Integrating MPC and RL for Coordinated Highway Traffic Control

N.F.A.J.J. Al Aeedanee

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Integrating MPC and RL for Coordinated Highway Traffic Control

Master of Science Thesis

N.F.A.J.J. Al Aeedanee

17th February 2025

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Traffic congestion remains a critical challenge with profound economic, safety, and environmental implications, exacerbated by the continuous growth of global population and vehicle densities. The expansion of road infrastructure is often impractical due to excessive costs and spatial constraints. In response, intelligent traffic management strategies have gained prominence, leveraging advanced control methodologies to optimize flow and mitigate congestion. Among these, Model Predictive Control (MPC) has been extensively applied due to its capacity to handle complex, constrained systems. However, the accuracy of the underlying prediction models inherently determines its effectiveness. Traditional system identification methods can have inherent uncertainties and are often only accurate to a certain confidence level, leading to potential performance degradation in dynamic traffic conditions. To overcome this limitation, learning-based MPC integrates machine learning techniques, particularly Reinforcement Learning (RL), that can enhance adaptability and closed-loop performance. This integration can result in more responsive and robust traffic management systems capable of handling uncertainties and dynamic environments.

The integration of MPC and RL has emerged as a compelling approach to control complex nonlinear systems, capitalizing on the strengths of both methodologies. RL constantly improves control policies based on observed system dynamics and performance feedback, while MPC automatically chooses the best control actions over a limited prediction horizon while still enforcing system constraints. This thesis investigates the integration of RL within the MPC framework for highway traffic flow optimization, specifically focusing on the dynamic regulation of Variable Speed Limits (VSL) and Ramp Metering (RM). The proposed approach is evaluated against three benchmark models: a theoretically optimal MPC model with perfect system knowledge, assuming exact parameters; an MPC controller with an imperfect prediction model, where these parameters deviate from their true values, leading to suboptimal control decisions; and the MPC-RL model, which dynamically adjusts the learnable parameters during training to improve overall system performance.

Contents

List of Figures

List of Tables

Preface & Acknowledgements

I would like to thank my supervisors dr. A. Dabiri and F. Airaldi for their support and valuable insights during the course of my thesis. Thank you for all the feedback and guidance. Their expertise has played an integral part to the success of this research. I would also like to acknowledge TU Delft for supporting this research and providing the necessary resources to carry out this work. On a personal level, I am grateful to my family and friends for their continuous support, patience, and faith in me during this journey. Their encouragement has given me strength and motivation. Finally, I would like to express my gratitude to who contributed to the succesfull completion of this thesis. Your direction and support have been helpful, and I have been grateful for the opportunity to learn and grow via this research.

Delft, University of Technology
17th February 2025

N.F.A.J.J. Al Aeedanee

Chapter 1

Introduction

This chapter serves as a general introduction to this thesis report. Firstly, the chapter provides some background information and explains the motivation behind applying Reinforcement Learning (RL) to highway traffic control. Then, the problem formulation and objectives are briefly discussed. Lastly, the scope and main focus of this thesis are stated.

1-1 Background and motivation

As the global population continues to rise alongside an increase in the number of vehicles, the capacity of roads is being exceeded. This results in significant costs due to unproductive time loss, which negatively impacts the economy. For instance, traffic congestion can result in employees arriving late for work and missing important meetings, which can ultimately affect business productivity and profitability. More importantly, traffic congestion increases the likelihood of accidents, adversely affecting public safety, and has detrimental effects on the environment due to increased air pollution and fuel wastage. In some cases, the negative environmental impacts of traffic congestion can lead to significant health problems, such as respiratory diseases and noise. One solution to address the ever-growing traffic congestion problem is to expand the road network by adding lanes or building new highway connections. However, this approach is relatively expensive and may not be feasible in certain cities due to space limitations or other logistical challenges. Dynamic traffic control is an alternative, cost-effective, and efficient solution.

Dynamic traffic control refers to the use of technology to manage traffic flow in real-time [7]. It is an important field of study in traffic engineering and transportation planning as it offers the potential to improve traffic safety, reduce congestion, and enhance mobility for road users without the need to increase the size and capacity of the current infrastructure. Dynamic traffic control systems use a combination of hardware and software tools, such as sensors, cameras, and algorithms, to collect and analyze real-time traffic data and make informed decisions about traffic control strategies. These traffic control strategies are often

orchestrated by a central controller, i.e., a control system that, given the real-time measured data, outputs the (possibly best) decision to be taken.

Variable Speed Limits (VSLs) and Ramp Metering (RM) are widely acknowledged as useful control mechanisms for reducing traffic congestion on highways [8]. RM controls the flow at the on-ramp via traffic lights with appropriate timings for red, green, and yellow.[1] VSL modifies speed limits on highway segments (via black displays) based on real-time traffic circumstances, resulting in a more balanced traffic flow across the network. Both systems aim to avoid traffic and reduce congestion by lowering the speed upstream of congestion, preventing bottlenecks from forming or increasing. Together, RM and VSL can improve highway network efficiency and safety.

Early implementations of RM and VSL were based on fixed-time methods, with control timings computed offline using past traffic data [9]. These solutions were unresponsive to real-time traffic variations. Nowadays, more advanced control strategies use real-time traffic data to increase performance. These real-time control methods are enabled by algorithms ranging from simple feedback loops to more advanced methods such as Model Predictive Control (MPC) [10, 1, 6] and machine learning [11, 12].

MPC is a widely used control method in traffic control due to its versatility in handling multivariate, nonlinear, and constrained systems [13]. However, the effectiveness of an MPC controller in satisfying constraints and achieving desired closed-loop performance relies heavily on the accuracy of its prediction model. In practice, the design of the prediction model often involves simplifying or approximating the complex dynamics of the real-world traffic system. Additionally, the parameters of the model are typically estimated through system identification techniques, which have inherent uncertainties and are often only accurate up to a certain confidence level. Another downside of classical system identification techniques is that it completely separates the identification procedure from the control design. In other words, classical estimation algorithms try to find the best model parameters to match the observed data, and only then the controller is designed on top of this estimated model [14], possibly in an iterative manner until desired closed-loop control performance is met. On the contrary, a much better idea is to estimate the model in such a way that the controller performance on the true model is directly optimized. This approach can lead to performance degradation in MPC-based traffic control systems, particularly in the face of unforeseen disturbances or model errors. Therefore, there is a need to develop reliable and adaptive methods for designing predictive controllers in MPC-based traffic control systems.

The advancements in machine learning have fueled a growing interest in control methodologies that aim to learn different aspects of the controller scheme from data, with the objective of reducing conservatism and improving closed-loop performance, as well as enhancing safety and robustness [15]. One example of such adaptive and learning-based methodologies is Learning-Based Model Predictive Control, which enables the automation of controller design and adaptation by leveraging data collected during operation. This approach facilitates a more dynamic and responsive traffic control system that can better handle changes in the system or the task at hand [16]. Prominent machine learning techniques combined with MPC include Reinforcement Learning (RL) [17, 5], Gaussian processes [18], and neural networks [19], which are used to enhance prediction accuracy, adaptability, and decision-making in dynamic environments.

[1] An on-ramp is a roadway section that allows vehicles to enter a highway.

1-2 Problem formulation and objectives

Combining MPC and RL has emerged as a promising approach for learning control policies for complex nonlinear systems, while leveraging the benefits of both techniques. MPC provides a framework for incorporating constraints and optimizing over a finite horizon, while RL can learn a policy that is tailored to the dynamics of the system and performance objectives. The combination of the two methods is particularly useful when a reliable prediction model of the system is not available. The combination of MPC and RL has the potential to revolutionize the way we approach control problems for complex nonlinear systems, including traffic control. The question now is how to combine RL with MPC for the application for traffic control. Therefore, the research question states:

> *How can the frameworks of MPC and RL be effectively combined to optimize highway traffic control and increase performance?*

Computer simulations will assess performance in a case study. The specific focus of this thesis is the application of an MPC-RL framework to highway traffic control, particularly for RM and VSL control. While the effectiveness of MPC-RL for RM has been demonstrated [5], the addition of VSLs to the control loop introduces new dimensions to the problem that have not yet been fully explored. The MPC-RL model will initially be configured with parameter deviations from the perfect MPC model to evaluate whether the RL agent can adjust and optimize these parameters. The goal is to determine if the RL agent can minimize the cost terms outlined in the MPC objective function, ultimately converging toward the performance of the perfect MPC model. This approach addresses the research question by demonstrating how the frameworks of MPC and RL can be effectively combined to optimize highway traffic control and drive performance improvements.

1-3 Thesis outline

This section outlines the remainder of the study to facilitate the reading experience. The sections of this thesis are listed as follows:

- **Chapter 2** provides the background information necessary to understand the integration of MPC and RL in coordinated highway traffic control. It starts with an overview of existing highway traffic control measures. Additionally, the chapter introduces key concepts in RL, discussing its applications in traffic management and its potential to enhance traditional control methods.
- **Chapter 3** presents the theoretical framework and methodology used in this research. The chapter starts by introducing traffic flow modeling, with a focus on the METANET model, which serves as the foundation for the highway traffic simulations. The mathematical formulation of the MPC-based control approach for coordinated VSLS and RM is then introduced, along with necessary modifications to the standard MPC formulation. Finally, the role of RL within this framework is detailed.

- **Chapter 4** describes the case study conducted to evaluate the proposed MPC-RL framework. The experimental setup, including the benchmark highway network, parameter configurations, and implementation details, is outlined. Additionally, the performance of the model is being discussed and analyzed.
- **Chapter 5** concludes this thesis by summarizing the key findings and addressing the research question. The chapter highlights the main contributions of this study and suggests directions for future research.

Finally, additional technical details and supporting theoretical derivations are included in the appendices for further reference.

Chapter 2

Background Information

This chapter provides the necessary background to understand the theory of RL and MPC for highway traffic control applications. The chapter begins with an overview of the current state of highway traffic control, including existing implementations and challenges. Next, the principles of RL and the applications are introduced in traffic scenarios with a focus on Q-learning. The chapter then explores the integration between RL and MPC, highlighting how these methods can be combined to create robust and adaptive traffic control systems. For the last part of this chapter, current unresolved challenges in the field are highlighted, which can be explored in future research.

2-1 Terminology

In order to understand the field of highway traffic control, it is required to first understand the most common terminology used in highway traffic control before diving into the main discussion of this section [1, 20]:

- *Delay (hour):* Delay refers to the additional travel time or waiting time experienced by drivers.
- *Congestion:* Congestion in highway traffic control refers to a situation in which the demand for travel on a highway exceeds the capacity of the highway, resulting in slower speeds, and delays.
- *Density (vehicles per kilometer):* Density refers to the number of vehicles present in a given section of highway at a particular time.
- *Critical density (vehicles per kilometer):* Critical density refers to the density of vehicles at which, traffic flow changes from free-flow to congested flow. At this point, the capacity of the highway is reached.

- *Highway network:* It is a system of connected highway links that allow vehicles to travel from one location to another.
- *Traffic flow (vehicles per hour):* Traffic flow refers to the movement of vehicles on a highway over time. It includes the number of vehicles that pass a certain point or section on the lane per unit of time, the speed at which they travel, and the density of vehicles on the lane at a given time.
- *In/Outflow (vehicles per hour):* Inflow / Outflow refers to the rate at which vehicles enter / exit a highway segment.
- *On-/Off-ramp:* An on-/off-ramp is a roadway section that allows vehicles to enter/exit a highway.
- *Ramp queue (vehicles):* A ramp queue is a line of vehicles waiting to merge into a highway that has formed on a on-ramp.
- *Downstream area:* A downstream area is a section of a highway that is located further down the traffic flow direction from a specific place of interest.
- *Upstream area:* An upstream area is a stretch of a highway that is positioned before a given a specific place of interest.
- *Critical area:* The critical area is a section of the highway, where the capacity of the roadway is most likely to be exceeded and congestion is most likely to occur.
- *Traffic breakdown:* When traffic flow becomes unstable and transitions from free flow to congested flow, it causes a sudden decrease in speed and throughput. At some point, traffic becomes too dense, and little changes in vehicle speed can set off a chain reaction that leads to congestion and, eventually, breakdown.
- *Bottleneck:* It is a section of a highway network where the traffic capacity is insufficient to meet the current traffic demand. The most notable characteristic of a bottleneck is its tendency to cause congestion and queue formation, which in turn aggravates travel delays and contributes to traffic congestion.

2-2 Current State of Highway Traffic Control

This section aims to discuss the current state of highway traffic control and the implementations used in traffic management. First, the highway traffic control problem will be formulated. Secondly, the modelling approaches for highway traffic control will be discussed. Finally, the common highway traffic control measures, their coordination, and the challenges associated with their implementation will be discussed.

2-2-1 The Highway Traffic Problem

The highway traffic control problem is the challenge of managing and regulating highways of vehicular traffic. It comes with several challenges:

- **Congestion:** As population and vehicle ownership continue to grow, the roads become increasingly congested. This can lead to increased fuel consumption and total travel time.
- **Safety:** Ensuring safety is a primary objective and factors such as speeding and inadequate infrastructure can pose significant safety risks. The higher the traffic volume, the higher the risk of crashing [21].
- **Environmental impact:** The impacts of the transportation sector can lead to air and noise pollution, and environmental degradation. Traffic congestion increases vehicle idling times and emissions. Effectively managing traffic while addressing environmental concerns calls for the adoption of sustainable transportation options and alternative travel methods [22].

These challenges involve balancing the following objectives:

- **Ensuring the safety of all road users:** The primary goal of traffic control is to ensure that all road users, including drivers, passengers, pedestrians, and cyclists, can travel safely on the roadways. This involves managing traffic flow, minimizing the risk of collisions, and providing clear and consistent signage and road markings [23]. Traffic control measures must adhere to safety requirements in order to prevent unsafe situations. In this sense, safety requirements can be viewed as a boundary condition or constraint that must be met when designing and implementing traffic control measures.
- **Maximizing the efficiency of the transportation system:** This involves minimizing delays, minimizing the environmental impact of transportation by reducing emissions and air pollution [24], and ensuring that traffic flow is optimized to meet demand [6].
- **Network reliability:** Ensuring that the transportation system is able to provide consistent and predictable travel times for users, even in the face of unexpected events or disruptions, such as incidents or weather conditions. This involves implementing strategies and technologies that can quickly identify and respond to disruptions, and improve the overall performance of the transportation network [25].

In general, safety and efficiency are two important objectives that are interconnected in highway traffic control. This means that improving safety can lead to greater efficiency, as there are likely to be fewer accidents due to less congestion on the roads. Conversely, improving efficiency can also lead to greater safety, as there may be fewer opportunities for accidents to occur. However, there are also cases where safety and efficiency may conflict with each other. For example, lower speed and densities could lead to higher safety, but leads to lower efficiency as the traffic slows down. Therefore, it is important to carefully consider the specific circumstances of each situation when designing traffic control strategies that aim to achieve both safety and efficiency goals.

In order to improve the performance of highway traffic control, the focus should be on maximizing the efficiency of the transportation system by minimizing the delays, reducing congestion, and optimizing the traffic flow to meet demand. For simplicity, ensuring safety of all road users and network reliability are not primary concerns in this project. However, it is important to note that in any practical application of traffic control, safety and network reliability are critical factors that must be considered.

2-2-2 Modelling Approaches for Highway Traffic Control

The problem of traffic control can be thought of as a traditional control problem, where the highway traffic system serves as the "plant" and feedback control is utilized (see Figure 2-1). The plant, which encompasses the highway traffic system, includes various actuators, such as traffic signals, RMs, and VSLs. These actuators are capable of interacting with and influencing the state of the traffic system by, e.g., blocking entering vehicles or slowing down incoming traffic.

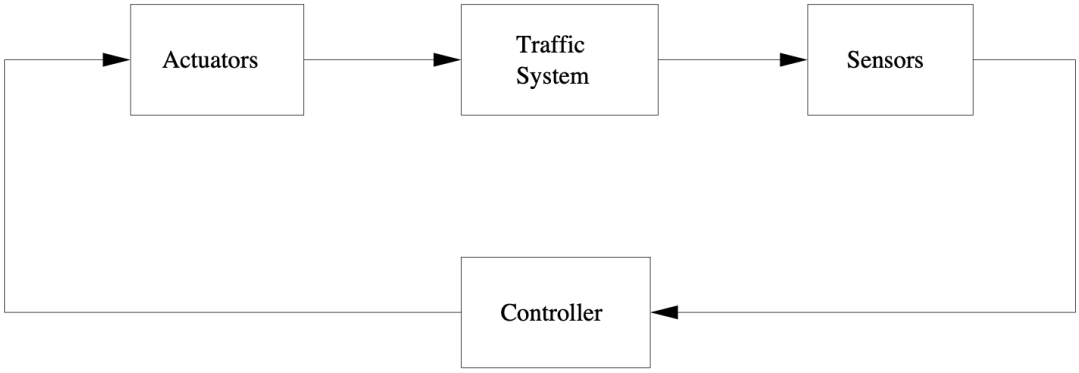


Figure 2-1: A schematic representation of the control loop for managing dynamic traffic. The actuators receive control signals from the controller, which are determined based on sensor measurements. With a closed control loop, any deviations from the desired traffic system behaviour are detected, and appropriate control actions are taken in response [1].

In order to design traffic control strategies effectively, it is important to have accurate models of traffic flow [26]. These models help predict the evolution of traffic congestion and form the basis of control decisions. There are various types of traffic flow models, ranging from simple empirical models to more complex physics-based models. Traffic models can be categorized based on the level of detail used to describe the traffic process, which includes three main types [27]:

- **Microscopic:** It focuses on modelling the behaviour of individual vehicles, taking into account their car-following and lane-changing behaviour. This behaviour is typically influenced by the distance and relative speed of surrounding vehicles, as well as the desired speed of the individual vehicle. In these models, each vehicle is modelled separately, allowing for the assignment of unique characteristics such as driving style, vehicle type, destination, and chosen route. These models are better suited for simulation due to their accuracy in capturing detailed vehicle behaviour, but their high dimensionality makes them less practical for control applications.
- **Macroscopic:** It describes traffic flow in terms of averaged variables such as density, speed, and flow rate computed over multiple vehicles. Macroscopic models are often used in traffic control applications as they provide a high-level view of traffic flow that can be easily incorporated into control algorithms.

- **Mesoscopic:** This traffic flow models describe the behaviour of vehicles in probabilistic terms without tracking them individually. These models include headway distribution models, cluster models, and gas-kinetic models.

For highway traffic control, the focus is on aggregate traffic flow variables such as density, speed, and flow rate. This means that macroscopic models are more desirable rather than other described models. Additionally, macroscopic models are better suited to capture the effects of traffic management strategies such as RM or VSLs, which can have significant impacts on overall traffic flow but may not be discernible at the individual vehicle level. Since macroscopic models focus on aggregate variables, they can be computationally efficient compared to microscopic models that track individual vehicles. The efficiency is crucial for RL algorithms, which often require numerous interactions with the environment (i.e., traffic system) to learn optimal control policies. Additionally, a simplified state space representation is preferred for RL algorithms to process, making macroscopic models beneficial. By providing a higher-level view of traffic flow, macroscopic models can help transportation planners and engineers to optimize traffic control strategies and improve the overall performance of highway systems. In this thesis project, a macroscopic model is employed to describe the dynamics of the system, which will be further discussed in the following chapter.

2-2-3 Highway Traffic Control Measures

Traffic congestion can often be expected in areas where the inflow of the main road and the on-ramp lane exceeds the capacity of the downstream area. Consequently, on-ramps are notorious for causing traffic jams and delays, making them a critical area for improvement. Therefore, it is important to focus on these specific road sections to find ways to reduce congestion and improve overall traffic flow. One potential solution to address the issue of traffic congestion is to synchronize the inflow of vehicles on the on-ramp with the gaps in the flow of the main lane traffic.

RM is a control measure that aims to achieve this synchronization. RM works by using traffic signals at on-ramps to control the rate at which vehicles enter the highway, thereby improving overall traffic flow and reducing congestion. The traffic signals use a combination of red, green, and amber lights to regulate the number of vehicles that can enter the highway at any given time. When the ramp meter is activated, the traffic signal at the on-ramp will display a red light, indicating that vehicles should come to a stop before the on-ramp. After a predetermined time, the signal will change to green, allowing a certain number of vehicles to enter the highway. The green light duration is typically adjusted to maintain a steady flow of traffic on the highway without causing congestion. The duration of the lights is depended on the applied RM strategy. A visual representation of an on-ramp is depicted in Figure 2-2.

In order to evaluate the performance and effectiveness of RMs in pursuing traffic and safety-related objectives (as per Section 2-1-1), literature has used various metrics and measures such as queue length, delay, and travel time. It is important to evaluate RM systems to ensure that they are meeting their objectives, which typically include improving traffic flow, reducing congestion, and improving safety. While the analysis of highway and ramp demands shows that RM can improve the total time spent in congested conditions, its effectiveness is limited if ramp congestion cannot be adequately controlled. The main reason for this is

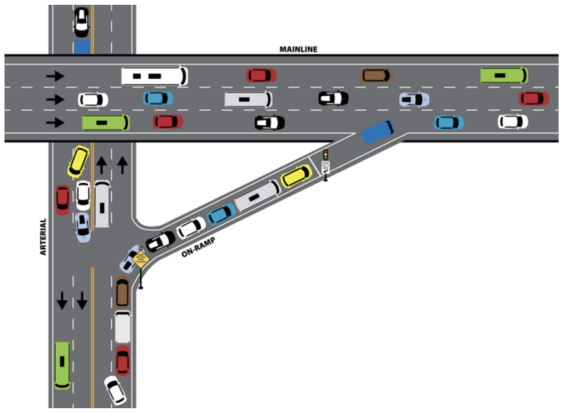


Figure 2-2: Schematic representation of the RM application [2]. It includes detector loops that feed information to the ramp meter controller, monitor the level of traffic and detect gaps in the traffic stream. Loops are also installed on the on-ramp to monitor on-ramp queue. Finally, an end of queue loop is placed at the entrance of the on-ramp to prevent backups onto local streets by releasing the queue when the ramp fills the capacity.

that the RM rate cannot drop below a certain threshold, and as a result, the inflow of the congested area cannot be restricted sufficiently [6]. Specifically, a ramp meter is only effective if ramp congestion is present and the following condition is satisfied:

$$d_{\text{hw}} < q_{\text{drop}} - q_{\text{r,min}}, \tag{2-1}$$

where d_{hw} is the highway demand, q_{drop} the outflow of the ramp congestion (post capacity drop), and $q_{\text{r,min}}$ the minimum ramp flow. This condition shows that RM must balance highway demand and congestion at the ramp. Without satisfying this relationship, the ramp meter cannot sufficiently control inflows, resulting in limited improvements to overall traffic flow and congestion management.

One of the main drawbacks of existing RM strategies is their reliance on predefined traffic models to predict traffic patterns and calculate control actions. These models may not be able to accurately capture unusual conditions, such as extreme weather or traffic incidents, and the parameters of the control algorithm must be fine-tuned. Furthermore, existing strategies have limited capability to improve control efficiency in the long-term. Therefore, a new strategy that improves RM and outperforms existing applications is desired.

As for VSLs, they are deployed as a control measure to achieve multiple objectives, including reducing congestion, increasing traffic flow, and enhancing safety. VSLs are electronic signs that display the current speed limit on a particular stretch of highway (see Figure 2-3) . Unlike traditional speed limit signs, which are fixed and do not change, VSLs can be adjusted to display different speeds in an effort to reflect changing traffic conditions or other factors that affect the flow and safety of the road. One of the main benefit of VSLs is that they improve safety. By reducing speed discrepancies between vehicles travelling in the same lane or adjacent lanes, the behaviour of the drivers becomes more synchronized and discouraged from changing lanes, leading to a decrease in the likelihood of collisions [28]. Another significant benefit of VSL signs is their ability to resolve traffic breakdowns. When a highway approaches

capacity, even minor disruptions to the traffic flow can quickly lead to congestion and traffic breakdown. By slowing down traffic in advance of bottleneck locations, VSL signs can help to prevent or delay the onset of traffic breakdowns, restoring the capacity of the highway and keeping traffic moving smoothly [1]. At last, it can also lead to environmental benefits. Congestion is often linked to increased fuel consumption and emissions, and by reducing congestion, VSL signs can help to minimize these negative impacts [29].

Figure 2-3: VSLs used on a A15/A16 highway in the Netherlands [3].

In the end, VSLs are considered to be effective control applications. However, it is not without limitations. It is still incapable of dealing against large variations in traffic flow. It is therefore of interest to find a new strategy that improves VSLs and outperform the existing applications.

2-2-4 Coordinated RM and VSL Control

Coordinated traffic control systems have emerged as a promising solution to manage traffic flow and reduce congestion, combining RM and VSL signs to regulate traffic volume and speed. Improving the synergy of these systems can yield an improvement of traffic flow and reduced congestion on highways, leading to reduced travel times and improved safety [6, 30]. Coordinated control systems for highway traffic control using RM and VSLs typically consist of several components that work together to achieve the desired traffic flow. These components include:

- **Traffic simulation model**: The model is used to simulate the traffic flow on the highways, including the vehicles, their speeds, and their behaviour.
- **Control input variables**: The data collected includes density, speed of the mainstream flow in the vicinity of the ramp, and queue length of the on-ramp, and is used as input for control and decision-making processes. This data can be collected by using sensors.
- **Decision making architecture**: The architecture is tasked with processing the data collected by the sensors to determine the appropriate actions to take. This is typically

done using control algorithms. These algorithms take into account factors such as traffic flow, capacity, and safety to determine the optimal RM rates and VSLs (control objectives). The controller can adjust the RM rate based on current traffic conditions and congestion levels, while VSLs can adjust the speed limit in real-time to improve traffic flow and reduce congestion. Once the appropriate RM rates and VSLs have been determined, these are implemented through various actuation mechanisms. Feedback mechanisms are commonly used to monitor the effects of the control actions and adjust them as needed. A requirement is to have a communication system that ensures that the controllers receive accurate information about the traffic conditions on the highway, allowing them to make the necessary adjustments to improve traffic flow.

There exist various approaches to integrate VSLs with RM. In addition to implementing the aforementioned components, this integration is determined by the adopted model and the design of the control strategy, and can be categorized as follows [30]:

- **RM rate is determined before determining VSL.** In this approach, the RM rate is initially determined based on the traffic conditions and then the VSLs are adjusted accordingly. The VSLs can be set to the maximum allowable speed if there is no congestion or set to an intermediate or lower value if congestion is present. This approach is simpler to implement but may result in suboptimal control as VSL decisions are constrained by prior RM control actions.
- **Determine VSL first before determining RM rate.** In this approach, the VSLs are determined first based on the traffic conditions and then the RM rate is adjusted accordingly. The VSLs can be set to a lower value to reduce the inflow of traffic to a congested area and then RM can be used to further manage the traffic flow. This approach smooths traffic flow upstream by prioritizing VSL but limits the flexibility of RM control actions.
- **Determine RM and VSL simultaneously with coupled dynamic model involving speed and density.** In this approach, a dynamic model that can consider both RMs and VSLs and can take into account the effect of their control on the traffic speed and density must be employed. The model is used to determine the optimal RM rate and VSL values simultaneously. This approach provides greater control flexibility and as a higher degree of freedom, but is computationally intensive.

Various control methods are used in traffic management, with a focus on controlling VSLs and RM. For instance, [31] combined VSL control with pre-determined RM rates using the macroscopic model METANET to optimize traffic conditions. Their approach led to a 31.8% reduction in total travel time and a 12.8% increase in total travelled distance. In another study, [8] also designed VSL controls based on factors such as bottleneck flow and onramp demand variation, using a linearized model to coordinate RM. This method facilitated efficient numerical optimization. Similarly, [1] integrated VSL and RM to reduce vehicle travel time in highway networks. They highlighted the need for distributed MPC to manage large networks effectively while maintaining computational efficiency. Their approach reduced total time spent by 14.3%, which was a better result compared to using RM alone. Other studies investigated the differences between distributed and centralized MPC control strategies.

While centralized control is more effective, it is computationally more expensive. In contrast, distributed systems that allow communication between controllers can enhance traffic performance with a lower computational cost [32].

2-2-5 Challenges

There are three aspects that make the application of a control mechanism challenging and computationally complex in highway traffic control:

- The dynamics of the highway traffic flow are often non-linear and are influenced by a variety of factors, including driver behaviour, vehicle characteristics, road design, and environmental conditions. These factors can lead to highly dynamic and complex traffic patterns, such as stop-and-go waves, and congestion.

- The VSLs are only permitted to take values from a discrete set, while the RM rates are continuous control inputs. VSLs are limited to a discrete set of speed limits, which are distinct and separate values often expressed as integers or whole numbers. In contrast, RM rates are continuous control inputs that can take any real value within a given range. This means that VSLs can only be set to specific speed limits, such as 60 km/h or 80 km/h, but not to intermediate values like 65 km/h. In contrast to RM, the RM rate can be set to 0.5 vehicles per second, 0.6 vehicles per second, or any value in between. The limitation of discrete control inputs, such as VSLs, restricts the range of control strategies that can be implemented. Continuous control inputs, such as RM rates, provide greater flexibility, allowing for a wider range of control strategies and potentially more effective traffic management. This could be challenging for optimization problems, since it must deal with the combination of continuous and discrete control inputs. This restriction limits the range of control strategies that can be implemented and can potentially make the optimization problem more complex. The formulation and complexity of the optimization problem will depend on the specific objectives and constraints of the control problem, as well as the number and type of control inputs involved. Mixed integer programming approach may be an appropriate optimization method for combining RM with VSLs in highway traffic control, particularly when the issue comprises both continuous and discrete variables. However, the effectiveness of the method is dependent on the specific characteristics of the problem, such as the size of the network, and the complexity of the control objectives.

- To achieve effective performance, MPC needs to be applied over a broad spatial scale in order to coordinate multiple control measures. In highway traffic networks, there are typically multiple control measures that need to be coordinated, such as VSLs and RM installations. Coordinating these control measures requires a large-scale approach that takes into account the interactions between different parts of the network. Moreover, the large spatial scale of the highway traffic network means that the prediction model used in the MPC controller needs to be able to accurately predict future system trajectories over a wide area. This requires a high level of computational power and accuracy, which can be challenging to achieve.

2-3 Reinforcement Learning and its Applications in Traffic Management

This section introduces the key concepts of RL, and emphasizes the importance of the Markov property in modelling the highway traffic control problem. It also explores Q-learning, a model-free RL algorithm well-suited to highway traffic scenarios, and discusses the role of hyperparameters in optimizing RL performance. It concludes with a review of RL applications in traffic control, such as RM and VSLs, highlighting both the successes and limitations of current methods.

2-3-1 Introduction to Reinforcement Learning

RL techniques are increasingly being recognized as a valuable approach for tackling intricate decision-making tasks. One reason for this is that RL has the capability to automatically develop novel strategies for challenging problems that may be difficult for humans to devise. It is a subfield of Artificial Intelligence (AI) that focuses on creating agents that can learn how to make optimal decisions in complex environments [33]. RL agents learn through trial and error by receiving feedback in the form of rewards or punishments for their actions. It consists of five main subelements:

- **Environment**: The environment is the external system in which the agent operates. It is a dynamic system that can change over time based on the actions of the agent. The environment can be anything from a simple game to a complex real-world scenario.
- **Agent**: The agent is the decision-making entity that interacts with the environment. It observes the state of the environment, selects actions based on its policy, and receives feedback from the environment in the form of rewards or punishments.
- **Reward signal**: The reward signals are the feedback mechanism used by the environment to provide feedback to the agent, after an action is taken. The goal of the learning agent is to maximize the total reward it receives over the simulation time. Rewards can be positive or negative, and they reflect the desirability of the action taken by the agent. As consequence, it can alter the policy. If an action selected by the policy is followed by low reward, then the policy may be changed to select some other action in that situation in the future.
- **Policy**: A policy can be seen as a guide that a learning agent follows to decide what actions to take in different situations, when interacting with the environment. It essentially describes the mapping from the states of the environment to actions to be taken when in those states.
- **Value function**: Whereas the reward signal indicates what is good in an immediate sense, a value function specifies what is good in the long run. The value of a state given a policy is the total reward an agent can expect to accumulate over the future by following that policy. Also, it captures the long-term desirability of states. For example, a state with low immediate rewards might still have high value if it leads to future states with substantial rewards. Conversely, a state with high immediate rewards might have low value if it leads to poor outcomes in the long run.

The agent interacts with the environment by taking actions that transition it from one state to another. Each action results in a reward or punishment from the environment, and the objective of the agent is to maximize the cumulative reward over time. To achieve this, the agent must choose actions that yield high rewards according to the objective function. The policy of the agent determines how it selects actions based on the current state, and the value of a state is the expected return that can be achieved by following the current policy. Depending on the methodology, these values may be stored for all states in a tabular form, or learned via function approximation.

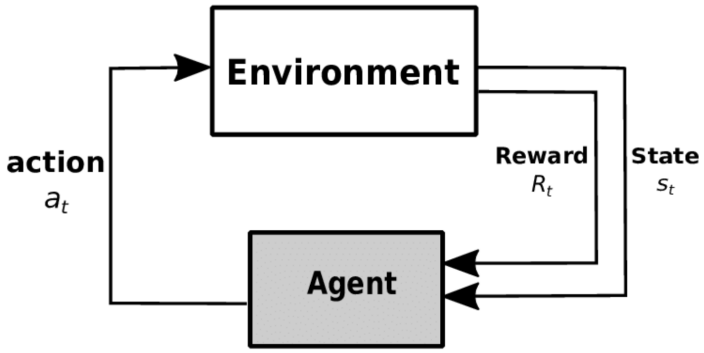


Figure 2-4: A schematic representation depicting how the agent interacts with the environment [4].

Note that an RL problem is essentially a closed-loop optimal control problem, since the actions of the agent influence the environment, and the resulting changes in the environment impact the future decisions of the agent. This continuous cycle of interaction allows the agent to learn from the consequences of its actions over time, adapting its policy to maximize long-term rewards.

One of the challenges that arises in RL is the trade-off between exploration and exploitation. To obtain high reward, an RL agent would intuitively prefer actions that it has tried in the past and found to be effective in producing reward. When exploiting, the agent relies on past experience, choosing actions it has already found to be effective. However, to discover such actions or better actions, it has to try actions that it has not selected before. By exploring, the agent gains new actions, which can help it make better decisions in the future. The agent has to exploit what it already knows in order to obtain reward, but it also has to explore in order to make better action selections in the future. The dilemma is that neither exploration nor exploitation can succeed on its own without failing at the task. Therefore, the agent should try and discover a variety of actions and progressively favour those that appear to be best in term of rewards [33].

2-3-2 Markov Property in Highway Traffic Control

The Markov property means that the future state of a system depends only on its current state and action, not on any past states. In other words, if the current state of the highway control system provides enough information to predict the future state, then the system has the Markov property. In the context of highway traffic control and depending on the model

that is used (METANET), the states represent the density and speed on each segment, and the queue at the on-ramp and mainstream origin. The actions are the control inputs such as the RMs and VSLs. If the state representation of the system includes all relevant information about the current states, the future state can be predicted without needing information about past states, thereby satisfying the Markov property.

It is beneficial to have a Markov property in highway traffic control, since it simplifies both the modelling and the application of RL algorithms, enabling more efficient and effective learning for controlling highway traffic dynamics. With the Markov property, the current state of the traffic system would be sufficient to predict the next state after applying a control action. This eliminates the need to track complex histories of past states, flows, and control actions, which simplifies both state representation and model design. Also, it reduces the computational complexity because the agent can treat each decision step independently of the past.

Note that the highway traffic control problem has a Markov property and consists of infinite states and actions. Therefore, the highway traffic control problem can be formulated as a Markov Decision Process (MDP).

2-3-3 Q-learning Algorithm

Q-learning is a model-free, off-policy reinforcement learning (RL) algorithm. Q-learning aims to find the optimal policy for an agent, which is the set of actions that maximizes the expected cumulative reward when interacting with the environment [33]. It achieves this by learning an action-value function, also called the Q-function. This function estimates the "quality" of each action in each state, which makes it possible to make decisions that maximize long-term rewards. This is described as:

$$Q(s_t, a_t) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s_t, a_0 = a_t\right], \tag{2-2}$$

where r_t describes the reward at time step t, s_t and a_t are the current state and action, respectively, and γ is the discount factor, which describes the importance of future rewards and has always a value between 0 and 1. The main objective is to learn an estimate of the Q-function through repeated interactions with the environment, in order to determine the optimal action in each state (learning the optimal policy). This involves updating the Q-values based on the Bellman equation:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right], \tag{2-3}$$

where α is the learning rate, r_{t+1} is the reward received at the next time step, and s_{t+1} and a_{t+1} represent the next state and action, respectively. The agent repeats these steps over many episodes, updating Q-values, and refining its policy. Note that this learning process is independent of the current behaviour of the agent and can learn from data generated by any other exploratory policy. This property, known as off-policy learning, allows the agent to improve its target policy (the one it is learning to optimize) while following a different policy for exploration.

Note that Q-learning algorithm operates at within the framework of a MDP problem. Since the highway traffic control problem can be formulated as MDP, the Q-learning algorithm is well-suited for this application. Although a highway traffic model is often available to predict traffic dynamics (and is model-based) and apply control actions, there are significant benefits to applying a model-free RL approach like Q-learning. A highway traffic control model is limited to unexpected behaviour or real-time variabilities, while Q-learning can learn policies based on observed data and rewards, adapting directly to real conditions without relying on model assumptions [34]. This has a potential to develop a more accurate highway traffic model than the existing one. However, it is important to note that training in a model-free fashion generally requires more interactions and significantly more time to learn effective policies compared to model-based approaches, which leverage predictive models to reduce learning time.

The step-by-step process of the Q-learning algorithm is depicted below [33].

- Initialize $Q(s_t, a_t)$, $\forall s \in \mathcal{A}(s)$, arbitrarily, and $Q(\text{terminal-state}, \cdot) = 0$
- Repeat (for each episode):
  - Initialize s_t
  - Repeat (for each step of episode):
    - Choose a_t from s_t using policy derived from Q
    - Take action a_t, observe r_{t+1}, s_{t+1}
    - $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(r_{t+1} + \gamma \max_a Q(s_{t+1}, a_t) - Q(s_t, a_t)\right)$
    - $s_t \leftarrow s_{t+1}$

  until s_t is terminal

2-3-4 Hyperparameters

In RL, hyperparameters directly influence the ability of the agent to explore in its environment, learn effectively from experience, and converge to an optimal policy. Unlike model parameters, which are learned during the training process, hyperparameters are predefined values and options that determines the behaviour and efficiency of the learning process. Improperly set hyperparameters can lead to slow convergence, suboptimal policies, or unstable training. It is therefore important to have optimal values for the hyperparameters to balance exploration and exploitation, adapt to varying environments, and ensure stability during training.

The key parameters are listed in Table 2-1, categorized into groups based on their tuning priority.

Table 2-1: Key parameters grouped by tuning priority

Group 1 (can be tuned)	**Group 2 (might be tuned)**	**Group 3 (Least favourite to tune)**
Learning rate (α)	Decay rate of the learning process	Update frequency
Exploration (strength and chance)	Discount factor (γ)	Additional constraint for the change update of the learning parameters
Disturbance at the demand	Number of episodes to learn	
Weights for the RL stage cost		

The learning rate determines how quickly the agent updates its knowledge (i.e., in the case of Q-learning, its estimate of the action-value function). A proper learning rate is crucial because a rate that is too high can cause the learning process to overshoot optimal solutions, while a rate that is too low can make the process extremely slow. As for the exploration, too much exploration can lead to inefficient learning, while too little can cause the agent to miss out on potentially better actions. Furthermore, introducing noise or disturbance can help in generalizing the learning process, making the agent robust to fluctuations. Lastly, fine-tuning the stage cost weights for the RL agent can optimize the performance of the agent, since it can influence the value for the TD-error and the update for the learnable parameters.

Although the parameters described in group 2 are not as immediately impactful as group 1 parameters, they play a critical role in fine-tuning the learning process for stability and efficiency. The decay rate of learning rate and exploration controls how quickly the learning rate decreases over time. Properly tuning the decay rate helps in preventing overshooting of optimal solutions and ensures stable convergence to the optimal policy. The discount factor determines how future rewards are valued compared to immediate rewards. Tuning this factor is crucial for balancing long-term and short-term gains. A well-tuned discount factor ensures that the agent makes decisions that optimize long-term performance without overly discounting future rewards. Lastly, tuning the number of episodes is important to ensure that the agent has sufficient number of interactions with the environment to learn effectively.

As for group 3, the parameters typically do not have as dramatic an impact on the overall learning performance as group 1 or group 2 parameters. The update frequency determines after how many episodes the learnable parameters of the agent are updated. In most cases, it is set to update after each episode, which is a common and effective default setting. However, this frequency can be adjusted depending on the learning environment or computational constraints. For example, in scenarios where updates after every episode lead to instability or inefficiency, a less frequent update schedule might be beneficial to improve convergence or reduce computational overhead. Another parameter in this group involves fine-tuning the maximum and minimum allowable changes for the learnable parameters during each update. This parameter provides additional control over the stability of the learning process by constraining the magnitude of adjustments made to the model. Such constraints can be useful in environments when precise control over learning dynamics is required. Since these parameters are less critical to the learning ability of the agent, they are typically tuned only when necessary.

2-3-5 RL in Highway Traffic Control

Numerous studies have explored the application of RL in highway traffic control to improve the performance of the system. Table 2-2 provides a summary of state-of-the-art research on the application of Q-learning to RM control.

Table 2-2: Summary of Q-learning applications in highway control for RM control.

Research paper	**Applied method**	**State variables/ Number of ramp meters**	**Action**	**Reward**	**Traffic model**
[35]	Tabular method	Density of bottleneck, ramp queue length, ramp demand, current metering rate/ 1	Whether to increase, decrease, or not change the current metering rate	Outflow, ramp queue length	Macro
[36]	Tabular method	Number of vehicles in mainline, number of vehicles entered from the ramp, ramp signal of the last step/ 1	Red/green signal	Deviation of density from critical density	Macro
[37]	Tabular method	Mainline speeds, ramp queue lengths, RM signal status/ 1	Red/green signal	Ramp queue length, mainline average speed	Macro
[38, 39]	Tabular method with state-space RM approximation by k-nearest neighbors	Density, ramp flow/ 2	Direct red phase lengths	Total time spent	Micro
[40]	Value function RM approximation by the deep neural network	Densities, ramp queue lengths, off-ramp presence and absence/ 3	Metering rates	Number of discharged vehicles	Macro

The primary goals of these studies were to minimize the total time spent by vehicles in transiting through the highway [38, 39], and reduce deviations of traffic density from the critical density (at which flow is maximal) in the control target section [35, 36, 40]. Based on the Q-learning method used, these studies can be classified into two categories: tabular methods and value function approximation-based methods. Some examples of tabular methods include studies [35, 36, 37, 39, 37] while value function approximation-based methods are exemplified

by study [40]. Most studies used tabular methods for RL, but as the state variables of highway control problems are usually continuous and grows exponentially in larger networks, this method is limited due to the computational cost and the curse of dimensionality of the state space. Some studies used more complex methods like k-nearest neighbours to approximate the continuous state spaces and make it less computationally heavy (such as [38, 39]).

To overcome challenges such as the high computational cost and the curse of dimensionality in the state space, value and policy function approximation methods, such as those employing artificial neural networks (ANNs), have been developed. To the extend of our knowledge, only [40] used value function approximation-based reinforcement learning methods, and this study used artificial neural networks as the approximate value functions. It employed an ANN as the approximated value function for Q-learning. At each time step, the ANN takes all network state variables as inputs and calculates the Q-values of all available actions as outputs. This means that the ANN maps the state vector to another vector, where each element represents the value of a state-action pair. ANNs provide a useful tool for function approximation in RL, especially when dealing with large or continuous state and action spaces. However, overfitting and generalization issues can arise if the ANN is too complex for the given task, or is trained for too long, and the performance is dependent on the assigned hyperparameters. Further research is needed to determine the limits and convergence of ANN-based function approximation methods in reinforcement learning for highway traffic control.

Several studies have also explored the application of Q-learning approaches to VSL control. Table 2-3 provides a summary of prior research on the application of Q-learning to VSL control.

Table 2-3: Summary of Q-learning applications in highway control for VSL control.

Research paper	**Applied method**	**State variables/ Number of VSLs**	**Action**	**Reward**	**Traffic model**
[41]	Tabular method	Densities of mainline and ramp/ 1	Speed limits	Deviation of density from critical density	Macro
[42]	Tabular method	Density of controlled VSL -, immediate -, and downstream section/ 1	Speed limits	Total crash risk	Macro
[43]	Tabular method with state-space RM approximation by k-nearest neighbour	Densities and speeds/ 3	Speed limits	Deviations of densities from critical density, times to collision	Micro
[44, 45]	Value function approximation by the deep neural network	Lane-specific occupancies in mainline and ramp/ 1	Lane-specific speed limits	Total time spent, bottleneck speed, emergency brake, emissions, reduce crash probability	Micro

Again, the primary goals of these studies were to minimize the total time spent by vehicles [45, 44], and reduce deviations of traffic density from the critical density in the control target

section [41, 43]. One paper has a different objective, which is to reduce the total crash risk [42]. Among the studies summarized in this section, [43, 44, 45] employed microscopic traffic simulation models such as MOTUS [46], and SUMO [47]. [41, 42] used macroscopic dynamic traffic flow models such as CTM as the simulation tools. As before, based on the Q-learning method used, these studies can be classified into two categories: tabular methods and value function approximation-based methods (see Table 2-3).

[44] and [45] developed a deep reinforcement learning (DRL) model for controlling differential VSLs (DVSL) that can accommodate dynamic and varying speed limits among lanes. The proposed DRL models use an actor-critic architecture where the actor generates the VSL control action and the critic evaluates the policy of the actor. The Q-value that is estimated by the critic is associated with the efficiency, safety and emission of vehicles in the transportation network. Both the actor and critic functions use multi-layer neural networks as function approximators. The objective of the actor is to maximize the value function Q, while the goal of the critic is to produce a perfect approximation of the value function. Since obtaining the true value function for a complex MDP requires trying out a large number of policies, the value function can be learned by bootstrapping from the current estimate of the value function. In this case, the goal of the training algorithm is to identify a parametrized policy using the actor that maximizes the expected reward return throughout the DVSL controlled time period. The Deep Deterministic Policy Gradient (DDPG) method is used for the training. DDPG uses a stochastic behaviour policy for exploration while estimating a deterministic target policy [48]. The authors stated that one of the primary challenges in reinforcement learning is balancing exploration, which involves actively seeking out actions that might yield high rewards and lead to long-term gains, with exploitation, which involves maximizing short-term rewards using the current knowledge of the agent. Without adequate exploration, the agent may not discover effective DVSL control strategies. DDPG, as an off-policy RL framework, has the benefit of exploration being independent of the learning algorithm.

Only one paper managed to discuss the application of RL to coordinated VSL and RM control. [49] employed a decentralized reinforcement learning approach to address both RM and VSL control simultaneously. A simple, microscopic traffic simulation model is employed for the evaluation of the relative effectiveness of the control policies proposed by the reinforcement learning agents. The RL algorithms for the RM and VSL problems were trained on four scenarios of varying traffic demand. Each scenario consisted of a rush hour period with a gradual increase in demand, a peak demand period lasting an hour, and a gradual decrease in demand followed by a 90-minute period to allow for the system to return to free-flowing traffic conditions. These scenarios were designed to simulate real-world traffic conditions and provide a suitable test environment for the RL algorithms. The authors applied multi-agent reinforcement learning (MARL), which is a specialized branch of reinforcement learning that investigates the behaviour of multiple learning agents operating in a shared environment [50]. Using a MARL approach to solve both the RM and VSL problems simultaneously has shown that there are possibilities for further reduction in Total Travel Time (TTS), although these improvements are relatively minor [49]. Moreover, it has been observed that the utilization of a MARL approach results in a more stable traffic flow in the presence of RM. Additionally, the MARL approach has shown to achieve a better balance between protecting the flow of traffic on the highway and maintaining an acceptable queue length at the on-ramp, compared to other RM approaches, hinting at the fact that some coordination between RM and VSL actions can be indeed beneficial. This is beneficial as it helps prevent congestion on the on-

ramp from spilling back into arterial systems. However, we note that a decentralized approach like MARL, while computationally lighter, can potentially lead to more suboptimal control policies compared to centralized approaches.

Tabular Q-learning has been the most commonly used algorithm in the studies discussed in this subsection, while Q-learning with value function approximation is still under development, especially for coordinated RM and VSL control. Interestingly, there has been no investigation into combining existing controllers with reinforcement learning, such as a MPC. Although it has been shown that the cooperation of ramp meters and VSLs has led to good performance, it is unclear which approach, reinforcement learning or traditional non-learning-based control methodologies, is better. Further research is needed to compare the performance of these two approaches. Overall, the potential benefits of using reinforcement learning in highway traffic control are substantial, and more research in this area is needed to develop advanced control algorithms and optimize reinforcement learning-based systems.

2-4 Combining RL and MPC

Combining RL and MPC requires an understanding of how the two methodologies can complement one another. RL learns from trial-and-error interactions with the environment, enabling the design of policies that maximize long-term rewards. However, RL often struggles with satisfying constraints, stability, and safety during the learning phase. MPC, on the other hand, is a model-based method that computes optimal control actions by solving optimization problems over a finite time horizon, taking explicit constraints into account. In general, predictive controllers such as MPC are heavily bound to the quality of the prediction model, which require a high level of accuracy during the system identification phase. The integration of RL and MPC often focuses on overcoming the limits of one method while leveraging the capabilities of the other. For example, the reliance on an accurate prediction model of the MPC can be mitigated by incorporating the ability of the RL to learn models or policies from data. Conversely, the exploration-driven learning process from RL can be guided by MPC to ensure that actions remain safe and feasible.

The integration of RL with MPC has been investigated in a variety of fields. [51] proposes an integration where Economic Non-linear Model Predictive Control (ENMPC) acts as a value function approximator within the RL framework. Additionally, it acts as a policy generator. ENMPC optimizes performance according to an economic objective while incorporating constraints and dissipativity conditions, providing a structured mechanism to ensure stability and constraint satisfaction. The ENMPC controller uses a parametrized stage cost and constraints that RL fine-tunes via interactions with the real system. This approach requires that the ENMPC controller can deliver the optimal control policy even when the underlying model deviates from reality. One of the critical results demonstrated in the paper is that ENMPC can approximate value - and action-value functions of the RL while maintaining safety guarantees. This is achieved through adaptive adjustments to the ENMPC parameters, including stage costs and terminal components, which RL learns to optimize based on observed system performance. The RL component, using methods such as Q-learning, systematically adjusts the parameters of the MPC controller. This iterative adjustment aims to uncover the optimal policy, either by direct computation or by refining the approximations of the action-value function. This synergy allows the system to leverage the strengths of both MPC and RL,

resulting in enhanced closed-loop performance. This approach can be described in Figure 2-5, where the MPC feeds actions to the real system (environment) according to its current control policy, while the RL component refines the MPC parameters. This feedback loop creates a data-driven mechanism that not only compensates for model inaccuracies but also ensures continuous improvement in system performance.

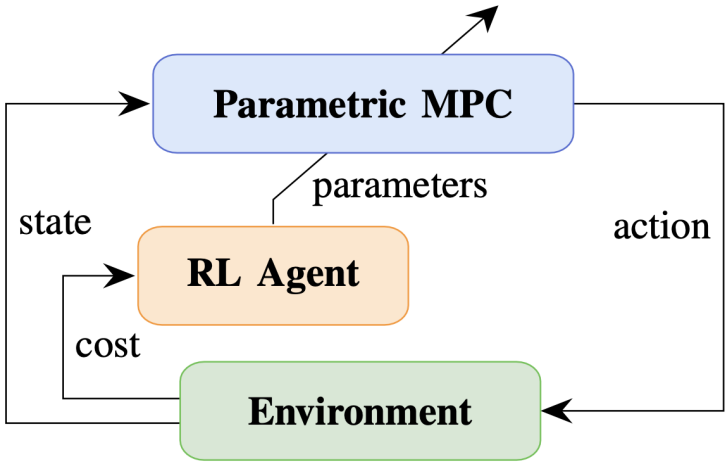


Figure 2-5: Representation of the hybird MPC-RL approach [5].

A key advantage of this integration lies in its capacity to handle discrepancies between the model used by the MPC scheme and the actual dynamics of the system. Even when the prediction model is imperfect, the framework enables the MPC controller to adapt and converge toward optimal policies and value functions through the learning-driven adjustments of its parameters. The effectiveness of this technique, however, is dependent on the expressiveness of the MPC parametrization. It should be rich enough to reflect the real and complete dynamics and aims of the underlying RL task. One of the standout benefits of this approach, especially when compared to Neural Network (NN)-based RL methods, is the ability of MPC to incorporate prior knowledge of the system through pre-defined imperfect prediction models. NN-based RL methods often function as black-box models, offering limited interpretability and posing difficulties in guaranteeing these critical control properties. Moreover, MPC controllers handle constraints on states and actions explicitly and in a structured manner, an area where NN-based approaches typically struggle [52]. This explicit constraint handling ensures a safer and more stable learning process, allowing the system to maintain feasibility even in the early stages of learning.

Traffic control is an ideal application domain for RL-MPC frameworks due to its complexity and dynamic nature. Traffic systems often involve high-dimensional state spaces, nonlinearity, and interactions between vehicles and infrastructure, making them challenging to model and control with traditional methods. For example, [5] used this framework to optimize RM strategies. The approach was designed to adapt dynamically to traffic conditions by embedding RL into the MPC architecture. Specifically, the control problem was formulated as an RL task using a stage cost function that accounts for traffic conditions, control action variability, and queue constraints. The MPC-RL controller utilized MPC for its predictive capabilities while allowing RL to refine the learnable parameters of the MPC and handle uncertainties

in the traffic model. This design overcame the limitations of traditional MPC, which relies heavily on accurate modelling, by enabling the system to learn improved policies directly from real-time interactions with the environment. Testing on a benchmark highway network found that the framework successfully decreased congestion and handled ramp queues. These findings highlight the effectiveness of MPC-RL frameworks in improving traffic flow and meeting real-world constraints.

In summary, combining RL and MPC involves integrating a differentiable parametrized MPC scheme, serving as both policy provider and function approximator, with an Q-learning algorithm. The Q-learning algorithm updates MPC parameters through gradient-based adjustments using observed closed-loop data, enabling the automatic learning of congestion-free, constraint-compliant policies. This method addresses model inaccuracies and uncertainties by bypassing the need for precise system identification and manual parameter tuning, instead learning (sub)optimal parameters through real-time interactions. It has been successfully implemented and tested on a benchmark highway network, efficiently reducing bottlenecks and managing queue lengths while incorporating constraint penalties into both the MPC objective and RL reward function.

2-5 Summary and Identified Research Gaps

While RL has shown promise in learning adaptive policies directly from data and interaction of the agent with the environment, and MPC excels at handling constraints and optimizing over finite horizons, integrating the two approaches has revealed significant opportunities for improving highway traffic control. However, despite these potentials, several challenges and gaps remain that requires further investigation.

In general, highway traffic control continues to face challenges, such as congestion, safety risks, and environmental concerns. Traditional control measures such as RMs and VSLs have demonstrated effectiveness to some extent, but often rely on static models that struggle to adapt to the dynamic and unpredictable nature of real-world traffic conditions. Coordinated RM and VSL control strategies have emerged as a solution, yet their computational complexity limits their scalability to large networks. Additionally, current RL methods often depend on tabular techniques, which become computationally expensive and impractical for large-scale traffic systems with continuous and high-dimensional state spaces. In contrast, value function approximation methods were introduced to address these challenges, making them well-suited for highway traffic control applications.

The integration of RL and MPC presents a promising alternative, leveraging the ability of RL to adapt and learn and the capability of MPC to ensure stability and satisfy constraints. RL-MPC frameworks can improve traffic flow and reduce congestion by enabling data-driven adaptation while maintaining the prediction and constraint-handling capabilities of the MPC scheme.

One of the primary research gaps lies in the scalability of RL methods for traffic networks. While RL has been successfully applied to smaller, simplified highway segments, extending its use to more complex, larger-scale networks remains a significant challenge. Neural network-based approximations have shown promise for addressing the curse of dimensionality, but their application in traffic control is still limited and underexplored. Similarly, the coordination

of RM and VSL using RL-based methods has yet to be thoroughly investigated. Although the combination of these control measures offers potential for improved traffic performance, few studies have explored the integration of RL into this domain, particularly in comparison to traditional control techniques. Safety and environmental considerations also remain underexplored in the context of learning-based control frameworks. While traffic efficiency is a primary focus, ensuring safety and minimizing environmental impacts, such as emissions and fuel consumption, are critical objectives that require further attention. Incorporating these metrics into RL-MPC designs could lead to more traffic control solutions. Lastly, real-time adaptation and learning bring additional difficulties. Such frameworks need to find a balance between making quick, reliable decisions and managing the challenge of RL of exploring new actions while using what it has already learned. Keeping the system stable and safe during this learning process, especially in real-world applications, is a major challenge for practical use.

Addressing these gaps will require a combination of theoretical innovation and simulations. Future research should aim to develop scalable RL-MPC frameworks capable of managing large networks efficiently while incorporating real-time adaptation and coordination between RM and VSL. It will also be important to expand the scope of objectives to include safety and environmental metrics and to validate these frameworks in real-world settings.

Chapter 3

Theoretical Framework and Methodology

The combination of MPC and RL for highway traffic management is a potential technique to reconciling real-time traffic needs with system-wide performance improvement. This chapter describes the theoretical framework and methodology for this integration. To demonstrate the proposed method, three preliminary components are essential. First, a modelling framework is required to construct a representation of the dynamical behaviour of the highway network. Secondly, a classical formulation of MPC is introduced that leverages this model for effective implementation. Lastly, a suitable RL algorithm shall be applied, where the MPC scheme can be used as a function approximation.

3-1 Traffic Flow Modeling

This section discusses the traffic model that is applied during the simulations. Different modelling approaches for investigating traffic phenomena on highway networks exist. This thesis project employs the macroscopic second-order METANET framework [53] to obtain a discrete-time dynamical representation of highway traffic behaviour under ramp metering and variable speed limit control. The METANET traffic flow model was selected for its optimal balance between simulation speed and accuracy. It is deterministic, discrete-time, and macroscopic, making it particularly suitable for model-based traffic control. The relatively large simulation time steps and segment lengths of the discretized highway enable fast model execution. In the remainder of this section, the basics of METANET are introduced.

3-1-1 Link Equations

The METANET model conceptualizes a network as a directed graph, where each link, indexed by m, corresponds to a stretch of highway. These links are characterized by uniform attributes, with no significant changes in geometry. Nodes are introduced where significant changes occur,

such as an on-ramp or off-ramp. Each highway link m is subdivided into N segments, indexed by i, each segment being L_m long.

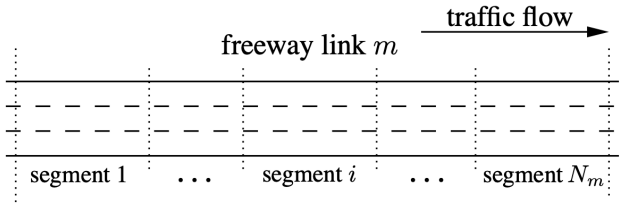


Figure 3-1: A highway link m that is divided into segments [6].

At time instant k, each segment i is defined by its traffic density $\rho_j(k)$, average speed $v_j(k)$, and the outflow $q_{m,i}(k)$. Conversely, each origin o is characterized by its queue length $w_o(k)$. To ensure stability, the segment length (L_m) and the simulation time step (T) must meet specific criterion for each link m:

$$L_m > v_{free,m}T, \tag{3-1}$$

where $v_{free,m}$ represents the average speed assumed by drivers when traffic conditions are free-flowing, i.e., there is no congestion in the link m. The density and average speed of the vehicles in each segment, and the queue length in each origin or on-ramp are considered as the states of the system. The density of a segment is computed by taking the previous density, adding the inflow from the upstream segment, and then subtracting the outflow of the segment, ensuring the conservation of vehicles:

$$\rho_{m,i}(k+1) = \rho_{m,i}(k) + \frac{T}{L_m\lambda_m}\left(q_{m,i-1}(k) - q_{m,i}(k)\right), \tag{3-2}$$

where the flow of each segment can be computed as the density multiplied by the mean speed and the number of lanes on that segment (λ_m):

$$q_{m,i}(k) = \rho_{m,i}(k)v_{m,i}(k)\lambda_m \ . \tag{3-3}$$

In the METANET model, the mean speed at simulation step $k+1$ is determined by three components: the mean speed at time k, a relaxation term indicating the tendency of the drivers to achieve their equilibrium speed $V(\rho)$, a convection term showing speed variation due to vehicle inflow, and an anticipation term depicting speed adjustments in response to downstream density changes:

$$\begin{aligned} v_{m,i}(k+1) = v_{m,i}(k) &+ \frac{T}{\tau}\left(V\left(\rho_{m,i}(k)\right) - v_{m,i}(k)\right) + \\ &\frac{T}{L_m}v_{m,i}(k)\left(v_{m,i-1}(k) - v_{m,i}(k)\right) - \\ &\frac{\eta T}{\tau L_m}\frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa}, \end{aligned} \tag{3-4}$$

where τ, η, and κ are model parameters. The equilibrium speed can be computed as

$$V\left(\rho_{m,i}(k)\right) = v_{\text{free},m} \exp\left[-\frac{1}{a_m}\left(\frac{\rho_{m,i}(k)}{\rho_{\text{crit},m}}\right)^{a_m}\right],$$

where a_m is a model parameter, and $\rho_{crit,m}$ is the critical density at which the traffic flow is maximal on a homogenous highway. It is important to note that the speed should also depend on the speed limit displayed on the VSLs. Taking this into account, the equation can be reformulated and extended as follows for segments equipped with VSLs [54]:

$$V_{\text{VSL}}\left(\rho_{m,i}(k)\right) = \min\left(v_{\text{free},m} \exp\left[-\frac{1}{a_m}\left(\frac{\rho_{m,i}(k)}{\rho_{\text{crit},m}}\right)^{a_m}\right], (1+\alpha) v_{control,m,i}(k)\right), \tag{3-5}$$

where $v_{\text{control},m,i}(k)$ is the speed limit for segment i of link m at simulation step k, and $(1+\alpha)$ is the non-compliance factor. If $(1+\alpha) > 1$, drivers aim to exceed the speed limit; if $(1+\alpha) < 1$, they drive below it. Thus, α is positive when target speeds are higher than the limit and negative when they are lower, allowing the model to account for both enforced and unenforced speed limits. The length of the queue at origin o at time k, denoted as $w_o(k)$, is determined by adding the demand $d_o(k)$ to the previous queue length and subtracting the outflow $q_o(k)$:

$$w_o(k+1) = w_o(k) + T(d_o(k) - q_o(k)) . \tag{3-6}$$

The outflow from origin o is dictated by the prevailing traffic conditions on the main roadway and, for a metered on-ramp, by the ramp metering rate $r_o(k)$, with $r_o(k)$ ranging from 0 to 1. The ramp flow $q_o(k)$ is computed as the ramp metering rate multiplied by the minimum of the available traffic at simulation step k (queue plus demand), the capacity of the on-ramp, and the maximum flow that can merge onto the highway given the current mainstream conditions:

$$q_o(k) = r_o(k) \min\left[d_o(k) + \frac{w_o(k)}{T}, C_o, C_o\left(\frac{\rho_{\max,m} - \rho_{m,1}(k)}{\rho_{\max,m} - \rho_{\text{crit},m}}\right)\right], \tag{3-7}$$

where C_o represents the on-ramp capacity (veh/h) under free-flow conditions, $\rho_{\max}$ (veh/km/lane) denotes the maximum density of a segment, and m is the index of the link to which the on-ramp is. To account for the speed reduction caused by merging phenomena at an on-ramp, the term

$$-\frac{\delta T q_o(k) v_{m,1}(k)}{L_m \lambda_m (\rho_{m,1}(k) + \kappa)} \tag{3-8}$$

is added to Equation (3-4) of the first segment of link m (i.e., Equation (3-8) is added exclusively to the segment connected to the ramp, while the remaining segments remain unchanged). Here, $v_{m,1}(k)$ and $\rho_{m,1}(k)$ represent the speed and density of the segment connected to the on-ramp, as illustrated in the figure below. The parameter δ is a model-specific constant.

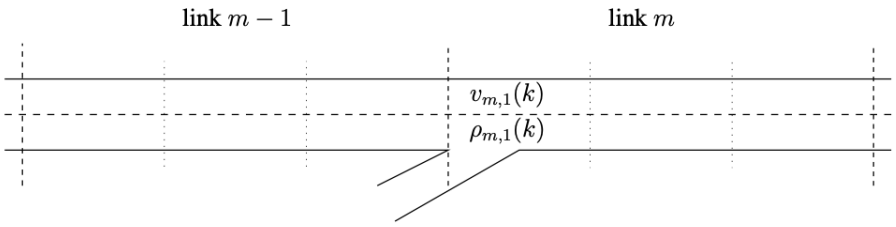


Figure 3-2: When there is an on-ramp connected to the highway. The speed $v_{m,1(k)}$ in the first segment of link m is reduced by merging phenomena.

In the case of a lane drop, as depicted in the figure below, the speed reduction resulting from weaving phenomena can be described as follows:

$$-\frac{\phi T \Delta\lambda_m \rho_{m,N_m}(k) v^2_{m,N_m}(k)}{L_m \lambda_m \rho_{\text{crit},m}}, \tag{3-9}$$

which is added to Equation (3-4), where $\Delta\lambda_m = \lambda_m - \lambda_{m+1}$ is the number of lanes being dropped, and ϕ is a model parameter.

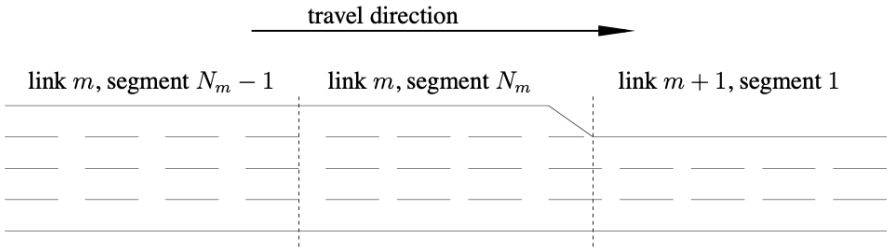


Figure 3-3: When a lane drop occurs, the speed $v_{m,N_m}(k)$ in the last segment of link m is reduced due to merging phenomena.

3-1-2 Mainstream Origin

To effectively distinguish a mainstream origin link $q_{o,main}$ from a standard on-ramp, an additional flow constraint is incorporated to a modified version of Equation (3-7):

$$q_{o,main}(k) = \min\left[d_o(k) + \frac{w_o(k)}{T}, q_{\text{lim},\mu,1}(k)\right], \tag{3-10}$$

where $q_{lim,\mu,1}(k)$ is the maximal inflow determined by the limiting speed in the first segment of link μ, that is attached to the origin. This modification is necessary because the queue at a mainstream origin represents the aggregate effect of upstream sections of the highway network being modelled. The segment inflow, and thus the outflow from the mainstream origin, may be constrained by either an imposed speed limit or the actual speed on the first segment, whichever is lower at critical density. Consequently, the maximum inflow is derived from the speed-flow relationship defined in Equations (3-3) and (3-5), using the smaller value

between the speed limit and the actual speed on the first segment. This is described by the following criteria:

$$q_{\text{lim},\mu,1}(k) = \begin{cases} q_{\text{speed},\mu}(k) & \text{if } v_{\text{lim},\mu,1}(k) < V(\rho_{\text{crit},\mu}) \\ q_{\text{cap},\mu} & \text{if } v_{\text{lim},\mu,1}(k) \geq V(\rho_{\text{crit},\mu}) \end{cases} \tag{3-11}$$

Here, the limiting speed $v_{\text{lim},\mu,1}(k)$ is defined as the minimum of the control speed $v_{\text{control},\mu,1}(k)$ and the actual speed $v_{\mu,1}(k)$:

$$v_{\text{lim},\mu,1}(k) = \min(v_{\text{control},\mu,1}(k), v_{\mu,1}(k)). \tag{3-12}$$

The speed that limits the flow, $q_{\text{speed},\mu}(k)$, is given by

$$q_{\text{speed},\mu}(k) = \lambda_\mu v_{\text{lim},\mu,1}(k)\rho_{\text{crit},\mu}\left[-a_\mu \ln\left(\frac{v_{\text{lim},\mu,1}(k)}{v_{\text{free},\mu}}\right)\right]^{\frac{1}{a_\mu}} \tag{3-13}$$

On the other hand, the capacity flow $q_{\text{cap},\mu}$ is expressed as

$$q_{\text{cap},\mu} = \lambda_\mu V(\rho_{\text{crit},\mu})\rho_{\text{crit},\mu} \tag{3-14}$$

By applying these equations, the maximal inflow can effectively be modelled into the first segment of link μ based on either the speed limit or the actual speed conditions.

3-1-3 Boundary Conditions

To effectively model the traffic network, it is essential to establish boundary conditions for both the entry and exit points. Since the state of a segment in METANET is influenced by the upstream speed, the outflow from the upstream node, and the downstream density, we must specify the upstream speed and inflow at the entry points of the network, and the downstream density at the exit points.

The incoming speed v_μ of the origin is decided to be the speed of the first segment of the leaving link:

$$v_\mu(k) = v_{m,1}(k), \tag{3-15}$$

where μ is the first link after the origin. As for the inflow, it can be assumed that the inflow for the first segment is equal to the main stream flow, which is described in Equation (3-14). Lastly, the downstream boundary conditions are here considered to be affected by no congestion. In cases where the last segment of the incoming link is experiencing congestion, the downstream density is defined as the critical density $\rho_{\text{crit},\mu}$. On the other hand, if there is no congestion, the downstream density is determined by the density of the final segment of the incoming link:

$$\rho_{\mu,N_\mu+1}(k) = \min\left(\rho_{\mu,N_\mu}(k), \rho_{\text{crit},\mu}\right), \tag{3-16}$$

where μ here instead is the last link connected to the destination.

3-2 The MPC scheme

This section discusses the fundamental of the MPC scheme, together with the formulation of the MPC scheme. The MPC [52] method have been proven to be very effective in controlling VSLs and RM installations together [1]. MPC is an advance control methodology in optimal control which is able to provide optimal control input sequences that minimize an arbitrary objective function. To achieve so, a prediction model is used to forecast future system behaviour along a prediction horizon. This allows for the determination of multiple control inputs and their impact on various parts of the network. Therefore, the MPC method is ideal for coordinating VSLs and RM installations. Additionally, an MPC scheme allows to impose constraints on states and inputs of the system and to systematically handle these during the optimization process. This ensures that maximum queue lengths at on-ramps are not exceeded, which helps to avoid congestion.

In general, the methods in the MPC framework use the following strategy [52] (depicted in Fig. 3-4):

- The future control inputs are determined by optimizing an objective function. The goal of this function is usually to minimize the difference between the process and a predetermined reference trajectory. The optimization process aims to find the set of control inputs that will bring the process as close as possible to the desired trajectory. However, this cost function can also include economic terms. For instance, instead of focusing solely on minimizing the tracking error between the process and the reference trajectory, the objective could be designed to optimize other practical performance metrics, such as reducing the total time spent on the highway by the driver, minimizing energy consumption, or maximizing operational efficiency. This alternative approach is often referred to as Economic MPC (EMPC) [55].
- Only the current optimal control input value, $u^\star(k|k)$, is sent to the system, while all other predicted input values for future time steps are discarded. This procedure is repeated at every time step for the remaining future control inputs. This is also known as a receding horizon strategy.
- At each discrete time instant k, the model predictive control algorithm predicts the future outputs $y(k+l|k)$ for $l = 1, \ldots, N_p$, where N_p is the prediction horizon length. This prediction depends on the past inputs and outputs as well as on the future control signal $u(k+l|k)$ for l ranging from 0 to $N_p - 1$, and the dynamic prediction model employed by the controller to estimate the evolution of the state. Note that this prediction model does not necessarily need to match the real system dynamics, although the closer it approximates the actual behaviour, the better the prediction performance. The predicted outputs are calculated for all samples within the prediction accuracy, and they are used to optimize the future control inputs that will bring the system as close as possible to a reference trajectory.

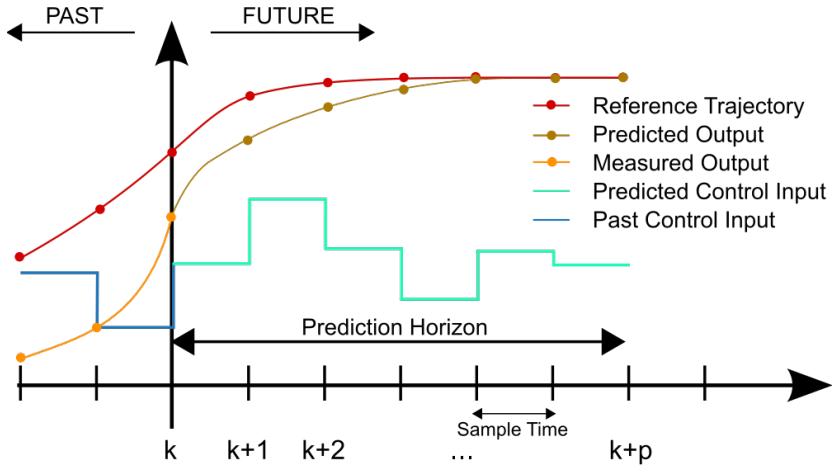


Figure 3-4: A presentation of the rolling horizon strategy that is used with MPC.

MPC is an effective method for traffic management because it combines several essential features, such as coordinating control actions, predicting their effects by the given prediction model, optimizing overall performance, utilizing feedback loops, and managing constraints. These properties make MPC particularly well-suited for complex traffic systems.

3-2-1 MPC Formulation for Coordinated Highway Traffic Control

Two critical elements for successful network control are coordination and prediction. MPC inherently incorporates both. Prediction is vital since the effects of control measures often take time to reach distant parts of the network. Additionally, in traffic systems, sometimes one have to deliberately reduce traffic flow in certain areas to prevent bigger issues, like congestion or gridlock, from occurring later. This might result in a temporary decrease in performance (for example, slowing traffic down temporarily) but leads to better performance overall, once the problem is resolved. An example of this is when traffic flow is intentionally slowed or limited to avoid gridlock, which would otherwise cause a major standstill. Once the congestion clears, the traffic can return to its usual speed and efficiency. MPC is naturally aligned with the challenges of traffic control. Most traffic models are based on discrete time, which fits well within the MPC framework, making it a practical tool for managing traffic networks.

Define N_p as the prediction horizon and N_c as the control horizon, with the condition that $N_c \leq \frac{N_p}{M}$, where M is a positive integer. This setup distinguishes the system simulation time step T from the control timescale $T_c = \frac{T}{M}$. In this framework, the system evolves M times faster than the controller, so the MPC algorithm is only executed once every M simulation steps rather than at every individual step. By allowing the system to evolve faster than the controller, the computational load is reduced because the optimization problem is solved less frequently, improving overall computational efficiency without sacrificing excessive control performance. Additionally, solving the control problem every M steps helps avoid abrupt changes in control actions. This smoothens transitions, leading to more stable traffic flows and minimizing potential disruptions caused by frequent control changes. After the optimization problem is solved, the first control action is applied for the following M simulation steps in a

rolling horizon manner. Control action indices on the MPC timescale are denoted by i_c, and are related to the system's time indices i by the relation

$$i_c(i) = \min\left\{\left\lfloor \frac{i}{M} \right\rfloor, N_c - 1\right\}. \tag{3-17}$$

The primary goal of traffic control is to enhance network performance, but this can be defined in various ways, leading to different objective functions. One of the most common objectives is minimizing the total time vehicles spend in the network (TTS), as it directly correlates with reducing the average travel time— a common goal in traffic management. This is described as

$$J_{\text{TTS}} = T\left(\sum_{m,i} \rho_{m,i}(j) L_m \lambda_m + \sum_{o} w_o(j)\right), \tag{3-18}$$

where $\rho_{m,i}(k)$ represents the vehicle density in segment i of link m at time step k, L_m denotes the length of a segment in link m, and λ_m refers to the number of lanes in link m. $w_o(k)$ corresponds to the number of vehicles queued at origin o.

After incorporating TTS as a key performance indicator in highway traffic control, it is essential to also consider control variability. Control variability refers to the rate at which control measures change over time. While minimizing TTS optimizes network efficiency, frequent or abrupt changes in control actions—such as ramp metering rates or speed limits—can cause drivers confusion, reduce compliance, and destabilize traffic flow. To address this, a control variability term can be added to the objective function, which penalizes large fluctuations in control actions. This results in smoother transitions, improved compliance of the driver, and enhanced traffic stability by reducing the risk of congestion. This is described as

$$J_{\text{VAR}} = \sum_{o \in O_{\text{ramp}}} \left(r_o(j) - r_o(j-1)\right)^2 + \sum_{(m,i) \in I_{\text{speed}}} \left(\frac{v_{\text{control},m,i}(j) - v_{\text{control},m,i}(j-1)}{v_{\text{free},m}}\right)^2, \tag{3-19}$$

where O_{ramp} represents the set of indices o corresponding to on-ramps where ramp metering is applied, and I_{speed} denotes the set of index pairs (m, i) for the links and segments where speed control is implemented.

The objective function is designed to balance these two objectives, with the first term targeting traffic efficiency and the second penalizing abrupt changes in control actions. This helps to enhance both the operational performance of the traffic network and the overall driving experience. The MPC objective can be expressed as follows:

$$\min_{\hat{u}_k, \hat{x}_k} \sum_{i=0}^{N_p} J_{\text{TTS}}(\hat{x}_{i|k}) + \eta \sum_{i=0}^{N_c-1} J_{\text{VAR}}(\hat{u}_{i|k}). \tag{3-20}$$

The weighting factor η balances the importance of optimizing traffic flow and ensuring smooth transitions in control actions. This trade-off is crucial because while minimizing TTS improves

traffic efficiency, control actions that change too abruptly can lead to driver discomfort, reduced compliance, and traffic instability.

The MPC controller begins with the current state of the traffic system, denoted by x_k, and uses this as the initial condition for predicting future states over a defined prediction horizon N_p:

$$\hat{x}_{0|k} = x_k.$$

It is assumed that the state x_k is fully measurable without noise. The predicted future states of the system evolve based on a state transition function $f(\cdot)$, which serves as prediction model for the MPC controller. This function accounts for the current state $\hat{x}_i|k$, control inputs $\hat{u}_i|k$, and any external disturbances d_k:

$$\hat{x}_{i+1,|k} = f(\hat{x}_{i|k}, \hat{u}_{i|k}, d_k), \quad i = 0, \ldots, N_p - 1.$$

The dynamics of the METANET model is used for describing the transition function. The states x_k, control inputs u_k, and disturbances d_k at time k can be compactly represented as vectors

$$x_k = \begin{bmatrix} \rho_{1,1}(k) \\ \rho_{1,2}(k) \\ \vdots \\ \rho_{1,n_i}(k) \\ \rho_{2,1}(k) \\ \vdots \\ \rho_{m,i}(k) \\ v_{1,1}(k) \\ v_{1,2}(k) \\ \vdots \\ v_{1,i}(k) \\ v_{2,1}(k) \\ \vdots \\ v_{m,n_i}(k) \\ w_1(k) \\ \vdots \\ w_{|O|}(k) \end{bmatrix}, \quad u_k = \begin{bmatrix} r_1(k) \\ \vdots \\ r_{|\mathcal{O}_{\text{ramp}}|}(k) \\ v_{\text{control},1}(k) \\ \vdots \\ v_{\text{control},|I_{\text{speed}}|}(k) \end{bmatrix}, \quad d_k = \begin{bmatrix} d_1(k) \\ \vdots \\ d_{|O|}(k) \end{bmatrix},$$

where O_{ramp} and I_{speed} are the sets of indices of ramps and speed limits, while $|O_{\text{ramp}}|$ and $|I_{\text{speed}}|$ represent their cardinality. Furthermore, n_i denotes for the number of segments for a given link m and $|O|$ denotes the number of origins for a given highway network. These variables are fed into the nonlinear traffic dynamics function f, which models the evolution of the traffic states over time. The decision variables in this optimization problem are the predicted control actions and states over both the control horizon N_c and the prediction horizon N_p, grouped together as:

$$\hat{u}_k = \left[\hat{u}_{0|k}, \ldots, \hat{u}_{N_c-1|k}\right] \in \mathbb{R}^{n_u \times N_c},$$

$$\hat{x}_k = \left[\hat{x}_{0|k}, \ldots, \hat{x}_{N_p|k}\right] \in \mathbb{R}^{n_x \times (N_p+1)},$$

where n_u is the number of control inputs and n_x is the number of states. At each time step, the system must respect a set of constraints, ensuring that the control actions and system states remain within safe and feasible bounds. These constraints could include limits on the states, ramp metering rates, and speed limits:

$$h(\hat{x}_{i|k}, \hat{u}_{i|k}) \leq 0, \quad i = 0, \ldots, N_p$$

In general, the states and the control actions are not allowed to take negative values. Furthermore, the ramp metering rate is always between 0 and 1. Lastly, the flow is not allowed to exceed the capacity of the network. All of these conditions, and others if needed, can be imposed as constraints in$h$. Overall, the final MPC formulation can be described as

$$\begin{aligned} \min_{\hat{u}_k, \hat{x}_k} \quad & \sum_{i=0}^{N_p} J_{\mathrm{TTS}}(\hat{x}_{i|k}) + \eta \sum_{i=0}^{N_c-1} J_{\mathrm{VAR}}(\hat{u}_{i|k}) \\ \text{s.t.} \quad & \hat{x}_{0|k} = x_k, \\ & \hat{x}_{i+1|k} = f(\hat{x}_{i|k}, \hat{u}_{i,c|k}, d_k), \quad i = 0, \ldots, N_p - 1, \\ & h(\hat{x}_{i|k}, \hat{u}_{i_c(i)|k}) \leq 0, \quad i = 0, \ldots, N_p, \end{aligned} \tag{3-21}$$

3-2-2 Modifications to the MPC Formulation

MPC is a powerful tool for optimizing control actions in dynamic systems. However, real-world applications, such as traffic management, often require modifications to address practical constraints and ensure safe and reliable operation. These modifications are necessary to handle safety-critical scenarios, enhance flexibility, and maintain feasibility during real-time operations.

Safety motivation

One of the primary motivations for modifying the MPC formulation is to manage safety-critical scenarios. In traffic control systems, for instance, it is essential to prevent conditions that may lead to unsafe or unstable behaviour, such as long vehicle queues at on-ramps. Excessive queue lengths can lead to gridlock that degrade overall network performance and create safety risks. To address this, additional constraints can be integrated into the MPC framework to ensure that the control actions prevent queue lengths from exceeding a critical threshold. For example, a constraint can be added to the optimization problem to ensure that the queue length $w_o(k)$ at on-ramp remains below a predefined maximum limit w_{max}, which can be described as

$$h_0(x_k, \sigma_k) := w_o(k) - w_{\max} - \sigma_k, \tag{3-22}$$

where σ_k is a slack variable to be included as additional optimization variable in the MPC optimization. This ensures that the system operates safely, without allowing congestion to build up in critical areas. Note that a soft constraint is applied here to provide flexibility in maintaining performance. Instead of strictly enforcing the maximum queue length $w_{\max}$, the slack variable σ_k allows for minor violations, which are penalized in the objective function. This ensures that the system can still operate efficiently, even under fluctuating traffic conditions, while discouraging excessive queue lengths. By penalizing violations rather than strictly forbidding them, the control system remains feasible and avoids suboptimal decisions during high-demand situations. Furthermore, for learning purpose, a mechanism is needed to both relax constraints during the learning process and penalise violations, in order to allow the agent to learn discerning between safe and unsafe policies. Note that this leads to an additional term in the cost function. Soft constraints are linked to a penalty function, which should be added to the objective function. In this minimization problem, the result is a trade-off between competing objectives: minimizing the primary objective while also reducing the penalties associated with constraint violations.

Ease numerical complexity

The MPC formulation can be more computationally simplified. Solving the current MPC formulation can be challenging, especially when using gradient-based solvers, as they can encounter difficulties due to the specific structure of the problem. Specifically, when solving the MPC, issues can arise from the min operators, described in Equation (3-7) and Equation (3-5). These operators can cause the gradient to become zero over large regions of the state-action space. When this happens, the gradient-based solver may struggle to find an optimal solution because it cannot sense which direction to move in to improve the performance. This problem is particularly significant when MPC is integrated into a RL framework, as the complexity of the control architecture increases, making optimization more difficult. To address this issue, the proposed control approach can be changed slightly.

First, instead of controlling the ramp metering rate $r_o(k)$, which indirectly influences the flow at the on-ramp $q_o(k)$, we directly control $q_o(k)$ (the flow at the on-ramp itself). To ensure that the MPC controller can always output a feasible control action for any on-ramp flow, additional constraints must be introduced. These constraints guarantee that for every flow value generated by the controller, there exists a corresponding metering rate that can achieve it. The third term in the minimum operator (Equation $(3-7)$) can represent the on-ramp flow limit based on a combination of $\rho_{\max}$, $\rho_j(k)$, and ρ_{crit}, which can be less or equal to C_0 under realistic traffic conditions. Moreover, since the on-ramp flow will always be less than or equal to the values determined by the minimum operator when the maximum ramp-metering rate is applied ($r_o(k) = 1$), the flow constraints can be simplified. Therefore, the separate term C_0 in the minimum operator can be disregarded. This leads to the introduction of the following additional constraints:

$$h_1(x_k, u_k) := q_o(k) - d_o^w(k) - \frac{w_o(k)}{T}, \tag{3-23}$$

$$h_2(x_k, u_k) := q_o(k) - C_o \left(\frac{\rho_{\max} - \rho_j(k)}{\rho_{\max} - \rho_{\text{crit}}} \right). \tag{3-24}$$

As for the VSL, $V_{\text{VSL}}(\rho_{m,i}(k))$ can be directly controlled. Controlling $v_{\text{control}}(k)$ is equivalent to controlling $V_{\text{VSL}}(\rho_{m,i}(k))$, provided that the constraint $V_{\text{VSL}}(\rho_{m,i}(k)) \leq V(\rho_{m,i}(k))$ is satisfied. In other words, if this constraint holds, for any given value of $V_{\text{VSL}}(\rho_{m,i}(k))$, the corresponding equivalent value of $v_{\text{control}}(k)$ can always be determined. Thus the constraint is

$$h_3(x_k, u_k) = V_{\text{VSL}}(\rho_{m,i}(k)) - V(\rho_{m,i}(k)). \tag{3-25}$$

These modifications simplify the control problem by bypassing two min operators, reducing the chances of encountering numerical issues and making the overall optimization process more efficient. The rest of the dynamics remain unchanged. Now, the action can be structured as

$$u_k = \begin{bmatrix} q_{0,1}(k) & q_{0,2}(k) & \dots & q_{0,|O_{\text{ramp}}|} & V_{\text{VSL},1}(k) & V_{\text{VSL},2}(k) & \dots & V_{\text{VSL},|I_{\text{speed}}|}(k) \end{bmatrix}^\top.$$

Note that implementing these modifications lead also to a change to the cost of the control variability:

$$\begin{aligned} J_{\text{VAR}} &= \sum_{o \in O_{\text{ramp}}} \left(\frac{q_o(j) - q_o(j-1)}{C_o} \right)^2 + \sum_{(m,i) \in I_{\text{speed}}} \left(\frac{V_{\text{VSL}}(\rho_{m,i}(k))(j) - V_{\text{VSL}}(\rho_{m,i}(k))(j-1)}{v_{\text{free},m}} \right)^2, \\ &= J_{\text{VAR, ramp}} + J_{\text{VAR, speed}}. \end{aligned} \tag{3-26}$$

The new MPC formulation retains its equivalence to the original but is easier to solve numerically. For clarity, the additional constraints can be ordered as $h(x_k, u_k, \sigma_k) = \begin{bmatrix} h_0(x_k, \sigma_k) & h_1(x_k, u_k) & h_2(x_k, u_k \end{bmatrix}$ As a result, the final MPC formulation becomes:

$$\begin{aligned} \min_{\hat{u}_k, \hat{x}_k, \sigma} \quad & \sum_{i=0}^{N_p} J_{\text{TTS}}(\hat{x}_{i|k}) + \eta \sum_{i=0}^{N_c-1} J_{\text{VAR}}(\hat{u}_{i|k}) + \zeta^\top \sigma, \\ \text{s.t.} \quad & \hat{x}_{0|k} = x_k, \\ & \hat{x}_{i+1|k} = f(\hat{x}_{i|k}, \hat{u}_{i,c|k}, d_k), \quad i = 0, \dots, N_p - 1, \\ & h(\hat{x}_{i|k}, \hat{u}_{i,c|k}, \sigma_k) \leq 0, \quad i = 0, \dots, N_p, \\ & \sigma \geq 0, \end{aligned} \tag{3-27}$$

where $\sigma = \begin{bmatrix} \sigma_0 & \dots & \sigma_{N_p} \end{bmatrix}^\top$ is the collection of slack variables along the prediction horizon, and $\zeta \in \mathbb{R}^{N_p+1}$ is the corresponding vector of positive weights for the slack penalty.

3-3 MPC-based RL for RM and VSLs

This section discusses how MPC can be integrated with RL. In model-based RL, MPC can be employed as a function approximator [56]. In this setup, the MPC controller acts as a policy, mapping the current state of the system to an optimal action based on predefined performance criteria, and it acts also as function approximation for the value functions of the underlying control problem. The RL algorithm, in turn, adjusts the parameters of the MPC controller through gradient-based methods, with the goal of discovering the optimal control policy that maximizes long-term performance. Techniques like sensitivity analysis can be used to enable RL methods, such as Q-learning or policy gradients, by differentiating the MPC problem with respect to its parameters and updating them iteratively.

MPC-based RL has several advantages over deep neural network-based approaches, as it leverages the structure of the underlying optimization problem, enhancing guarantees on stability and safety features. However, maintaining safety and stability remains a challenge. If the RL updates are not carefully controlled, the MPC parameters could be driven to unsafe configurations, leading to constraint violations and suboptimal performance [17].

The system to be controlled (e.g., the METANET model in this work) can be described as a discrete Markov-Decision Process (MPD) having the (possibly) stochastic state transition dynamics

$$\mathbb{P}[s_+ \mid s, a], \tag{3-28}$$

where s, a is the current state-input pair and s_+ describing the next state. The goal of the RL algorithm is to find the optimal policy for the system, which is described as

$$\pi_\theta^* = \arg\min_\theta J(\pi_\theta), \tag{3-29}$$

where $J(\pi_\theta)$ is the performance for a given policy π_θ parametrized in $\theta \in \mathbb{R}^{n_\theta}$ (in this work, this policy is provided by the MPC controller). For performance, it is essential in highway traffic control to minimize the stage cost for a given state-action pair. The more effectively the RL algorithm can minimize the cost accumulated at each time step, the better it performs in terms of reducing congestion and improving traffic flow. Therefore, the performance is described as

$$J(\pi_\theta) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L\left(s_k, \pi_\theta(s_k)\right)\right], \tag{3-30}$$

where $L(s_k, \pi_\theta(s_k))$ is the stage cost function for a given policy $\pi_\theta(s_k)$ and $\gamma \in [0, 1)$ is the discount factor.

In order to find the optimal policy π_θ^* that dictates the best action to take in every possible state to maximize the expected reward over time, the optimization problem (3-29) should be solved. Note that, in general, the true unknown optimal value functions and policy are impossible to find and characterise. To tackle this problem, function approximation techniques can be considered [57]. The key advantage is that approximation can be made for π_θ, Q_θ, and

V_θ, which are used to solve the minimization problem via iterative gradient updates of the parametrization. This can be solved directly via the policy gradient methods, which explicitly parametrise and learn the approximation π_θ of the optimal policy itself. Briefly, these method try to estimate the gradient of the objective, described in Equation (3-30), with respect to the learnable parameters θ, and updates the learnable parameters using gradient descent

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta). \tag{3-31}$$

Instead of directly optimizing the policy π_θ, value-based methods focus instead on estimating value functions (V_θ, Q_θ) from the underlying RL task. By learning these value approximations, the optimal policy can be then recovered indirectly. This will be further discussed in Section 3-3-3.

3-3-1 The RL task

To effectively implement an RL algorithm for task of controlling RM and VSL actuators, it is essential to first establish a suitable stage cost function that can evaluate the performance of a policy in managing effectively the traffic process, which is described in Equation (3-30). This is decribed as:

$$L(x_k, u_k) = c_{\mathrm{TTS}} J_{\mathrm{TTS}}(x_k) + c_{\mathrm{VAR}} J_{\mathrm{VAR}}(x_k) + c_{\mathrm{CV}} J_{\mathrm{CV}}(x_k), \tag{3-32}$$

where $J_{\mathrm{CV}}(x_k)$ is defined as:

$$J_{\mathrm{CV}}(x_k) = \max\left\{0, w_o(k) - w_{\max}\right\}. \tag{3-33}$$

In Equation (3-32), the first and second terms encourage low TTS and control variability respectively, as described in Section 3-2-2, while the third term penalizes any violations of the queue length constraint, encouraging the agent to develop control policies that respect this limit. Note that the weights c_{TTS}, c_{VAR} and c_{CV} are included that balance each term differently. It is ultimately the task of the designer to select these values in a way that effectively incorporates the learning objective into the RL stage cost.

3-3-2 MPC as Function Approximator in RL

In MPC, parametric approaches involve defining adjustable parameters, such as weights in the cost function or constraints on states and inputs, that influence the optimization problem. These parameters play a critical role in ensuring that the controller adapts effectively to dynamic environments. However, fine-tuning these parameters manually is challenging, particularly in complex traffic systems modeled such as METANET, due to their nonlinear and high-dimensional nature. This difficulty in fine-tuning motivates the use of RL as an adaptive mechanism to adjust the MPC parameters automatically. By learning from real-time data, RL can continuously refine the performance of the MPC, eliminating the need for manual intervention in complex closed-loop systems.

To do this, the cost function, the prediction model and some of the constraints should first be parameterized. The overall goal is to enhance the closed-loop performance (3-30), and this can be done by the RL algorithm, which refines the parameter settings by utilizing the observed state transitions and corresponding rewards. The parametrization θ of the MPC controller that is used in this thesis is the following:

$$\begin{aligned}\min_{\hat{u}_k,\hat{x}_k,\sigma} \quad & \sum_{i=0}^{N_p}\gamma^i\Big(\theta_T J_{\text{TTS}}(\hat{x}_{i|k})+\theta_{C,i}\sigma_i\Big)+\sum_{i=0}^{N_c-1}\gamma^{Mi}\Big(\theta_{V,\text{ramp}}J_{\text{VAR,ramp}}(\hat{u}_{i|k}) \\ & +\theta_{V,\text{speed}}\frac{v_{\text{free}}^2}{\tilde{v}_{\text{free}}^2}J_{\text{VAR,speed}}(\hat{u}_{i|k})\Big) \\ & +\lambda_\theta(\hat{x}_{0|k})+\sum_{i=1}^{N_p-1}\gamma^i\ell_\theta(x_k)+\gamma^{N_p}\Gamma_\theta(x_{N_p}), \end{aligned} \tag{3-34a}$$

$$\text{s.t.} \quad \hat{x}_{0|k}=x_k, \tag{3-34b}$$

$$\hat{x}_{i+1|k}=f_\theta(\hat{x}_{i|k},\hat{u}_{i_c(i)|k},d_k), \quad i=0,\dots,N_p-1, \tag{3-34c}$$

$$h_\theta(\hat{x}_{i|k},\hat{u}_{i_c(i)|k},\sigma_k)\le 0, \quad i=0,\dots,N_p, \tag{3-34d}$$

$$\sigma\ge 0. \tag{3-34e}$$

Note that the MPC parameters θ_T, θ_V, and θ_C are distinct from the RL stage cost weights c_{TTS}, c_{VAR}, and c_C in Equation (3-32). Adding additional learnable cost terms to the MPC objective increases the flexibility and degree of freedom of the function approximation, allowing the controller to better handle complex, dynamic systems. This flexibility helps balance competing objectives, such as J_{TTS} and J_{VAR}, while ensuring stability and compliance with constraints. Furthermore, these learnable cost functions enable the controller to approximate the true system costs, which are often uncertain or non-linear, by adjusting parameters during learning. The initial learnable cost function is considered as

$$\lambda_\theta(x_k)=\sum_{m,i}\left(\theta^\rho_{\lambda,m,i}\frac{\rho_{m,i}(k)}{\rho_{\max}}+\theta^v_{\lambda,m,i}\frac{v_{m.i}(k)}{v_{\max}}\right)+\sum_o\theta^w_{\lambda,o}\frac{w_o(k)}{w_{\max}}, \tag{3-35}$$

where $\rho_{\max}$, $v_{\max}$ and $w_{\max}$ are here used as fixed, non-learnable normalization constants in order to improve the stability of the learning process. For the stage and terminal costs, Huber Loss functions are considered. The goal is to ensure that densities and speeds closely track the fixed, non-learnable setpoints ρ_{sp} and v_{sp}, while minimizing queue lengths as much as possible. Importantly, deviations from the setpoints are not penalized symmetrically. High densities and low speeds are penalized more heavily than low densities and high speeds. To achieve this, the left-sided Huber Loss function ϕ_l is applied to track the density setpoint, while the right-sided Huber Loss function ϕ_r is used to track the speed setpoint. A detailed definition of the Huber Loss function can be found in Appendix A . Accordingly, the stage and terminal costs are defined as

$$\ell_\theta(x_k) = \sum_{m,i} \left(\theta^{\rho}_{\ell,m,i} \phi_l(\rho_{m,i}(k) - \rho_{\text{sp}}; \theta^{\rho}_{\ell,\text{lin}}) + \theta^{v}_{\ell,m,i} \phi_r(v_{m,i}(k) - v_{\text{sp}}; \theta^{v}_{\ell,\text{lin}}) \right) + \sum_o \theta^{w}_{\ell,o} \left(\frac{w_o(k)}{w_{\max}} \right)^2 \tag{3-36}$$

$$\Gamma_\theta(x_k) = \sum_{m,i} \left(\theta^{\rho}_{\Gamma,m,i} \phi_l(\rho_{m,i}(k) - \rho_{\text{sp}}; \theta^{\rho}_{\Gamma,\text{lin}}) + \theta^{v}_{\Gamma,m,i} \phi_r(v_{m,i}(k) - v_{\text{sp}}; \theta^{v}_{\Gamma,\text{lin}}) \right) + \sum_o \theta^{w}_{\Gamma,o} \left(\frac{w_o(k)}{w_{\max}} \right)^2 \tag{3-37}$$

The transition function f_θ mentioned in Equation (3-34) include three learnable parameters: $\tilde{\rho}_{crit}$, $\tilde{a}$, and $\tilde{v}_{free}$. This approach enables the agent to partially tune the prediction model based on performance metrics (3-30) rather than solely relying on system identification. Consequently, the learned parameters may diverge from the values of the true underlying dynamics. Additionally, the constraints h_θ is also parametrized, since $\tilde{\rho}_{crit}$ also appear in the constraint Equation (3-24). The whole parametrization is summarized in Table 3-1.

Learnable parameters	Scope	Space
$\tilde{\rho}_{\text{crit}}$	prediction model, cost, constraint	$\mathbb{R}$
$\tilde{a}$	prediction model	$\mathbb{R}$
$\tilde{v}_{free}$	prediction model	$\mathbb{R}$
θ_T	cost - TTS weight	$\mathbb{R}$
$\theta_{V,\text{ramp}}$	cost - control variability weight for ramp flow	$\mathbb{R}^{\lvert O_{\text{ramp}} \rvert}$
$\theta_{V,\text{speed}}$	cost - control variability weight for equilibrium speed	$\mathbb{R}^{\lvert I_{\text{speed}} \rvert}$
θ_C	cost - slack weights	$\mathbb{R}^{N_p}$
$\theta^{\{\rho,v\}}_{\{\lambda,\ell,\Gamma\}}$	{initial, stage, terminal} cost - $\{\rho, v\}$ weights	$\mathbb{R}^{\sum_{i=1}^{m} n_i}$
$\theta^{w}_{\{\lambda,\ell,\Gamma\}}$	{initial, stage, terminal} cost - w weights	$\mathbb{R}^{\lvert O \rvert}$
$\theta^{\{\rho,v\}}_{\{\ell,\Gamma\},\text{lin}}$	{stage, terminal} Huber Loss cost - $\{\rho, v\}$ linear slope value	$\mathbb{R}^{\sum_{i=1}^{m} n_i}$

Table 3-1: parametrization θ of the MPC function approximation described by Equation (3-34)

Finally, the parametrized vector θ can be considered as

$$\theta = \begin{bmatrix} \tilde{\rho}_{\text{crit}} \\ \tilde{a} \\ \tilde{v}_{\text{free}} \\ \theta_{\{T, V_{\{\text{ramp,speed}\}}, C\}} \\ \theta^{\{\rho,v,w\}}_{\{\lambda,\ell,\Gamma\}} \\ \theta^{\{\rho,v\}}_{\{\ell,\Gamma\},\text{lin}} \end{bmatrix}. \tag{3-38}$$

The parametrized MPC scheme described in Equation (3-34) can be considered as value function V [56]:

$$\begin{aligned} V_\theta(x_k) = \min_{\hat{u}_k, \hat{x}_k, \sigma} \quad & \text{(3-34a)} \\ \text{s.t.} \quad & \text{(3-34b) – (3-34e)} \end{aligned}, \tag{3-39}$$

followed by the Q-function and policy defined as

$$\begin{aligned} Q_\theta(x_k, u_k) = \min_{\hat{u}_k, \hat{x}_k, \sigma} \quad & \text{(3-34a)} \\ \text{s.t.} \quad & \text{(3-34b) – (3-34e)}, \\ & \hat{u}_{0|k} = u_k. \end{aligned} \tag{3-40}$$

$$\pi_\theta(x_k) = \arg\min_u \, Q_\theta(x_k, u) \tag{3-41}$$

As discussed further in the next subsection, the optimal action received by solving Equation (3-39) is used as initial condition $\hat{u}_{0|k}$ when solving Equation (3-40). In order to find the optimal policy π^* by the MPC scheme, θ should be rich enough [56]. As consequence, many combinations of parametrization choice will be possible to yield a valid solution.

3-3-3 Second-Order LSTD Q-learning

Linear least-squares techniques have been effectively applied to prediction tasks in reinforcement learning. While they do not offer the same level of generalization as more complex methods like deep RL algorithms, they are significantly simpler to implement and troubleshoot [58]. Additionally, Least Square Temporal Difference (LSTD) Q-learning make an efficient use of data and tend to converge faster than more basic temporal-difference learning methods.

In order to use this algorithm, one seeks an indirect (or value-based) methods that aim to find the optimal policy by first estimating the state- and action-value functions V_θ, Q_θ from the underlying RL task. The policy is then implicitly derived from these value functions. The objective of Q-learning can then be expressed as minimizing the least squares of the Bellman residual error with respect to θ:

$$\min_\theta \mathbb{E}\left[\|Q^\star(s,a) - Q_\theta(s,a)\|^2\right], \tag{3-42}$$

where Q_θ, obtained from the MPC scheme in Equation (3-40), serves as an approximation of the true unknown action-value function $Q^\star$. In this equation, the goal is to find the parametrization that best aligns the action-value function Q_θ with the observed data. By approximating the unknown optimal $Q^\star$ through this process, the aim is to indirectly derive the optimal policy. Consider an approximation of the Bellman optimality equation, which is commonly used in Temporal Difference (TD)-based learning approaches:

$$Q^\star(s,a) \approx L(s,a) + \gamma Q_\theta(s_+, \pi_\theta(s_+)) \approx L(s,a) + \gamma V_\theta(s_+), \tag{3-43}$$

where $L(s,a)$ denotes the stage cost in the reinforcement learning (RL) scheme. By substituting Equation (3-43) into Equation (3-42), the least squares problem for first-order LSTD Q-learning can be solved using data collected from the transitions $s_i, a_i \to s_{i+1}$ as follows:

$$\mathbb{E}\left[\delta \nabla_\theta Q_\theta(s_i, a_i)\right] = 0, \tag{3-44}$$

$$\delta = L(s_i, a_i) + \gamma V_\theta(s_{i+1}) - Q_\theta(s_i, a_i), \tag{3-45}$$

where δ is the temporal difference error (TD-error). Both value and action-value functions, respectively are obtained from the MPC formulations. At last, a Newton's method can be used to enhance Equation (3-45) as first order solution to Equation (3-42). Leveraging second-order Newton's method, combined with an experience replay buffer that stores past transitions [59], and reformulating the Q-fitting problem as a least-squares optimization, leads to improved convergence speed and greater sample efficiency compared to standard first-order approaches. The second order LSTD Q-learning scheme can be described as:

$$\theta \leftarrow \theta - \alpha A^{-1} b, \tag{3-46}$$

$$A = \mathbb{E}\left[\delta \nabla_\theta^2 Q_\theta + \nabla_\theta Q_\theta \nabla_\theta^\top \delta\right], \quad b = \mathbb{E}\left[\delta \frac{\partial Q_\theta}{\partial \theta}\right], \tag{3-47}$$

where $\alpha \geq 0$ is the learning rate and b is the gradient, and A the Hessian. Evaluating the gradient and Hessian of Q_θ can be done by introducing the Lagrange function $\mathcal{L}_\theta$ corresponding to the MPC optimization problem (3-40). The Lagrange function combines the cost of the MPC problem and the constraints. In simplified terms, the Lagrange function is described as

$$\mathcal{L}_\theta = \Phi_\theta + \boldsymbol{\lambda}^\top G_\theta + \boldsymbol{\mu}^\top H_\theta, \tag{3-48}$$

where Φ_θ is the cost function for the MPC, and $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are Lagrange multipliers vectors that help enforce constraints. Specifically, $\boldsymbol{\lambda}$ is associated with equality constraints, while $\boldsymbol{\mu}$ is associated with inequality constraints. G_θ and H_θ represent the equality and inequality constraints of Equation (3-40), respectively.

Let the primal variables (state and control inputs) be labeled as $\mathbf{p} = \{\mathbf{X}, \mathbf{U}\}$, where $\mathbf{X}$ represents the state trajectories and $\mathbf{U}$ represents the control input trajectories. The combined primal-dual variables are represented as $\mathbf{z} = \{\mathbf{p}, \lambda, \mu\}$, with λ and μ being Lagrange multipliers for equality and inequality constraints, respectively. To compute the gradient $\nabla_\theta Q_\theta$ and Hessian $\nabla_\theta^2 Q_\theta$ required for optimization in Equation (3-47), a sensitivity analysis is conducted [60, 61]. The gradient of Q_θ with respect to θ is derived from the Lagrange function $\mathcal{L}_\theta(s, a, \mathbf{z}^\star)$, where $\mathbf{z}^\star$ is the optimal solution for the problem. This gives:

$$\frac{\partial Q_\theta}{\partial \theta} = \frac{\partial \mathcal{L}_\theta(s, a, \mathbf{z}^\star)}{\partial \theta} \tag{3-49}$$

Additionally, the Hessian is calculated as follows:

$$H(Q_\theta) = \frac{D}{D\theta}\left(\frac{\partial \mathcal{L}_\theta(s, a, \mathbf{z}^\star)}{\partial \theta}\right) \approx \frac{\partial^2 \mathcal{L}_\theta(s, a, \mathbf{z}^\star)}{\partial \theta^2}, \tag{3-50}$$

where $\mathbf{z}^\star$ is the primal-dual solution of Equation (3-40) and D is the total derivative.

As for the update for the learnable parameters, one can impose lower and upper bounds on the parametrization to enhance stability and ensures that the parameters change incrementally while remaining within a range that foster stability of the learning process. Note that the Hessian should be modified to a semi-postive definite matrix, since it is necessary to guarantee

that the optimization process will converge to a minimum. This is achieved by incrementally adding identity matrices to the Hessian matrix until it becomes numerically feasible for Cholesky decomposition [62]. Therefore, the update rule for the learnable parameters can be described as

$$\begin{aligned} \Delta\theta^\star = \arg\min_{\Delta\theta} \quad & \tfrac{1}{2}\Delta\theta^\top \hat{H}\Delta\theta + \alpha b^\top \Delta\theta \\ \text{s.t.} \quad & \theta_{\text{lb}} \leq \theta + \Delta\theta \leq \theta_{\text{ub}} \\ & \Delta\theta_{\text{lb}} \leq \Delta\theta \leq \Delta\theta_{\text{ub}}, \end{aligned} \tag{3-51}$$

where $\hat{H}$ is the modified semi-definite Hessian. This formulation limits the rate of change for each parameter using the bounds $\Delta\theta_{\text{lb}}$ and $\Delta\theta_{\text{ub}}$. The parameters are then updated according to the rule $\theta \leftarrow \theta + \Delta\theta^\star$.

To prevent the Q-learning agent from getting stuck in very suboptimal local minima, one can add enough exploration on top of the policy. It can therefore discover better actions and improve its long-term performance. This can be done by adding a perturbation term to the MPC objective (3-34a), which can be described as $q^\top u_0$, where q is randomly chosen from a normal distribution.

Chapter 4

Case Study: Using MPC as Function Approximation in RL for Highway Traffic Control

This chapter focuses on applying the proposed method for highway traffic control to a benchmark highway network, by using MPC as function approximation for RL. The goal is to evaluate the effectiveness of the algorithm in managing traffic flow and reducing congestion. This chapter details the setup of the highway network benchmark, the configuration of the learning-based controller and algorithm, and the results obtained from the simulations.

4-1 Setup

This section presents the algorithmic framework for integrating MPC with RL to enhance highway traffic control. The approach leverages predictive capabilities of MPC and adaptability of RL to dynamically optimize traffic flow and improve performance.

4-1-1 Algorithm Approach

The foundation of this framework is a macroscopic traffic model (METANET) that simulates highway dynamics, including vehicle flow, density, and speed across multiple segments. The MPC serves as both control methodology and function approximation within the RL framework. It computes optimal control actions by minimizing the cost function over a finite prediction horizon. The RL component updates the MPC learnable parameters by using the second-order LSTD Q-learning method. Here is how the algorithm works step-by-step:

1. The traffic model is initialized with baseline configurations, including steady-state conditions, control inputs, and demand profiles. Additionally, the learnable MPC parameters

are initialized to reasonable but poorly tuned values, e.g., the MPC prediction model is different from the real traffic model (as explained later in Section 4-1-3).

2. The value function V_θ is solved for the initial state, initial learnable MPC parameters and control action, yielding the first optimal action and corresponding objective value, before the learning episodes begin.

3. The learning loop begins. The optimal action derived from the value function is applied to the environment, resulting in the new state and the RL stage cost.

4. The first action-value function Q_θ is solved using the current state and the current values of the learnable MPC parameters. Additionally, a_0 is updated based on the action determined by the value function at the current time step. This step is crucial for computing the objective value of the action function and its sensitivities.

5. The value function V_θ for the next time step is computed using the new state and the action derived from the value function of the current time step. This yields the optimal action for the next time step and the objective value of the next time step.

6. The computed objective values and the RL stage cost are used to calculate the TD error.

7. The learnable parameters are updated using the TD error and the sensitivities.

8. Steps 3 through 7 are repeated until the learning episode concludes.

The nonlinear optimisation problems were formulated and solved with the CasADi framework. IPOPT (Interior Point OPTimizer) is used as a Non-linear Programming solver to solve the optimization problem.

4-1-2 Network Environment and Configuration

A simplified highway traffic network is used as benchmark, adapted from [1], which serves as the RL environment for this study. The network comprises two origins (a mainstream and an on-ramp), two highway links, and one destination. The main origin, O_1, features two lanes with a total capacity of 2000 veh/h/lane. This is followed by the highway link L_1, which also has two lanes and spans 4 km, divided into four 1 km segments. The third and fourth segment of link L_1 consists of VSLs. At the end of L_1, a single-lane on-ramp O_2 with a capacity of 2000 vehicles per hour merges with the mainline. After the on-ramp, the network transitions to link L_2, which consists of two lanes divided into two 1-km segments and terminates at destination D_1, where outflow is open and uncongested . The queue length at O_2 is restricted to a maximum of 100 vehicles to ensure smooth traffic operations on adjacent roads. The network is depicted in Figure 4-1.

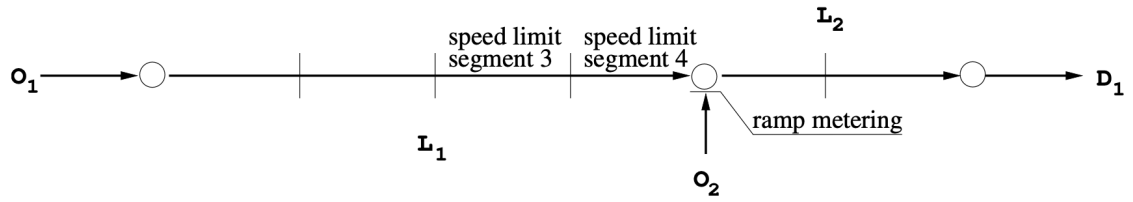


Figure 4-1: Network that consist RM and VSLs [1]

The METANET network parameters are adopted from [54] (see Table 4-1).

Table 4-1: Traffic model parameters for the METANET model

Parameter	Initial Value	Unit
T	10	s
τ	18	s
κ	40	veh/km/lane
η	60	km^2/h
ρ_{max}	180	veh/km/lane
δ	0.0122	-
a	1.867	-
v_{free}	102	km/h
ρ_{crit}	33.5	veh/km/lane

Note that the parameters in Table 4-1 are the parameters used to simulate the real evolution of the METANET dynamics, and are fixed. Conversely, in the MPC prediction model, a, v_{free} and ρ_{crit} are made learnable, i.e., are adjusted by the RL algorithm. To simulate a more challenging learning scenario, these parameters in the MPC model are initialized with a deviation with respect to the real ones in the METANET dynamics (which will be further discussed in Section 4-1-3).

Demand Scenario

To investigate the impact of combining RL and MPC for highway ramp metering and coordinated highway traffic control (including VSLs), a representative traffic demand scenario is considered. In this scenario, the mainline traffic flow starts at a consistently high level, then decreases sharply after 120 minutes, reaching a lower, steady rate. Meanwhile, the on-ramp demand rises, remains stable for 15 minutes, and then declines to a constant low rate. Since RL often struggles with overfitting to specific demand patterns or scenarios, stochasticity is added to the demand profiles. stochasticity acts as a regularizer, forcing the agent to generalize its policy to handle both typical and atypical demand fluctuations. The demand at each time step is perturbed by stochasticity from a normal distribution with zero mean and standard deviation proportional to the demand value, calculated as 20% of the original demand. After adding the stochasticity, the data is smoothed using a third-order Butterworth filter applied with the forward-backward filtering method to preserve the phase of the signals

characteristics. Finally, a non-negativity constraint is enforced, ensuring that all demand values remain zero or positive. Figure 4-2 shows the distribution of these demand profiles.

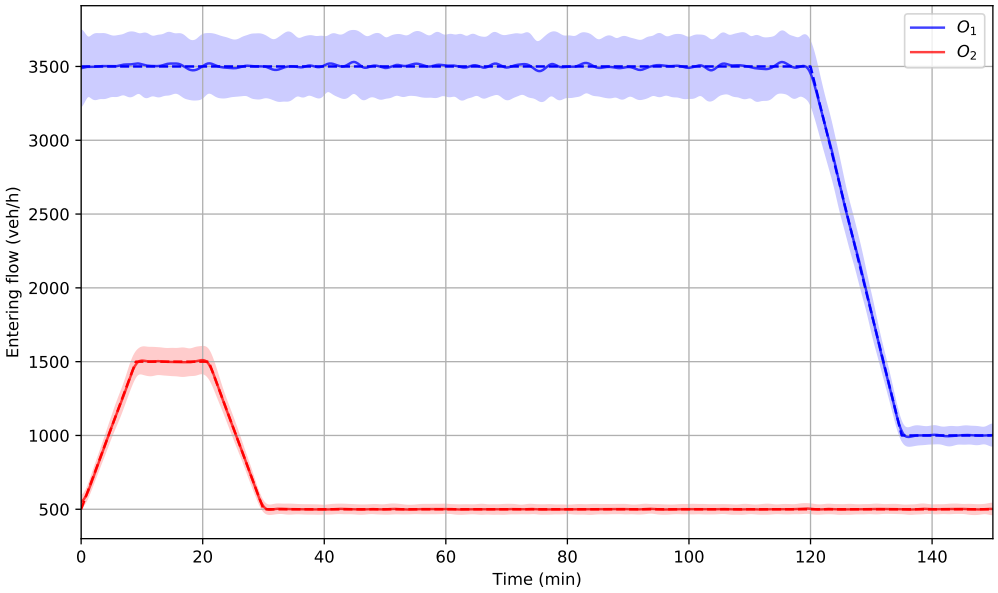


Figure 4-2: The demand scenario considered in the simulation experiments. The dashed lines represent the original deterministic demand profiles, while the solid lines represent the mean of the noisy demand profiles after applying stochasticity and filtering. The shaded regions represent the standard deviation ranges from which the random demands are sampled from.

4-1-3 MPC parameters

The controller (3-34) is employed to manage the control actions, with the primary objective of preventing congestion within the traffic network. The sampling interval for the process dynamics is set to 10 seconds, while the controller operates every 60 seconds ($M = 6$). Hence, the sampling time of the controller is equal to 1 minute. The prediction horizon is defined as $N_p = 7$, and the control horizon is set to $N_c = 5$.

The initial values of the objective function parameters for the MPC optimization problem are left untuned, simulating a design process with limited expertise and poor identification phase. Furthermore, to evaluate the influence of RL on the performance of the MPC controller, the parametrization of the controller is initialized with suboptimal values, introducing a 30% error relative to their true values. This approach mimics the effects of an imperfect system identification process. The initialization focuses on the MPC learnable parameters that directly impact the performance, specifically:

$$\begin{cases} \tilde{\rho}_{\text{crit}} = 0.7\rho_{\text{crit}}, \\ \tilde{a} = 1.3a, \\ \tilde{v}_{\text{free}} = 1.3v_{\text{free}}. \end{cases}$$

The initial values of the learnable parameters introduced by Equation (3-34) are summarized in Table 4-2.

Table 4-2: Initial values, dimensions, and parameters of the parametrization.

Symbol	Initial Value	Bounds	Dimension
$\tilde{\rho}_{\text{crit}}$	$0.7\rho_{\text{crit}}$	$[10, 180]$	veh/km/lane
$\tilde{a}$	1.3a	$[1.1, 6]$	-
$\tilde{v}_{free}$	$1.3v_{\text{free}}$	$[10^{-3}, 150]$	km/h
θ_T	1	$[10^{-3}, \infty)$	$\text{veh}^{-1}\text{h}^{-1}$
$\theta_{V,\text{ramp}}$	500	$[10^{-3}, \infty)$	veh^2/h^2
$\theta_{V,\text{speed},\{1,2\}}$	50	$[10^{-3}, \infty)$	km^2/h^2
θ_C	5	$[10^{-3}, \infty)$	veh^{-1}
$\theta^{\{\rho,v,w\}}_{\{\lambda\}}$	1	(∞, ∞)	-
$\theta^{\{\rho,v,w\}}_{\{\ell,\Gamma\}}$	1	$[10^{-3}, \infty)$	-
$\theta^{\{\rho,v\}}_{\{\ell,\Gamma\},\text{lin}}$	2	$[10^{-3}, \infty)$	-

The remaining fixed parameters in Equation (3-34) are assigned the same values as those used in the real system. At last, the setpoints and the normalization coefficients are described in Table 4-3.

Table 4-3: Setpoints and normalization coefficients for the parametrized MPC scheme

Symbol	Value	Dimension
ρ_{sp}	$0.7\rho_{\text{crit}}$	veh/km/lane
v_{sp}	$1.3v_{\text{free}}$	km/h
$\rho_{\max}$	180	veh/km/lane
$w_{\max}$	100	veh
$v_{\max}$	$1.3v_{\text{free}}$	km/h

4-1-4 RL parameters

A single simulation episode lasts 150 minutes and the RL agent is trained for 80 episodes. Note that at the end of each episode and when a new episode starts, new random demands are generated (as described in Section 4-1-2) and the state is reset to steady-state. The RL hyperparameters that were considered to tune are described in Table 2-1.

The weight c_{TTS} is assigned to the term $J_{\text{TTS}}(x_k)$, while the weight c_{VAR} includes $c_{V,\text{ramp}}$ and $c_{V,\text{speed}}$, which are assigned to the term $J_{\text{VAR}}(x_k)$. Here, $c_{\text{V,ramp}}$ accounts for the variability in each on-ramp flow, whereas $c_{\text{V,speed}}$ addresses the variability in each equilibrium speed action. The weight c_{CV} is assigned to the penalty term $J_{\text{CV}}(x_k)$, ensuring compliance with queue length constraints at on-ramps.

As for the update of the learnable parameters θ, the maximum change of each parameter is equal to 30% of its current value, hence $\Delta\theta = 0.3\theta$. The learnable parameters of the MPC are updated using a second-order LSTD Q-learning that relies on information stored in a buffer, whose size corresponds to the number of times the MPC is solved within a single episode. At

each timestep, the Q-function is evaluated for the current state-action pair, and its gradient (b), Hessian (A) and corresponding TD error are stored in the buffer. Before a parametrization update (3-51) is carried out, these terms are then averaged over the buffer to stabilize the updates and mitigate the effects of noisy gradients or outliers. The update of θ happens at the end of each learning episode. The learnable parameters are incrementally updated to minimize the TD error. To ensure numerical stability and feasibility during optimization, the Hessian term A is made positive semi-definite (PSD) using Cholesky factorization with added regularization. Finally, the learning rate (α) is decayed after each update, reducing the step size as training progresses to stabilize convergence. In this case study, different learning rates are considered for the learnable parameters. Specifically, the learning rate for $\tilde{\rho}_{\text{crit}}$ ($\alpha_{\rho_{\text{crit}}}$) and the learning rate for $\tilde{a}$ (α_a) are distinct from each other and also differ from the learning rate assigned to the remaining learnable parameters (α_θ). However, all parameters share the same learning rate decay α_{decay}, ensuring a consistent reduction in step size over time.

To facilitate exploration during training, the objective function of the MPC framework is perturbed with random stochasticity. This happens when solving the value function, described in Equation (3-39). Specifically, a perturbation term, $q^\top$, is introduced, which is randomly generated, multiplied by the first control action, u_0, and added to the objective function. The resulting product is further scaled by the maximum control actions, ensuring that the magnitude of the perturbation remains proportional to the scale of the control inputs. During each training step, random stochasticity is independently generated for the on-ramp flow and the VSLs. The stochasticity is drawn from a normal distribution with zero mean and a standard deviation that decays over time. The exploration strength is determined by the standard deviation, where $\sigma_{\text{q,ramp}}$ determines the exploration strength for the on-ramp flow action, and $\sigma_{q,\text{speed}}$ determines the exploration strength for the desired speed control action. For the on-ramp flow, the stochasticity is scaled by the ramp's capacity, while for the VSLs, it is scaled by v_{free}. Note that the share the same exploration decay, which is determined by σ_{decay}. The primary purpose of this exploration mechanism is to allow the controller to discover new, potentially superior control strategies and improve performance. Early in training, the higher standard deviation in the perturbations facilitates broader exploration of the solution space, enabling the RL agent to gather more diverse data for learning. As training advances, the decaying variance reduces exploration, allowing the MPC controller to refine and exploit the learned optimal policies.

Note that infeasibilities in the nonlinear MPC problems for V_θ and Q_θ can occur if the solution cannot satisfy the given constraints or poor initialization of the decision variables. Therefore, a simulation result is considered valid if the number of infeasibilities remains below 10% of the total number of MPC solver calls during a learning episode. The RL parameters that were considered are described in Table 4-4.

Table 4-4: RL parameters for MPC-RM-VSL case study

Symbol	Value
c_{TTS}	5
$c_{\text{V,ramp}}$	16
$c_{\text{V,speed}}$	1
c_{CV}	5
γ	0.98
$\alpha_{\rho_{\text{crit}}}$	0.5
α_a	0.1
α_θ	0.9
α_{decay}	0.91
$\Delta\theta$	0.3
$\sigma_{q,\text{ramp}}$	1
$\sigma_{q,\text{speed}}$	0.5
σ_{decay}	0.9

4-2 Results

This section discusses the results obtained from the simulations. The performance of all the previously mentioned methods is evaluated within the traffic network environment. To account for variability caused by stochastic factors such as exploration, the results are averaged over 5 simulations, each initialized with a different random seed. It is important to note that the same random seed number was used across all simulations for every model to ensure consistency of the results. The RL stage cost, as previously mentioned, includes the TTS, control variability, and constraint violations. These components collectively serve as valuable metrics for assessing the performance of the learning. The results are compared against three distinct models: an MPC with a perfect prediction model, which assumes exact knowledge of the learnable parameters influencing the dynamics (a, ρ_{crit}, and v_{free}); an MPC with a imperfect model, where these parameters are initialized with incorrect values that deviate from their true values as specified in Table 4-2; and the RL-MPC agent, which dynamically adjusts the learnable parameters during the training process to enhance performance.

4-2-1 Cost Objective Terms

The results for the evolution of the TTS during learning are shown in Figure 4-3. The MPC with the perfect prediction model, which has precise knowledge of the system parameters, consistently achieves low TTS and exhibits stable performance across all episodes. On the other hand, the MPC with the imperfect prediction model, despite starting with suboptimal parameters, performs surprisingly well, maintaining TTS levels close to those of the perfect model. However, keep in mind that, although the MPC with imperfect model outperforms the MPC with the perfect prediction model in terms of TTS, this does not necessarily imply that the former is superior overall. The objective function used in the evaluation considers multiple performance metrics as shown in Equation (3-32), including constraint violations and control variability. The MPC-RL model starts with high TTS due to its suboptimal initial parametrization, and demonstrates significant improvement over the learning episodes.

Thanks to RL, the controller adapts its parameters and progressively learns more effective policies. As a result, the TTS for the MPC-RL controller steadily decreases and starts to approach the performance of the perfect model. This highlights the ability of the RL framework to enhance the performance of the controller through adaptation, even when starting with a poor initialization. It can also be observed from the plot that the MPC-RL model has the potential to outperform the MPC with the perfect prediction model if more learning episodes are considered. However, achieving this requires careful consideration of the initial hyperparameter values, particularly the learning rate and its decay. It is crucial to ensure that, toward the end of the learning process, the learning rate has decayed sufficiently to allow for smaller updates. This promotes stability and increases the chance of convergence to an optimal value. The MPC-RL model has reached an averaged value of 1372.46veh $\cdot$ h in the last episode, and has showed a 46.24% reduction in comparison with its initial value.

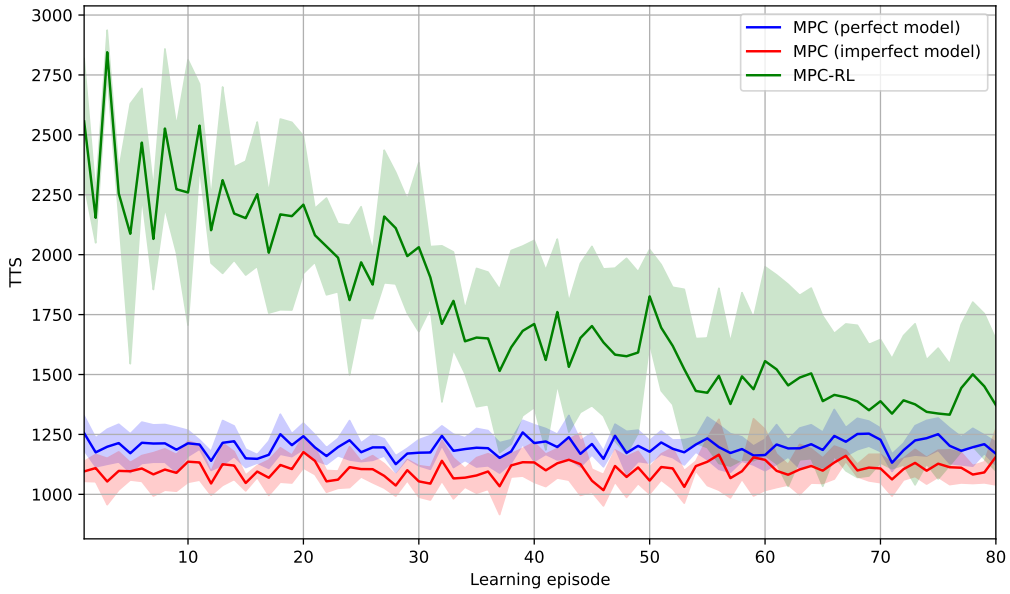


Figure 4-3: Evolution of the TTS over the learning episode. The solid line represents the mean result over 5 agents, while the shaded areas indicate the standard deviation.

The results for the constraint violation are shown in Figure 4-4 and is depicted in logarithmic scale. The MPC with the perfect prediction model exhibits consistently low violations values throughout the episodes, reflecting its ability to adhere strictly to system constraints. In contrast, the MPC with the imperfect prediction model shows occasional significant spikes in constraint violation values, particularly around the 45th, 60th, and 80th episodes. These spikes indicate episodes where the imperfect model struggles to meet constraints, likely due to inaccuracies in its parameter estimates and high randomness in the demand. However, between these episodes, the imperfect MPC demonstrates performance closer to the perfect model, suggesting that it can operate effectively under certain conditions despite its imperfections. The MPC-RL model is not plotted, as its constrain violation values remain consistently zero across all learning episodes. This highlights the effectiveness of the RL component in dy-

namically adjusting the learnable parameters to fully satisfy constraints, even in the presence of initial parameter deviations. This means that the queue length did not excel the maximum allowable queue length on the on-ramp.

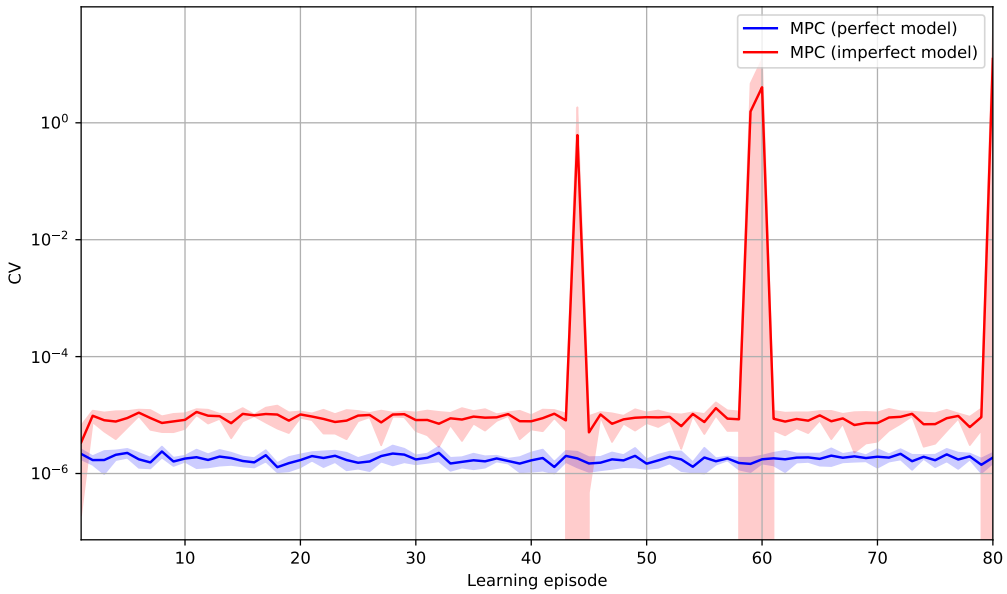


Figure 4-4: Evolution of the constraint violation over the learning episode

The results for the control variability are shown in Figure 4-5. The MPC model with perfect parameters consistently demonstrates the lowest control variability, as it has exact knowledge of the dynamics, allowing it to make smooth and stable control decisions. In contrast, the MPC with the imperfect prediction model exhibits higher variability due to parameter inaccuracies, which lead to less precise predictions and more aggressive or oscillatory control actions. The MPC-RL model starts with a relatively high control variability, reflecting the exploration phase of the learning process. However, as the learning progresses, the control variability decreases significantly, indicating that the model is refining its policy and moving toward more stable control actions. By the later episodes, the MPC-RL model achieves a level of variability lower than the MPC with the imperfect prediction model and approaches that of the MPC with the perfect prediction model. After the 40th learning episode, exploration significantly decreases, resulting in oscillations around a steady value.

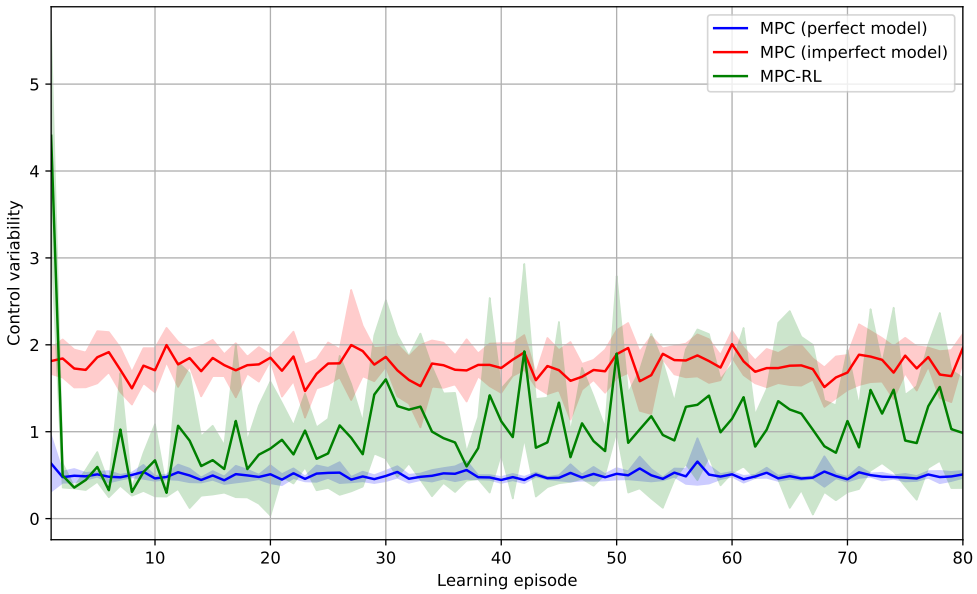
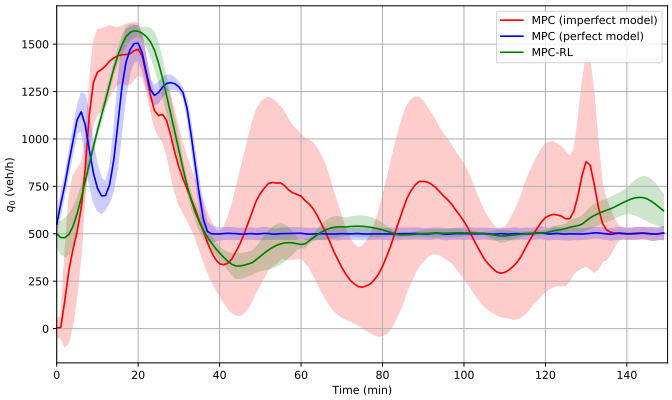


Figure 4-5: Evolution of the control variability over the learning episode

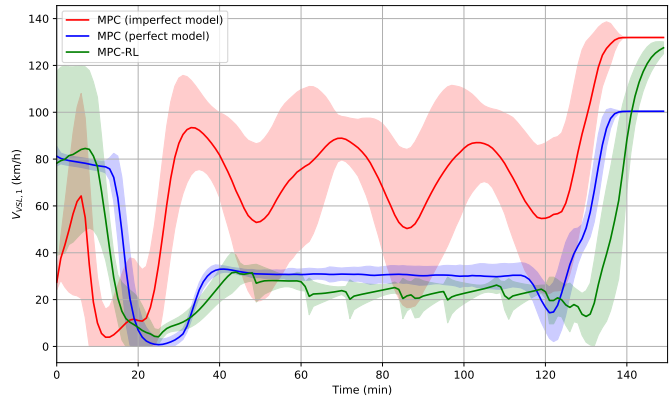
The control variability reaches a relatively low value, indicating smooth and stable control actions over time. This implies that adjustments to control variables are minimal and consistent. This can be observed in the results of the control actions, depicted in Figure 4-6. In general, it is desired to have stable and smooth control actions, in order to guarantee safety traffic conditions.

4-2-2 Control Actions

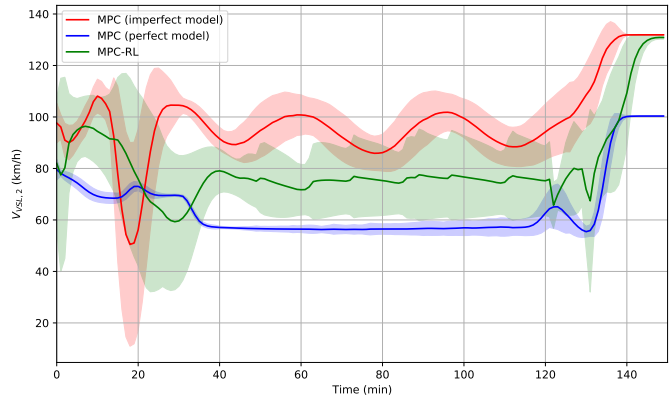
The results of the control actions are depicted in Figure 4-6. A single MPC model (both with perfect and imperfect prediction model) was simulated 80 times using a specific random seed that introduced stochasticity in the demand. The actions from these 80 simulations were averaged for that particular random seed. This process was repeated for five different random seeds, and the resulting averaged actions were further averaged across these five simulations to obtain the final result. For the MPC-RL model, the last action from each learning episode was considered. These final actions were then averaged across the five different simulations to derive the overall result.



(a) Ramp flow (q_0).



(b) Equilibrium speed ($V_{\text{VSL},1}$).



(c) Equilibrium speed ($V_{\text{VSL},2}$).

Figure 4-6: Results of the control action on average

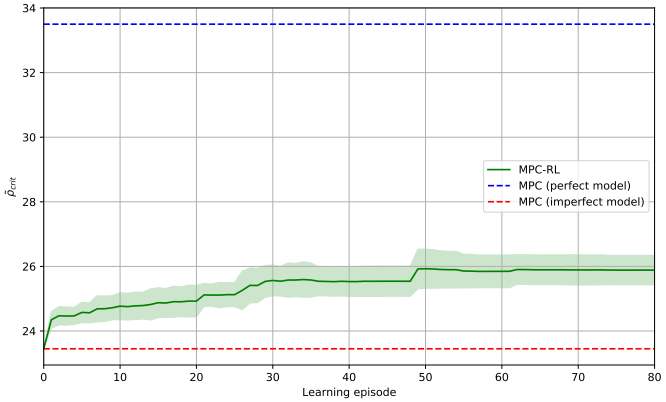
For ramp flow, all models show significant variability during the initial stages as the system responds to dynamic traffic demands. The MPC with the imperfect prediction model exhibits oscillations, particularly during the first 90 minutes. In contrast, both the MPC with the perfect prediction model and MPC-RL demonstrate smoother ramp flow trajectories, with fewer oscillations. Toward the end of the simulation, all models converge to similar ramp flow values, with the MPC-RL aligning closely with the MPC with the perfect prediction model. It can also be observed that during the high demand at the beginning of the simulation lead to sharp oscillations in the MPC with the perfect prediction model, whereas the MPC-RL model shows a smoother response. This suggests that the MPC-RL model effectively incorporates safety conditions into its decision-making process, even under conditions of elevated demand. Consequently, the MPC-RL model outperforms the MPC model in managing ramp flow and maintaining safety conditions in such scenarios.

The equilibrium speed of the MPC with the imperfect prediction model show similar oscillations trend. It struggles to maintain stable speed limits, showing erratic and large oscillations. This can lead to unsafety conditions, hence a high chance to number of accidents. On the other hand, the MPC with the perfect prediction model manages more consistent speed limit control, with minor transient deviations. The MPC-RL model shares the same performance, achieving stable and smooth VSL adjustments after the initial transient phase. Both equilibrium speeds show that the MPC-RL model adapts effectively to the traffic environment, providing results that closely align with those of the perfect model. Furthermore, it can be observed that the equilibrium speeds in the MPC-RL model are higher than those of the MPC with the perfect prediction model toward the end of the simulation. This behaviour is lead to the MPC-RL model prioritizing higher speeds during periods of low demand, while lowering the equilibrium speeds more significantly than the MPC with the perfect prediction model during high-demand phases. Additionally, while some spiky oscillations are present, they remain within a reasonable range and do not compromise safety conditions, indicating that the system maintains stable and controlled adjustments.

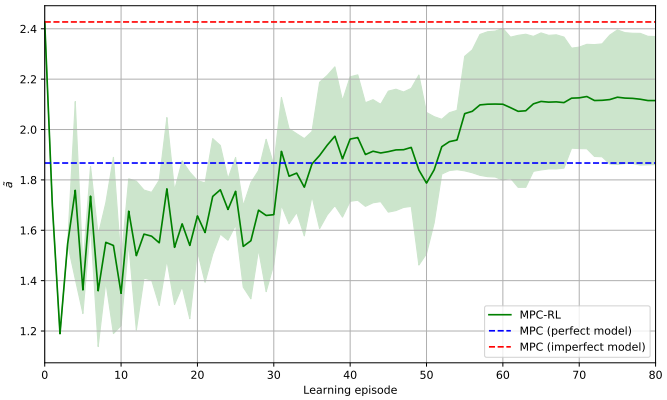
4-2-3 Learnable MPC Parameters

The results for the learnable MPC parameters that influence the prediction model are shown in Figure 4-7. The critical density parameter, $\tilde{\rho}_{\text{crit}}$, starts at the initialization of the MPC with the imperfect prediction model. Over the learning episodes, the RL algorithm adjusts $\tilde{\rho}_{\text{crit}}$, converging on average toward an optimal value of approximately 26 veh/km/lane. However, it can be observed that this learnable parameter does not reach the value of the MPC with the perfect prediction model. This means that the MPC-RL model adopts a more pessimistic stance toward congestion. Specifically, the transition to congested flow is predicted to occur at lower densities compared to the real ρ_{crit}-value given by the MPC with the perfect prediction model. As consequence, the MPC-RL model overestimates congestion. Moreover, in the context of constraint h_2 (3-24), a lower $\tilde{\rho}_{\text{crit}}$ tightens the constraint, limiting the control actions (ramp flow) to prevent congestion. The parameter $\tilde{a}$ exhibits a distinct learning trajectory. Initially, the MPC-RL model starts at the MPC with the imperfect prediction model initialization of $1.3a$, but steadily approaches value of the MPC with the perfect prediction model (a). By the 40th learning episode, the averaged parameter stabilizes around 2.1, slightly higher than the perfect value. This implies that V_{VSL} favours prediction of higher speeds at lower densities and lower speeds for higher densities compared to the a-value given by the MPC

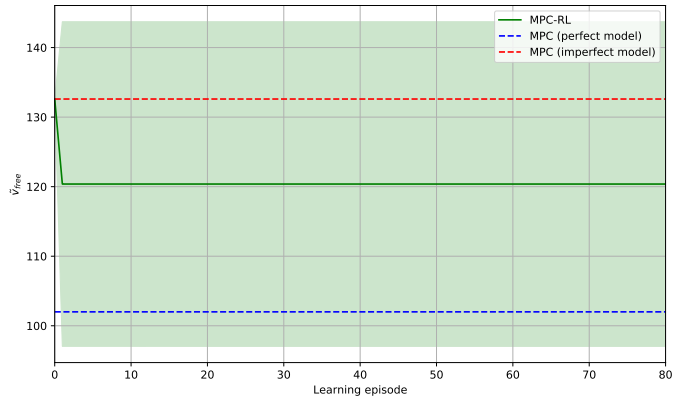
with the perfect prediction model, according to the relation described in Equation (3-5).



(a) Leanable MPC parameter: $\tilde{\rho}_{\text{crit}}$



(b) Leanable MPC parameter: $\tilde{a}$



(c) Leanable MPC parameter: $\tilde{v}_{\text{free}}$

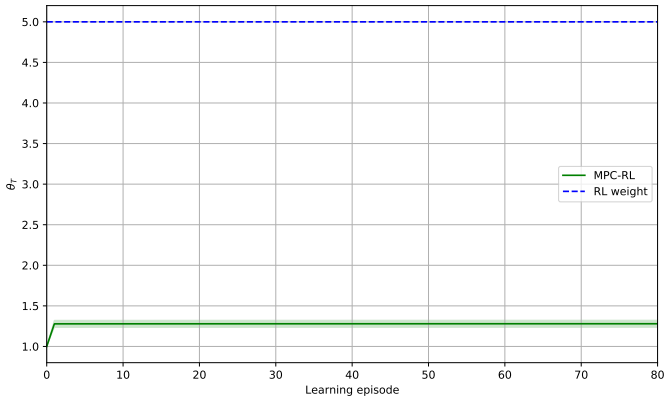
Figure 4-7: Evolution of the learnable MPC parameters that influence the prediction model

The free-flow speed, $\tilde{v}_{\text{free}}$, starts at 132.6 km/h, corresponding to $1.3v_{\text{free}}$ from the MPC with the imperfect prediction model. Unlike $\tilde{\rho}_{\text{crit}}$ and $\tilde{a}$, v_{free} demonstrates minimal changes during learning. The MPC-RL model maintains a consistently higher value compared to the value of the MPC with the perfect prediction model ($v_{\text{free}} = 102\,\text{km/h}$). This behaviour highlights the preference of the MPC-RL model for maintaining higher free-flow speeds, prioritizing faster traffic flow when safety and performance constraints are satisfied. Eventually, this learnable parameter reaches a value of 120.38km/h on average. Note that the standard deviation for this result is very high. This outcome is attributed to five simulations, where one simulation converged to a value of 150, two simulations converged to 133.89, and the remaining two settled at 92.81. This convergence already happened at the start of the learning episode. This suggests that the learning process is dominated by initialization and early dynamics, possibly resulting in suboptimal solutions. Addressing this issue would require further fine-tuning the learning mechanisms to prevent premature convergence.

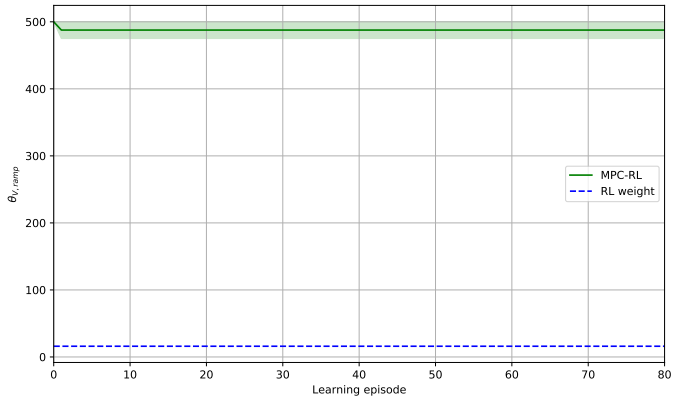
4-2-4 Parametrized Cost Weights

The result for θ_{T} is shown in Figure 4-8a. The parameter θ_{T}, which emphasizes the importance of minimizing TTS in the objective function of the MPC function approximator, converges to a value close to 1.25. This indicates that the RL framework has determined a relatively balanced weight for TTS minimization in the objective. The convergence to a value higher than the initial value reflects the increasing emphasis on TTS reduction as a critical performance metric during learning, which can also be seen in Figure 4-3. The evolution of the parameter $\theta_{V,\text{ramp}}$ and $\theta_{V,\text{speed}}$ plays a crucial role in influencing control variability, as depicted in Figures 4-8b and 4-8c. For the parameter $\theta_{V,\text{ramp}}$, which is associated with the variability of the ramp metering control action, the RL agent adjusts its value rapidly during the initial episodes. The parameter stabilizes early, indicating that the RL process effectively fine-tunes the balance between reducing variability and maintaining control stability at an early stage of learning. Similarly, the parameters $\theta_{V,\text{speed.1}}$ and $\theta_{V,\text{speed,2}}$, which influence variability in equilibrium speed, exhibit quick convergence to stable values within the first 2 episodes. The control variability plot depicted in Figure 4-5 further corroborates these observations, as the MPC-RL model demonstrates a significant reduction in control variability compared to the MPC with the imperfect prediction model at the start of the learning episode due to rapid changes of $\theta_{V,\text{ramp}}$ and $\theta_{V,\text{speed}}$. The convergence of $\theta_{V,\text{ramp}}$ and $\theta_{V,\text{speed}}$ directly translates to smoother and more stable control actions, highlighting the effectiveness of the learned parameters in mitigating unnecessary oscillations or abrupt changes in traffic control measures. This behaviour is particularly evident in the reduced variability of both ramp metering and speed adjustments under the MPC-RL framework. This means that a proper initialization or a different learning rate is necessary to see if this learnable parameter changes in value. As for θ_C, this value remained 5 and has not changed over the learning episodes. While this could initially suggest that the parameter has found a global optimum or is stuck at a local minimum, a closer analysis reveals a different explanation. Specifically, if the Lagrangian formulation is examined, the sensitivity of θ_C is proportional to the optimal values of the slack variables. However, since the MPC-RL agent never encounters constraint violations, these slack variables remain zero throughout training. As a result, the gradient of the Lagrangian with respect to θ_C is always zero, preventing any updates and causing the parameter to remain fixed at its initial value. This indicates that θ_C is not necessarily at an

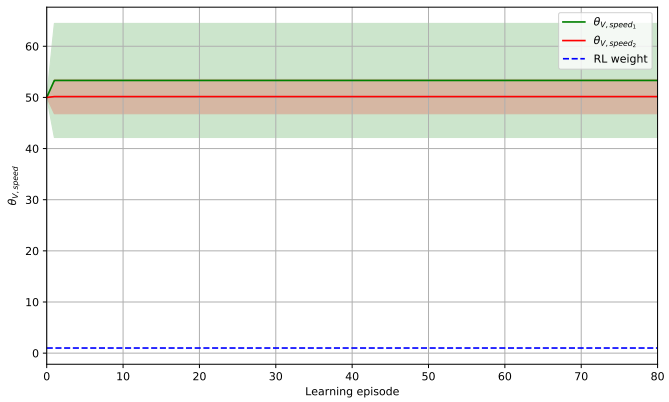
optimum but is instead in a stationary state due to the structure of the optimization problem.



(a) Leanable MPC parameter: θ_{T}



(b) Leanable MPC parameter: $\theta_{V,\mathrm{ramp}}$



(c) Leanable MPC parameter: $\theta_{V,\mathrm{speed},1}$ and $\theta_{V,\mathrm{speed},2}$

Figure 4-8: Evolution of the parametrized cost weights

The results for the other learnable parameters can be seen in the Appendix C.

4-2-5 TD error

The results for the TD error are shown in Figure 4-9. Initially, the TD error is high and exhibits significant oscillations, which is expected as the model begins with limited knowledge and explores various actions to improve its policy. As the learning progresses, a clear downward trend in the TD error can be observed, indicating that the model is gradually refining its predictions and learning the learnable MPC parameters of the traffic environment. This is a good indication that the MPC scheme (3-27) is able to provide a reliable approximation Q_θ of the true unknown action-value function. By the 40th episode, the TD error begins to stabilize, showing a noticeable reduction in standard deviation, as reflected by the narrowing shaded region. This reduction in standard deviation suggests that the learning process has become more consistent, and the model is making more accurate predictions about the outcomes of its actions over 5 simulations. Toward the end of the episodes, the TD error stabilizes at a lower value, approximately around 40, signifying that the MPC-RL model has reached a steady policy. Unfortunately, the TD error does not reach an absolute zero and this can be due to factors such as the imperfect or insufficient parametrization of the model and the inherent stochasticity of the environment. Variations in traffic demand introduce oscillations that make predicting the value functions more complex and challenging, preventing the TD error from reaching a perfect zero.

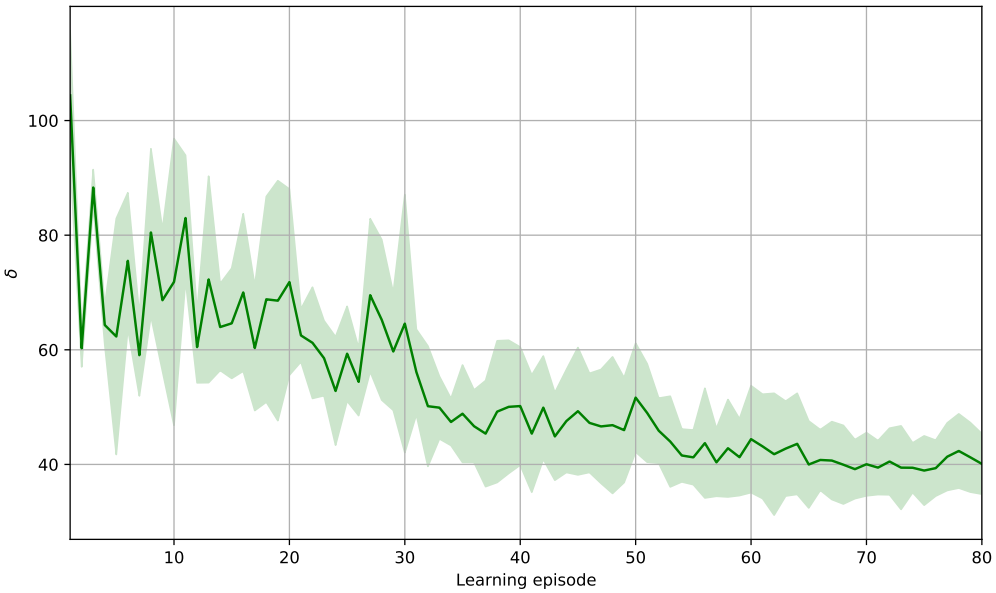


Figure 4-9: Evolution of the TD error over the learning episode

4-3 Discussion

From the results, it can be observed that the proposed methodology shows an ability in learning to improve the traffic control performance while maintaining the prescribed safety condition for given wrong model parameters and a poorly tuned initial controller. The MPC-RL model exhibits promising results compared to the MPC with the perfect and imperfect prediction models. The TTS has lowered significantly, demonstrating a clear decreasing trend and has the potential to outperform over the MPC with the perfect prediction model if longer learning episodes were considered. However, extending the learning process would require adjusting the learning rate decay to ensure a sustainable and effective learning progression over the extended time. As consequence, this can influence the results, as different learning rates might lead to the discovery of new local minima due to the highly nonlinear nature of the system dynamics. Such variations in the optimization could influence the overall performance.

As for the variability, this is stable and lead to less oscillations in comparison with the MPC with the imperfect prediction model. This stability directly contributes to improved safety conditions. However, it can be observed that the standard deviation is very high for the second control action of the equilibrium speed, in comparison with other control actions. Furthermore, as observed in Figure 4-6, there are slight oscillations occurring around the 120 minutes. While these oscillations are minor, they could potentially lead to slightly unsafe behaviour. In order to lower the standard deviation and to have a more smoother control action with even less oscillations, one could consider to adjust the weight for this particular control action. By fine-tuning the weighting, the standard deviation can be reduced, leading to more stable and safer system behaviour over time.

In general, exploring alternative initial values for the learnable parameters may significantly influence the results, as it provides the RL agent with a new region to explore, increasing the likelihood of discovering new local minima or global minimum. During the simulations, it was observed that many learnable parameters converged too quickly, either becoming stuck in local minima. To address this, assigning distinct initial values to each learnable parameter and fine-tuning them systematically could help ensure the RL agent more effectively identify the global minimum. Similarly, one could consider to assign different learning rates and learning rate decays for each learnable parameters to escape from local minima. Another idea is to consider multi-starting the Nonlinear Programming solver for the MPC problems, as it solves the optimization problem multiple times form different initial guesses and selecting the best solution [52]. This also helps to counter infeasibilities for a particular time-step. These approaches are particularly relevant for learning $\tilde{v}_{\text{free}}$, as the results revealed the highest standard deviation for this parameter, with varying convergence values across different simulations. This also holds for learning $\tilde{a}$, as the standard deviation is also high at the end of the learning episode. However, the disadvantage of these approaches lies in its computational intensity. Fine-tuning multiple parameters requires testing numerous combinations, and the process becomes even more time-consuming and complex as the parametrization grows.

Lastly, it is worth noting that the RL algorithm can only process a finite amount of data during training. This is an additional factor that influence the suboptimality of the MPC-based RL solution. As the amount grows infinite, the theoretical optimality will be achieved [33].

Chapter 5

Conclusion

5-1 Summary

This thesis project explored the combination of MPC and RL for optimizing highway traffic control, specifically through RM and VSLs. The goal was to investigate how these frameworks can be effectively integrated to improve traffic flow and overall system performance. Recall that the research question posed in this thesis was:

> *How can the frameworks of MPC and RL be effectively combined to optimize highway traffic control and increase performance?*

The study demonstrated that the MPC-RL approach successfully enhances traffic control performance by leveraging the predictive capabilities of MPC and the adaptability of RL. Simulation results indicated that this combination leads to significant reduction in TTS in traffic compared to the initial value from the imperfect model, demonstrating the effectiveness of the RL component in optimizing control strategies over time. Additionally, the MPC-RL framework exhibited a smoother and more stable traffic flow with fewer oscillations in control variables compared to an MPC with an imperfect prediction model. While the RL framework initially showed high control variability due to exploration, it successfully converged toward stable control actions, ensuring improved highway safety and efficiency. The model also avoided significant constraint violations. Furthermore, the MPC-RL model showed the potential to outperform the MPC with a perfect prediction model if more learning episodes were considered, highlighting the benefits of adaptive learning. This requires to increase the learning decay rate value in order to still have learning at the nearly end of the episodes. Note that this can lead to different suboptimality solutions, and this could influence the overall performance. Meaning that the whole initial values for the hyperparameters and parametrization should be reconsidered.

The learnable MPC parameters that influence the prediction model differ from the values given by the MPC with the perfect prediction model. The critical density parameter $\tilde{\rho}_{\text{crit}}$

gradually increases from its initial value but does not reach that of the MPC with the perfect prediction model, leading to a conservative congestion estimation that limits control actions. The parameter $\tilde{a}$ follows a steady learning trajectory, converging slightly above the true a value, suggesting that the model predicts higher speeds at lower densities and lower speeds at higher densities. The free-flow speed $\tilde{v}_{\text{free}}$ remains relatively unchanged, stabilizing at a higher value than the MPC with the perfect prediction model, indicating a preference for maintaining higher speeds when safety constraints allow. However, the high standard deviation in $\tilde{v}_{\text{free}}$ across simulations suggests that early-stage learning heavily influences final parameter values, potentially leading to suboptimal solutions. Addressing this issue through improved learning mechanisms could enhance the consistency of the model and reliability in traffic management. Exploring alternative initial values for learnable parameters appears to be a promising avenue for improving model performance. Assigning distinct initial values to each parameter and fine-tuning them systematically could help the RL agent identify global minimum more effectively. Additionally, employing different learning rates and learning rate decays for each parameter might enhance the ability of the system to escape local minima. Another promising approach is the use of multi-start optimization, which solves the MPC problem multiple times from different initial guesses, thereby improving feasibility across time steps. However, these approaches come at a computational cost, as fine-tuning multiple parameters significantly increases processing time and complexity.

Overall, the study confirms the potential of the MPC-RL framework for improving highway traffic control, balancing efficiency and stability. While challenges remain in fine-tuning learning processes and managing computational complexity, these findings pave the way for further enhancements in adaptive traffic management systems.

5-2 Future outlook

The findings of this study highlight several promising directions for future research and improvements in the integration of RL with MPC for highway traffic management. While the current methodology has demonstrated effectiveness, various aspects can be further explored and refined to enhance its performance.

- **Enhancing learning optimality:** One of the primary challenges observed in the study is the rapid convergence of learnable parameters, sometimes leading to local minima. Future research should focus on refining learning rates, parameter initialization, and introducing multi-start optimization techniques to ensure that the model identifies global optimum rather than suboptimal solutions.
- **Incorporating safety and environmental considerations:** While the primary focus of this study was optimizing traffic efficiency, future research should incorporate additional performance metrics, such as accident risk reduction, emissions control, and fuel efficiency. Integrating these factors into the reward function could lead to a more sustainable and safety-conscious traffic control strategy.
- **Exploring complex parametrization:** A key avenue for future research is the development of more complex parametrizations of the MPC scheme, such as incorporating neural networks. Given the nonlinear nature of the METANET framework, neural

networks have the potential to better capture the system's dynamics and improve prediction accuracy. By leveraging deep learning for action-value function approximation, the MPC-RL framework could achieve more precise and adaptive control policies. This approach would enable the system to handle higher-dimensional traffic conditions while improving optimization performance. Another idea is to the extend the framework to include multiple RL agents that cooperatively manage different highway sections.

- **Validation in real-world application:** While simulation-based results provide valuable insights, real-world testing is necessary to validate the effectiveness of MPC-RL model. Future research should focus on deploying the framework in actual highway environments, collecting empirical data, and refining the approach based on real-world constraints and uncertainties.
- **Exploring alternative solvers for finding the optimal policy:** Another promising direction for future research is investigating alternative solvers for finding the optimal policy within the MPC-RL framework. While MPC acts as a policy function approximator, different optimization techniques could enhance efficiency and convergence. Methods such as Bayesian Optimization and gradient-based approaches, including policy gradient methods, could refine the learning process by enabling iterative updates to the policy parameters.

Appendix A

Huber Loss Functions

In this appendix, the formulation of the left-sided and right-sided Huber loss penalties is described, providing their mathematical definitions as mentioned in Section 3-3-2.

A-1 Left-sided Huber Loss Penalty

Let $\rho \in \mathbb{R}^n$ be a vector of densities in n links at a given time step (e.g., stage or terminal) as an example. Define the following objective:

$$J(\rho) = \sum_{i=1}^{n} w_i \phi_\ell \left(\frac{\rho_i - s_i}{M}, \delta_i \right), \tag{A-1}$$

where ϕ_ℓ is the left-sided Huber loss function:

$$\phi_\ell(r; p) = \begin{cases} -p(2r + p), & \text{if } r \leq -p, \\ r^2, & \text{if } r > -p, \end{cases} \tag{A-2}$$

and $w \in \mathbb{R}^n$, $s \in \mathbb{R}^n$, and $\delta \in \mathbb{R}^n$ are the weights, the setpoints, and the knee-points of each penalty term, respectively. $M \in \mathbb{R}$ is a scalar value used for normalization.

Note: For $J(\rho)$ to be convex and meaningful, we need $w_i \geq 0$, $\delta_i \geq 0$, $i = 1, \ldots, n$, and $M > 0$.

A-2 Right-sided Huber Loss Penalty

Let $\mathbf{v} \in \mathbb{R}^n$ be a vector of velocities in n links at a given time step (e.g., stage or terminal). Define the following objective:

$$J(\mathbf{v}) = \sum_{i=1}^{n} w_i \phi_r \left(\frac{v_i - s_i}{M}, \delta_i \right), \tag{A-3}$$

where ϕ_r is the right-sided Huber loss function:

$$\phi_r(r; p) = \begin{cases} p(2r - p), & \text{if } r \geq p, \\ r^2, & \text{if } r < p, \end{cases} \tag{A-4}$$

and $\mathbf{w} \in \mathbb{R}^n$, $\mathbf{s} \in \mathbb{R}^n$, and $\boldsymbol{\delta} \in \mathbb{R}^n$ are the weights, the setpoints, and the knee-points of each penalty term, respectively. $M \in \mathbb{R}$ is a scalar value used for normalization.

Note: For $J(\mathbf{v})$ to be convex and meaningful, we need $w_i \geq 0$, $\delta_i \geq 0$, $i = 1, \ldots, n$, and $M > 0$.

Appendix B

Parameter Evolution in MPC-RL for Coordinated Highway Traffic Control

In this appendix, the evolution of all learnable parameters in the MPC-RL model for coordinated highway traffic control is presented, as detailed in Section 3-2-2.

B-1 Parametrised Cost weights

B-1-1 Initial cost

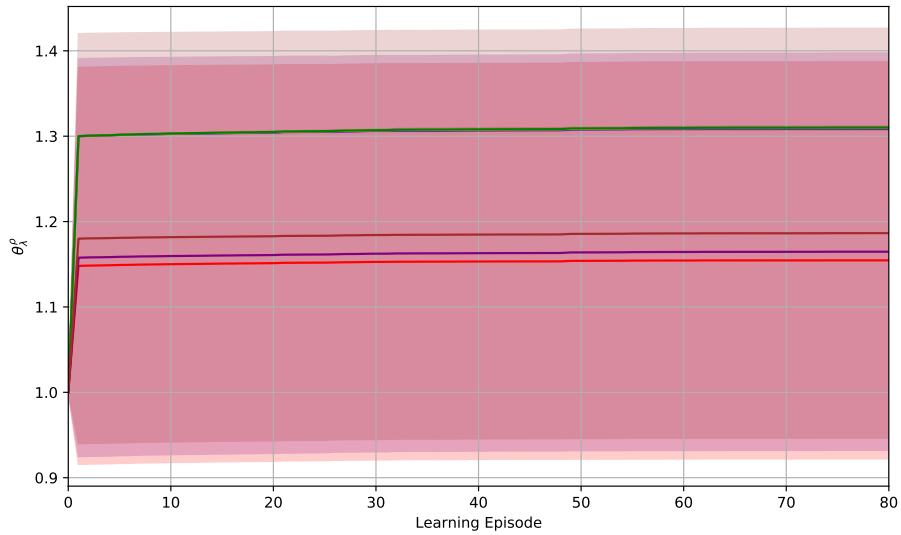


Figure B-1: θ_λ^ρ

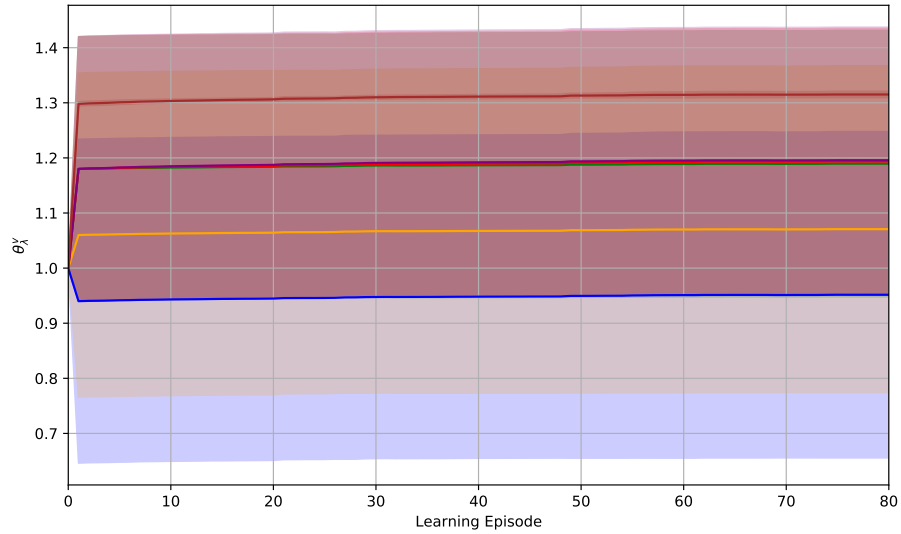


Figure B-2: θ_λ^v

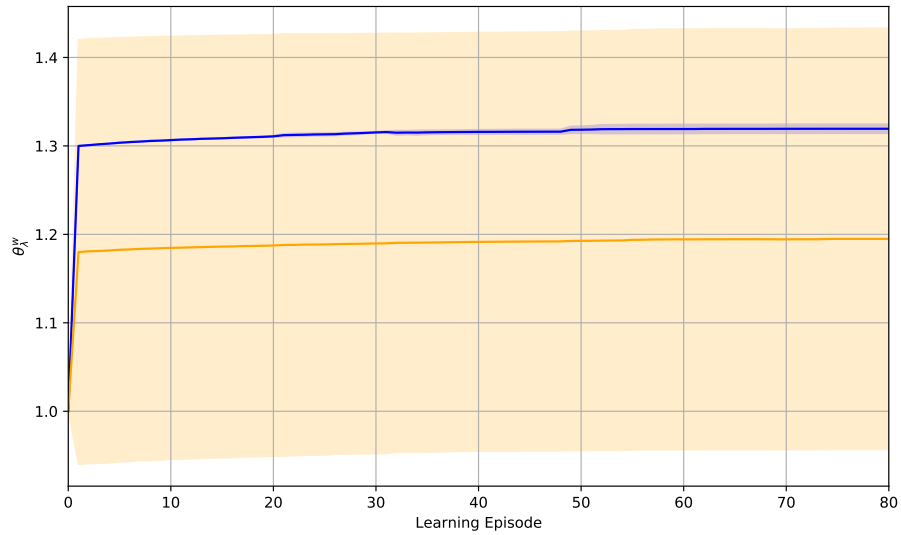


Figure B-3: θ_λ^w

B-1-2 Stage cost

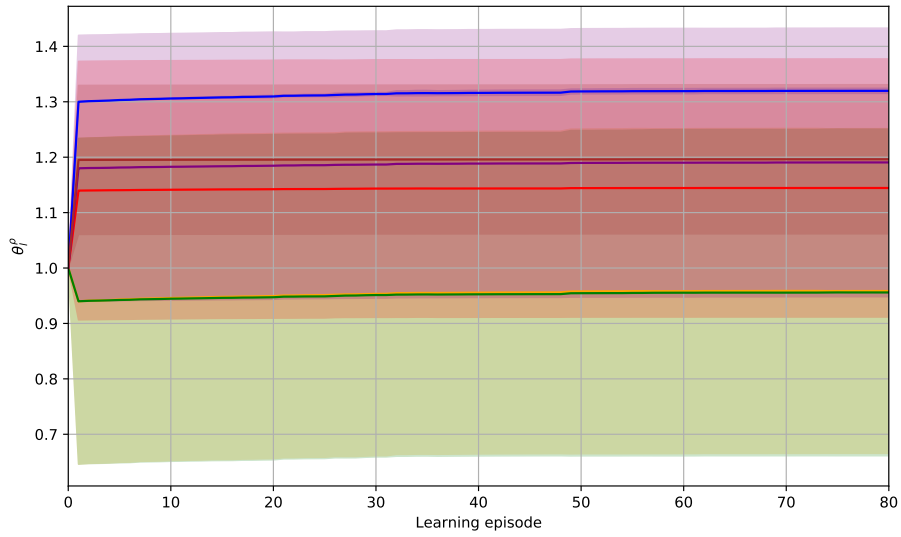


Figure B-4: θ_l^ρ

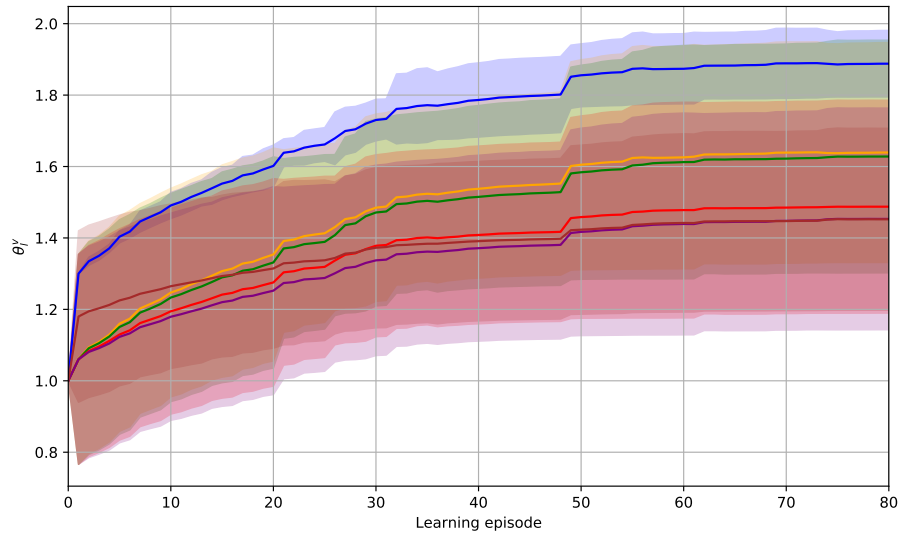


Figure B-5: θ_l^v

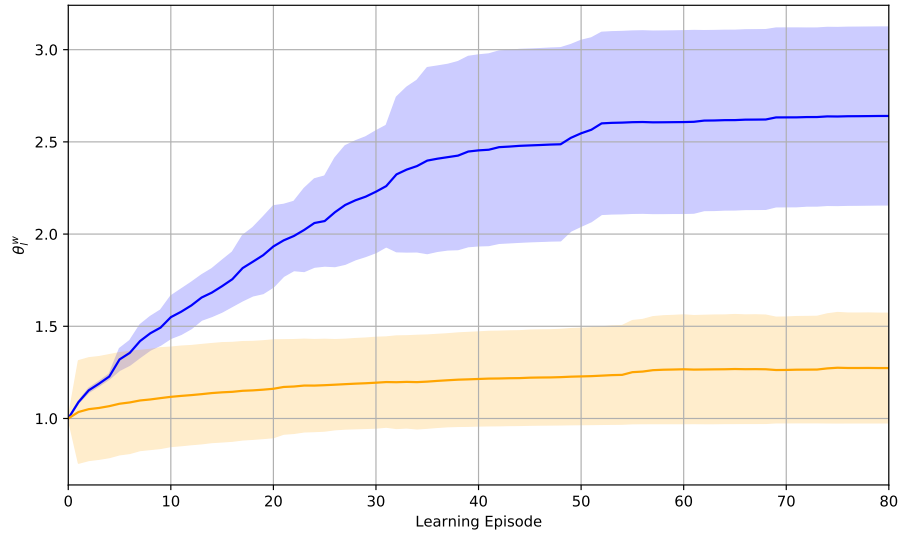


Figure B-6: θ_l^w

B-1-3 Terminal cost

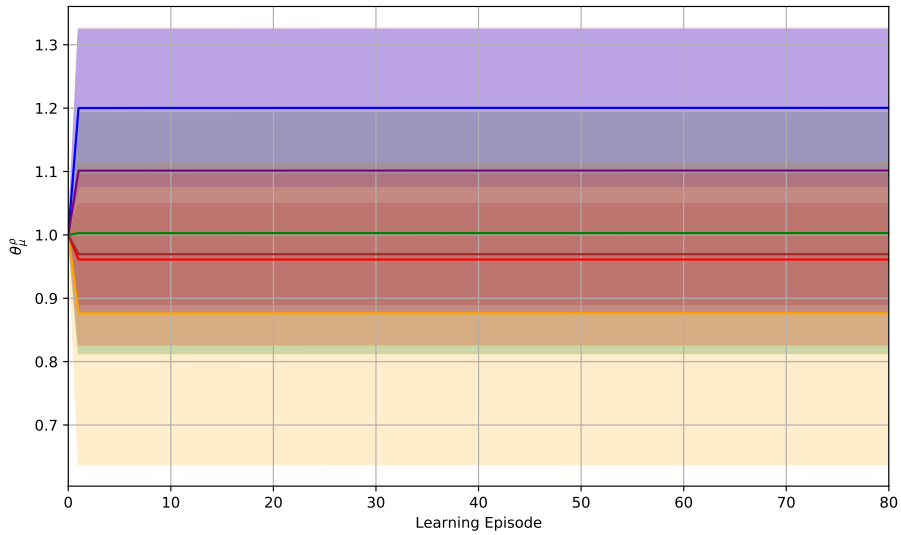


Figure B-7: θ_Γ^ρ

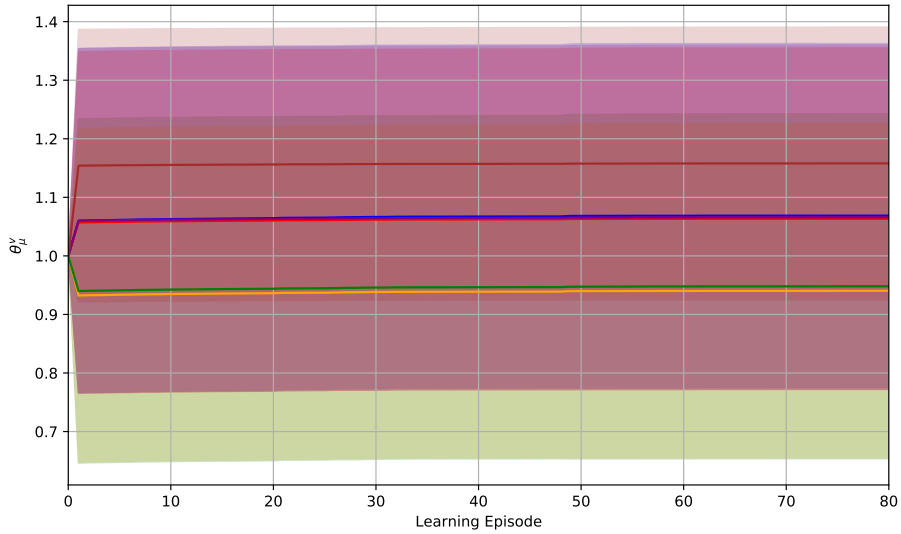


Figure B-8: θ_Γ^v

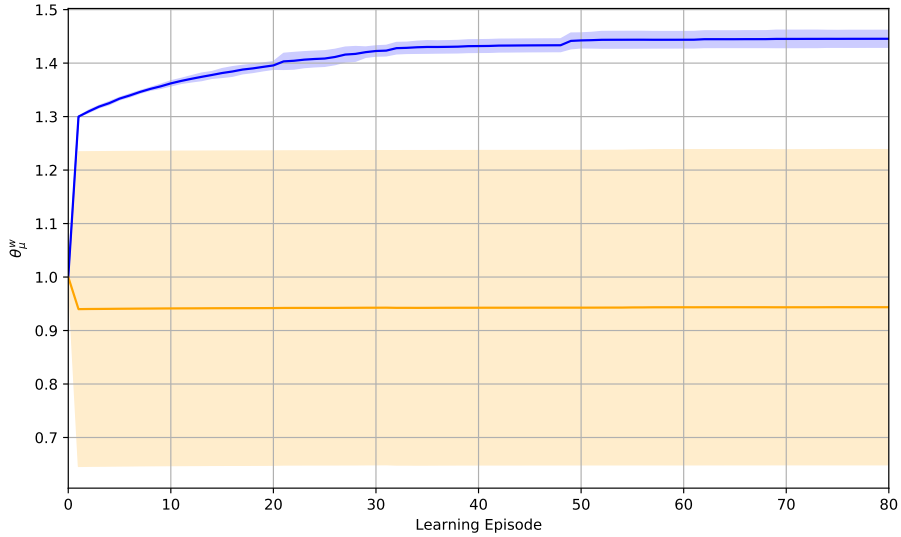


Figure B-9: θ_Γ^w

B-2 Linear Slope Parameters

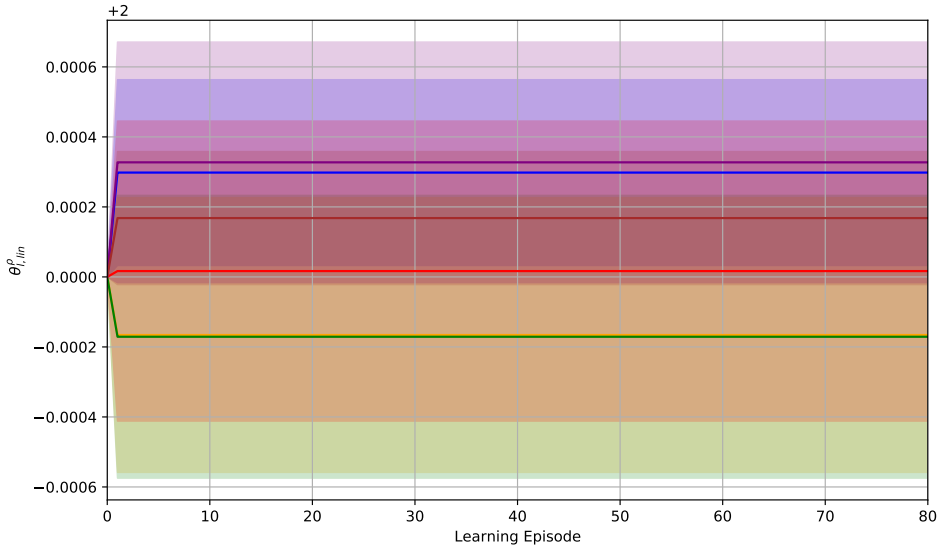


Figure B-10: $\theta^{\rho}_{l,\text{lin}}$

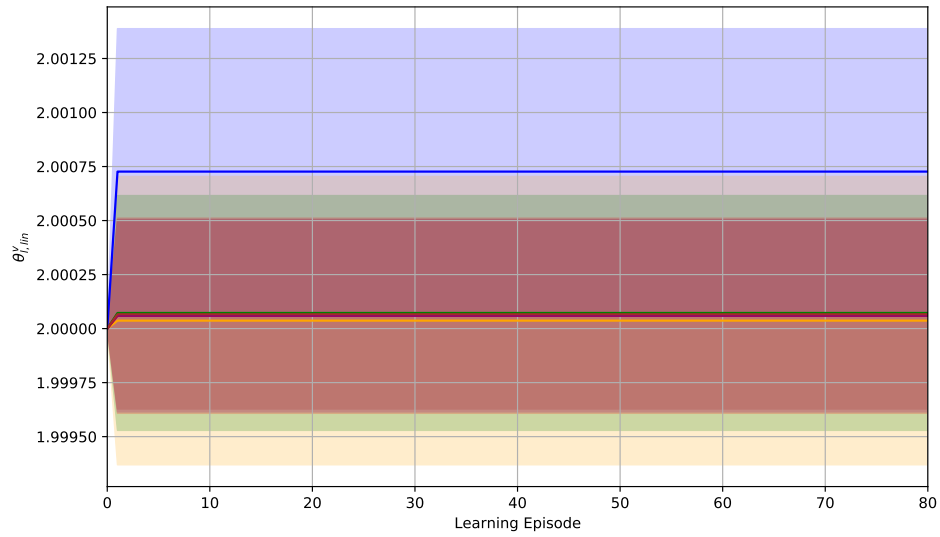


Figure B-11: $\theta^{v}_{l,\text{lin}}$

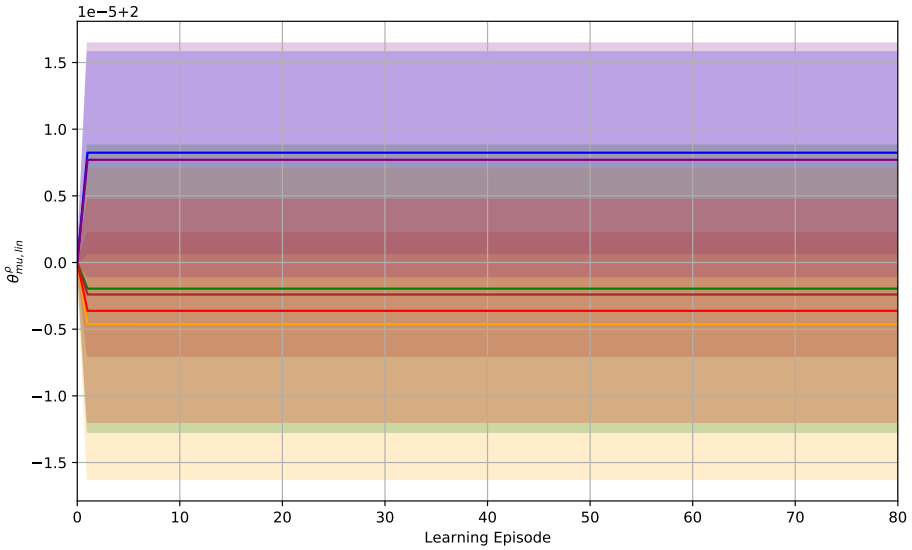


Figure B-12: $\theta^{\rho}_{\Gamma,\text{lin}}$

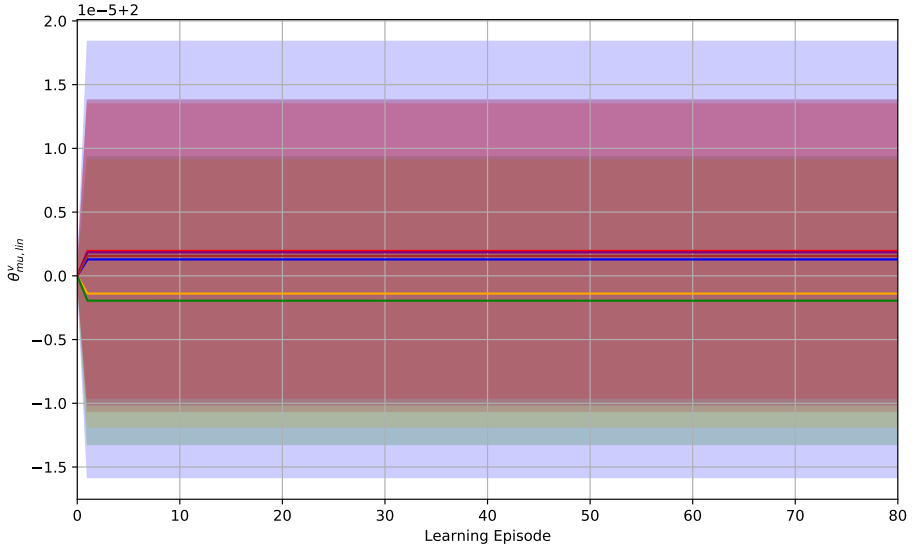


Figure B-13: $\theta^{v}_{\Gamma,\text{lin}}$

Appendix C

States Evolution in MPC-RL for Coordinated Highway Traffic Control

In this appendix, the evolution of key traffic state variables in MPC-RL for coordinated highway traffic control is presented. The analysis focuses on differences in traffic density and speed across six network segments, comparing the first and last episodes of the learning process. The presented figures illustrate the averaged results from 5 simulation runs, highlighting the adaptation of the system and performance improvements over time.

C-1 Density

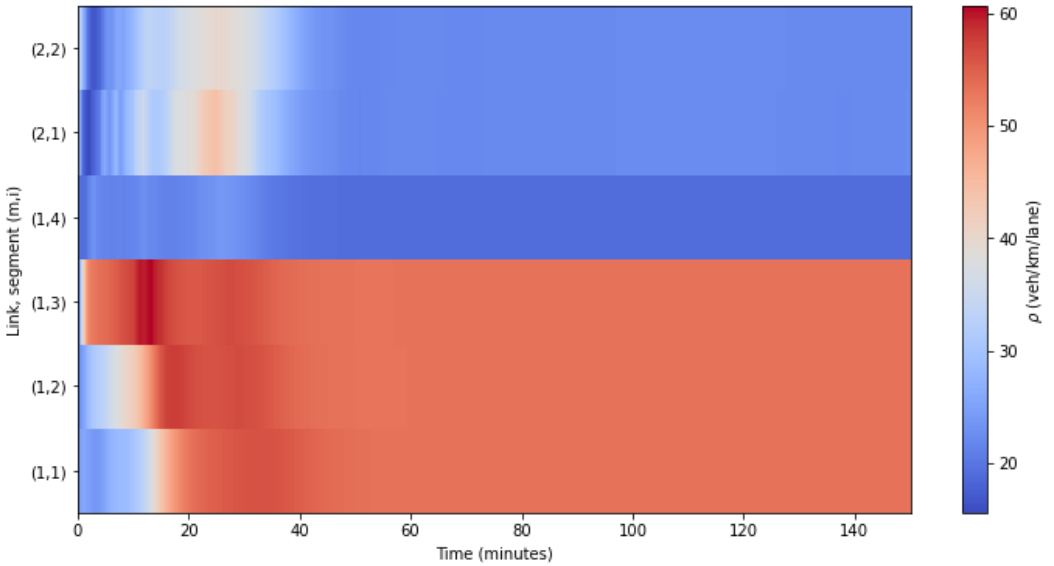


Figure C-1: Average density of the first learning episode

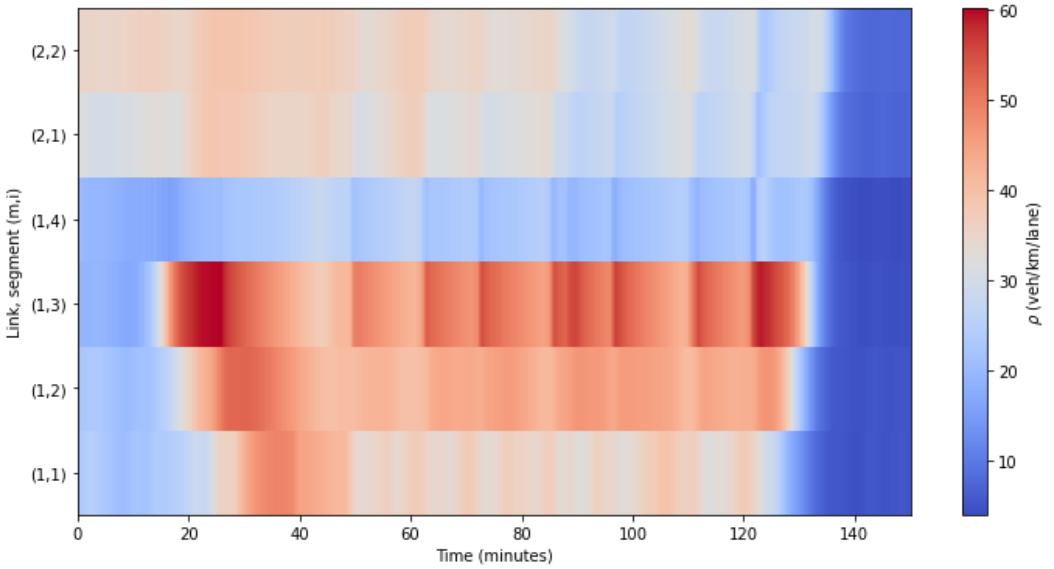


Figure C-2: Average density of the last learning episode

C-2 Speed

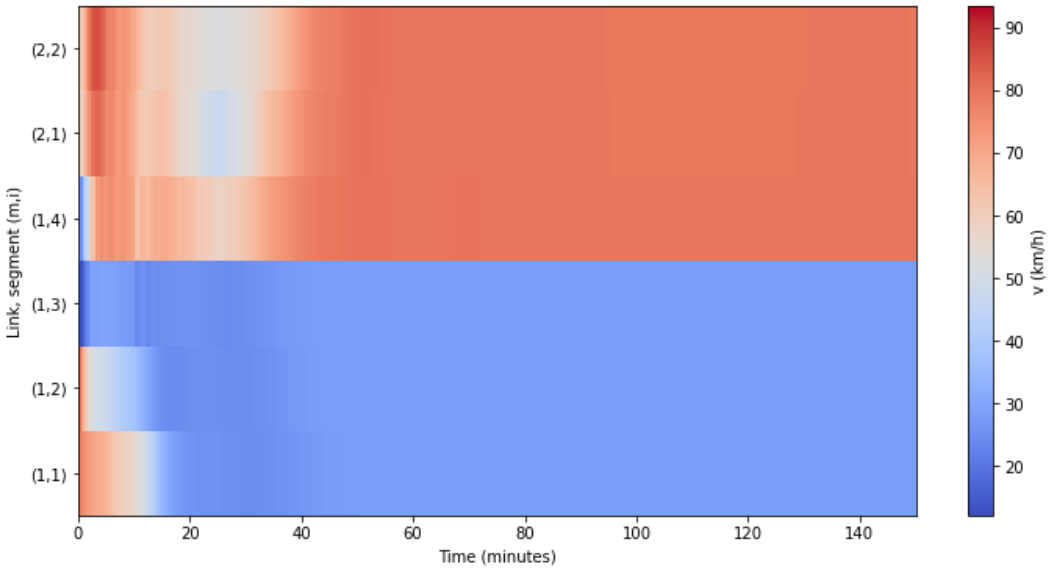


Figure C-3: Average speed of the first learning episode

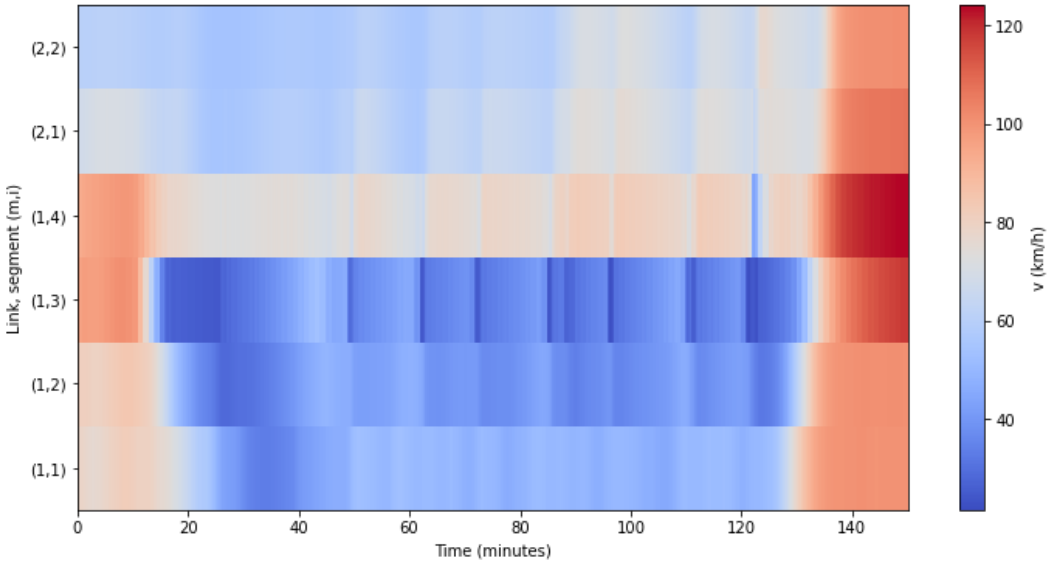


Figure C-4: Average speed of the last learning episode

Bibliography

[1] Andreas Hegyi, Bart De Schutter, and Hans Hellendoorn. Model predictive control for optimal coordination of ramp metering and variable speed limits. *Transportation Research Part C: Emerging Technologies*, 13(3):185–209, 2005.

[2] U.S. Department of Transportation Federal Highway Administration (FHWA). Freeway Management and Operations Handbook - Chapter 2: Basic Concepts and Definitions. https://ops.fhwa.dot.gov/publications/fhwahop14020/sec2.htm. Accessed on March 19, 2023.

[3] DFDS Ferry Company. Motorcycling in Holland. https://www.dfds.com/en-gb/passenger-ferries/destinations/holland/motorcycling-in-holland. Accessed on March 19, 2023.

[4] Roohollah Amiri, Hani Mehrpouyan, Lex Fridman, Ranjan Mallik, Arumugam Nallanathan, and David Matolak. A machine learning approach for power allocation in hetnets considering qos. 03 2018.

[5] Filippo Airaldi, Bart De Schutter, and Azita Dabiri. Reinforcement learning with model predictive control for highway ramp metering. *arXiv preprint arXiv:2311.08820*, 2023.

[6] András Hegyi. *Model predictive control for integrating traffic control measures*. Netherlands TRAIL Research School, 2004.

[7] Hong K Lo, Elbert Chang, and Yiu Cho Chan. Dynamic network traffic control. *Transportation Research Part A: Policy and Practice*, 35(8):721–744, 2001.

[8] Xiao-Yun Lu, Pravin Varaiya, Roberto Horowitz, Dongyan Su, and Steven E Shladover. A new approach for combined freeway variable speed limits and coordinated ramp metering. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 491–498. IEEE, 2010.

[9] Joseph A Wattleworth. Peak-period analysis and control of a freeway system. Technical report, Texas Transportation Institute San Antonio, TX, USA, 1965.

[10] Andreas Hegyi, Bart De Schutter, Hans Hellendoorn, and T Van Den Boom. Optimal coordination of ramp metering and variable speed control-an mpc approach. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, volume 5, pages 3600–3605. IEEE, 2002.

[11] Chong Wang, Yang Xu, Jian Zhang, and Bin Ran. Integrated traffic control for freeway recurrent bottleneck based on deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):15522–15535, 2022.

[12] Yu Han, Meng Wang, Linghui Li, Claudio Roncoli, Jinda Gao, and Pan Liu. A physics-informed reinforcement learning-based strategy for local and coordinated ramp metering. *Transportation Research Part C: Emerging Technologies*, 137:103584, 2022.

[13] Bao-Lin Ye, Weimin Wu, Keyu Ruan, Lingxi Li, Tehuan Chen, Huimin Gao, and Yaobin Chen. A survey of model predictive control methods for traffic signal control. *IEEE/CAA Journal of Automatica Sinica*, 6(3):623–640, 2019.

[14] Dario Piga, Marco Forgione, Simone Formentin, and Alberto Bemporad. Performance-oriented model learning for data-driven mpc design. *IEEE control systems letters*, 3(3):577–582, 2019.

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

[16] Ali Mesbah, Kim P Wabersich, Angela P Schoellig, Melanie N Zeilinger, Sergio Lucia, Thomas A Badgwell, and Joel A Paulson. Fusion of machine learning and mpc under uncertainty: What advances are on the horizon? In *2022 American Control Conference (ACC)*, pages 342–357. IEEE, 2022.

[17] Sebastien Gros and Mario Zanon. Learning for mpc with stability & safety guarantees. *Automatica*, 146:110598, 2022.

[18] Jie Wang and Youmin Zhang. A tutorial on gaussian process learning-based model predictive control. *arXiv preprint arXiv:2404.03689*, 2024.

[19] Bruno RO Floriano, Alessandro N Vargas, João Y Ishihara, and Henrique C Ferreira. Neural-network-based model predictive control for consensus of nonlinear systems. *Engineering Applications of Artificial Intelligence*, 116:105327, 2022.

[20] JianCheng Long, ZiYou Gao, HuaLing Ren, and AiPing Lian. Urban traffic congestion propagation and bottleneck identification. *Science in China Series F: Information Sciences*, 51(7):948–964, 2008.

[21] Gang-Len Chang and Hua Xiang. The relationship between congestion levels and accidents. Technical report, 2003.

[22] Gerald Berger, Peter H Feindt, Erling Holden, and Frieder Rubik. Sustainable mobility—challenges for a complex transition, 2014.

[23] IE Agureev, KP Andreev, EV Ionov, A Ya Svistunova, and VV Terentyev. The use of intelligent systems when regulating road traffic. In *IOP Conference Series: Materials Science and Engineering*, volume 832, page 012090. IOP Publishing, 2020.

[24] Michel André and Ulf Hammarström. Driving speeds in europe for pollutant emissions estimation. *Transportation Research Part D: Transport and Environment*, 5(5):321–335, 2000.

[25] Michael AP Taylor. Dense network traffic models, travel time reliability and traffic management. ii: Application to network reliability. *Journal of Advanced Transportation*, 33(2):235–251, 1999.

[26] Muhammad Sameer Sheikh and Yinqiao Peng. A comprehensive review on traffic control modeling for obtaining sustainable objectives in a freeway traffic environment. *Journal of Advanced Transportation*, 2022(1):1012206, 2022.

[27] Jaume Barceló. Models, traffic models, simulation, and traffic simulation. *Fundamentals of traffic simulation*, pages 1–62, 2010.

[28] Bidoura Khondaker and Lina Kattan. Variable speed limit: an overview. *Transportation Letters*, 7(5):264–278, 2015.

[29] Solomon Kidane Zegeye, Bart De Schutter, Johannes Hellendoorn, and Ewald A Breunesse. Variable speed limits for area-wide reduction of emissions. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 507–512. IEEE, 2010.

[30] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Yibing Wang. Review of road traffic control strategies. *Proceedings of the IEEE*, 91(12):2043–2067, 2003.

[31] Xiao-Yun Lu, Tony Z Qiu, Pravin Varaiya, Roberto Horowitz, and Steven E Shladover. Combining variable speed limits with ramp metering for freeway traffic control. In *Proceedings of the 2010 american control conference*, pages 2266–2271. IEEE, 2010.

[32] José Ramón Domínguez Frejo and Eduardo Fernández Camacho. Global versus local mpc algorithms in freeway traffic control with ramp metering and variable speed limits. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1556–1565, 2012.

[33] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[34] Marco A Wiering et al. Multi-agent reinforcement learning for traffic light control. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML'2000)*, pages 1151–1158, 2000.

[35] Mohsen Davarynejad, Andreas Hegyi, Jos Vrancken, and Jan van den Berg. Motorway ramp-metering control with queuing consideration using q-learning. pages 1652–1658, 2011.

[36] Ahmed Fares and Walid Gomaa. Freeway ramp-metering control based on reinforcement learning. pages 1226–1231, 2014.

[37] Edouard Ivanjko, Daniela Koltovska Nečoska, Martin Gregurić, Miroslav Vujić, Goran Jurković, and Sadko Mandžuka. Ramp metering control based on the q-learning algorithm. *Cybernetics and Information Technologies*, 15(5):88–97, 2015.

[38] Kasra Rezaee, Baher Abdulhai, and Hossam Abdelgawad. Application of reinforcement learning with continuous state space to ramp metering in real-world conditions. pages 1590–1595, 2012.

[39] Kasra Rezaee, Baher Abdulhai, and Hossam Abdelgawad. Self-learning adaptive ramp metering: Analysis of design parameters on a test case in toronto, canada. *Transportation research record*, 2396(1):10–18, 2013.

[40] Francois Belletti, Daniel Haziza, Gabriel Gomes, and Alexandre M Bayen. Expert level control of ramp metering based on multi-task deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 19(4):1198–1207, 2017.

[41] Zhibin Li, Pan Liu, Chengcheng Xu, Hui Duan, and Wei Wang. Reinforcement learning-based variable speed limit control strategy to reduce traffic congestion at freeway recurrent bottlenecks. *IEEE transactions on intelligent transportation systems*, 18(11):3204–3217, 2017.

[42] Zhibin Li, Chengcheng Xu, Yanyong Guo, Pan Liu, and Ziyuan Pu. Reinforcement learning-based variable speed limits control to reduce crash risks near traffic oscillations on freeways. *IEEE Intelligent Transportation Systems Magazine*, 13(4):64–70, 2020.

[43] Chong Wang, Jian Zhang, Linghui Xu, Linchao Li, and Bin Ran. A new solution for freeway congestion: Cooperative speed limit control using distributed reinforcement learning. *IEEE Access*, 7:41947–41957, 2019.

[44] Yuankai Wu, Huachun Tan, and Bin Ran. Differential variable speed limits control for freeway recurrent bottlenecks via deep reinforcement learning. *arXiv preprint arXiv:1810.10952*, 2018.

[45] Yuankai Wu, Huachun Tan, Lingqiao Qin, and Bin Ran. Differential variable speed limits control for freeway recurrent bottlenecks via deep actor-critic algorithm. *Transportation research part C: emerging technologies*, 117:102649, 2020.

[46] Delft University of Technology. Motus. http://homepage.tudelft.nl/05a3n/, 2015.

[47] Daniel Krajzewicz, Georg Hertkorn, Christian Rössel, and Peter Wagner. Sumo (simulation of urban mobility)-an open-source traffic simulation. In *Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002)*, pages 183–187, 2002.

[48] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[49] Thorsten Schmidt-Dumont and Jan H van Vuuren. Decentralised reinforcement learning for ramp metering and variable speed limits on highways. *IEEE Transactions on Intelligent Transportation Systems*, 14(8):1, 2015.

[50] Stefano Albrecht and Peter Stone. *Multiagent learning: foundations and recent trends*, volume 223. 2017.

[51] Mario Zanon and Sebastien Gros. Safe reinforcement learning using robust mpc. *IEEE Transactions on Automatic Control*, 66(8):3638–3652, 2021.

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

[53] Markos Papageorgiou, Ioannis Papamichail, Albert Messmer, and Yibing Wang. Traffic simulation with metanet. *Fundamentals of traffic simulation*, pages 399–430, 2010.

[54] Serge Paul Hoogendoorn. Multiclass continuum modelling of multilane traffic flow. *PhD thesis*, 1999.

[55] Matthew Ellis, Jinfeng Liu, and Panagiotis D Christofides. Economic model predictive control. *Springer*, 5(7):65, 2017.

[56] Sébastien Gros and Mario Zanon. Data-driven economic nmpc using reinforcement learning. *IEEE Transactions on Automatic Control*, 65(2):636–648, 2019.

[57] Lucian Busoniu, Robert Babuska, Bart De Schutter, and Damien Ernst. *Reinforcement learning and dynamic programming using function approximators*. CRC press, 2017.

[58] Michail G Lagoudakis, Ronald Parr, and Michael L Littman. Least-squares methods in reinforcement learning for control. In *Methods and Applications of Artificial Intelligence: Second Hellenic Conference on AI, SETN 2002 Thessaloniki, Greece, April 11–12, 2002 Proceedings 2*, pages 249–260. Springer, 2002.

[59] Steven D Whitehead and Dana H Ballard. Learning to perceive and act by trial and error. *Machine Learning*, 7:45–83, 1991.

[60] Christof Büskens and Helmut Maurer. Sensitivity analysis and real-time optimization of parametric nonlinear programming problems. *Online Optimization of Large Scale Systems*, pages 3–16, 2001.

[61] Hossein Nejatbakhsh Esfahani, Arash Bahari Kordabad, and Sébastien Gros. Approximate robust nmpc using reinforcement learning. In *2021 European Control Conference (ECC)*, pages 132–137. IEEE, 2021.

[62] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.