subsections.

3.5.3 Case 2: Uniform SPaT with 20 Vehicles

In this case, an additional 10 vehicles are incorporated into the simulations, making the situation more complex compared to the previous example. Some vehicles may not be able to pass through the intersection in the first green phase. As described in Section 3.3.2, these vehicles will choose the next green phase and calculate their reference speed accordingly. Figure 3.6 shows that the first group of 11 vehicles have selected the first green phase. Meanwhile, to fit as many vehicles as possible into the green phase and achieve better overall mobility, the gap distances between the leading vehicles were sacrificed. After passing the first intersection, the group of vehicle reaches the stable state of cruising.

For the second group of 9 vehicles, despite the second green phase has enough space, the upper bound of their reference velocity is smaller than the optimal cruising speed $v^*$. Such that, to have better fuel efficiency, their reference velocity is set to be the upper bound of the green phase, which is to arrive at the intersection at the beginning of the green phase. It should be noted that only the first vehicle of the group can meet the reference velocity, so other vehicles must be penalized by the cost function. Consequently, the distances between vehicles of the second group are reduced for compensation of velocity tracking violation. In addition, by adjusting the desired distance $R$ and weights of cost functional terms, the balance of fuel efficiency and safe inter-vehicle distance can be customized.
Figure 3.6: Vehicle trajectories of Case 2.
3.5.4 Case 3: Random SPaT

In the real-world application, the SPaT is not always uniform. A sequence of SPaT for multiple intersections is randomly generated in this example. Specifically, the duration of green phase varies from 10 to 30 s, and the signal cycle length is constantly 45 s. As shown in Fig. 3.7, when the green phase is long enough, the vehicles may keep a desired distance between each other. Otherwise, the distance between vehicles is reduced until the minimum inter-vehicle distance requirement is meet. In this way, more vehicles can pass the intersection during the same green phase.

The results of non-uniform SPaT are compared to those of uniform SPaT. Unlike uniform SPaT, leading vehicles need to frequently adapt their speed and distance to changing SPaT for following vehicles, allowing more vehicles to pass through the intersection during the same green phase. It can be seen that the last few vehicles have more stable tracking of the optimal cruising speed, meaning that leading vehicles sacrifice their optimality for the benefit of following vehicles. While determining the exact impact on overall fuel efficiency is challenging, the improvement in traffic flow mobility is evident, especially in heavy traffic where reducing the chance of congestion is desirable.

3.5.5 Comparison with Baseline Simulation in SUMO

The above results indicate that the proposed speed trajectory optimization algorithm not only generates near-optimal trajectories for individual vehicles in real-time, but also enhances the performance of a group of vehicles. To further validate the effectiveness of the method, simulations were run in SUMO (Simulation of Urban MObility) Lopez et al. (2018), a microscopic traffic simulation software that provides detailed information about simulated objects and allows for adjustments to parameters at every time step. The SUMO simulations were conducted with no V2I or V2V communications, to simulate the behavior of real-world drivers, while other vehicle parameters remained the same.
Figure 3.7: Vehicle trajectories of Case 3.
The SUMO simulations, shown in Fig. 3.8, reveal that the vehicles frequently stop at traffic lights due to red lights. This results in a significant portion of the green phase being wasted and slowing down the entire group of vehicles. The average velocity of the traffic flow in all simulations is listed in Table 3.3. Although maximizing the average velocity is not one of the objectives of the proposed trajectory optimization algorithm, the results show that tracking the reference velocity to avoid idling at traffic signals can still improve the mobility of the traffic flow.

The results of the fuel consumption comparison are shown in Fig. 3.9. It can be seen that the proposed CAV-based optimization method significantly improves the fuel efficiency in all three cases. For instance, in case 1, the average fuel consumption rate of the proposed method is 58.49 miles per gallon (MPG), while the rate obtained from the SUMO simulation is 38.45 MPG, resulting in a 34.54% reduction compared to the proposed method. These results indicate that the CAV-based speed control method can effectively reduce fuel consumption while improving traffic mobility.

### 3.6 Conclusion

In this Chapter, a CAV-based optimal speed control approach was proposed for vehicles traveling in signalized corridors to avoid idling at the intersections, keep a safe inter-vehicle distance, and minimize fuel consumption. The vehicles’ velocities are adjusted based on SPaT information obtained through V2I communications, to reduce the need for frequent deceleration and acceleration when passing intersections. Additionally, V2V information exchange among neighboring vehicles enables speed coordination, improving traffic mobility and safety.

The proposed approach employs a pseudospectral discretization method and a sequential convex programming method to develop a real-time, onboard algorithm with strong potential. The MPC control framework was also utilized to generate speed control commands at each time step, ensuring collision avoidance and improved
Table 3.3: Average velocity of traffic flow.

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Method</td>
<td>14.25 m/s</td>
<td>13.26 m/s</td>
<td>14.23 m/s</td>
</tr>
<tr>
<td>SUMO</td>
<td>13.83 m/s</td>
<td>12.69 m/s</td>
<td>13.69 m/s</td>
</tr>
</tbody>
</table>

Figure 3.9: Comparison of Fuel Consumption.
inter-vehicle coordination. The algorithm’s accuracy and convergence were verified by comparison with optimal solutions from IPOPT.

The effectiveness of the approach was validated through simulations of cases with both uniform and random SPaT. The comparison with SUMO simulations showed improvement in both traffic mobility and fuel efficiency. While the solution is sub-optimal, it effectively handles safety constraints in dynamic traffic environments, improving traffic mobility and reducing fuel consumption with real-time performance.

The ultimate goal of this study is to create a comprehensive control framework for CAVs in traffic systems to enhance road safety, improve traffic flow, and increase energy efficiency. The persistent problem of traffic congestion and idling results in excessive energy consumption, and a systematic optimization approach is imperative to minimize energy usage and promote sustainable traffic systems. In the future, the plan is to implement a cooperative control strategy for both CAVs and signalized intersections, aligning the motion planning of CAVs with the optimization of traffic signals to alleviate congestion and enhance energy efficiency.
Chapter 4

Study II: Pseudospectral Convex Optimization for On-Ramp Merging Control of Connected Vehicles
4.1 Abstract

It can be a challenging task for human drivers to merge onto highways due to the complex vehicle negotiations and potential risks within limited time and space. Connected vehicle (CV) technologies offer a solution to this problem and provide numerous benefits for road safety, traffic mobility, and energy efficiency. However, the real-time optimal control of CVs remains a challenge due to the nonlinear vehicle dynamics, non-convex fuel consumption models, and highly dynamic uncertain inter-vehicle interactions. The motion control system of CAVs plays a crucial role in ensuring road safety; however, real-time updates to vehicle maneuvers in response to dynamic surroundings can be challenging due to the complex interactions between vehicles and nonlinear dynamics. Conventional optimization methods may struggle to find a solution in the non-convex solution space imposed by many constraints including speed coordination and nonlinear dynamics.

To address these issues, a novel real-time optimal control approach is proposed for onboard application, balancing computational efficiency and solution optimality. The approach integrates the pseudospectral collocation method with a sequential convex programming approach to develop two new optimization algorithms within a model predictive control framework, generating real-time optimal merging speed profiles. The first algorithm uses a line search technique to improve convergence, while the second leverages the trust region method for improved computational efficiency. The optimality and convergence of both proposed algorithms were investigated by comparing their solutions to a popular non-linear solver. The simulation results show that the proposed methods outperform the benchmark in terms of computational cost, fuel consumption, and traffic efficiency. On average, the proposed fuel-efficient merging rule can save 57.1% fuel consumption across four different traffic volumes, while the proposed optimal control algorithms can reduce 2.2% travel time compared to the “first-in-first-out” merging rule.
4.2 Introduction

According to a report from the National Highway Traffic Safety Administration Administration (2020), vehicle crash fatalities on urban highways have been higher than those on rural roadways since 2016. In particular, the number of fatalities on urban highways has increased 48% since 2011 and was 30% higher than rural areas in 2020. Therefore, as a bottleneck challenge of highway transportation and one of the most challenging scenarios for ground vehicles, on-ramp merging control has attracted lots of interest from many researchers. The merging process is a silent interactions involving complex traffic negotiations, where the traffic situation needs to be assessed and merging decisions and vehicle operations need to be made within a very limited time and distance. Any mistake in a single step would cause crash.

The goal of on-ramp merging control is to coordinate the vehicles on two different roads to pass through the merging area as safely and smoothly as possible. Ramp metering has been investigated to improve traffic flow Mizuta et al. (2014), which mitigates the freeway congestion around merging areas by controlling the on-ramp inflow rate. However, it suffers drawbacks including increased travel delay and high fuel consumption Systematics (2001).

The basic idea of optimal on-ramp merging is to locate or produce a sufficient gap between vehicles on the main road for the on-ramp vehicles to merge, while minimizing or eliminating the braking operations caused by the merging maneuvers. With the help of CV technologies, vehicles can communicate with the infrastructure (V2I) and other vehicles (V2V) about their states (e.g., location, velocity, and acceleration), planned trajectories and other information. The availability of these data provides CVs with the opportunity to be coordinated to accomplish safe and smooth merging maneuvers.

A number of studies have demonstrated that the motion of CVs can be coordinated for cooperative merging and be optimized to improve road safety, reduce travel delay and energy consumption, and increase throughput Rios-Torres and Malikopoulos.
However, most related works focus on generating feasible merging trajectories for CVs, while ignoring the real-time execution of the reference trajectory. Since the motion control system of CVs is safety-critical and requires real-time update of maneuvers to react to the dynamic surrounding environment, the challenges of the current optimal control methods for vehicle on-ramp merging include the high computational cost of existing optimization solvers and inability of the control algorithms to cope with nonlinear vehicle dynamics, dynamic and uncertain environments, and nonconvex constraints.

To address these challenges, this study develops a new merging control approach that balances the computational efficiency and solution optimality while maintaining real-time performance and safe merging operations. In particular, a pseudospectral convex optimization formulation with hard collision-avoidance constraints is devised for the merging of CVs at roadway on-ramps. With advantages of globally optimal solutions and polynomial solution time, the proposed sequential convex programming (SCP) algorithms address the nonlinear vehicle dynamics and nonconvex motion constraints with guaranteed real-time performance. Furthermore, the performance of the merging control algorithm is enhanced by a line-search technique and a trust-region method, thus leading to two improved SCP algorithms. Moreover, the proposed algorithms are implemented under a model predictive control (MPC) framework to deal with errors and uncertainties for better inter-vehicle coordination.

4.3 Problem Formulation

The aim of this research is to design an optimal speed control model for CAVs merging onto highways. By considering a standard single-lane highway merging scenario (as shown in Fig. 4.1), the objective of the control problem is to minimize fuel consumption, alleviate congestion, and avoid collisions during merging at roadway on-ramps. To accomplish this, a centralized controller placed within or near the control
Figure 4.1: Roadway on-ramp merging scenario in this study.
zone assigns each CV a reference speed based on a predetermined merging sequence. In real-time, each vehicle solves an optimal control problem (OCP) to determine its speed profiles and regulate its movement with minimal acceleration and deceleration. In the merging zone, the merging sequence is fixed, and the OCPs are solved to coordinate the vehicles’ speeds and distances to complete the merging maneuver. The generated vehicle trajectories are assumed to be shared with surrounding vehicles through V2V communication. In this section, we will describe the vehicle dynamic model, the process of determining the merging sequence, and the formulation of the OCP in detail.

4.3.1 Vehicle Dynamics

In this study, the longitudinal vehicle dynamics are considered as follows Vahidi et al. (2005):

\[ m \dot{v} = \frac{T_e}{r_g} - F_b - F_{aero} - F_{other} \]  

(4.1)

where \( m \) is the mass, \( v \) denotes velocity, and \( T_e \) represents the engine torque. \( r_g \) is computed by the total gear ratio dividing the wheel radius, and we assume that \( r_g \) is constant. \( F_b \) is the braking force at the wheels. \( F_{aero} \) represents the force produced by aerodynamic resistance, and it is defined by:

\[ F_{aero} = \frac{1}{2} \rho A C_D v^2 \]  

(4.2)

where \( A \) denotes the frontal area of vehicle, \( C_D \) represents the coefficient of aerodynamic drag, and \( \rho \) is the air density. \( F_{other} \) represents a combination of forces due to the rolling resistance and road grade, and it is determined by:

\[ F_{other} = mg (\sin \theta + \mu \cos \theta) \]  

(4.3)

where \( \mu \) represents the coefficient of rolling resistance, \( g \) denotes the gravitational constant, and \( \theta \) is the road gradient. In the simulations, \( \theta \) is assumed to be constant.
In this study, all the vehicles are assumed to be identical. The fuel consumption model is adopted from Kamal et al. (2011) with the rate of fuel consumption $\dot{m}_f$ given by:

$$
\dot{m}_f = \begin{cases} 
\alpha_0 + \alpha_1 v + \alpha_2 v^2 + \alpha_3 v^3 \\
+a(\beta_0 + \beta_1 v + \beta_2 v^2), & a > 0 \\
\alpha_0, & a \leq 0
\end{cases}
$$

(4.4)

where $a$ is acceleration of the vehicle, the constants of third-order polynomial function $\alpha_i$ and $\beta_i$ are determined by fitting the experiment data. If vehicles are idling or braking, the fuel consumption rate is a constant value of $\alpha_0$. The designated values of vehicle model parameters are listed in Table 4.1.

### 4.4 Merging Sequence Determination

When there are vehicles on both the on-ramp road and mainline road, it is necessary to determine which vehicle passes through the merging zone first. The simplest and most common approach is the “first-in-first-out” (FIFO) rule, where the merging order is based on the minimum arrival time or distance to the merging zone. While this approach prioritizes travel-time efficiency, it neglects fuel efficiency. Other studies such as Jing et al. (2019) and Chen et al. (2020) aim to balance travel-time and fuel efficiency by using optimization techniques to establish the merging sequence. However, as the number of merging vehicles increases, the computational load of finding the optimal sequence grows factorially, making a real-time solution infeasible. Another approach to determine the merging sequence is to establish explicit rules that prioritize specific traffic control goals. For instance, Ding et al. (2019) proposed a rule-based vehicle merging algorithm that encourages vehicles to merge in groups to reduce the potential for alternating merging and resultant delays.
Table 4.1: Values of vehicle model parameters.

<table>
<thead>
<tr>
<th>Parameter (Unit)</th>
<th>Value</th>
<th>Parameter (Unit)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$ (kg)</td>
<td>1200</td>
<td>$u_e^{\text{max}}$ (m/s$^2$)</td>
<td>3</td>
</tr>
<tr>
<td>$\rho$ (kg/m$^3$)</td>
<td>1.184</td>
<td>$u_p^{\text{max}}$ (m/s$^2$)</td>
<td>3</td>
</tr>
<tr>
<td>$A$ (m$^2$)</td>
<td>2.5</td>
<td>$v^{\text{max}}$ (m/s)</td>
<td>20</td>
</tr>
<tr>
<td>$C_D$</td>
<td>0.32</td>
<td>$v^{\text{min}}$ (m/s)</td>
<td>0</td>
</tr>
<tr>
<td>$g$ (m/s$^2$)</td>
<td>9.8</td>
<td>$\mu$</td>
<td>0.015</td>
</tr>
<tr>
<td>$\alpha_0$ (mL/s)</td>
<td>0.1569</td>
<td>$\beta_0$ (mLs/m)</td>
<td>0.0724</td>
</tr>
<tr>
<td>$\alpha_1$ (mL/m)</td>
<td>2.450e-2</td>
<td>$\beta_1$ (mLs$^2$/m$^2$)</td>
<td>9.681e-2</td>
</tr>
<tr>
<td>$\alpha_2$ (mLs/m$^2$)</td>
<td>-7.415e-4</td>
<td>$\beta_2$ (mLs$^3$/m$^3$)</td>
<td>1.075e-3</td>
</tr>
<tr>
<td>$\alpha_3$ (mLs$^2$/m$^3$)</td>
<td>5.975e-5</td>
<td>$R_0$ (m)</td>
<td>2.5</td>
</tr>
<tr>
<td>$t_{\text{hd}}$ (s)</td>
<td>1.3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4.4.1 Optimization-based Approach

In this study, various methods for determining the merging sequence are examined. The optimization-based approach formulates the merging problem as a constrained optimization to minimize objectives such as total travel time, fuel consumption, or a combination of both. The centralized merging controller detects a new vehicle and evaluates all possible merging sequences by formulating and solving a trajectory optimization problem for each vehicle. The performance indices of all sequences are calculated and compared, and the best one is selected and broadcast to vehicles within the control zone.

The first step in determining the merging sequence is to form all the possible sequences and assign an arrival time to each vehicle. For safety reasons, the difference in arrival time between two consecutive vehicles must be greater than a safety headway $\Delta t_a$. According to Algorithm 4.1, if a vehicle $j$ intends to merge in front of another vehicle $j - 1$, its arrival time $t_{arrival}^j$ is set to be $t_{arrival}^{j-1} - \Delta t_a$. If the minimum arrival time of vehicle $j$ is greater than $t_{arrival}^{j-1} - \Delta t_a$, $t_{arrival}^j$ is set to its minimum arrival time.

Furthermore, the minimum arrival time $t_{min}^j$ of vehicle $j$ is calculated by assuming it accelerates to its maximum speed with maximum acceleration and then cruises to the merging zone, following the parameters and limits described in Table 4.1.

The second step is to choose the preferred merging sequence by evaluating performance. If the objective of the control is to minimize travel time, the simplest approach would be to add up the arrival times assigned to all vehicles and choose the one with the smallest total. However, in this study, the aim is to enhance fuel efficiency, so it is necessary to determine the speed profile using a trajectory optimization method. The trajectory optimization problem for vehicle $i$ can be formulated by applying the optimal control method and constraints outlined in Chapter 3.

As previously discussed in Chapter 3, while minimizing acceleration and deceleration may not directly lead to minimizing fuel consumption, fuel consumption
Algorithm 4.1 Pseudo-code for assigning arrival time to the vehicles

1 Inputs: Merging sequence M, states of all vehicles
2 Outputs: The assigned arrival time of all vehicles
3 for each vehicle $i$ in M from the last one do
4   if the arrival time of preceding vehicle $i-1$ is later than the arrival time of vehicle $i$ minus $\Delta t_a$ then
5     if the minimal arrival time of preceding vehicle $i-1$ is smaller than the arrival time of vehicle $i$ minus $\Delta t_a$ then
6       Assign a new arrival time to vehicle $i-1$ as $t_i^{-}\text{arrival} - \Delta t_a$
7     else
8       Assign a new arrival time to vehicle $i-1$ as its minimum arrival time $t_i^{-1}_{\text{min}}$
9   for each vehicle $j$ in M from the first one: do
10     if the arrival time of following vehicle $j+1$ is earlier than the arrival time of vehicle $j$ plus $\Delta t_a$ then
11       if the maximal arrival time of following vehicle $j+1$ is smaller than the arrival time of vehicle $j$ plus $\Delta t_a$ then
12         Return The merging sequence M is infeasible
13       else
14         Assign a new arrival time to vehicle $j+1$ as $t_j^{+}\text{arrival} + \Delta t_a$

and vehicle acceleration are related. This is because fuel consumption is influenced by factors such as mechanical efficiency, aerodynamic drag, road grade, rolling resistance, etc. However, it can be stated that fuel consumption increases with vehicle acceleration when the vehicle’s speed is fixed. Given the fuel-optimal solution by solving an optimal speed control problem, the total fuel consumption can be calculated using equation Eq. (4.4). Then the optimal merging sequence is chosen as the one with the minimum fuel consumption.

4.4.2 Rule-based Approach

Despite the optimization-based approach producing the optimal merging sequence with the best fuel efficiency, its computational cost is prohibitive and unrealistic for real-time applications. For instance, if there are 10 on-ramp and 10 mainline vehicles waiting to merge, the number of possible merging sequences would be 184,756. Although infeasible candidates are eliminated, finding the answer can still take hours
or even days. As a result, this study also includes the development of a rule-based algorithm to determine the desired merging sequence in real-time.

Based on a safety-first paradigm, a set of cooperative rules are proposed to determine the merging sequence of vehicles on the on-ramp road and the mainline road, taking into account different control goals. The rules are outlined in Algorithm 4.2 and illustrated in Fig. 4.2. The vehicles on the on-ramp road must give way to those on the mainline road, which often leads to reduced traffic flow speed on the on-ramp road. To enhance the efficiency and capacity of on-ramp roads, these vehicles should aim to merge into the first available gap with minimal deceleration. Meanwhile, the vehicles on the mainline road should not be affected by the merging vehicles, as this can cause congestion, extra fuel consumption, and even collisions. When there is insufficient space for merging, the vehicles on the mainline road may voluntarily accelerate to create gaps if feasible. When traffic volume is low and there is no conflict between vehicles, they should be allowed to travel at their preferred speed.

In order to schedule the arrival time and reference speed of the newly joined vehicle, once the merging sequence is determined by Algorithm 4.2, different cases for calculating the vehicles’ arrival time to the merging zone are listed in Table 4.2. The assigned arrival time of the preceding vehicle is represented by $t_{\text{front, arrival}}$, the assigned arrival time of the following vehicle is represented by $t_{\text{back, arrival}}$, and the time headway for safety is represented by $t_{\text{hd}}$. When gaps are made by vehicle accelerations, the recalculation of arrival times is passed from the newly joined vehicle either backward or forward to all the involved vehicles in the merging sequence. On the other hand, if the vehicles are not impacted by the newly joined vehicle, their original arrival times are maintained.
Figure 4.2: A rule-based cooperative merging strategy.
Table 4.2: Cases for vehicles in control zone to recalculate arrival time when new vehicles join.

<table>
<thead>
<tr>
<th>Vehicle</th>
<th>Case</th>
<th>Arrival time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mainline</td>
<td>1: no conflict or not able to accelerate</td>
<td>not change</td>
</tr>
<tr>
<td></td>
<td>2: accelerate to catch up to the preceding vehicle</td>
<td>$t_{\text{arrival}} + t_{\text{hd}}$</td>
</tr>
<tr>
<td></td>
<td>3: accelerate to make a gap</td>
<td>$t_{\text{arrival}} - t_{\text{hd}}$</td>
</tr>
<tr>
<td></td>
<td><strong>Ramp</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1: no conflict or not able to accelerate</td>
<td>not change</td>
</tr>
<tr>
<td></td>
<td>2: accelerate to merge into a gap without adjustment to the mainline vehicles</td>
<td>$t_{\text{arrival}} + t_{\text{hd}}$</td>
</tr>
<tr>
<td></td>
<td>3: accelerate to merge into a gap created by accelerating the preceding vehicles</td>
<td>$t_{\text{arrival}} - t_{\text{hd}}$</td>
</tr>
</tbody>
</table>
Algorithm 4.2 Pseudocode for the determination of merging sequence

1 function DETERMINE Merging Sequence
2     if New vehicle is on mainline road then
3         if The last vehicle in merging sequence is on mainline road then
4             The new vehicle merges last
5         else
6             if The last vehicle is slower than the desired speed and cannot make enough space then
7                 loop The $N$ ramp vehicles at the end of merging sequence
8                     if The vehicle $i$ is faster than the desired speed then
9                         The new vehicle merges after vehicle $i$
10                        Break loop
11                     else if Loop Ends then
12                         The new vehicle merges before all the $N$ ramp vehicles
13                 else
14                     The new vehicle merges last
15             else
16                 if The last vehicle in merging sequence is on ramp road then
17                     The new vehicle merges last
18             else
19                 if There is no ramp vehicle in the merging sequence then
20                     loop All the vehicles in merging sequence
21                         if The new vehicle can merge before vehicle $i$ then
22                             if There is enough space before vehicle $i$ or possible to make space then
23                                 The new vehicle merges before vehicle $i$
24                                 Break loop
25                         else if Loop Ends then
26                             The new vehicle merges last
27                     else
28                     loop The mainline vehicles at the end of merging sequence
29                         if The new vehicle can merge before vehicle $i$ then
30                             if $i = 1$ then
31                                 if There is not enough space before vehicle $i$ and not possible to make space then
32                                     loop The ramp vehicles in front of vehicle $i$
33                                         if The vehicle $i$ can merge before vehicle $j$ then
34                                             The new vehicle merges at position $i$ and vehicle $i$ merges before vehicle $j$
35                                             Break loop
36                             else
37                                 The new vehicle merges before vehicle $i$
38                         else
39                             if There is enough space before vehicle $i$ or possible to make space then
40                                 The new vehicle merges before vehicle $i$
41                                 Break loop
42                         else if Loop ends then
43                             The new vehicle merges last
44                     else if Loop ends then
45                     The new vehicle merges last
4.5 Optimal Control Problem

Once the merging sequence and arrival times to the merging zone have been determined, the vehicles need to be guided through the control zone with reference speed profiles. These profiles can be obtained by solving a nonlinear optimal control problem (OCP) that balances multiple driving requirements such as fuel efficiency, safety distance from the preceding vehicle, desired speed, and passenger comfort. Instead of using a terminal constraint, the arrival time is converted into a reference speed for the vehicle to track. This allows the speed control problem to be solved iteratively over a rolling time horizon $T$, as will be described in Section 4.7. The speed control problem can be formulated as follows:

Problem 5.

\[
\begin{align*}
\text{minimize} \quad & J = \int_t^{t+T} \omega_2 R(t)^2 + \omega_3 (v(t) - v_d)^2 \\
& \quad + \omega_4 (u(t) - u_d(t))^2 dt \\
\text{s.t.} \quad & \dot{x}(t) = v(t) \\
& u(t) = u_e(t) - u_b(t) \\
& \dot{v} = u(t) - \frac{\rho AC_D v(t)^2}{2m} - (\mu \cos \theta + \sin \theta)g \\
& u_d(t) = \frac{\rho AC_D v(t)^2}{2m} + (\mu \cos \theta + \sin \theta)g \\
& R(t) = R_0 + v(t) t_{hd} + x(t) - x_p(t) \\
& x_p(t) - x(t) \geq R_0 \\
& 0 \leq u_e(t) \leq u_e^{\max} \\
& 0 \leq u_b(t) \leq u_b^{\max} \\
& -u_b^{\max} \leq u(t) \leq u_e^{\max} \\
& v^{\min} \leq v(t) \leq v^{\max}
\end{align*}
\]
In this problem, the position of the vehicle is represented by $x(t)$, and the control inputs $u(t)$ consist of two components: the vehicle’s acceleration generated by the engine power $u_e = \frac{T_e}{m g}$ and the deceleration due to the braking power $u_b = \frac{F_b}{m}$. The cost function, given in Eq. (4.5), takes into account multiple driving requirements, such as maintaining a safe distance with the preceding vehicle, tracking the reference speed, and minimizing control efforts. The first term ensures a safe distance by penalizing deviation from the desired safety distance $R$, which is calculated using Eq. (4.10). In Eq. (4.10), $R_0$ is the minimum requirement of distance between the vehicles, $x_p$ denotes the position of its preceding vehicle, and $t_{hd}$ represents the safety requirement of headway measured in time.

The second term helps the vehicle to track the reference speed, which is computed based on its arrival time to the merging zone. The last term of Eq. (4.5) is a penalty term for minimizing control efforts and avoiding control jitters. At most of the time, the vehicles travel with stable speeds and have stable fuel consumption rates as well. Therefore, it is helpful to include a compensation term $u_d(t)$ to the cost function to have better optimality. The lower and upper bounds for the control variables are defined in Eqs. (4.12) to (4.14), and Eq. (4.15) provides the constraints for vehicle speed.

In addition, each term in Eq. (4.5) is weighed by an adjustable weight $\omega_i$ to enhance decision-making. The weight $\omega_2$, which ensures safe inter-vehicle distances, is defined as $\omega_2 = \gamma e^{\alpha(x-x_p)}$ Kamal et al. (2010), where $\alpha$ and $\gamma$ are adjustable parameters. This exponential definition ensures that the closer a vehicle is to its preceding one, the larger $\omega_2$ becomes, preventing collisions in advance. If the vehicle-to-vehicle distance exceeds the desired value, the entire penalty term increases as well. In this way, the vehicles can have a suitable gap between each other for better traffic and fuel efficiency. On the other hand, if the vehicle is far from its preceding vehicle, $\omega_2$ becomes negligibly small, avoiding unnecessary acceleration.

In summary, the objective of Problem 5 is to balance tracking a reference speed while ensuring safety through maintaining a safe distance from the preceding vehicle,
minimizing fuel consumption, and providing comfortable driving. The reference speed, $v_d$, can be calculated either by solving a TPBVP with a fixed terminal time, or by dividing the distance to the merging zone by the planned arrival time. It’s important to note that $v_d$ is updated in each control cycle. In normal circumstances, the vehicle can travel at either its desired speed or the speed of its preceding vehicle, whichever is lower. To minimize fuel consumption, the fuel-optimal speed is selected as the desired speed for each vehicle in this study. However, if necessary, $v_d$ can be set to any preferred speed specifically for an individual vehicle. The fuel-optimal speed, $v^*$, is the speed at which fuel consumption per unit distance is minimized, and it can be derived as a solution of Eq. (4.16).

$$\frac{d}{dv} \left( \frac{\dot{m}}{v} \right) = -\frac{\alpha_0}{v^2} + 2\alpha_3 v + \alpha_2 = 0.$$  (4.16)

As discussed above, the goal of Problem 5 is to track a reference speed while having a safe distance with the preceding vehicle and comfort driving maneuvers. The reference speed $v_d$ is computed as the distance to the merging zone divided by the scheduled arrival time. If the vehicle is not assigned an arrival time, it can use the fuel-optimal speed or the same speed as the preceding vehicle, depending on which one is smaller. As such, tracking the reference speed would result in a fuel-efficient driving strategy.

Based on the fuel consumption model in Eq. (4.4), the optimal cruising speed $v^*$ can be derived at the minimum fuel consumption rate per unit distance. To be specific, $v^*$ can be obtained as the solution of Eq. (4.16).

## 4.6 Merging Preparation

Upon reaching the merging zone with assigned speeds, it becomes imperative to coordinate the vehicles’ speeds and inter-vehicle distances before the merging maneuver in order to guarantee a safe merging process. This can be achieved by
solving Problem 5 with an appropriate $R(t)$ and $v_d$. To minimize travel delay, a straightforward approach would be to use the maximum permitted speed as $v_d$. However, in this research, the fuel-optimal speed $v^*$ is incorporated into the formulation of $v_d$ to improve fuel efficiency as:

$$v_d = \begin{cases} 
  v^* + (v^*_f - v_d)e^{\alpha(R_0 + v^*_f t_{ud} - d_f)} & \text{if } d_f < d_0 \text{ and } v_d < v^*_d \\
  v^* & \text{otherwise}
\end{cases}$$

(4.17)

subject to $v_{\min} \leq v_d \leq v_{\max}$

where $d_f$ is the distance between ego-vehicle and its following vehicle, and $v^*_d$ denotes the reference speed of the following vehicle. The exponential weight $\alpha$ is applied to promote safe inter-vehicle distances, and $d_0$ serves as the operation range, which can be either a constant value or a variable dynamically linked to real-time speeds.

The solution of Problem 5, using the $v_d$ computed by Eq. (4.17), allows vehicles to maintain a safe inter-vehicle distance while approaching the merging speed. During the time spent in the merging zone, the vehicles will be prepared for the merging maneuver. It’s worth noting that instead of relying solely on the fuel-optimal speed ($v^*$), the reference speed can also be set to any desired value to accommodate different traffic scenarios and demands.

### 4.7 Optimal Control Method

Problem 5 is a complex nonconvex OCP that can be solved using well-known nonlinear programming (NLP) packages, such as SNOPT Gill et al. (2005) and IPOPT Wächter and Biegler (2006). However, these NLP solvers can be time-consuming and may not guarantee convergence. Additionally, they often require a good initial guess from the user, making it challenging to implement them onboard directly. Thus,
it’s necessary to reformulate the problem to ensure faster computation and practical onboard application.

Recently, pseudospectral optimal control Ross and Karpenko (2012); Fahroo and Ross (2008) and convex optimization Boyd et al. (2004); Wang and Grant (2017); Li et al. (2021a) have emerged as two new techniques that greatly improve the computational efficiency of solving optimization problems. To enable fast yet optimal solution approaches, a combination of these two techniques has been developed into a new pseudospectral convex optimization approach for the on-ramp merging control problem. In particular, a pseudospectral method is applied to discretize the original OCP and then nonconvex terms are convexified into tractable optimization formulations that can be solved for approximate optimal solutions using SCP. Finally, the algorithms are implemented under an MPC framework.

4.7.1 Pseudospectral Discretization

Recently, pseudospectral methods have been applied with success to many trajectory optimization problems and integrated into several general-purpose commercial optimal control software packages. Their advantages over traditional direct and indirect methods, such as high accuracy, fast convergence, and low sensitivity to the initial guess, have led to their widespread use. In this study, a pseudospectral optimal control method is used to discretize the original OCP from a continuous-time problem into a parameter optimization problem. The pseudospectral method is briefly summarized in this section, with more detailed information available in references Ross and Karpenko (2012); Fahroo and Ross (2008); Gong et al. (2008). It is noteworthy that, unlike many studies that directly use the GPOPS software package Patterson and Rao (2014), this study implements the pseudospectral collocation method from scratch for use in the convex optimization algorithms.

The pseudospectral discretization method approximates the continuous trajectory using \( N \)th-order weighted interpolating polynomials at orthogonal collocation points.
This approach is determined by three design elements: the domain transformation $\Gamma$, the $N$th-order collocation points $y^N$, and the weight function $W$. These elements are central to the implementation of a pseudospectral method, as discussed in Ross and Karpenko (2012).

This study employs Lagrange polynomials to approximate the state and control variables at Legendre-Gauss-Lobatto (LGL) points, which are defined within the interval $[-1, +1]$. The non-uniform, dense distribution of the LGL points near the endpoints helps prevent the Runge phenomenon in the approximation of Lagrange polynomials Garg (2011). For finite-horizon OCPs, the time domain $[t_0, t_f] \ni t$ is transformed into the computational domain $[-1, +1] \ni \tau$ via the following affine transformation:

$$
\begin{align*}
    t &= \Gamma(\tau) = \frac{t_f-t_0}{2} \tau + \frac{t_f+t_0}{2} \\
    \tau &= \Gamma^{-1}(t) = \frac{2}{t_f-t_0} - \frac{t_f+t_0}{t_f-t_0}
\end{align*}
$$

The Lagrange polynomials are used to approximate the state trajectory $X(\tau)$ on the discretized Legendre-Gauss-Lobatto (LGL) points, $\pi^N := \tau_i, i = 0, ..., N$. The approximation is expressed as follows:

$$
X(\tau) \approx X^N(\tau) = \sum_{i=0}^{N} X_i L_i(\tau), \quad -1 \leq \tau \leq 1,
$$

where $L_i(\tau)$ represents the $N$th-order Lagrange interpolating polynomial:

$$
L_i(\tau) = \prod_{j=0, j \neq i}^{N} \frac{\tau - \tau_j}{\tau_i - \tau_j}
$$

After discretizing the time domain and mapping the continuous states and control variables to a discrete points, the next step is to transcribe the dynamics of the system in Eqs. (4.6) and (4.8) into approximation equations by differentiating the interpolating polynomial as:
\[
\dot{X}(\tau_k) \approx \dot{X}^N(\tau_k) = \sum_{i=0}^{N} X_i \dot{L}_i(\tau_k) = \sum_{i=0}^{N} D_{ki} X_i \tag{4.21}
\]

where \( k = 0, 1, ..., N \), and \( D \) is the Gauss pseudospectral differentiation matrix with constant elements for particular polynomials Fahroo and Ross (2008):

\[
D_{ki} = \begin{cases} 
-\frac{N(N+1)}{4}, & k = i = 0 \\
\frac{L_N(\tau_k)}{L_N(\tau_i)(\tau_k - \tau_i)}, & k \neq i, 0 \leq k, i \leq N \\
N(N+1)/4, & k = i = N \\
0, & 1 \leq k = i \leq N
\end{cases} \tag{4.22}
\]

By combining Eqs. (4.6), (4.8) and (4.21), the state-space constraints for the collocated control problem can be expressed as:

\[
\sum_{i=0}^{N} D_{ki} X_i = \frac{t_f - t_0}{2} f(X_k, u_k, \tau_k) \tag{4.23}
\]

where \( f(X_k, u_k, \tau_k) \) denotes the dynamics function in the state-space representation of Eqs. (4.6) and (4.8).

The last step is to approximate the cost functional \( J \) using Gauss–Lobatto quadrature rule:

\[
J = \int_{t_0}^{t_f} g(X(t), u(t)) dt \approx \frac{t_f - t_0}{2} \sum_{i=0}^{N} g(X_i, u_i) w_i \tag{4.24}
\]

where \( g(X(t), u(t)) \) denotes the same function as the terms in Eq. (4.5), and \( w_i \) are the LGL integration weights given by:

\[
w_i = \frac{2}{N(N + 1) L_N^2(\tau_i)}, \quad 0 \leq i \leq N \tag{4.25}
\]
To successfully implement the pseudospectral method, it is essential to have accurate computation of the collocation points. While explicit expressions for the LGL points do not exist, they can be determined numerically through the use of the Newton method. This method finds the roots $L_N'(\tau)$ through a recursive process, with the iteration rule and initial approximation given as Shen et al. (2011):

$$
\tau_{i}^{k+1} = \tau_{i}^{k} - \frac{(1 - \tau_{i}^{2})L_{N}'(\tau_{i})}{2\tau_{i}L_{N}'(\tau_{i}) - N(N+1)L_{N}(\tau_{i})},
$$

for $k \geq 0$, $i = 1, 2, ..., N - 1$ (4.26)

$$
\tau_{i}^{0} = \frac{\sigma_{i} + \sigma_{i+1}}{2}, \ i = 1, 2, ..., N - 1
$$

(4.27)

$$
\sigma_{i}^{0} = \left[1 - \frac{N - 1}{8N^{3}} - \frac{1}{384N^{4}} \left(39 - \frac{28}{\sin^{2} \theta_{i}}\right)\right] \cos \theta_{i}
$$

$$
+ O(N^{-5}), \ i = 1, 2, ..., N
$$

(4.28)

$$
\theta_{i} = \frac{4i - 1}{4N + 2} \pi, \ i = 1, 2, ..., N
$$

(4.29)

After the root-finding iteration converges to a specified criterion, such as maximum absolute difference between subsequent iterations less than a given tolerance ($|\tau_{i}^{k+1} - \tau_{i}^{k}| < \epsilon$), the corresponding differential matrix $D$ and the weights $W$ can be calculated using the formulas in Eq. (4.22) and Eq. (4.25), respectively.

### 4.7.2 Convex Optimization

Recently, with the advancement of optimization theory and computational techniques, convex optimization has gained increasing popularity in the field of optimal control. Compared to nonlinear programming algorithms, convex optimization presents several compelling benefits. For instance, a convex optimization problem can be solved efficiently in polynomial time, as demonstrated in Boyd et al. (2004). Additionally, as long as the feasible set is non-empty, a globally optimal solution
can be obtained without the requirement of any user-provided initial guesses. These advantages make convex optimization well-suited for onboard applications.

To transform Problem 5 into a convex optimization problem, the equality constraints must be made convex. This is achieved by linearizing the nonlinear terms in Eq. (4.5) and Eq. (4.8) through first-order Taylor series expansions with respect to the solution of the previous SCP iteration. The successive linearization for the $k$-th iteration is given as follows:

$$
\dot{v}(t) \approx u(t) - \frac{\rho AC_D}{2m} (2v_{k-1} - (v_{k-1})^2) - (\sin \theta + \mu \cos \theta)g \quad (4.30)
$$

The nonlinear term $u_d$ in Eq. (4.9) is also linearized in reference to the previous iteration’s solution of the speed profile $v_{k-1}(t)$, as outlined below:

$$
u_d(t) \approx \frac{\rho AC_D}{2m} (2v_{k-1} - (v_{k-1})^2) + (\sin \theta + \mu \cos \theta)g \quad (4.31)$$

With the use of the SCP method, Problem 5 is now transformed into a convex OCP as follows:

**Problem 6.**

$$
\begin{align*}
\text{minimize} & \quad x, v, u \\
\text{subject to} & \quad \text{Eq. (4.6), Eq. (4.10), Eq. (4.11), Eq. (4.12), Eq. (4.13), Eq. (4.14), Eq. (4.30)}, \text{ and Eq. (4.31).}
\end{align*}
$$

### 4.7.3 Sequential Convex Programming Algorithms

To find an approximate optimal solution to the original optimal control problem, Problem 6 is discretized into a finite-dimensional optimization problem by approximating the continuous-time states and controls with polynomials, and then the problem can be solved by the following two SCP algorithms.
**Algorithm 4.3** Line-search SCP Algorithm

1) Initialize $k = 0$. Generate an initial trajectory, $\hat{z}^0$, by integrating the equations of motion in Eqs. (4.6) and (4.8) using a constant control, based on the vehicle’s current state of $x(t_0) = x_0$ and $v(t_0) = v_0$. Increment $k$ by 1.

2) For $k \geq 1$, parameterize Problem 6 using the previous iteration’s trajectory, $\hat{z}^{k-1}$, and solve Problem 6 under the dynamic constraint Eq. (4.23) and the cost functional integration rule Eq. (4.24) to find the updated solution, $z^k = \{x^k, v^k, u^k\}$.

3) Evaluate the convergence criteria:

$$\begin{align*}
\sup_{t_0 \leq t \leq t_f} |x^k(t) - x^{k-1}(t)| &\leq \epsilon_1 \\
\sup_{t_0 \leq t \leq t_f} |v^k(t) - v^{k-1}(t)| &\leq \epsilon_2 , \quad k > 1 
\end{align*}$$

(4.32)

where $\epsilon_1$ and $\epsilon_2$ are prescribed tolerances. If the above criteria are met, proceed to step 5; otherwise, go to step 4.

4) Calculate the search direction for the next iteration with $p^k = z^k - \hat{z}^{k-1}$. Find the step length, $\alpha^k$, by reducing $\alpha^0 = 1$ using a contraction factor, $c_1$, such that $\alpha^k = c_1 \alpha^{k-1}$, until the cost functional, $J$, demonstrates a sufficient decrease as described below:

$$J(z^k) + (1 - c_2) \alpha^k \nabla J_k^T p^k \leq J(z^k + \alpha^k p^k) \leq J(z^k) + c_2 \alpha^k \nabla J_k^T p^k$$

subject to $0 < c_2 < 1/2$ (4.33)

(4.34)

Update the reference trajectory with $\hat{z}^k = \hat{z}^{k-1} + \alpha^k p^k$, increment $k$ by 1, and go back to step 2.

5) The procedure is completed and the optimal solution is found to be $z^* = \{x^*, v^*, u^*\} = \{x^k, v^k, u^k\}$.
In general, Algorithm 4.3 utilizes a line-search method and leverages the solution from the previous iteration to formulate a feasible search direction. It starts by generating a feasible search direction and then determining the appropriate step length along that direction in subsequent iterations for stable convergence. The Goldstein conditions (Nocedal and Wright, 2006, p. 36), as stated in Eq. (4.33), are characterized by the constant $c_2$. These inequalities ensure that the step length $\alpha^k$ obtains an adequate reduction in the cost function, without being overly restrictive.

Another SCP algorithm designed to solve the CV merging control problem employs the trust-region method with the aim of improving the convergence of the SCP algorithm, similar to the line-search method. The trust-region method starts by defining a trustworthy region around the initial guess for the solution. An approximate solution within this region is then obtained as a step. If the step fails to meet the convergence criteria, the accuracy of the approximate solution is evaluated. If it results in a sufficient decrease in the cost function, the size of the trust-region can be increased or kept unchanged. If not, another SCP iteration is carried out with a smaller trust-region. The detailed trust-region SCP algorithm is described as illustrated in Algorithm 4.4.

In the definition of model ratio $\nu^k$, $J(z^k)$ represents the original cost functional as defined in Eq. (4.5) with nonlinear dynamics, while $J'(z^k)$ is the parameterized cost functional with linearized dynamics. The model ratio measures the quality of the approximate solution $z^k$ by comparing the actual decrease in the cost functional $(J(\hat{z}^{k-1}) - J(z^k))$ to the predicted decrease $(J(\hat{z}^{k-1}) - J'(z^k))$.

If $\nu^k > 0.75$, it indicates that the solution $z^k$ is of good quality and the trust-region radius $\delta$ can be increased by multiplying it with the constant $\beta_2$. Conversely, if $\nu^k < 0.25$, it means that $z^k$ is not an acceptable solution and the trust-region radius $\delta$ needs to be decreased by multiplying it with the constant $\beta_1$. If $\nu^k$ is between 0.25 and 0.75, $\delta$ remains unchanged.

If $z^k$ is of sufficient quality, meaning that $\nu^k > \eta$, the reference trajectory $\hat{z}^k$ will be updated to $z^k$, $k$ will be incremented to $k + 1$, and the algorithm will proceed to
Algorithm 4.4 Trust-region SCP Algorithm

1) Set \( k = 0 \). Generate an initial trajectory \( \hat{z}^{0} \) by applying the equations of motion in Eqs. (4.6) and (4.8) with a constant control input based on the current state of the vehicle, \( x(t_0) = x_0 \) and \( v(t_0) = v_0 \). Select appropriate values for the trust-region size \( \delta = \delta_0 \), the constants \( 0 < \eta < \beta_1 < 1 < \beta_2 \). Increment \( k \) by 1.

2) For \( k \geq 1 \), formulate Problem 6 using \( \hat{z}^{k-1} \). Find a solution \( z^k = \{ x^k, v^k, u^k \} \) for Problem 6 subject to the trust-region constraint in Eq. (4.35), the dynamic constraint in Eq. (4.23), and the cost functional integration rule in Eq. (4.24).

\[
|z^k - \hat{z}^{k-1}| \leq \delta \quad (4.35)
\]

3) Check the convergence criteria defined as follows:

\[
\begin{align*}
\sup_{t_0 \leq t \leq t_f} |x^k(t) - x^{k-1}(t)| & \leq \epsilon_1 \\
\sup_{t_0 \leq t \leq t_f} |v^k(t) - v^{k-1}(t)| & \leq \epsilon_2, \quad k > 1
\end{align*} \quad (4.36)
\]

where \( \epsilon_1 \) and \( \epsilon_2 \) are prescribed tolerances. If the criteria are met, proceed to step 5. Otherwise, move on to step 4.

4) Calculate the model ratio \( \nu^k \) using \( z^k \) and \( \hat{z}^{k-1} \) as follows:

\[
\nu^k = \frac{J(\hat{z}^{k-1}) - J(z^k)}{J(\hat{z}^{k-1}) - J'(z^k)} \quad (4.37)
\]

5) The algorithm terminates and the optimal solution is found as \( z^* = \{ x^*, v^*, u^* \} = \{ x^k, v^k, u^k \} \).
step 2. On the other hand, if $\nu \leq \eta$, the solution $z^k$ will be discarded and step 2 will be repeated with a smaller trust-region size.

In summary, the Trust-region SCP algorithm iteratively updates the solution by solving the problem within a trust-region constraint, adjusting the trust-region size based on the quality of the solution, and using the best solution so far as the guess for the next iteration. The algorithm stops when the solution satisfies the convergence criteria.

The trust-region constraint (as described in Eq. (4.35)) is introduced in step 3 of Algorithm 4.4 in order to improve the convergence of the SCP method. In step 4, Algorithm 4.4 finds an appropriate trust-region radius $\delta$ for the next iteration, which is crucial for the convergence rate. If $\delta$ is too small, the algorithm may require more iterations to find the optimal solution, and if $\delta$ is too large, it may not be able to find a step that is close enough to the optimal solution. Algorithm 4.4 reduces computational expense compared to Algorithm 4.3 as it doesn’t require computing the derivatives of $J$, but in certain scenarios with strong artificial infeasibility or large model errors, it may take more iterations to converge (as stated in Wang and Lu (2020); Wang and McDonald (2020)). For more information on trust-region methods, see (Nocedal and Wright, 2006, p. 66).

4.7.4 Model Predictive Control

The proposed SCP algorithms have been implemented within the framework of Model Predictive Control (MPC) to enable their potential application in real-world driving conditions. This framework has been made possible due to recent advancements in computing hardware and numerical algorithms that can handle nonlinear optimization in real-time. MPC, as described in detail in Camacho and Alba (2013), solves the optimal control problem over a fixed time horizon, but only implements a portion of the control solution at a time, re-solving the problem with updated system information at each iteration. The MPC law for the solution of Problem 6, utilizing
pseudospectral discretization, is expressed as \( u(t) = u_0^* \), and the subsequent vehicle state is estimated through the integration of the original dynamics using the Runge-Kutta method Dormand and Prince (1980).

To sum up, the proposed real-time speed control framework is outlined in the flowchart shown in Fig. 4.3. It consists of three main components: 1) Pseudospectral collocation that improves the optimization algorithms’ convergence and accuracy, 2) Convex optimization that reduces the computational complexity for real-time control, and 3) MPC that enables online control and provides robust response to disturbances and uncertainties from the environment. At each time step, vehicles solve their OCPs to generate the predicted trajectory and control commands, which are then shared with other vehicles through V2V communication.

### 4.8 Simulation Results

In this section, a series of simulation results are presented to validate the performance of our proposed on-ramp merging control method. The optimization algorithms were implemented using the YALMIP environment on the MATLAB platform. The SCP steps utilized the Gurobi Gurobi Optimization (2021) convex optimization solver, while IPOPT Biegler and Zavala (2009) was employed as the nonlinear optimization solver for comparison.

In this study, it is assumed that the slopes of both the mainline and ramp roads are constant and equal to 0%. The vehicles in the simulation are considered to have identical parameters, as listed in Table 4.1. The fuel-optimal speed was determined by solving Eq. (4.16) and was found to be \( v^* = 13.46 \text{ m/s} \). The maximum speed limit for both roads was set to \( v^{\text{max}} = 30 \text{ m/s} \) and the minimum allowable speed was set to \( v^{\text{min}} = 0 \text{ m/s} \).
Figure 4.3: Flowchart of convex optimization-based real-time speed control.
4.8.1 Convergence Analysis

While a formal proof for the convergence of the SCP method is yet to be established, numerical simulations have provided convincing evidence for its convergence. The accuracy and validity of the SCP method’s converged solution can be confirmed through numerical evaluation by applying its control variables to the dynamic functions, and then comparing the resulting trajectory with the one generated by the SCP method.

In the simulation experiments, the proposed algorithms are firstly evaluated by considering a test problem involving a single vehicle driving 400 m to merge at $t = 20$ s, showcasing the optimality and convergence of the algorithms. The nonlinear optimization solver IPOPT was used to obtain a baseline solution, and the resulting time-space trajectory was uniformly discretized with a very small time step of 1ms. As shown in Fig. 4.4, the blue and green lines with markers represent the trajectories generated by Algorithm 4.3 and Algorithm 4.4, respectively, while the red lines represent the baseline solution from IPOPT. The comparison reveals that all three profiles are in good agreement, confirming the accuracy and optimality of the solutions from Algorithms 2 and 3. However, minor differences in the control profiles may be attributed to the different linear approximations of the dynamics, convergence criteria, and discretization strategies used by each algorithm. The absolute errors of velocity and position in the final state were $3.20e^{-5}$ m/s and $1.27e^{-2}$ m, respectively.

The convergence of Algorithm 4.3 and 3 is shown in Fig. 4.5. Both SCP algorithms converged in just 3 iterations, with convergence tolerances set at $\epsilon_1 = \epsilon_2 = 1e^{-6}$. The convergence and results of both algorithms are similar, indicating little difference between using trust-region and line-search for this scenario. It’s worth mentioning that in this example, the initial trajectory was set as a constant zero control, causing the solution of the first iteration to deviate significantly from the initial guess. However, within an MPC framework, the initial trajectory can be taken from the
Figure 4.4: Comparison of optimal control profiles.

Figure 4.5: Convergence of vehicle states and objective functions.
solution of the previous time step, allowing the SCP algorithms to typically converge within two iterations.

In terms of computational cost, the convex optimization approach requires significantly less time compared to the nonlinear optimization solver. For instance, the baseline trajectory using IPOPT took around 127 seconds to generate on a MacBook Pro with a 64-bit Mac OS and an Intel Core i7 2.2 GHz processor. On the other hand, the solution of Problem 6 took only about 50 milliseconds for each iteration when solved using the Gurobi solver. Additionally, the computation time for the SCP algorithms can be reduced by increasing the convergence tolerances or decreasing the number of discretization nodes. The computational efficiency could further be improved by implementing the SCP algorithms in a compiled programming environment or using a more powerful computation device. Most importantly, convex optimization solvers provide guaranteed convergence within a finite number of iterations without the need for initial guesses, as long as the problem is feasible.

### 4.8.2 Case Study: Coordination of 40 Vehicles

To visually illustrate the merging process, two simulation scenarios (balanced and unbalanced traffic flow) featuring multiple vehicles were conducted using the proposed SCP algorithms under varying vehicle arrival rates. A rolling time horizon of 10 seconds was chosen. Each vehicle entered the simulation randomly, starting from position 0, with a probability based on the traffic volume setting. If a preceding vehicle was not within 60 meters, the vehicle’s initial speed ranged from 10 to 20 m/s, otherwise, its initial speed was randomly selected from a standard normal distribution centered around the speed of its preceding vehicle, with a standard deviation of 10% to mitigate collision risk.

1) Balanced Traffic Conditions: The merging process under different traffic conditions is illustrated in Fig. 4.6a, Fig. 4.6b, Fig. 4.6c, and Fig. 4.6d. In a light traffic condition with 360 vehicles per hour in each leg (Fig. 4.6a), the on-ramp vehicles
Figure 4.6: Vehicles trajectories of case study with balanced traffic in control zone.

(a) 360 vehicles per hour on each leg
(b) 720 vehicles per hour on each leg
(c) 900 vehicles per hour on each leg
(d) 1,080 vehicles per hour on each leg
seamlessly merge onto the mainline road without affecting the existing vehicles. However, under moderate traffic (720 vehicles per hour in each leg, Fig. 4.6b), the ramp vehicles must travel faster to maintain traffic flow. In a saturated traffic condition (900 vehicles per hour in each leg, Fig. 4.6c), some mainline vehicles adjust their speeds to accommodate merging vehicles, resulting in cooperative merging to increase road capacity. In an over-saturated traffic condition (1,080 vehicles per hour in each leg, Fig. 4.6d), traffic congestion occurs, and the merging order alternates similar to a FIFO strategy, The priority of the vehicles on the mainline road is evident as expected. The vehicle trajectories entering the merging zone for all cases are shown in Fig. 4.7. It can be observed that, after a period of time, all vehicles maintain stable and safe inter-vehicle distances, making them ready for merging.

2) Unbalanced traffic conditions: The performance of the proposed method is evaluated by studying two opposite traffic scenarios. As seen in Fig. 4.8a, the mainline road experiences a heavy traffic volume of 1,080 vehicles per hour while the ramp road experiences a lighter volume of 540 vehicles per hour. This results in difficulties for merging vehicles to find sufficient gaps without adjustments. In contrast, in the scenario shown in Fig. 4.8b where the traffic volumes are reversed, the majority of the mainline vehicles are able to travel at their fuel-optimal speed while the ramp vehicles actively maneuver into available gaps between them. The vehicle trajectories in the merging zone, shown in Fig. 4.9, also demonstrate the successful coordination of merging speeds and distances by the proposed method under unbalanced traffic conditions.

4.8.3 Case Study: Continuous Simulation

The results above show that the proposed method is effective in guiding the vehicles to merge cooperatively with adequate safety distances. To quantify the performance of the proposed method, average vehicle speed and fuel consumption were compared to those generated by the FIFO merging strategy using Algorithms 2 and 3.
Figure 4.7: Vehicles trajectories of case study with balanced traffic in both control and merging zone.
Figure 4.8: Vehicles trajectories of case study with unbalanced traffic in control zone.

(a) 360 (veh/h) on ramp road and 720 (veh/h) on mainline road

(b) 720 (veh/h) on ramp road and 360 (veh/h) on mainline road
Figure 4.9: Vehicles trajectories of case study with unbalanced traffic in both control and merging zone.

(a) 540 (veh/h) on ramp road and 1,080 (veh/h) on mainline road

(b) 1,080 (veh/h) on ramp road and 540 (veh/h) on mainline road
Additionally, to verify the fuel efficiency of the proposed method, it was compared with the vehicle trajectories generated by SUMO (Simulation of Urban Mobility) using the Krauss model Lopez et al. (2018); Krauß (1998). This model is designed to drive as quickly as possible while ensuring optimal safety. The comparison is presented in Fig. 4.10.

It is worth noting that each box plot in Fig. 4.10 displays the results of continuous simulations conducted over 3,600 seconds. To optimize computation time, 26 discretization nodes were utilized in the pseudospectral method, as opposed to the example in Fig. 4.4. Thus, the control cycle for MPC used a time step of 0.056 seconds, requiring a solution of the OCP to be computed 532 times for a vehicle traveling 30 seconds. Each solution took between 50 milliseconds and 150 milliseconds to run on a CPU.

The results in Fig. 4.10a and Fig. 4.10b show that the proposed method with a rule-based merging strategy results in lower average vehicle speed, owing to its focus on energy efficiency. However, compared to the SUMO simulation, the proposed vehicle motion control algorithm with the FIFO strategy has a higher average vehicle speed in the first three traffic conditions, leading to reduced traffic delays. In the case of saturated traffic volume (900 veh/h), the proposed method results in a lower average speed compared to the SUMO simulation, which may be attributed to its tighter safety constraints and larger headway requirements. The results also show that the proposed rule-based merging strategy is able to adapt to different traffic volumes, with faster speeds required from on-ramp vehicles in high traffic conditions while main-line vehicles maintain lower speeds for fuel efficiency.

Additionally, as illustrated in Fig. 4.10c and Fig. 4.10d, the average fuel consumption rate of the proposed vehicle merging control algorithm with the FIFO strategy is comparable to that of the SUMO simulations for the first three different cases of traffic volumes. This indicates that the former has a more efficient fuel consumption compared to the latter, considering its higher average vehicle speed.
Figure 4.10: Performance comparison with baselines.
The comparisons also confirm that the proposed rule-based merging strategy is fuel-
economical, as anticipated.

4.9 Conclusion

In this Chapter, a real-time and efficient merging control strategy is proposed for
on-ramp merging using a CV-based optimal speed control approach. A set of
cooperative rules have been proposed to ensure the safe and efficient merging of
vehicles on the on-ramp road and the mainline road. These rules prioritize safety
while taking into account different control goals, such as minimizing deceleration and
avoiding congestion. By utilizing sequential convex programming and pseudospectral
discretization methods, the proposed line-search and trust-region SCP algorithms
achieve high computational efficiency and stable convergence, making them suitable
for onboard applications. To ensure safe and coordinated merging, the MPC
framework is used to continuously update vehicle maneuvers in response to dynamic
traffic situations. The proposed algorithms’ optimality and stability were confirmed
through comparison with solutions from the nonlinear solver IPOPT. The results
show that the proposed method is effective in handling safety constraints under
dynamic traffic environment with significant improvements in traffic mobility and
fuel efficiency.
Chapter 5

Study III: A Novel Deep Reinforcement Learning Approach to Traffic Signal Control with Connected Vehicles
5.1 Abstract

Real-time traffic data from CVs can help optimize traffic signals to reduce congestion, increase fuel efficiency, and enhance road safety. The success of CV-based signal control relies on an accurate and computationally efficient model that accounts for the stochastic and nonlinear nature of traffic flow. Reinforcement Learning (RL) provides a promising solution for acquiring control policies without the need for prior knowledge of the traffic system’s model architecture. This study presents a novel data-driven traffic signal control method that leverages the latest in deep learning and reinforcement learning techniques. By incorporating a compressed representation of traffic states, the proposed method overcomes limitations in defining the action space and allows for more practical and flexible signal phases. Simulation results show the convergence and robust performance of the proposed method compared to existing benchmark methods in terms of average vehicle speed, queue length, wait time, and traffic density.

5.2 Introduction

The CV technologies creates a dynamic and interconnected environment for drivers, vehicles, and traffic infrastructure. Through wireless communication, vehicles can communicate real-time data such as location, speed, and acceleration with both other vehicles (V2V) and infrastructure (V2I). This real-time data enables the optimization of signal phase and timing (SPaT) plans for enhanced road safety and sustainability. However, the complexity of network-level SPaT optimization, taking into account realistic driving behavior and multiple objectives, presents a significant challenge and remains an open research area due to the NP-complete nature of the problem Wünsch (2008); Al Islam and Hajbabaie (2017), and the “curse of dimensionality” associated with an increasing number of vehicles and traffic lights in the network.
With the large amounts of real-time data generated by CVs, it is possible to understand the interactions between vehicles and traffic infrastructure components, thus enabling the development of data-driven traffic control strategies. Non-parametric learning approaches, particularly reinforcement learning (RL), are well suited for characterizing the stochastic and non-linear nature of traffic flow. These techniques allow the signal controller to learn policies by observing traffic state transitions, without the need for a prior knowledge of the system’s model structure Sutton and Barto (2018). In other words, RL-based signal control approaches eliminate the need to construct complex decision-making models for highly dynamic, nonlinear, and stochastic traffic systems.

However, constructing and training a learning-based controller directly from raw data presents a major challenge. Without proper design of the learning models and algorithms, the controller may not be able to effectively learn a control strategy. To tackle the issue of “curse of dimensionality”, many researchers have had to simplify the training model by restricting the action and state space, which reduces the realism and optimality of the resulting controllers. This article aims to overcome these limitations by defining the action space in a way that allows for more practical and versatile signal timings, and by restructuring the state space to enhance the learning performance.

The main contributions of the work in this Chapter are as follows: 1) A new traffic signal control framework using deep reinforcement learning (DRL) is proposed, by incorporating a novel convolutional autoencoder network to reduce the dimensionality of the input traffic states. This results in a condensed representation of comprehensive traffic information that can aid in the acquisition of effective SPaT plans. 2) The proposed framework extends the action space by incorporating both phase duration and cycle length, allowing for increased adaptability to dynamic traffic flow. With the combinatorial action space of multiple phase duration and cycle lengths, this method can effectively handle unbalanced traffic flow with varying traffic volumes. 3) To improve the learning efficiency of the proposed DRL algorithm, several state-of-the-art techniques such as target network Van Hasselt et al. (2016), dueling network
Wang et al. (2015), and experience replay Schaul et al. (2015) are implemented. Through simulations on the widely used Simulation of Urban MOBility (SUMO) traffic simulator, the superiority of the proposed method are demonstrated by comparing to several existing traffic signal control methods.

5.3 Methodology

The deep reinforcement learning (DRL) approach has gained attention for its capability in learning high-level decision making processes, as demonstrated in Vinyals et al. (2019). As a data-efficient method, it enables learning of decision-making by agents through interactions with the environment. A typical DRL-based control framework consists of three components: environment sensing, the action space of the agent, and a learning goal for the agent Sutton and Barto (2018). As demonstrated in previous studies, data-driven methods applied to traffic signal control problems outperform conventional methods in simulations involving large amounts of data.

The objective of this study is to enhance the overall performance of data-driven signal control frameworks by enhancing training outcomes and yielding more adaptable SPaT results. As highlighted in the Chapter 2, the current DRL-based approaches face various challenges in terms of learning efficiency and optimality. In particular, the learning structure needs to be revised to accommodate dynamic traffic conditions and improve learning efficiency. With this in mind, this study presents a novel data-driven optimization framework for traffic signal control that leverages innovative DRL techniques. The structure of the proposed method is depicted in Fig. 5.1.

5.3.1 Scenario and Simulation Environment

In this study, the problem of traffic signal control at a typical four-way signalized intersection is considered. The intersection has one left-turn lane, one through
Figure 5.1: Proposed data-driven optimization framework for traffic signal control.
lane, and one right-turn lane in each direction, as shown in Fig. 5.2. The isolated intersection and traffic are simulated by SUMO Lopez et al. (2018), a microscopic, space-continuous traffic simulation software, which allows to retrieve details of simulated objects and to adjust their parameters at every time step. The traffic signal at the intersection is managed by a DRL-based actor, also referred to an agent. This agent continually receives traffic state information and a reward signal from the simulated environment and makes decisions based on the current traffic state.

**Traffic States**: The traffic state is represented by discrete encoding of position and speed information of vehicles around the intersection Genders and Razavi (2016). The simulated intersection is divided into squared mesh grids with equal length $c$, which can be represented by an $N \times N$ matrix. Each grid in the matrix has two values: one binary value that indicates the presence of a vehicle, and another that stores the speed of the existing vehicle. An example of a 30 x 30 traffic state matrix is shown in Fig. 5.3, where the yellow grids denote vehicles and the numbers show their speeds in $m/s$. Blank grids indicate the absence of vehicles at those positions. In real world implementations, vehicle mobility information can be obtained through a vehicular network or other devices. Jeong et al. (2021).

**Action Space**: This research focuses on two main types of discrete action spaces for RL-based traffic signal control. The first option involves selecting a signal phase from a fixed set of choices at predetermined time intervals, with the duration of each phase limited to a multiple of the time interval. The second option involves fixing the phase sequence of a signal cycle and adjusting the duration of each phase in the subsequent cycle at the end of the current cycle. As depicted in Fig. 5.4, a typical four-phase signal cycle is considered comprised of two straight and two left-turn phases. To discretize the action space, a combination approach is used to choose the signal cycle length and phase splits.

The selection of the signal cycle length plays a crucial role in handling traffic volumes. A longer cycle length increases road capacity and prevents loss of green time that could occur with delayed response to the green light Transportation.
Figure 5.2: The simulated intersection.

Figure 5.3: An example of traffic state matrix.
Figure 5.4: A typical signal cycle with four phases.
Research Board and National Academies of Sciences, Engineering, and Medicine (2015). However, excessively long cycle lengths can result in increased congestion and long waiting queues. There is a trade-off between road capacity and traffic delay to consider when selecting the cycle length. To balance this, the selection of cycle length is limited to 10s, 20s, 30s, 40s, 50s, and 60s to prevent extremely long cycle lengths from slowing down traffic flow. The available selections for phase duration range from 0 to the maximum cycle length, in increments of 5 s. This results in a total of 1,035 possible actions. Additionally, during the last 3 s of each phase, the green lights turn yellow.

5.3.2 Compressed Representation of Traffic States

It is convenient to use the position and speed information of vehicles to build a traffic state matrix for input to the DRL training algorithm. However, the large space of the resulting traffic states makes it difficult for the DRL algorithm to identify a direct relationship between the traffic state and the signal control action. To address this issue, finding an appropriate traffic state representation method is crucial. In this study, an autoencoder neural network is used to represent the complex traffic states of the whole intersection as a concise representation Baldi (2012). Having a state representation that contains rich information is vital for the control agent to make informed decisions, without being affected by the “curse of dimensionality”. To achieve this, the dimension of the state representation must be reduced while preserving as much information as possible. Additionally, the compressed state representation allows the underlying traffic pattern to be extracted as features by the autoencoder. Although it is challenging to know the exact features learned by the autoencoder, feature extraction has been shown to be a useful approach in improving reinforcement learning in various applications Hakenes and Glasmachers (2019).

Autoencoder is a type of neural network that learns a compact representation of the input data Goodfellow et al. (2016). Convolutional neural networks, such as
Visual Geometry Group (VGG) neural networks Simonyan and Zisserman (2014), can extract features from the spatial information in the input data. By organizing a convolutional neural network into an encoder-decoder architecture, the network can encode a static traffic state into a fixed-length vector, serving as input to the reinforcement learning model.

The proposed Convolutional Autoencoder (CAE) network, shown in Fig. 5.5, has a mirror structure of two components: the encoder and decoder. The objective of CAE is to reconstruct the original input to its output through a bottleneck layer $h$. As illustrated in the figure, the input and output of the CAE are traffic state matrices with a shape of $64 \times 64 \times 2$. The encoder part consists of two pairs of convolution-pooling layers followed by two fully-connected layers, while the decoder is the reverse of the encoder structure. The notation numbers define the dimensions of the outputs at each respective layer. The size of the hidden layer $h$ can be determined through training experiments. Upon completion of training, the encoder network will act as the state representation compressor and generate the input vector for the DRL neural network.

The proposed CAE is implemented and trained using Tensorflow Abadi et al. (2015). The optimization process employs the Adam algorithm Kingma and Ba (2014) with Mean Squared Error (MSE) as the cost function. The hyper-parameters, such as the number of filters and neurons in the CAE, are determined through cross-validation training experiments. The size of the reduced representation vector of the traffic states is chosen to be 8, resulting in a compression ratio of 1,024 to 1. Moreover, the input state matrix is normalized by scaling the vehicle speed to the range of 0 to 1 based on the maximum allowable speed of the road.

The proposed method is evaluated using a validation set, which constitutes 10% of the available dataset. The simulation of the entire traffic signal control system is carried out in SUMO, and the training dataset is generated by continuously running simulations in SUMO. Vehicles are randomly initialized with a specified flow rate. An example training session is shown in Fig. 5.6 using a dataset of one million samples.
Figure 5.5: The structure of proposed Convolutional AutoEncoder (CAE) for traffic state representation.

Figure 5.6: Training history of the proposed convolutional autoencoder.
The minimum reconstruction errors in this example are $5.07 \times 10^{-4}$ for the training error and $6.54 \times 10^{-4}$ for the validation error. It’s important to note that the purpose of using CAE is not only to reduce the dimension of the traffic state but also to extract inherent features within the traffic information. Hence, the effectiveness of CAE should not be solely judged by its ability to perfectly reconstruct the input sample but validated through RL control experiments. An increase in the size of the hidden layer $H$ may lower the reconstruction loss, but a large input size for the RL algorithm may have negative consequences.

### 5.3.3 Deep Reinforcement Learning Structure

The well-trained CAE is then combined with the DRL algorithm to form the final model. The overall structure of the model is shown in Fig. 5.7. The CAE’s encoder network generates a compressed representation of traffic states, which is then fed into the fully connected neural network to approximate the Q-value function as described in Sutton and Barto (2018). At the end of each control cycle, the neural network computes the Q-values for all available actions for the given traffic state representation. The agent then selects the action with the highest Q-value, which is expected to result in the maximum reward. After executing the selected action, a new control cycle begins and the agent continues to learn how to maximize rewards through interaction with the environment. To improve learning efficiency and reduce possible overestimations, the model incorporates techniques such as the target network Van Hasselt et al. (2016), dueling network Wang et al. (2015), and prioritized experience replay Schaul et al. (2015). The proposed DRL training algorithm is detailed in Algorithm 5.1.

**Rewards Signal**: The design of the reward signal is critical to the success of the RL-based traffic signal control. The reward signal serves as a guide for the agent to learn the objective of its control actions, which is to enhance the intersection’s throughput and minimize vehicle waiting time. The reward provides feedback to the
Figure 5.7: Proposed deep reinforcement learning model.
agent, evaluating its past actions, and it’s important to note that unlike other RL applications, the traffic signal control problem has no terminal state. This means that the reward signal must reflect the performance of each action taken by the agent, as there is no terminal reward to learn from.

**Algorithm 5.1** Pseudo-code for training algorithm of DRL-based agent

**Input:** mini batch size B, pre-train step $t_p$, training episode length $N$, learning rate $\alpha$, greedy $\epsilon$, discount factor $\gamma$, target network update rate $\tau$, target network update frequency K

Initialize primary network $Q_{\theta}$, target network $Q_{\theta^-}$, replay memory D with capacity M

**Loop** for each episode:
- Initialize simulator environment
- Initialize time step $t = 0$
- Observe current state $S_t$
  **while** time step $t < N$:
  - With probability $\epsilon$ select action $A_t$ randomly
  - otherwise select $A_t \leftarrow \text{argmax}_{a} Q_{\theta}(S_t, a)$
  - Execute action then observe next state $S_{t+1}$ and reward $R_t$
  - Store $(S_t, A_t, R_t, S_{t+1})$ in replay memory D
  - $S_t \leftarrow S_{t+1}$
  - if current step $t \neq$ pre-training step $t_p$:
    - Sample a minibatch of B experience tuples $(S_t, A_t, R_t, S_{t+1})$ from D
    - Compute target Q values for each experience:
      $$Q^*(S_t, A_t) \approx R_t + \gamma Q_{\theta^-}(S_{t+1}, \text{argmax}_{a'} Q_{\theta}(S_{t+1}, a'))$$
    - Perform a gradient descent step with loss:
      $$\frac{1}{B} \| Q^*(S_t, A_t) - Q_{\theta}(S_t, A_t) \|^2$$
    - Update target network $\theta^-$ every K steps:
      $$\theta^- \leftarrow \tau \theta + (1 - \tau) \theta^-$$
    - $t \leftarrow t + 1$

The selection of a suitable performance index is crucial in RL-based signal control, as it guides the agent in learning the desired control objectives. Commonly used performance indices in the field of traffic signal control are travel delay, queue length, and average vehicle speed (as seen in Guo et al. (2019); Wang et al. (2022); Hong et al. (2022)). Some approaches focus on reducing road congestion and only use average
waiting time or queue length as the reward signal. However, this could result in the agent learning a strategy that frequently changes the signal, leading to shorter queue times but lower speeds. On the other hand, using a shorter cycle length reduces average vehicle speed and increases fuel consumption.

A better approach is to use average vehicle speed as the reward signal. This allows the agent to improve overall traffic mobility and reduce average travel delay, while also promoting fuel efficiency, which is largely influenced by vehicle speed and idle time. Therefore, the average vehicle speed ($\bar{V}$) of the entire intersection is used, calculated at the end of each control cycle, as the reward signal ($R$) defined below:

$$R = \bar{V} = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} v_i,$$  \hspace{1cm} (5.1)

where $v_i$ is the velocity of vehicle $i$, $T$ is the length of the current signal cycle, $N$ is the total number of vehicles in the control zone, and $t$ is time step of simulation.

### 5.4 Simulation Results

In this section, the effectiveness of the proposed methodology is examined through simulated experiments.

#### 5.4.1 Simulation Parameters

The simulation takes place in a SUMO environment where a $320 \times 320$ intersection has been established (as shown in Fig. 5.2). The parameters of the simulated intersection and vehicles are listed in detail in Table 5.1. The vehicles are randomly generated with a 10% probability per second. The Krauss car-following model Krauß (1998) is employed to ensure that vehicles move as fast as possible while maintaining perfect safety requirements. The simulation assumes a 100% CV penetration rate. Further evaluation of other penetration levels will be conducted in future work.
Table 5.1: Adopted parameters of simulation environment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lane length</td>
<td>160 meters</td>
</tr>
<tr>
<td>Vehicle length</td>
<td>5 meters</td>
</tr>
<tr>
<td>Time step</td>
<td>1 second</td>
</tr>
<tr>
<td>Maximum vehicle speed</td>
<td>20 (m/s)</td>
</tr>
<tr>
<td>Maximum vehicle acceleration</td>
<td>3 (m/s²)</td>
</tr>
<tr>
<td>Maximum vehicle deceleration</td>
<td>4.5 (m/s²)</td>
</tr>
<tr>
<td>Minimum gap between vehicles</td>
<td>2 meters</td>
</tr>
<tr>
<td>Car following model</td>
<td>Krauss Following Model</td>
</tr>
<tr>
<td></td>
<td>Krauß (1998)</td>
</tr>
<tr>
<td>Duration of yellow phase</td>
<td>3 seconds</td>
</tr>
<tr>
<td>Traffic volume</td>
<td>480 vehicles per lane and per hour</td>
</tr>
<tr>
<td>Left turning vehicles ratio</td>
<td>25% of total</td>
</tr>
<tr>
<td>Right turning vehicles ratio</td>
<td>25% of total</td>
</tr>
</tbody>
</table>
5.4.2 Hyper-parameter of Deep Reinforcement Learning Network

The implementation of the DRL network was carried out using Tensorflow Abadi et al. (2015) and integrated with the SUMO simulation environment through a Python interface. The training was conducted in episodes, with each episode consisting of 3,600 time steps of 1 second each, totaling one hour per episode. The random seed for the vehicle simulation was changed in every episode. The critical hyperparameters are listed in Table 5.2, with the values determined through a process of trial and error.

5.4.3 Convergence of DRL-based Signal Controller Training

The convergence of the proposed DRL training algorithm is demonstrated by evaluating the accumulated rewards for each episode. As shown in Fig. 5.8, the rewards increase rapidly at first and then level off as the training progresses. The average vehicle speed and average waiting time in each episode are also plotted to show the improvement and convergence of traffic measurements. As previously noted, the average waiting time is not part of the optimization objective due to its conflicting relationship with the average vehicle speed in the signal control policy. As a result, the average waiting time increases slightly at the end of the training process.

5.4.4 Comparison with Baselines

The proposed method is compared with existing methods by implementing the DRL-based traffic signal controller in Liang et al. (2019) under the same simulation conditions. The DRL training algorithm used is similar to the one outlined in Algorithm 5.1. It is important to note that the reward signal in Liang et al. (2019) is the average waiting time and their signal control strategy involves adding or subtracting 5 seconds from the duration of one of the current phases. As shown
Table 5.2: Hyper-parameters of deep reinforcement learning network.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulated time steps for each episode</td>
<td>3,600</td>
</tr>
<tr>
<td>Replay memory size</td>
<td>20,000</td>
</tr>
<tr>
<td>Minibatch size</td>
<td>64</td>
</tr>
<tr>
<td>Pre-train steps</td>
<td>2,000</td>
</tr>
<tr>
<td>Target network update interval</td>
<td>64 control cycles</td>
</tr>
<tr>
<td>Target network update rate</td>
<td>0.001</td>
</tr>
<tr>
<td>Discount factor</td>
<td>0.99</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam Kingma and Ba (2014)</td>
</tr>
<tr>
<td>Learning rate</td>
<td>$1e^{-4}$</td>
</tr>
<tr>
<td>Initial probability of exploration</td>
<td>1</td>
</tr>
<tr>
<td>Final probability of exploration</td>
<td>0.01</td>
</tr>
<tr>
<td>Ending step for exploration probability</td>
<td>40,000</td>
</tr>
</tbody>
</table>

Figure 5.8: Convergence of the proposed DRL network.
in Fig. 5.9, the performance of both average waiting time and average vehicle speed improve as the training process progresses, but its convergence is slower and more fluctuated compared to our proposed method in this study.

In addition to the proposed DRL-based traffic signal controller, two baseline methods have been implemented for comparison. These include a fixed-timing signal controller and an actuated signal controller. To determine the best fixed-timing strategy, various combinations of cycle lengths and phase duration were explored and the one with the best performance was selected. The selected cycle length is 60 seconds, and the four phase duration are 20 seconds, 10 seconds, 20 seconds, and 10 seconds, respectively. The actuated signal controller operates on a time-gap basis, allowing the green phase to be extended if there is a continuous flow of traffic. When the time gap between successive vehicles meets a predefined criterion, the signal switches to the next phase. Actuated controllers are known to perform better than fixed-timing controllers in dynamic traffic conditions, but in this study, the traffic flow was steady and thus the actuated signal controller performed similarly to the fixed-timing controller. The maximum phase duration was set to match the fixed-timing controller, while the minimum phase duration was set at 5 seconds.

To evaluate the performance of the proposed DRL-based control method, five common traffic mobility metrics are selected: 1) Average vehicle speed, which reflects the overall mobility of the intersection, both in the present moment and over a certain period of time, and represents the average travel time of all vehicles to complete the trip when computed over an episode. 2) Average waiting time, which is calculated by dividing the total waiting time of vehicles by the number of vehicles present at each time step, providing an insight into travel delays from the road user’s perspective. 3) Average queue length, which is the average of the total number of lanes and measures the congestion level of the intersection at each time step. 4) Average queue time, which is calculated by dividing the total waiting time of vehicles caused by a queue by the total number of lanes at each time step and offers a similar perspective as average waiting time, but with a focus on traffic congestion. 5) Average vehicle
Figure 5.9: Training history of a reference DRL-based traffic signal controller.
density, which is the average number of vehicles in each lane at each time step and represents the density of vehicles approaching the intersection.

The performance comparison of the proposed DRL-based method with the baseline methods is shown in Fig. 5.10. These metrics were obtained through 100 repetitions of statistical testing using the same vehicle initialization file. Each data point represents the traffic state at a single time step of the simulations and the median values are indicated by the white labels in the middle of the boxes. The proposed method outperforms the other methods in terms of average vehicle speed, average waiting time, average vehicle density, average queue length, and average queue time. The wide distribution of the average vehicle speed and average waiting time confirms the stochastic nature of traffic flow, while the lower median values indicate that the DRL-based controller has acquired a special control policy that may prioritize overall performance over individual vehicles, as verified later in this section. However, there is room for further improvement of the control policy by tuning the learning algorithm, such as by adding a maximum constraint for the vehicle's waiting time, which will be explored in future research.

The performance of the proposed DRL-based control method is demonstrated in Fig. 5.11 through five common traffic mobility metrics, each with its average value and confidence interval in a time series. As evident from the comparison, the proposed method outperforms the baseline controllers in all five metrics. Comparing Figs. 5.11a to 5.11e, it can be seen that the curves have similar trends, implying that these metrics are interconnected. For example, longer queues lead to increased average queue time, slower average vehicle speed, and slower vehicle discharge. The figure shows that the proposed method has a much shorter average queue length than the other three baselines. The curve of the proposed method increases at the beginning and then drops sharply at the end, whereas the other methods do not follow this trend, demonstrating the effectiveness of the proposed controller in recovering from saturation and clearing the intersection quickly when the traffic volume decreases.
Figure 5.10: Performance comparisons with baselines.
Figure 5.11: Simulation comparisons with baselines.
In contrast, the other methods cannot fully recover from congestion within the simulation time. Additionally, compared to the fixed-timing control, the curves of the proposed method are less oscillating and more stable, indicating the adaptation and harmonization of the method to real-time traffic conditions.

To understand the reason behind the wide distribution of queue length standard deviation simulated by the proposed method, the lane-wise simulation data is plotted in Fig. 5.12. The comparison of the average queue length and queue time of each lane reveals that the lanes in the north and south approaches have similar curves with lower values compared to the west and east lanes. This suggests that the DRL-based control method has adopted an asymmetric traffic flow policy, even though the traffic volumes are balanced, to optimize the performance of the entire intersection. As a comparison, the traffic flow simulated by other considered methods are plotted in Fig. 5.13. It can be seen that their traffic flows are balanced, as evidenced by the similar trends and values of average queue length and queue time for each lane. This indicates that the DRL algorithm learned a balanced control policy.

The performance comparisons presented above showcase the effectiveness of the proposed method. To further validate the acquired control policy by the proposed method, we trained a controller with symmetric signal phases and timings. Specifically, each signal cycle had equal lengths for the first and third phases, while the second and fourth phases were identical. The cycle lengths were selected from the set \{10 s, 20 s, 30 s, 40 s, 50 s, 60s\}, resulting in 27 total actions. The purpose of this experiment was to understand the control policy learned by the DRL algorithm with symmetric SPaT. As shown in Fig. 5.14, the results for the symmetric policy have more noticeable periodic oscillations, indicating periodic traffic congestion. However, despite not being as good as the original DRL-based controller with flexible SPaT, it still outperforms the fixed-timing controller. There is no significant difference in traffic flow between directions for the symmetric SPaT policy, hence a plot was not included. This comparison confirms our speculation that the original DRL controller
Figure 5.12: Traffic flow simulated by proposed method in each lane.

Figure 5.13: Traffic flow simulated by referenced method in each lane.
Figure 5.14: Simulation comparisons with the DRL-based controller trained using symmetric SPaT.
had learned a policy with asymmetric traffic flows to maximize the performance of the entire intersection.

5.4.5 Robustness Analysis

The challenge in data-driven control methods is ensuring robustness and reliability with discrete data points. On one hand, to achieve optimal control in a complex environment, the agent must visit each state-action pair enough times, which may result in overfitting, i.e., poor control performance under the unseen traffic conditions. On the other hand, to perform robustly on unseen states, the solution is either to train the agent with a large dataset that includes as many state-action variations as possible or to simplify the state and action space, but both approaches may compromise convergence and optimality of the controller. Our solution tackles this challenge by using a CAE to reduce the dimensionality of the state space while increasing the action space to enhance control performance.

The previous simulations were conducted with a constant traffic flow rate of 480 vehicles per hour per lane. To test the robustness of the proposed DRL-based method, the controller was evaluated under different traffic volumes with varying flow rates. The results of the previously trained controller with traffic volumes of 400, 600, 720, and 800 vehicles per hour per lane are shown in the green box-plots in Fig. 5.15. The orange box-plots represent the performance of controllers specifically trained with the corresponding traffic flow rates. For example, the orange box-plots for 400 vehicles per hour per lane show the simulation data generated by a DRL-based controller trained with a constant traffic flow rate of 400 vehicles per hour per lane. The fixed-time and actuated controllers were also included for comparison. The comparison shows that the proposed DRL-based controller performs well in unseen traffic scenarios. However, retraining the controller for specific volumes does result in improved performance, but at a higher training cost and without guaranteed results. The control policy depends on the traffic volume, but the agent cannot determine the traffic volume from a single
Figure 5.15: Robustness analysis by comparing the performance of DRL-based controller.
control cycle’s traffic state. To overcome this limitation, future investigation could focus on incorporating temporal information into the traffic state inputs to achieve optimal performance in various traffic scenarios.

5.5 Conclusion

In this Chapter, a novel DRL-based traffic signal controller is developed for a typical four-way intersection. The proposed method leverages the use of CAE to capture traffic states into compact representations, enabling a more flexible design of the action space and increased responsiveness to dynamic traffic conditions. The simulation results demonstrate the robust performance of the proposed method, outperforming three baseline methods across five commonly used performance metrics. The proposed DRL-based controller also exhibits more consistent training results compared to existing DRL methods. To further validate the control policy learned by the DRL algorithm, the traffic flows with different SPaT plans are analyzed. Additionally, the proposed controller is tested for robustness against varying traffic volumes and compared with controllers retrained for specific traffic conditions. The results indicate that the proposed DRL agent is capable of handling unseen traffic scenarios effectively.

The proposed method has a limitation in that it incurs a high training cost due to the expanded action space. Additionally, it has only been tested on a single four-phase intersection with 100% CV penetration rate. Future work will aim to extend the method to more complex scenarios and scale it up for corridor/network-level signal control with joint optimization of signal timing. This presents a greater challenge as the traffic state and action space dimensions increase exponentially. One potential solution to this challenge is to utilize the multi-agent reinforcement learning approach, which addresses the control problem of multiple autonomous, interactive agents in a common environment by distributing the global control to multiple local RL control agents Bu et al. (2008). The sharing of information among intersections
can help individual signal controllers to learn and work together to optimize the overall performance of the traffic network.

Furthermore, the proposed method uses the position and speed information of CVs at the end of a control cycle to construct the traffic state matrix as input, ignoring the temporal information. Utilizing recurrent neural networks, such as Long Short-Term Memory (LSTM), has the potential to capture the complex dynamics within the temporal information. Integrating an LSTM-autoencoder into the Encoder-Decoder network architecture can learn a representation for time series sequence data, enabling the DRL controller to make more accurate traffic state estimations and improve control strategies. These are promising avenues for future research.
Chapter 6

Study IV: Data-Driven Optimization Framework for On-Ramp Merging Control with Connected and Automated Vehicles
6.1 Abstract

In this Chapter, a Deep Reinforcement Learning (DRL) approach is proposed for coordination of Connected and Autonomous Vehicles (CAVs) at merging roadways. The method is designed to guide the merging of CAVs in a cooperative manner with sufficient safe distance, aiming to improve traffic efficiency, road safety, and reduce fuel consumption and emissions. The DRL network is trained with traffic data obtained by simulating CAVs in a SUMO simulation environment. The simulation results demonstrate that the proposed method effectively improves traffic efficiency and reduces fuel consumption compared to the default Krauss model used in SUMO.

6.2 Introduction

As discussed above, on-ramp merging control has been a topic of interest for many researchers as it presents a bottleneck challenge in highway transportation and is considered one of the most difficult scenarios for human drivers. The merging process involves complex traffic negotiations that require correct assessment of the traffic situation and making merging decisions and vehicle operations within a very limited time and distance. A single mistake can lead to a crash, which makes it a daunting task for human drivers, particularly in bad weather or light conditions.

The objective of on-ramp merging control is to facilitate safe and smooth passage of vehicles on two different roads through the merging area. Optimal on-ramp merging involves creating a sufficient gap between the vehicles on the main road for the on-ramp vehicles to merge, while minimizing or eliminating the braking operations resulting from the merging maneuvers. With the use of CV technologies, such as V2I and V2V communication, vehicles can share information about their location, velocity, acceleration, planned trajectories, and other relevant data. This information can be used to coordinate CVs to achieve safe and seamless merging maneuvers. Different from the rule-based approach to speed control of merging vehicles that has been
studied in Chapter 4, a data-driven method is investigated in this Chapter for the traffic control of the merging scenarios.

As introduced previously, Deep reinforcement learning (DRL) has gained attention due to its ability to effectively tackle complex control tasks by learning a high-level decision making process Vinyals et al. (2019). DRL has been utilized for traffic signal control Shi et al. (2023) and has shown superior performance in simulations, compared to traditional model-based methods. By interacting with the environment continuously, DRL employs a data-efficient approach to train a decision-making agent through experience. As a step toward achieving the optimal merging controls of CVs based on a data-driven method, this research adds a virtual traffic signal at the merging point to only manage the traffic flow on the ramp road, with the assumption that the traffic on main road has priority.

The virtual traffic signal is controlled by the DRL controller. Ideally, when there is a suitable gap between the vehicles on the main road, the signal will release vehicles on the ramp road to merge, otherwise they have to wait for a clear merging gap. It is similar to the idea of ramp metering Mizuta et al. (2014), but without the requirement of building complicated mathematical models to characterize the highly dynamic, nonlinear, stochastic traffic states.

The stochastic and nonlinear nature of traffic flow makes data-driven control approaches particularly suitable for learning policies by observing traffic state transitions. Unlike model-based optimization approaches, data-driven methods do not require prior knowledge of the traffic system and are less computationally intensive for generating merging sequences. In addition, compared to rule-based methods, they eliminate the need for building complex decision-making models and offer better optimality and adaptability. These advantages make data-driven approaches promising for addressing various challenges in traffic control.
6.3 Methodology

The aim of this research is to develop a DRL-based merging control method for CAVs. As shown in Fig. 4.1, this study considers a typical single lane roadway merging scenario with 100% penetration rate and a centralized controller placed within or around the control zone. In general, the DRL-based signal controller collects traffic data from the simulation environment and determines a sequence of merging windows with a constant control cycle. After receiving the sequence information, CAVs decide which merging window to pass through based on their own control strategy. Then, each CAV is assigned with a reference speed based on the selected merging window. Next, each vehicle solves an optimal control problem in real-time to determine the speed profiles, regulating its movement with minimum acceleration and deceleration operations. Additionally, the generated vehicle trajectories are assumed to be shared with the surrounding vehicles through V2V communication to enhance road safety and energy efficiency.

6.3.1 Deep Reinforcement Learning-based Merging Control

The DRL-based control approach typically consists of three fundamental components: environmental perception, agent’s action space, and control objectives for the agent to acquire Sutton and Barto (2018). The agent gathers traffic states and reward signals from the simulation environment, and then takes action based on the present traffic state. By maximizing the expected reward with an appropriate learning algorithm, the agent learns how to generate the desirable control actions to achieve the traffic control objectives. With the aid of vast traffic data from CAVs, the agent can consistently train and identify the inherent relationships between vehicles, resulting in the optimal control policy.
Environmental Perception

At first, the traffic state is represented as a discrete traffic encoding, which consists of position and speed information of vehicles around the merging zone Genders and Razavi (2016). To be specific, each lane is divided into mesh grids with a length of c, resulting in a $3 \times N$ matrix that represents the entire merging area. Each element of the matrix contains two values: one is binary, indicating the presence of a vehicle, and the other stores the speed of the vehicle. Figure 6.1 displays an example of the traffic state matrix, where yellow grids represent vehicles, and the numbers indicate their speeds in m/s. Blank grids signify the absence of vehicles at those positions. In real-world scenarios, mobility information of each vehicle can be obtained through a vehicular network or other devices Jeong et al. (2021).

Although using the position and speed information of vehicles to build a traffic state matrix as input to the DRL training algorithm is convenient, the resulting space of the traffic states is too large for the DRL algorithm to establish a direct relationship between the traffic state and the control action. Therefore, an autoencoder neural network (Baldi, 2012) can be used to compress the complex traffic states of the entire merging junction into a concise representation. The control agent requires a state representation with rich information to make better decisions while avoiding the “curse of dimensionality”. Essentially, the solution is to reduce the dimension of state representation while retaining as much of the remaining information as possible. Furthermore, using a compressed state representation may enable the autoencoder network to extract the underlying traffic pattern as features. Feature extraction has proven to be an effective method for improving reinforcement learning in numerous applications Hakenes and Glasmachers (2019).

An autoencoder is a type of neural network that can generate a compressed representation of input data. Meanwhile, convolutional neural networks, such as the Visual Geometry Group (VGG) neural networks, are capable of learning the intrinsic features of the spatial information in input data. When the convolutional neural
Figure 6.1: An example of traffic state matrix for on-ramp merging.
network is arranged in an encoder-decoder architecture, it can encode a static traffic state into a fixed-length vector, which serves as the compressed representation. This compressed representation can be utilized for other tasks, such as serving as the input to a reinforcement learning model.

The convolutional autoencoder (CAE) network used in this study has an architecture consisting of an encoder and decoder, each with two pairs of convolution-pooling layers followed by two fully-connected layers. The CAE is designed to reconstruct the input data through a bottleneck layer, represented by $H$. The input and output data for the CAE are the traffic state matrices, each with a shape of $3 \times 100 \times 2$. The dimensions of the output at each layer are determined by the number of notations in Fig. 6.2. The size of the hidden layer $h$ is determined through training experiments. Once the training is complete, the encoder network will be used as a state representation compressor to generate the input vector for the DRL neural network. The architecture of the CAE is illustrated in Fig. 6.2.

The CAE is trained using Tensorflow Abadi et al. (2015), with the Adam optimization algorithm and Mean Squared Error (MSE) as the cost function. Various hyper-parameters, such as the numbers of filters and neurons, are determined through cross-validation training experiments. The size of the reduced representation vector is chosen to be 64. Additionally, the input state matrix is normalized using a normalization technique that scales the vehicle speed between 0 to 1 based on the maximum allowable speed of the road.

To evaluate the performance of the CAE structure, a validation set consisting of 10% of the total training dataset is used. The training dataset is generated by continuously running simulations with randomly initiated vehicles at a specified flow rate. Fig. 6.3 shows the training history using a dataset of eight million samples, resulting in minimum reconstruction errors of $5.71e-4$ and $5.52e-4$ for the training and validation errors, respectively. The purpose of using CAE is not only to reduce the dimension of the traffic state but also to extract intrinsic features from the traffic information. Increasing the size of the hidden layer can reduce the reconstruction loss.
Figure 6.2: The structure of the proposed Convolutional AutoEncoder (CAE) for traffic state representation.

Figure 6.3: Training history of the proposed convolutional autoencoder.
but may not be beneficial for the RL algorithm. Therefore, the effectiveness of the trained CAE needs to be further validated by the RL control experiments.

**Deep Reinforcement Learning Algorithm**

Once the CAE has been trained, the next step is to integrate it with the DRL algorithm. The DRL training algorithm structure is similar to the one shown in Fig. 5.7. In this process, the encoder generates the compressed representation of the traffic states, while the fully connected neural network approximates the Q-value function. By using the compressed traffic representation as input, the neural network calculates the Q-values for all possible actions. The agent then chooses the action with the highest Q-value, indicating the maximum reward. After selecting the action, a new control cycle begins. By continually interacting with the environment, the agent learns to maximize the rewards. To improve learning efficiency and minimize overestimation, target network Van Hasselt et al. (2016), dueling network Wang et al. (2015), and prioritized experience replay Schaul et al. (2015) techniques are implemented in this Chapter for the merging problem. Algorithm 5.1 provides the pseudocode for the proposed DRL training algorithm.

Having a well-defined reward signal is crucial for RL-based signal control as it guides the agent in learning the goal of control actions, which is to increase traffic throughput and reduce vehicle waiting time. The reward signal serves as feedback to the agent, evaluating the effectiveness of its prior actions. Unlike other RL-based applications, traffic signal control has no terminal state, so the reward signal must reflect the performance of every action the agent takes.

As discussed for the intersection scenarios, there is no deterministic rule for selecting the most suitable performance index in RL-based traffic control methods. Commonly used performance indices in this field include travel delay, queue length, vehicle density, and average vehicle speed, as noted in previous studies Guo et al. (2019); Wang et al. (2022); Hong et al. (2022). We conducted experiments with various performance indices and their combinations, and discovered that if using the
average vehicle speed as the reward signal, the agent can learn a policy to improve the overall traffic mobility as well as the average travel delay. Moreover, higher average vehicle speed is associated with better fuel efficiency, as fuel consumption is primarily related to vehicle speed and idle time. Therefore, we compute the average vehicle speed $\bar{V}$ of the entire control zone as:

$$\bar{V} = \frac{1}{N} \sum_{i=1}^{T} \frac{d_i}{t_i}$$  \hspace{1cm} (6.1)

where $d_i$ is the distance traveled by vehicle $i$ during the current control cycle of length $T$, $N$ is the total number of vehicles in the control zone, and $t$ is the simulation time step.

The second reward signal that we experimented with is the cumulative or average waiting time between two control cycles. It measures the road congestion, and at each time step, if a vehicle’s speed is lower than 0.1 m/s, the cumulative waiting time gains +1. However, this reward signal has limitations in handling dynamic traffic states. For instance, when the traffic condition is heavy or unbalanced, the agent may receive a large punishment signal even if it has taken an optimal action, while it could always receive positive rewards regardless of the decision it makes when the traffic load is light.

The third reward signal that we experimented with is the average vehicle density of the entire merging area. It can be calculated as the total number of vehicles divided by the length of roads or directly using the number of vehicles. This signal is conceptually similar to the metric of traffic throughput, but instead of being calculated over a period of time, it can be a real-time measurement of traffic state. However, when the traffic volume changes, it may not reflect the real performance of the DRL controller.

**Merging Sequence Determination**

The “first-in-first-out” (FIFO) rule is a popular and straightforward approach to determine the merging sequence, which prioritizes vehicles based on their arrival
time or distance to the merging zone. This approach is appropriate when travel-time efficiency is the sole control objective, without considering fuel efficiency. Some studies, such as Jing et al. (2019) and Chen et al. (2020), have introduced a merging performance indicator and optimization techniques to optimize the merging sequence. However, with the growing number of merging vehicles, the computational load of finding the optimal merging sequence increases factorially, making it impractical for real-time applications. To balance the computational efficiency and solution optimality, we develop a series of cooperative rules in Chapter 4 and illustrated in Algorithm 4.2. However, the rule-based method may not be able to achieve optimal performance under different traffic conditions or handle complex and dynamic traffic scenarios.

By leveraging the learning ability and adaptability of DRL, this study aims to achieve more efficient and safer merging operations in complex traffic environments, thereby improving the overall traffic flow and reducing fuel consumption and emissions. Specifically, DRL-based merging control methods have the ability to adapt to changing traffic conditions in real-time, and handle complex and uncertain traffic scenarios that rule-based methods may struggle with. DRL methods can also learn from experience and improve their performance over time, whereas rule-based methods are typically static and cannot improve without manual adjustments. Overall, DRL-based merging control methods have the potential to improve the efficiency and safety of traffic flow in merging areas compared to rule-based methods.

To manage the merging process, we divide the time horizon into equally-sized windows, each representing the duration that a single vehicle from different roads can pass through the merging point. For instance, if the merging window is 3-second long, the DRL agent generates a sequence of these windows for each control cycle. The CAVs receive this sequence and choose which merging window to pass through based on their availability and preference. To enable the CAVs to plan their speed profiles more effectively, the sequence must be generated well in advance, when the vehicles are still far from the merging point. Accordingly, we generate the sequence
at the start of the previous control cycle and execute it when it ends. We determine
the merging window length and the number of windows in each sequence through
experimental training and cross-comparison. The latter determines the size of the
DRL agent’s action space, which increases exponentially with the number of windows
in a sequence. Since there are only two options for each merging window, the total
number of actions is the number of windows to the power of 2. For example, a
sequence of 10 consecutive merging windows makes 1,024 selections of action.

6.3.2 Optimal Merging Speed Control of CAVs using DRL

Results

After determining the merging windows using the DRL-based controller, the vehicles
need to choose which ones to pass through the merging point. Additionally, they
require reference speed profiles to guide them through the control zone. These
profiles can be obtained by solving a nonlinear OCP that minimizes a cost function
considering various driving requirements, such as fuel consumption, safety distance
with the preceding vehicle, desired speed, and passenger comfort. Instead of enforcing
a terminal constraint on the OCP, the merging window is transformed into a reference
speed for the vehicle to track. Thus, the optimal speed control problem can be
solved iteratively over a short rolling time horizon $T$ for the implementation of model
predictive control (MPC), as discussed in detail in Chapter 4.

In this Chapter, we establish distinct control objectives for vehicles on the on-ramp
road and the mainline road. As the vehicles on the on-ramp road are required to yield
to those on the mainline road, merging maneuvers often result in reduced traffic flow
speed on the on-ramp road. To enhance the traffic efficiency and road capacity of
on-ramp roads, vehicles on the on-ramp road should attempt to merge into the first
available merging window, avoiding unnecessary deceleration. On the other hand,
vehicles on the mainline road should not be disrupted by merging vehicles, as it may
cause congestion, increased fuel consumption, and collisions. Therefore, the vehicles
on the mainline road are supposed to travel at a desired speed unless they need to accelerate to create voluntary gaps for the other vehicles to merge, if feasible.

There are two scenarios that vehicles must consider when selecting a merging window. The first scenario occurs when vehicles enter the control zone, where they must choose an available merging window. As shown in Algorithm 6.1, newly joined vehicles on the mainline road should aim to travel at a desired speed to improve fuel efficiency, while those on the on-ramp road must select the first available merging window to enhance traffic efficiency and road capacity. If a merging window is not allocated to a specific road, vehicles from either road may select it. The second scenario arises after the traffic controller determines a new sequence of merging windows, which must be allocated to vehicles without an assigned window. Similar to the first scenario, vehicles on the on-ramp road will be assigned to the first available window, while those on the mainline road must accelerate to the nearest available window if they are on a window that is not allocated to them.

**Algorithm 6.1 Pseudocode for the selection of merging window**

```plaintext
function Select Merging Window
  if New vehicle is on mainline road then
    if The optimal merging window is available or can be made available then
      Select the optimal merging window
    else
      Select the first available one after the optimal window
  else
    if The last vehicle in merging sequence is on ramp road then
      Select the first available window
    else
      loop All the available merging windows i after the last on-ramp vehicle
        if There is a vehicle v in front then
          if There is enough space after vehicle v or able to make space then
            Select the window i
            Break loop
          else if Loop Ends then
            Select the window i
        end loop
  end if
end function
```
6.4 Preliminary Simulation Results

The preliminary simulations consider a typical three-legged highway junction, as shown in Fig. 4.1. The parameters for the simulated junction are listed in detail in Table 6.1. All vehicles are randomly initialized with a 20% probability of emitting a vehicle per second. The slopes of both the mainline and ramp roads are assumed to be constant at 0% throughout the simulation. All simulated vehicles have identical parameters, which are listed in Table 4.1. The desired speed for each vehicle is the fuel-optimal speed, which is obtained by solving Eq. (3.20) and is calculated as \( v^* = 13.46 \text{ m/s} \). The maximum speed limit and minimum allowable speed of both roads are set to \( v_{\text{max}} = 30 \text{ m/s} \) and \( v_{\text{min}} = 0 \text{ m/s} \), respectively.

6.4.1 Convergence of DRL-based Signal Controller Training

To validate the proposed methodology, the DRL network was trained with the traffic simulated by SUMO (Simulation of Urban MObility) Lopez et al. (2018). The DRL network is implemented using Tensorflow Abadi et al. (2015) and integrated with the SUMO simulation environment via Python interface. The training was performed in episodes, each comprising 3,600 time steps with a duration of 1 second, resulting in a total of one hour per episode. The random seed for simulating vehicles was varied for each episode. Important hyper-parameters are listed in Table 5.2, and their assigned values were determined through trial and error.

The effectiveness of the proposed DRL training algorithm is assessed by examining the received rewards, which are accumulated after each control cycle. As illustrated in Fig. 6.4, the cumulative rewards sharply increase initially and then plateau as the training progresses. The average vehicle speed and waiting time for each episode are also plotted to demonstrate the traffic measurements’ improvement and convergence. It should be noted that the average waiting time is not included in the optimization objective due to conflicts with the signal control policy’s average vehicle speed. As
Table 6.1: Adopted parameters of simulation environment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lane length</td>
<td>500 m</td>
</tr>
<tr>
<td>Vehicle length</td>
<td>2.5 m</td>
</tr>
<tr>
<td>Maximum vehicle speed</td>
<td>30 (m/s)</td>
</tr>
<tr>
<td>Maximum vehicle acceleration</td>
<td>3 (m/s²)</td>
</tr>
<tr>
<td>Maximum vehicle deceleration</td>
<td>3 (m/s²)</td>
</tr>
<tr>
<td>Minimum gap between vehicles</td>
<td>4.5 m</td>
</tr>
<tr>
<td>Traffic volume</td>
<td>720 vehicles per lane and per hour</td>
</tr>
</tbody>
</table>

Figure 6.4: Convergence of the proposed DRL network.
a result, the average waiting time slightly increases towards the end of the training process.

6.4.2 Case Study: Merging with CAVs

To intuitively demonstrate the merging process, Fig. 6.5 shows the simulated trajectories of CAVs controller by the optimal control method presented in Chapter 4. Two simulation scenarios with different traffic volumes are examined using the proposed SCP algorithm and the trained DRL-based controller. The rolling time horizon of length $T$ was set to 10 s. Each vehicle entered the simulation randomly with a probability based on the traffic volume setting, starting from position 0. The initial speed of the vehicle ranged from 10 to 20 m/s if there was no preceding vehicle in a distance of 60 m. Otherwise, its initial speed was randomly selected from the standard normal distribution around the speed of its preceding vehicle, with a 10% standard deviation to avoid collision risks.

As seen in Fig. 6.5a, in a light traffic condition (360 vehicles per hour in each leg), the ramp vehicles merge actively into the mainline road with minimal impact on the existing vehicles on the mainline road. Conversely, in a saturated traffic condition (720 vehicles per hour in each leg), as shown in Fig. 6.5b, some mainline vehicles must adjust their speeds to create space for the ramp vehicles. In this case, vehicles on both legs merge cooperatively to increase the road capacity. Overall, the results of the DRL-based merging control method seem to be inferior to the rule-based merging control method presented in Chapter 4. The most likely reason is that the DRL-based controller was trained using SUMO simulation data, which does not perfectly align with the optimal control strategy of CAVs. In future work, the DRL-based controller will be trained using traffic data obtained by simulating CAVs with optimal control.
Figure 6.5: Vehicles trajectories of case study with balanced traffic in control zone.
6.4.3 Case Study: Continuous Simulation

The results above demonstrate the effectiveness of the proposed method in guiding the merging of vehicles with sufficient safe distance. To quantitatively evaluate its performance, the average vehicle speed and fuel consumption rate were compared with those obtained from SUMO simulation, where vehicles are not controlled by the optimal control method. The default Krauss model Krauß (1998) is used in SUMO, which prioritizes driving as fast as possible while maintaining safety. Due to the time limit and high computation cost, the merging simulation based on CAVs with optimal control will be included in future work. The comparison results are presented in Fig. 6.6, where each box plot represents the statistics of continuous simulations for 3,600 seconds.

As shown in Fig. 6.6, the DRL-based merging control results in lower vehicle speeds due to regulations imposed on merging maneuvers. However, the fuel consumption rate is significantly improved compared to the SUMO simulation. This indicates that by resolving conflicts between merging vehicles, unnecessary accelerations/decelerations can be avoided, leading to better fuel efficiency. Therefore, it can be concluded that the DRL-based merging control is effective in improving traffic efficiency and road safety, as well as reducing fuel consumption and emissions. In future work, additional traffic scenarios and more rigorous comparisons will be considered.

6.5 Conclusion

In this Chapter, the DRL method is extended to address the merging control problem, and preliminary results showed that the DRL-based approach can successfully improve the traffic efficiency at merging roadways. The compressed traffic states have also shown to be helpful to improve the training performance. However, more work needs to be done to further improve the merging performance for more
Figure 6.6: Performance comparison with SUMO simulation.
complicated traffic scenarios. Currently, the DRL-based controller is trained with SUMO simulations, which do not involve optimal control methods. However, the goal of this study is to develop a DRL-based merging control method for CAVs.

CAVs with optimal control are able to merge cooperatively to resolve the conflicts between merging vehicles, improve the traffic efficiency and road safety, and reduce fuel consumption and emissions. Moreover, with the cooperative control of CAVs, the future traffic state could be more predictable, which is also favorable to the training of DRL agent. However, simulating traffic with CAVs controlled by real-time optimal control is computationally expensive, and DRL agent training requires a large amount of data. As a result, the DRL-based controller trained using traffic data obtained by simulating CAVs with optimal control would probably be a promising direction in the future work.
Chapter 7

Conclusion and Future Work

In this study, a comprehensive control framework for CAVs in traffic systems was proposed with the goal of enhancing road safety, improving traffic flow, and increasing energy efficiency. The approach consists of four main components: (1) A CAV-based optimal speed control approach to reduce idling at intersections, keep a safe inter-vehicle distance, and minimize fuel consumption; (2) A real-time merging control strategy for on-ramp merging; (3) A DRL-based traffic signal controller for a typical four-way intersection; and (4) A DRL-based approach to control of virtual signals at merging roadways.

The CAV-based optimal speed control approach adjusts the velocity of vehicles based on SPaT information obtained through V2I communications, and V2V information exchange among neighboring vehicles. The proposed method employs a pseudospectral discretization method and sequential convex programming method to develop a real-time, onboard algorithm with strong potential. The Model Predictive Control (MPC) framework was used to generate speed control commands at each time step, ensuring collision avoidance and improved inter-vehicle coordination. The simulation results showed that the proposed method was effective in handling safety constraints under dynamic traffic environments, with significant improvements in traffic mobility and fuel efficiency.
The real-time merging control method proposes a set of cooperative rules to ensure the safe and efficient merging of vehicles on the on-ramp road and the mainline road. These rules prioritize safety while taking into account different control goals, such as minimizing deceleration and avoiding congestion. This study utilizes sequential convex programming and pseudospectral discretization methods, with the MPC framework used to continuously update vehicle maneuvers in response to dynamic traffic situations. The results showed that the proposed method was effective in handling safety constraints under dynamic traffic environments with significant improvements in traffic mobility and fuel efficiency.

The DRL-based traffic signal controller leverages the use of CAE to capture traffic states into compact representations, enabling a more flexible design of the action space and increased responsiveness to dynamic traffic conditions. The simulation results demonstrated the robust performance of the proposed method, outperforming three baseline methods across five commonly used performance metrics. The proposed controller also exhibited more consistent training results compared to existing DRL methods. The results indicate that the proposed DRL agent is capable of handling unseen traffic scenarios effectively.

Despite the promising results, the proposed methods have limitations, including a high training cost and limited testing on a single four-phase intersection. Future work will aim to extend the methods to more complex scenarios, scale them up for corridor/network-level signal control, and utilize recurrent neural networks to capture the complex dynamics of temporal information. The multi-agent reinforcement learning approach may also be utilized to optimize the overall performance of the traffic network. Going forward, the plan is to introduce a collaborative control approach for both CAVs and signalized intersections. This will synchronize the movement planning of CAVs with the optimization of traffic signals, aiming to reduce congestion and increase energy efficiency. Overall, this study presents promising avenues for future research to create a comprehensive control framework for CAVs in traffic systems.


164


Hao, P., Boriboonsomsin, K., Wu, G., Gao, Z., LaClair, T. J., and Barth, M. (2019). Deeply integrated vehicle dynamic and powertrain operation for efficient plug-in hybrid electric bus. Transportation Research Board. 15


166


167


170


172


176


177


178


Yang Shi was born and raised in China. After completing his bachelor’s degree in Mechanical Engineering at Hefei University of Technology, he decided to work as an manufacturing engineer at CRRC Nanjing Puzhen. After years of work, he realized that he was not completely fulfilled by the knowledge he had acquired in the industry. He felt a strong urge to continue his education and gain a deeper understanding of the latest, cutting-edge techniques in engineering field.

With a passion for lifelong learning and a desire to push the boundaries of knowledge, Yang embarked on a journey to further his education in United States. He completed a Master of Science program in Mechanical Engineering at University of South Carolina, Columbia. His research focused on the topics of machine learning and feature recognition. Throughout his studies, Yang threw himself into research with fervor, taking on coursework and consistently striving for excellence. His efforts finally bore fruit when he finished four journal papers during the Master’s.

Building upon the foundation laid by his Master’s degree, Yang set his sights even higher and enrolled in a PhD program in Mechanical Engineering at University of Tennessee, Knoxville. With the guidance and support of his advisor, Yang persevered through the challenges and obstacles that inevitably arise in a PhD program. He worked tirelessly to expand his knowledge and expertise in the field of optimal control. Yang’s hard work paid off when he was awarded the Dr. Thomas E. Shannon and Mrs. Patricia Shannon Endowed Graduate Fellowship and a special one-time Spring 2022 Fellowship for his hard work and excellent performance.