UNIVERSITY OF OKLAHOMA
GRADUATE COLLEGE

OPTIMAL CONTROL FOR SELF-DRIVING PATH PLANNING AND THERMAL MANAGEMENT

A DISSERTATION
SUBMITTED TO THE GRADUATE FACULTY
in partial fulfillment of the requirements for the
Degree of
DOCTOR OF PHILOSOPHY

by ZIBA ARJMANDZADEH

Norman, Oklahoma

2025

Downloaded from SHAREOK. Persistent Handle URL: https://hdl.handle.net/11244/341786

OPTIMAL CONTROL FOR SELF-DRIVING PATH PLANNING AND THERMAL MANAGEMENT

A DISSERTATION APPROVED FOR THE
SCHOOL OF AEROSPACE AND MECHANICAL ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Bin Xu, Chair

Dr. Arif Sadri

Dr. Dong Zhang

Dr. Pejman Kazempoor

Dr. Hamidreza Shabgard

Dr. Iman Ghamarian

©Copyright by ZIBA ARJMANDZADEH 2025

All Rights Reserved.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Dr. Bin Xu, for his invaluable guidance, continuous support, and encouragement throughout my PhD journey. His insightful feedback, patience, and dedication have been fundamental to the completion of this dissertation and to my growth as a researcher.

I am also sincerely thankful to the members of my dissertation committee — Dr. Sadri, Dr. Zhang, Dr. Ghamarian, Dr. Kazempoor, and Dr. Shabgard — for their time, constructive feedback, and thoughtful suggestions, which have greatly enriched the quality of this work.

I would like to extend my appreciation to my colleagues and friends in the Mobility Intelligence Lab (MiLa) for their collaboration, discussions, and camaraderie, which made this experience both intellectually stimulating and personally rewarding.

Finally, I owe my deepest gratitude to my family for their unwavering love, understanding, and encouragement throughout this long journey. Their constant support has been my greatest source of strength and motivation.

Table of Contents

Acknowledgments iv

List of Tables vii

List of Figures viii

Abstract x

1 Introduction 1
1.1 Autonomous Vehicle Local Path Planning 1
1.2 Battery Thermal Management 6

2 A Comparative Study on Autonomous Vehicle Local Path Planning Through Model Predictive Control and A Sampling-based Method . 12
2.1 Problem Description 12
2.2 Environment Model 13
2.3 Model Predictive Control 15
2.4 Sampling-based Method 17
2.5 Experimental Data Collection 21
2.6 Image Processing 22
2.7 Performance Appraisal 23
2.8 Results Analysis 24
2.9 Summary 30

3 A Lyapunov Optimization-Based Approach to Autonomous Vehicle Local Path Planning 31
3.1 Environment 31
3.2 Path Planning Optimization Problem 32
3.3 Reformulation 34
3.4 Lyapunov Optimization Model 36
3.5 Sampling-based Method 40
3.6 Simulation and Results 44
3.7 Summary 49

4 Lyapunov Optimization-based Approach Validation in CARLA Simulator . . . 50
4.1 CARLA, ROS, and CARLA-ROS bridge . . . 50
4.2 The Dynamic Model . . . 51
4.3 Extended Kalman Filter . . . 55
4.3.1 EKF Formulation . . . 56
4.4 The Lyapunov Model with Kalman Filter . . . 58
4.4.1 Optimization Problem . . . 61
4.4.2 Finding p_k^r and p_k^l . . . 62
4.4.3 Reformulation . . . 65
4.4.4 Lyapunov Optimization Model . . . 67
4.5 Results . . . 71

5 Electric Vehicle Battery Thermal Management Under Extreme Fast Charging with Deep Reinforcement Learning . . . 77
5.1 Thermal System Model . . . 78
5.2 Electrical System Model . . . 83
5.3 Optimization Approaches . . . 84
5.3.1 Markov Decision Process (MDP) . . . 84
5.3.2 Modeling Thermal System with MDP . . . 85
5.3.3 Proximal Policy Optimization . . . 88
5.3.4 Model Predictive Control . . . 89
5.3.5 Proportional–Integral–Derivative (PID) Controller . . . 90
5.4 Simulation and Results . . . 91
5.4.1 PPO Hyperparameter Selection . . . 92
5.4.2 Training The PPO Agent . . . 93
5.4.3 Results . . . 94
5.4.4 Validation . . . 101
5.4.5 Model Error . . . 102

6 Conclusion . . . 105

Appendix A. Proof of Theorem 3.1 . . . 108

Appendix B. Proof of Theorem 3.2 . . . 112

Appendix C. Proof of Theorem 4.1 . . . 117

Appendix D. Proof of Theorem 4.2 . . . 122

Reference List . . . 126

List of Tables

2.1 The encoded data of the bird-eye-view of the surrounding environment of the ego vehicle 13

3.1 The computation time of one step (each step is Δt long), smoothness, and safety of the three approaches. The computation time is normalized. 49

5.1 Thermal System Parameters 80

5.2 Parameters of Capacity Loss and Resistance Rise Equations Cordoba-Arenas et al. (2015) 87

5.3 The results of PPO, MPC, and PIDs with battery temperature setpoints of 36.5℃ and 35℃. 99

5.4 Percentage difference between the results of PPO and other methods . 100

List of Figures

2.1 Sample image from data 14

2.2 Road geometry 15

2.3 R_k components $L^i_{x,k}$ and $L^i_{y,k}$ 17

2.4 a) Cartesian coordinate, and b) Frenet coordinate 18

2.5 Environment perception sensors, human-machine interface and computation unit of the Nissan Leaf test car at The University of Oklahoma Mobility and Intelligence Lab 22

2.6 The results of scenario 1 24

2.7 The results of scenario 2 28

2.8 The results of scenario 3 29

3.1 r_k components $l^i_{x,k}$ and $l^i_{y,k}$ 35

3.2 The results for scenario 1 43

3.3 The results for scenario 2 46

3.4 The results for scenario 3 47

4.1 Model's output vs CARLA's data 54

4.2 Model's output vs CARLA's data after applying EKF 60

4.3 Adding the obstacle to the road boundary 63

4.4 Finding the rotation angle 64

4.5 Rotated Spline curves. The blue, thick, stepwise curve is the reference line, while the pink curve is the optimal path found by LO. Note that the optimal path respects the bounds of p_k^r and p_k^l. 64

4.6 Ego vehicle in CARLA 72

4.7 The LO local path planning result in CARLA's Town 5 environment. . 73

4.8 The PID local path planning result in CARLA's Town 5 environment. . 74

4.9 LO and PID local path planning and target speed tracking results. . . . 75

5.1 EV TMS configuration including A/C system (i.e., compressor, condenser, and reservoir), indirect battery cooling loop (i.e., chiller, radiator, and battery pack). 79

5.2 CC-CV profile used in this chapter 79

5.3 Sensitivity Analysis Results 92

5.4 PPO reward convergence during train. 93

5.5 Battery Temperature Elevation. 95

5.6 Coolant and refrigerant mass flow rates, as well as pump and compressor power consumption in all cases. 96

5.7 Fresh battery aging after one fast charging session in different cases . . 99

5.8 PPO results with 6C charging current 101

5.9 The impact of model error on MPC results 103

5.10 Battery aging after 1000 cycles in PPO, MPC with 10% and 1% model errors 104

Abstract

In the first part of this dissertation, a novel local path planning approach for autonomous vehicles (AVs) is proposed, based on Lyapunov Optimization (LO). The performance of the developed LO-based planner is benchmarked against two established techniques: Model Predictive Control (MPC) and a sampling-based method. To support real-world validation, an AV platform was developed using a modified 2015 Nissan Leaf S. The vehicle was equipped with a drive-by-wire system, perception sensors, and onboard computation units. Data collection was carried out in Norman, Oklahoma, enabling a comparative evaluation of the three control algorithms under real traffic conditions, including both straight and curved urban roads. In order to reduce system cost and enhance deployment feasibility, a vision-only perception solution was adopted for object detection and bird's-eye view coordinate generation. Using the collected data, each method—LO, MPC, and the sampling-based approach—was independently deployed to generate safe and smooth driving paths. The comparison focused on key performance metrics: path smoothness, safety, and computation time. Results show that the proposed LO method significantly outperforms the baseline approaches, achieving up to a 97.5% less computation time. Additionally, the LO approach was tested in the high-fidelity CARLA simulation environment, providing further validation prior to real-world deployment.

It is worth noting that, during the initial phase of this research, MPC and the sampling-based method were directly compared through real-world experiments. Findings revealed that MPC consistently outperforms the sampling-based method, particularly in terms of speed and safety.

In the second part of the dissertation, a new optimization strategy for Battery

Thermal Management Systems (BTMS) in electric vehicles (EVs) is introduced. The proposed method employs a model-free deep reinforcement learning (RL) framework designed to operate under extreme fast charging (XFC) conditions. The optimization objective is to minimize both battery degradation and the power consumption of the BTMS. The thermal dynamics of the system are modeled in detail, accounting for the air-conditioning refrigerant loop and the indirect liquid cooling loop for battery thermal regulation. The proposed RL method is implemented on a real battery pack and compared against two benchmark strategies: MPC, representing optimal control, and Proportional-Integral-Derivative (PID) control, representing traditional tracking control. The results show that, when provided with an accurate system model, MPC can perform comparably to the proposed RL approach. However, the RL method demonstrates a significant advantage in computational efficiency—operating up to 48 times faster than MPC during testing. This performance gain is attributed to the model-free nature of the RL algorithm, which eliminates the need for complex system modeling. Moreover, the RL approach outperforms PID control in both key metrics: battery degradation is reduced by up to 1.05%, and BTMS power consumption is lowered by up to 43.68%.

Chapter 1

Introduction

1.1 Autonomous Vehicle Local Path Planning

Developments in vehicle technology, together with continuous advancements in driver assistance systems, have accelerated the emergence of autonomous vehicles (AVs) (Ammour et al., 2022). AVs are gaining significant traction due to their potential to improve road safety, enhance traffic efficiency, and reduce environmental pollution through optimized driving behavior and energy consumption (Ho et al., 2023; Han et al., 2023). This dissertation focuses on AV local path planning, which represents one of the three essential pillars of autonomous navigation—alongside environment perception and control—that collectively form the foundation of AV intelligent behavior (Wang et al., 2022b).

In addition to methodological contributions, this research includes the creation of a real-world dataset collected using an autonomous 2015 Nissan Leaf S platform operating under urban driving conditions. The vehicle, equipped with sensors and control actuators, was driven manually around Norman, Oklahoma, while sensor data were recorded for algorithm development and testing. This contrasts with many existing studies that rely on publicly available datasets and predominantly focus on highway scenarios, where driving environments are relatively structured and predictable (Huang et al., 2023a,b, 2021). By comparison, urban environments present greater variability

and uncertainty, requiring more sophisticated decision-making capabilities.

Autonomous vehicles must handle diverse and dynamic driving scenarios such as curved roads, intersections, pedestrian crossings, obstacles, and highway ramp merging. These complex cases have been investigated in numerous studies (Huang et al., 2023b; Buehler et al., 2009; Claussmann et al., 2019; Li and Xu, 2019), all seeking to generate safe, comfortable, and collision-free trajectories that follow a reference path while accounting for static and dynamic obstacles.

A variety of path planning approaches have been explored in the literature (Ammour et al., 2022), with the most prominent categories including sampling-based methods, Model Predictive Control (MPC), and deep learning-based techniques. Each approach offers distinct advantages and challenges in terms of optimality, computational efficiency, and real-time feasibility.

Sampling-based methods (Werling et al., 2010) are widely used due to their flexibility and ability to emulate human-like driving behavior. A key advantage of sampling-based approaches lies in the use of Frenet frame coordinates, which effectively decouple the lateral and longitudinal motions of the vehicle, thereby simplifying the mathematical representation of curved trajectories. The pioneering work in (Werling et al., 2010) introduced sampling-based optimal trajectory generation, laying the foundation for numerous subsequent studies. For example, (Huang et al., 2023a) applied a Frenet-frame-based sampling method for path planning that accounts for static obstacles, while (Wang et al., 2022a) designed a collision-free sampling-based model validated through simulation. Despite their conceptual simplicity and versatility, these methods often suffer from high computational overhead because multiple candidate trajectories are generated at every time step, with only one ultimately selected for execution. This redundancy poses a serious limitation for real-time implementation in complex urban environments.

Model Predictive Control (MPC) addresses some of these limitations by directly optimizing the vehicle's trajectory through a constrained optimization framework that explicitly considers system dynamics, nonlinearities, and physical limits. MPC's weighted cost function structure provides flexibility in tuning various performance aspects such as comfort, energy efficiency, and safety, making it an attractive choice for AV control. Numerous studies have applied MPC to AV path planning and control. For instance, (Liu et al., 2017) integrates potential field theory and Farkas' lemma to achieve collision avoidance, while (Reiter et al., 2023) develops a nonlinear MPC formulation to reduce computational delay. Ref. (Lattarulo and Pérez Rastelli, 2021) introduces a two-stage path planning approach, where a preliminary local path is generated by ignoring obstacles and subsequently refined through MPC optimization. Likewise, (Zhang et al., 2021) employs a gradient descent optimizer in the Frenet frame to compute a collision-free path, followed by a PI–MPC controller for trajectory tracking. A hybrid Cartesian–Frenet MPC framework is proposed in (Xing et al., 2022), where the control model operates in Cartesian space while road constraints are defined in Frenet coordinates, enabling comprehensive testing on delivery and passenger vehicles. Further, an improved particle swarm optimization–based MPC method is proposed in (Zuo et al., 2020) for cooperative control in local planning and path tracking of intelligent vehicles. Despite these advancements, MPC often requires long prediction horizons to achieve acceptable control performance, significantly increasing the computational burden and hindering real-time application in fast-changing traffic environments.

Deep learning-based approaches have emerged as a modern alternative to classical optimization techniques. Once trained, deep neural networks enable near-instantaneous inference, making them suitable for real-time path planning tasks. A detailed survey in (Teng et al., 2023) reviews the latest developments in deep learning-based trajectory generation, highlighting applications in collision avoidance, overtaking, lane changing,

and safe navigation (Cheng et al., 2014; Ming et al., 2021; Iberraken et al., 2019; Mo and Lv, 2023). However, these methods introduce new challenges. One major limitation is the lack of interpretability of neural networks, which complicates safety validation and legal accountability in the event of system failures. Additionally, the large data requirements for training such models restrict their adaptability and pose difficulties in data collection, labeling, and processing, particularly for rare or safety-critical driving scenarios.

To overcome the computational inefficiencies of sampling-based and MPC approaches, as well as the data dependency and opacity of deep learning methods, this dissertation proposes a novel local path planning solution based on the Lyapunov Optimization (LO) framework. The LO method is distinguished by its computational efficiency, mathematical transparency, and theoretical guarantees of system stability (Abbasi et al., 2022). These attributes make LO particularly suitable for real-time AV path planning in dense urban traffic, where quick and reliable decision-making is crucial. In this dissertation, the LO approach is employed for local path planning and compared against both MPC and sampling-based benchmarks to evaluate performance, efficiency, and robustness.

Originally introduced in (Neely, 2010), the LO framework decomposes an optimization problem into multiple independent subproblems, each corresponding to a specific time step. This decomposition significantly enhances computational speed (Abbasi et al., 2022) and reduces dependency on future information, making the method highly scalable. The core of LO lies in the concept of a Lyapunov function, a positive-definite function that increases as a system moves toward instability. By minimizing the Lyapunov drift, the system can be kept stable over time. The foundational work in (Neely, 2010) extends this principle by introducing a drift-plus-penalty formulation, where a penalty term is added to the Lyapunov drift to jointly ensure stability and performance

optimization.

Applying LO to the AV local path planning problem involves reformulating the planning task as a queue stability problem by constructing virtual queues to represent system dynamics. The LO framework then minimizes the upper bound of the drift-plus-penalty expression, balancing stability and performance in each decision step. This structure enables a single-step optimization approach, reducing redundancy while ensuring convergence toward near-optimal solutions.

Several studies have explored LO in related domains. For example, (Bao et al., 2016) investigates Lyapunov-based stability and synchronization of fractional-order complex-valued neural networks with time delay, while (Udhayakumar et al., 2022) develops finite- and fixed-time stability theorems for fractional-order impulsive discontinuous systems using Lyapunov functional methods. In transportation and networked systems, (Wang et al., 2014) applies LO to traffic control for optimizing global path utilization, and (Wan et al., 2019) uses LO for online edge-processing scheduling with unmanned aerial vehicles acting as mobile edge servers. Similarly, (Clark, 2023) employs LO in robotic network localization under harsh environmental conditions, and (Wei et al., 2021) applies it to intelligent traffic control systems. However, to the best of our knowledge, the LO method has never been applied to AV local path planning, making this dissertation the first to bridge that gap.

In this research, the LO algorithm transforms the local path planning problem into a queue stability framework, incorporating queue backlogs directly into the objective function through a positive-definite Lyapunov function. This leads to the construction of a drift-plus-penalty term whose upper bound is minimized at each decision step. The theoretical foundations of this approach are presented in two stages: the first theorem derives the upper bound of the drift-plus-penalty term, and the second theorem demonstrates that minimizing this bound yields a solution that closely approximates

the global optimum.

In summary, this dissertation contributes to the field of autonomous vehicle research by introducing a Lyapunov Optimization-based local path planning strategy that achieves rapid computation, theoretical stability, and high interpretability. The primary contributions are summarized as follows:

- The problem of the local path planning of an AV is solved via LO for the first time.
- The optimization problem is transformed into a queue stability problem in order to apply LO.
- The standard LO algorithm is modified to change the desired state values dynamically from one time step to another.
- The proposed approach drastically reduces computation time.

1.2 Battery Thermal Management

Electric vehicles (EVs) have garnered considerable attention in recent years as a promising solution to the challenges of energy sustainability and environmental pollution. Governments across the globe are investing heavily to accelerate EV adoption and infrastructure development. For instance, the U.S. Bipartisan Infrastructure Bill allocates $15 billion toward the development of low-emission buses and ferries, as well as the establishment of a nationwide network of plug-in EV chargers (House, 2021; CNN, 2021). The consensus in the academic community is that EVs can substantially reduce carbon footprints and greenhouse gas emissions, particularly when the electricity that powers them is derived from renewable energy sources (Abbasi et al., 2022). Furthermore, the decline in EV purchase prices—partly due to the falling cost

of batteries—alongside the expansion of fast-charging infrastructure to enable long-distance travel, is expected to mitigate consumer range anxiety and further accelerate EV adoption (Xu and Arjmandzadeh, 2023).

Lithium-ion (Li-ion) batteries are the predominant energy storage technology for EVs because of their high energy and power densities, low self-discharge rate, and long cycle life (Tan et al., 2021). Nevertheless, one of the major challenges impeding the widespread adoption of EVs is the extended charging time of Li-ion batteries compared to the short refueling duration of conventional internal combustion engine vehicles (Abbasi and Zhang, 2021). While increasing charging power can reduce charging time, it simultaneously accelerates battery degradation and shortens the battery's lifespan (Abbasi and Zhang, 2021). Battery degradation is strongly correlated with temperature, making thermal management a critical aspect of EV performance and longevity (Fan et al., 2021).

The optimal operating temperature range for Li-ion batteries typically lies between 25°C and 40°C, with a maximum permissible temperature gradient of approximately 5°C across the battery pack (Akbarzadeh et al., 2021). Deviations from this range, particularly during high-rate or fast-charging events, can lead to excessive heat generation (Chen et al., 2019), thereby increasing the risk of thermal runaway, a catastrophic failure mode that threatens both battery safety and system integrity. Consequently, an effective Battery Thermal Management System (BTMS) is indispensable for the safe and efficient operation of EVs. The BTMS not only safeguards the battery pack from overheating but also plays a pivotal role in ensuring overall vehicle energy efficiency and safety.

From a systems engineering perspective, BTMSs are generally classified as active or passive systems (Deng et al., 2018). Active BTMSs consume external energy to heat or cool the battery pack, whereas passive systems rely on natural heat dissipation

mechanisms and phase change materials to manage temperature without direct power input (Patel and Rathod, 2020). Although passive systems offer energy savings, active methods are often preferred for their controllability and reliability under dynamic operating conditions. Various BTMS configurations have been explored, including air cooling, liquid cooling, phase change material (PCM) cooling, heat pipe cooling, and hybrid combinations thereof (Tan et al., 2021; He et al., 2023). Among these, liquid cooling has emerged as the most widely adopted approach in modern EVs due to its superior thermal performance and compact design flexibility.

Liquid-cooled BTMSs can be categorized as direct or indirect systems (Wu et al., 2019). In direct systems, the battery cells are immersed in a dielectric coolant that provides uniform temperature distribution. Conversely, indirect systems employ cooling plates or embedded channels through which the coolant circulates, extracting heat from the cells via conduction. Different EV manufacturers employ distinctive liquid cooling topologies. For example, Tesla's "ribbon cooling system" circulates coolant through a serpentine metallic tube interwoven between cells to achieve uniform thermal distribution (Bower, 2022). The Audi e-Tron (Audi, 2021), Ford Mustang Mach-E (Munro, 2021), and Chevrolet Bolt (Morris, 2018) adopt bottom cooling architectures, where a cold plate is mounted beneath the battery pack to facilitate heat transfer. Meanwhile, Mercedes-AMG applies a direct immersion cooling design, enabling superior heat removal and thermal uniformity (Dinh, 2021).

Given the critical role of BTMSs in EV performance, numerous research efforts have been devoted to improving their design, modeling, and control. For instance, (Xu and Arjmandzadeh, 2023) presents a parametric study on vehicle-level thermal management employing five PID controllers that regulate cabin and battery temperatures, refrigerant superheat, and subcool temperatures. However, this approach is not formulated as an optimization problem and does not consider battery fast-charging

conditions. Similarly, (Hannan et al., 2019) proposes a robust controller using particle swarm optimization (PSO) integrated with fuzzy logic. The authors demonstrate performance improvements in overshoot and undershoot by 0.49% and 0.97%, respectively, compared to a standard PID controller. Nevertheless, their work focuses on tracking control rather than optimizing BTMS behavior, and their model comprises only four battery modules totaling 40 cells.

Another study, (Xia et al., 2018), introduces a fuzzy control-based BTMS for temperature regulation; however, it suffers from limited flexibility and requires significant expert knowledge for fuzzy rule design. In (Afzal and Ramis, 2020), the authors combine a genetic algorithm (GA), PSO, and fuzzy logic to maintain battery temperature within the desired range, but the method's applicability is constrained by long computation times. A nonlinear Model Predictive Control (MPC)-based BTMS is presented in (Lopez-Sanz et al., 2016), benchmarked against a finite-state-machine controller. The MPC approach reduces operational costs by 30%, yet it is only evaluated under standard drive cycles, not under fast-charging conditions, and lacks detailed battery aging characterization.

Air-cooling-based BTMS designs have also been explored. For instance, (Fan et al., 2021) proposes an air-cooling control strategy to enhance the battery's state of health. However, its degradation model only accounts for discharge cycles and assumes slow overnight charging, excluding fast-charging effects. Similarly, (Yan et al., 2023) develops a control framework combining a modified GA and support vector regression, yet the model omits thermal dynamics and fails to address BTMS performance during fast charging. In (Sundararaju et al., 2023), an air-cooling BTMS is simulated to maintain the battery within an acceptable temperature range, but the work lacks detailed mathematical modeling and reveals no component-level simulation details.

Further research, such as (Fan et al., 2023), examines concurrent fast charging and

thermal management; however, the study focuses on a small 16-cell pack, which limits real-world scalability. In (Chen et al., 2021), neural networks are employed to optimize charging current and coolant mass flow rate, but the study restricts the flow rate to three discrete levels and limits charging to 2.5C—well below the extreme fast charging targets of 6C or higher set by the U.S. Department of Energy (Michelbacher et al., 2017). Reinforcement learning (RL)-based controllers have also been explored. For example, (Choi et al., 2022) introduces an RL-driven BTMS controller, but omits system dynamics, while (Huang et al., 2022) applies RL for EV energy management and battery degradation minimization. Nonetheless, (Huang et al., 2022) neglects the interconnection between thermal subsystems and evaluates performance only under drive cycles, not during fast charging. In (Zhao et al., 2023), a two-stage BTMS control framework combines dynamic programming (DP) for determining target temperatures with a fuzzy-PID controller to maintain them. However, DP is computationally expensive and unsuitable for real-time control, and again, the approach is validated only for drive cycles, not fast-charging conditions.

The above literature review reveals several important gaps. First, most studies treat BTMS control as a temperature tracking problem, rather than optimizing the underlying thermal and energy behavior of the system. Second, the majority of works focus on thermal management during vehicle operation, neglecting scenarios of extreme fast charging where heat generation is most intense. Third, while many papers (Su et al., 2021; Xu et al., 2022; Shan et al., 2023) investigate the design and parametric optimization of BTMS hardware—such as coolant type, channel geometry, or heat exchanger shape—fewer explore optimal control strategies that explicitly balance thermal safety, energy efficiency, and battery degradation.

Accordingly, two major research gaps are identified in the current literature: i) A lack of comprehensive studies on the optimal control of EV BTMS that jointly minimize

battery degradation and BTMS energy consumption, and ii) A lack of research focusing on BTMS optimization under extreme fast charging conditions, where temperature management is most critical.

To address these challenges, this dissertation considers an indirect active liquid-cooling BTMS architecture and introduces a deep reinforcement learning (RL)-based control framework. The proposed controller dynamically optimizes BTMS operation during extreme fast charging to simultaneously minimize battery aging and reduce BTMS power consumption.

The main contributions of this dissertation can be summarized as follows:

- A deep RL algorithm is developed and trained under battery pack extreme fast charging using the proximal policy optimization (PPO) method to minimize the battery degradation and power consumption of the BTMS.
- A detailed BTMS model is constructed, which integrates the battery pack, chiller, refrigerator, and radiator.
- The battery aging characteristics are considered through a capacity fade model and a power fade model.
- A nonlinear MPC-based controller is also developed as a baseline method by which the performance of the proposed deep RL algorithm is evaluated.
- The results of the proposed deep RL and MPC are compared with those of a PID controller to showcase the performance of optimization-based algorithms in contrast with tracking-based control of a BTMS under extreme fast charging.

Chapter 2

A Comparative Study on Autonomous Vehicle Local Path Planning Through Model Predictive Control and A Sampling-based Method

2.1 Problem Description

In this chapter, the local path planning of an AV is presented using MPC. Moreover, a real AV is constructed using a Nissan Leaf and is used to obtain data. The data is gathered through a mounted camera on the vehicle. The raw data is then processed with the help of image processing in Python to detect other vehicles as well as pedestrians on the road. Furthermore, the processed images are converted to a bird-eye view of the surrounding environment of the ego vehicle. The bird-eye view of the environment is stored in csv files containing a matrix whose elements correspond to the pixels of the processed image. In addition, as shown in Table. 2.1, each cell of the spreadsheet data has a value of 0, or 1, ..., or 4, where zero, one, two, three, and four mean nothing, road surface, expert trajectory, vehicle object, and pedestrian, respectively. Note that "expert trajectory" is the path the vehicle had taken when it was driven by a human

Data	0	1	2	3	4
Physical Meaning	Nothing	Road Surface	Expert Trajectory	Vehicle Object	Pedestrian

Table 2.1: The encoded data of the bird-eye-view of the surrounding environment of the ego vehicle

driver.

Fig. 2.1 shows a sample raw image, its corresponding processed image, and the bird-eye-view constructed data from the processed image. In the processed image, rectangular green boxes are used to show the boundaries of the detected objects. A pedestrian is detected on the right of the road. Additionally, green color is also utilized to paint the surface of the ego vehicle's lane and/or the lanes that the ego vehicle is permitted to drive on. In the bird-eye-view version, the grayscale colormap is exploited to represent the data as an image. The vehicles and/or pedestrians are illustrated in dark black, while the road surface where the ego vehicle is permitted to drive and the expert trajectory are shown in two different shades of gray.

2.2 Environment Model

As depicted in Fig. 2.2, suppose that x and y show the positions of each point in the Cartesian coordinate, and θ and v represent the heading angle and speed of the vehicle, respectively. The dynamic equation of the ego vehicle and surrounding cars can be shown as follows:

$$z_{k+1} = z_k + f(z_k, u_k)dt, \tag{2.1}$$

where the state of the problem at time k is $z_k = [x_k, y_k, \theta_k, v_k]^T$, and $f(z_k, u_k) = [v_k cos\theta_k, v_k sin\theta_k, \omega_k, a_k]^T$. Here, $u_k = [a_k, \omega_k]$ is the input of the problem, in which a_k

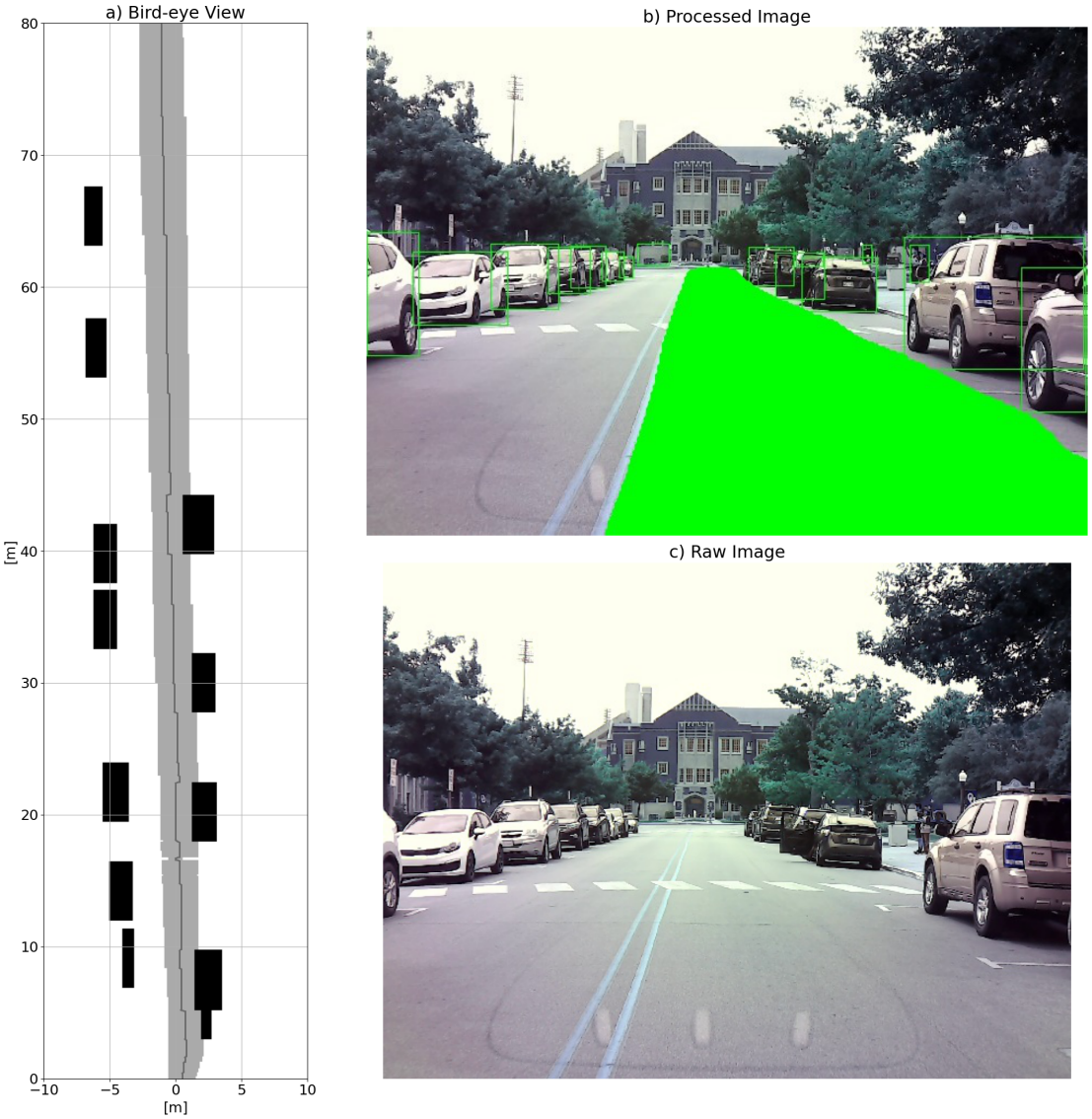


Figure 2.1: Sample image from data

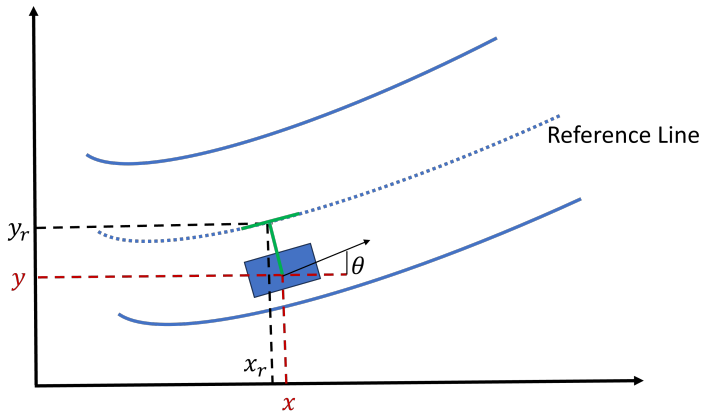


Figure 2.2: Road geometry

and ω_k are the linear acceleration and angular velocity, respectively, and dt is the time step. Additionally, the environmental information such as road waypoints, lane markings, and the surrounding vehicles' positions are obtained through image processing. For instance, in Fig. 2.1, the surrounding vehicles are detected, and a green rectangular bounding box is plotted around each of them.

2.3 Model Predictive Control

MPC is an optimal control method that optimizes a cost function under the constraints of system dynamics in a receding horizon fashion. The goal of the optimization problem is to bring comfort, guarantee the safety of the passengers, avoid collisions, and maintain the desired speed while following the reference trajectory provided by local path planning. Accordingly, distance from the reference line, angular and longitudinal velocity, acceleration and jerk are included in the objective function. Let N be the prediction horizon of MPC, and the constrained non-linear local path planning optimization problem is defined as follows:

$$\min \quad \sum_{k=1}^{N} \left(w_{gd} D_k^2 + w_{ym} R_k^2 + w_v \left(v_d - v_k\right)^2 + w_a a_k^2 + w_y \omega_k^2 + w_j \left(a_k - a_{k-1}\right)^2 \right) \tag{2.2a}$$

$$s.t. \quad z_{k+1} = z_k + f(z_k, u_k)dt, \tag{2.2b}$$

$$-1 \leq a_k \leq 1, \tag{2.2c}$$

$$-5 \leq \omega_k \leq 5, \tag{2.2d}$$

where, D_k denotes Euclidian distance from the reference line, shown in green in Fig. 2.2, and R_k is proposed to safely avoid collision with other vehicles. $\left(v_d - v_k\right)^2$ is the difference between the actual speed and the desired one. a_k^2 and ω_k^2 penalize the large control input, $\left(a_k - a_{k-1}\right)^2$ minimizes longitudinal jerk, improving the comfort of passengers in the vehicle, and constants w_{gd}, w_{ym}, w_v, w_a, w_y and w_j are the weighting factors. R_k^2 regulates the distance from the dynamic obstacle as follows,

$$R_k = \sum_{i=1}^{N_o} \frac{1}{L_{x,k}^i + L_{y,k}^i}, \tag{2.3}$$

$$L_{x,k}^i = \begin{cases} |x_k - p_1^i| & x_k < p_1^i, \\ |x_k - p_2^i| & x_k > p_2^i, \\ 0 & \text{O.W.}, \end{cases} \tag{2.4}$$

where, as illustrated in Fig. 2.3, p_1^i and p_2^i represent the start and end of the i-th object. N_o is the number of objects that are captured by the camera at this time instant. For example, in Fig. 2.1, 11 objects or cars are detected. In addition, R_k ensures the trajectory is collision-free, seeing that when $L_{x,k}^i + L_{y,k}^i$ goes to zero, the value of R_k tends to infinity which creates an extremely large penalty that dwarfs other factors

of the cost function to the point that, for instance, ego vehicle would prefer to stop, $v_k = 0$, to prevent collision if there is no way to go around the obstacle. Consequently, the objective function is designed so that the AV follows the reference line while the trajectory is collision-free, shortest, and smooth.

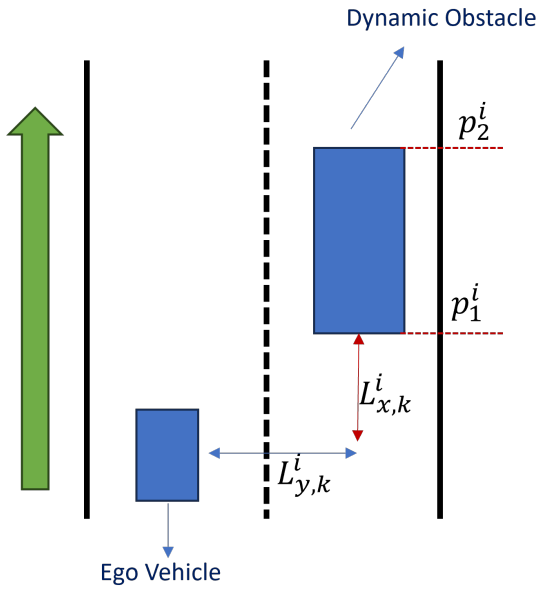


Figure 2.3: R_k components $L_{x,k}^i$ and $L_{y,k}^i$

2.4 Sampling-based Method

Frenet frame coordinate enables a straightforward representation of a reference path since its run length is described with the s axis. This definition of reference path is appropriate for curved roads. In Frenet frame, the longitudinal and lateral displacements, which are represented by s and d, are used for describing the position of any arbitrary point of the reference path. It should be noted that the coordinates s and d are perpendicular to each other at every point of the reference line. This method comprises the following two main steps:

- Constructing several candidate trajectories in each step.

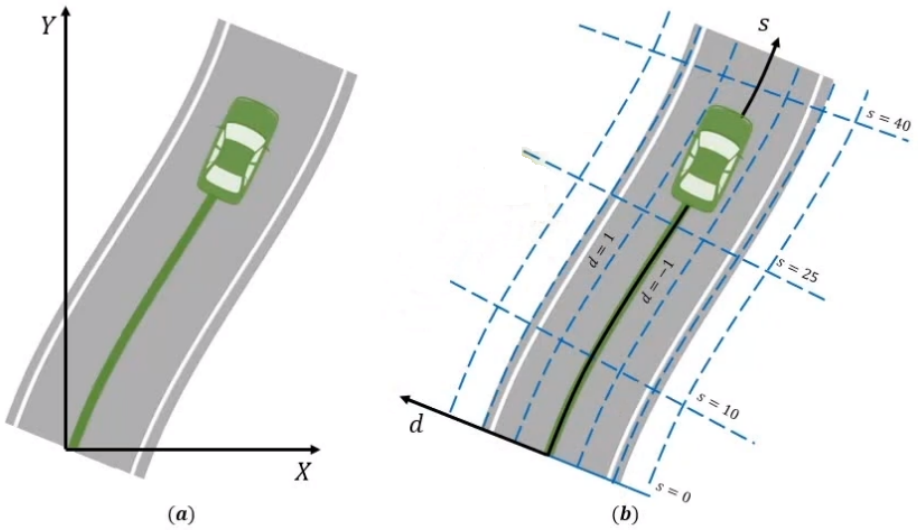


Figure 2.4: a) Cartesian coordinate, and b) Frenet coordinate

- Selecting the best path for the ego vehicle, according to a cost function.

In Frenet coordinate, ego vehicle movement is described by the tangential vector, $\vec{t}_r$, that is at the same direction of longitudinal displacement, and normal vector, $\vec{n}_r$, that is parallel to lateral displacement.

Using Frenet frame, the lateral and longitudinal motion of the vehicle is decoupled, and simultaneously, the driving state space is discretized to generate a set of target states of the vehicle. Based on the initial and target states, a set of lateral and longitudinal motion trajectories are generated by exploiting quintic and quartic polynomials, respectively. In addition, a cost function is formed that comprehensively considers the comfort, safety, and deviation from the reference line. As part of the cost function, a novel method is proposed to quantify the safety of candidate trajectories considering the size of obstacles.

Fig. 2.4 shows the difference between the Frenet coordinate and the Cartesian coordinate. The Frenet coordinate is illustrated in Fig. 2.4b, while the Cartesian

coordinate is displayed in Fig. 2.4a. In Frenet coordinate, a curved path is described by s and d coordinates, described by the following polynomials:

$$d(t) = c_{d_0} + c_{d_1}t + c_{d_2}t^2 + c_{d_3}t^3 + c_{d_4}t^4 + c_{d_5}t^5, \tag{2.5}$$

$$s(t) = c_{s_0} + c_{s_1}t + c_{s_2}t^2 + c_{s_3}t^3 + c_{s_4}t^4, \tag{2.6}$$

where t is time. Each trajectory starts at t_s and ends at t_e under the assumption that t_s and t_e are known. Considering equations (2.5) and (2.6) and their derivatives, with respect to time, at the start and end points, the following set of equations are achieved:

$$\begin{bmatrix} d(t_s) \\ \dot{d}(t_s) \\ \ddot{d}(t_s) \\ d(t_e) \\ \dot{d}(t_e) \\ \ddot{d}(t_e) \end{bmatrix} = \begin{bmatrix} 1 & t_s & t_s^2 & t_s^3 & t_s^4 & t_s^5 \\ 0 & 1 & 2t_s & 3t_s^2 & 4t_s^3 & 5t_s^4 \\ 0 & 0 & 2 & 6t_s & 12t_s^2 & 20t_s^3 \\ 1 & t_e & t_e^2 & t_e^3 & t_e^4 & t_e^5 \\ 0 & 1 & 2t_e & 3t_e^2 & 4t_e^3 & 5t_e^4 \\ 0 & 0 & 2 & 6t_e & 12t_e^2 & 20t_e^3 \end{bmatrix} \begin{bmatrix} c_{d_0} \\ c_{d_1} \\ c_{d_2} \\ c_{d_3} \\ c_{d_4} \\ c_{d_5} \end{bmatrix}, \tag{2.7}$$

$$\begin{bmatrix} s(t_s) \\ \dot{s}(t_s) \\ \ddot{s}(t_s) \\ \dot{s}(t_e) \\ \ddot{s}(t_e) \end{bmatrix} = \begin{bmatrix} 1 & t_s & t_s^2 & t_s^3 & t_s^4 \\ 0 & 1 & 2t_s & 3t_s^2 & 4t_s^3 \\ 0 & 0 & 2 & 6t_s & 12t_s^2 \\ 0 & 1 & 2t_e & 3t_e^2 & 4t_e^3 \\ 0 & 0 & 2 & 6t_e & 12t_e^2 \end{bmatrix} \begin{bmatrix} c_{s_0} \\ c_{s_1} \\ c_{s_2} \\ c_{s_3} \\ c_{s_4} \end{bmatrix}. \tag{2.8}$$

To solve matrix equations (2.7) and (2.8), assume $t_s = 0$ and $t_e - t_s = t_e = T$, where T is a constant, e.g., $T = 5$ [s]. By this assumption, we have $c_{d_0} = d(0), c_{d_1} = \dot{d}(0)$, and $c_{d_2} = \ddot{d}(0)$. By substituting these values in (2.7), the matrix equation is simplified into three equations and three unknowns. Solving the simplified matrix equation yields the values of c_{d_3}, c_{d_4}, and c_{d_5}. Note that, $d(t_s)$, $\dot{d}(t_s)$, and $\ddot{d}(t_s)$ are known, and we generate the trajectories such that $\dot{d}(t_e) = \ddot{d}(t_e) = 0$ while $d(t_e)$ has a reasonable value to keep the vehicle on the road. Likewise, considering the same assumption, i.e., $t_s = 0$ and $t_e = T$, for (2.8) we obtain $c_{s_0} = s(0), c_{s_1} = \dot{s}(0)$, and $c_{s_2} = \ddot{s}(0)$. Similarly, the matrix equation (2.8) can be simplified into a two-by-two matrix equation that yields c_{s_3} and c_{s_4}. Again, $s(t_s)$, $\dot{s}(t_s)$, and $\ddot{s}(t_s)$ are known, and we generate the trajectories such that $\ddot{s}(t_e) = 0$ while $\dot{s}(t_e)$ can take different values close to or equal to the desired speed. Now, at each step, several trajectories are generated, and the best one is selected according to a cost function. Several trajectories are generated by changing the values of $T \in [4, 5]$, $d(t_e) \in [d(t_s) - LW, d(t_s) + LW]$, and $\dot{s}(t_e) \in [v_d - 5/3.6, v_d + 5/3.6]$, where LW is lane width and v_d is the desired speed. In other words, each generated trajectory plans for the next 4 to 5 seconds, while the ego vehicle can stay in its lane or change one lane to the left or right, with the final speed no more than ± 5 [km/h] different than the desired speed v_d.

The cost function to be minimized is defined as the following,

$$C = C_d + C_s, \tag{2.9a}$$

$$C_d = K_j \sum_t \dddot{d}(t)^2 + K_t T + K_d d(T)^2, \tag{2.9b}$$

$$C_s = K_j \sum_t \dddot{s}(t)^2 + K_t T + K_d \left(v_d - \dot{s}(t)\right)^2, \tag{2.9c}$$

where C_d and C_s denote the lateral and longitudinal costs, and K_j, K_t, and K_d are the corresponding weighting parameters. The cost function aims to minimize the lateral and longitudinal jerks, reduce the lateral deviation from the reference line, and align the velocity with the desired value. Moreover, the term $K_t T$ introduces a bias toward selecting smaller $T \in [4, 5]$ values, provided that jerk minimization, reference-line adherence, and target-velocity tracking are not degraded. This promotes better maneuverability by enabling the selection of more points along the global path.

2.5 Experimental Data Collection

The relevant data is collected using a 2015 Nissan Leaf S battery electric vehicle in Norman, Oklahoma, USA. The test vehicle hardware configuration is shown in Fig. 2.5. Environment perception sensors include Velodyne VLP-16 Puck lidar, Texas Instruments AWR1642BOOST mmWave radar, Swift Piski Multi GNSS GPS, Xsens MTi-30 IMU, and cameras. The steering wheel and two pedals are modified to achieve a drive-by-wire function. Two powerful computation stations are mounted on the trunk floor, including an Alienware R13 and a Dell T3500. Image data collected from the dashcam are utilized in this chapter. The camera data is broadcasted to the ROS2 Humble via a v4l2 camera driver and saved in a ROS bag. The image data were collected from Asp Ave for straight roads and 24th Ave NW for curly roads, Norman, Oklahoma, USA

73019. Three scenarios are analyzed, with one straight road scenario (i.e., scenario 1) and two curly road scenarios (i.e., scenarios 2 and 3).

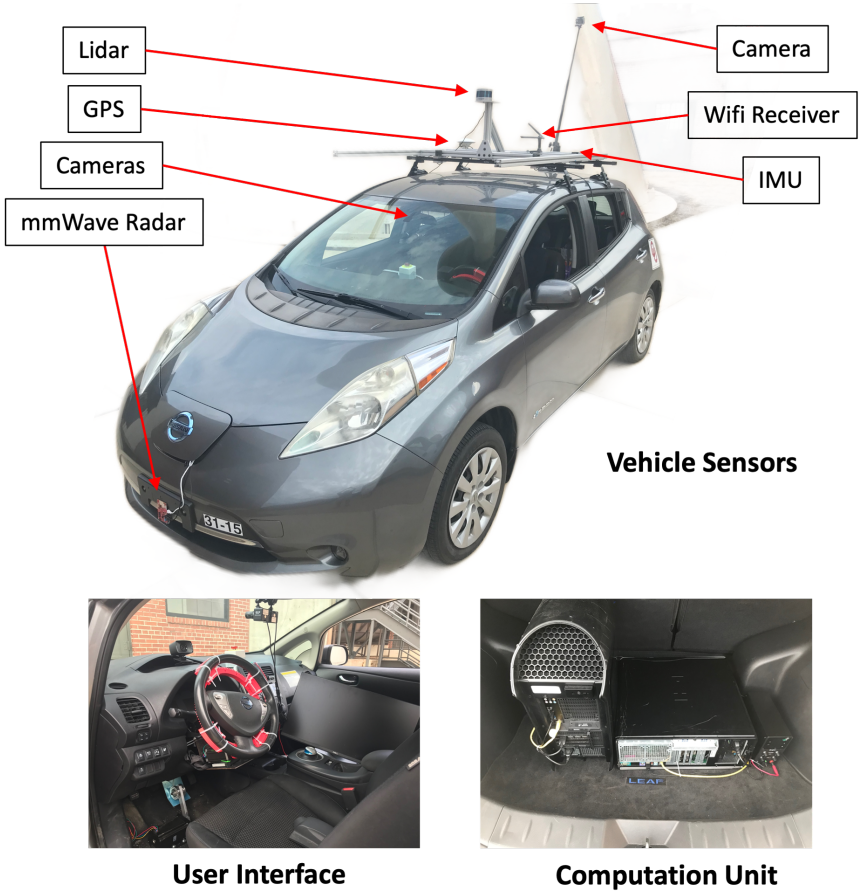


Figure 2.5: Environment perception sensors, human-machine interface and computation unit of the Nissan Leaf test car at The University of Oklahoma Mobility and Intelligence Lab

2.6 Image Processing

The collected images are processed for object detection, road detection, road boundary detection, depth prediction, image to bird-eye-view coordinates conversion and reference trajectory generation. YOLOv5 library is utilized for the object detection. Only vehicle, truck and pedestrian classes are activated in the object detection process. Road

surface and road boundary detection are achieved with the YOLOPv2 library. Due to the limited detection accuracy by the YOLOPv2 library, the raw detected boundaries are strengthened with curve fitting for better smoothness. The object depth prediction is achieved via a table-based method, which predicts the real-world lateral and vertical distance of any pixel in the image. The data based on Bird-eye-view coordinates are then generated from the raw image. The reference trajectory is generated as the central line of two detected lane boundaries.

2.7 Performance Appraisal

In order to compare the proposed MPC with the sampling-based method, the two algorithms are implemented on several instances containing various obstacles, including parked cars and pedestrians. The focus of this chapter is on extreme scenarios, such as narrow streets, as opposed to highways with clear lane marks, which is primarily addressed by many other studies. The following factors are used to compare the two approaches: computation time, comfort, collision avoidance, and smoothness. In this respect, computation time and comfort can be quantitatively measured, where comfort is quantified by the longitudinal jerk of the vehicle. On the other hand, collision avoidance and smoothness can not be easily measured in a qualitative manner, that is, one cannot judge collision avoidance (or staying clear from other vehicles) and smoothness by the value of the cost function. The reason lies in the fact that increasing the distance from other vehicles while overtaking may also increase the distance from the reference line, which is unfavorable. Moreover, returning to the reference line after a large deviation in order to maximize the gap from surrounding cars may lead to a high jerk which deteriorates the value of the cost function. Hence, it is not facile to assess the efficacy of obstacle avoidance and smoothness according to the cost function

value.

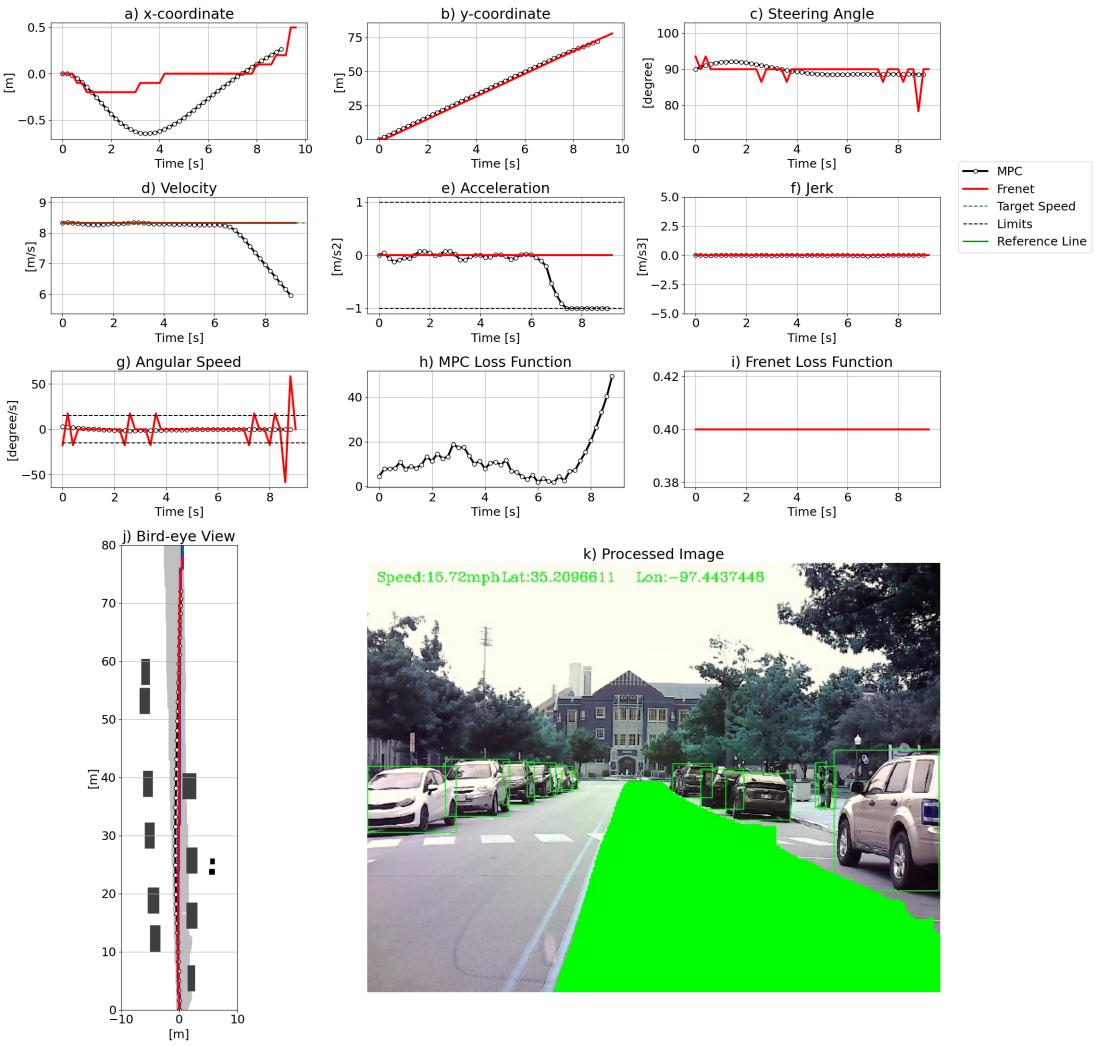


Figure 2.6: The results of scenario 1

2.8 Results Analysis

In this section, we compare the results of implementing MPC and the sampling-based method in the three scenarios. In this regard, each bird's-eye view image is converted to a data frame or an Excel file whose cells represent the pixels in the image. In the conversion process, each pixel is assigned a value of 0, 1, 2, 3, or 4, where 0 means

nothing, 1 means road surface, 2 means expert trajectory, 3 means vehicle object, and 4 means pedestrian. An example of the bird-eye view image where the road surface, vehicles/pedestrians, and expert trajectory are illustrated is shown in Fig. 2.1.

The path planning results of MPC and the sampling-based method are assessed based on the following metrics. First and foremost, the computation time of each method is compared as a too-long processing time precludes real-time implementation of the underlying algorithm. Moreover, the reference line comes from image processing. In terms of computation time, MPC is faster than the sampling-based method. Each iteration of MPC requires around 0.5 [s] to run, while the sampling-based method needs between 1 to 5 [s] for each iteration. Besides, we demonstrate the results based on two different scenario categories: The first category explains a straight road with some obstacles, including parked cars next to the curb and pedestrians. The second category consists of two different curved roads containing several other vehicles.

Fig. 2.6, subplot (f), shows both MPC and the sampling-based approach yield a smooth trajectory for a path with a near-zero jerk. Besides, despite the near identical behavior of the two models in following the reference line as shown in Fig. 2.6, subplot (j), MPC tends to maintain a larger lateral gap from the other cars parked next to the curb. Consequently, MPC finds a path with a lower risk of encountering obstacles. Moreover, according to Fig. 2.6, subplot (d), both MPC and the sampling-based algorithm reach the same destination with different speeds. In order to explain the reason for the MPC's speed drop in Fig. 2.6, subplot (d), firstly, it is emphasized that the test travel ends at the end of the path shown in Fig. 2.6, subplot (j,k). Secondly, since MPC has a prediction horizon, it realizes the destination is approaching and accordingly decreases the speed before reaching the destination point. MPC gently applies braking and reaches the deceleration limit of -1 $[m/s^2]$, Fig. 2.6, subplot (e). Subsequently, the speed reduction causes the loss function in Fig. 2.6, subplot (h), to

rise. On the other hand, Fig. 2.6, subplot (c,g), demonstrate MPC handles steering of the vehicle in a softer manner resulting in a smooth steering angle change (Fig. 2.6, subplot (c)) and much smaller angular speed (Fig. 2.6, subplot (g)) when compared with the sampling-based method. In fact, the sampling-based technique violates the limits set up for the angular speed in the MPC case. The limits are essential hardware and technical limitations and safety.

The second scenario results are displayed in Fig. 2.7. While the path of MPC and the sampling-based method are almost overlapping, that of the sampling-based method exhibits a greater longitudinal jerk (Fig. 2.7, subplot (f)). The other significant differences between the two methods are related to steering angle and velocity, as illustrated in Fig. 2.7, subplots (c) and (d), respectively. Similar to the previous scenario, the steering angle's variations for MPC are far smoother than the sampling-based method, which is important from the hardware point of view. The sudden changes of the steering angle by the sampling-based method have led to a ragged steering curve in Fig. 2.7, subplot (c), and dramatic fluctuations of angular speed in Fig. 2.7, subplot (g). Moreover, the velocity of the AV under the MPC decreases from the beginning, whereas the sampling-based approach maintains a constant speed, as shown in Fig. 2.7, subplot (d). This is because, in Fig. 2.7, the path planning is performed for roughly two seconds. MPC's prediction horizon covers the entire two seconds, helping MPC to notice that at the end, there is another vehicle. To avoid colliding with the front vehicle, MPC decelerates (Fig. 2.7, subplot (e)). Conversely, the sampling-based method's shortsightedness prevents it from decelerating, and thus, the sampling-based strategy keeps going at a constant speed (Fig. 2.7, subplots (d,e)). Furthermore, the deceleration by MPC leads to the monotonically ascending loss function curve as depicted in Fig. 2.7, subplot (h). In contrast, the sampling-based method's loss function in Fig. 2.7, subplot (i), takes on a trend similar to the sampling-based method's jerk in Fig. 2.7, subplot

(f). The differences between MPC and the sampling-based method loss functions stem from their definitions in (2.2a) and (2.9), respectively. the sampling-based algorithm's loss function depends on jerk and target speed. Since the target speed is being followed by the sampling-based method, jerk is the only factor causing changes in the sampling-based method's loss. MPC, on the other hand, minimizes the loss function for the entire prediction horizon. To avoid collision, it reduces the speed, which results in the rise of the loss. Should MPC allow collision to happen and maintain the speed, the loss function will soar at a higher rate as collision generates a huge penalty. On a similar note, in the previous scenario in Fig. 2.6, the the sampling-based method's loss is constant as its speed and jerk do not alter.

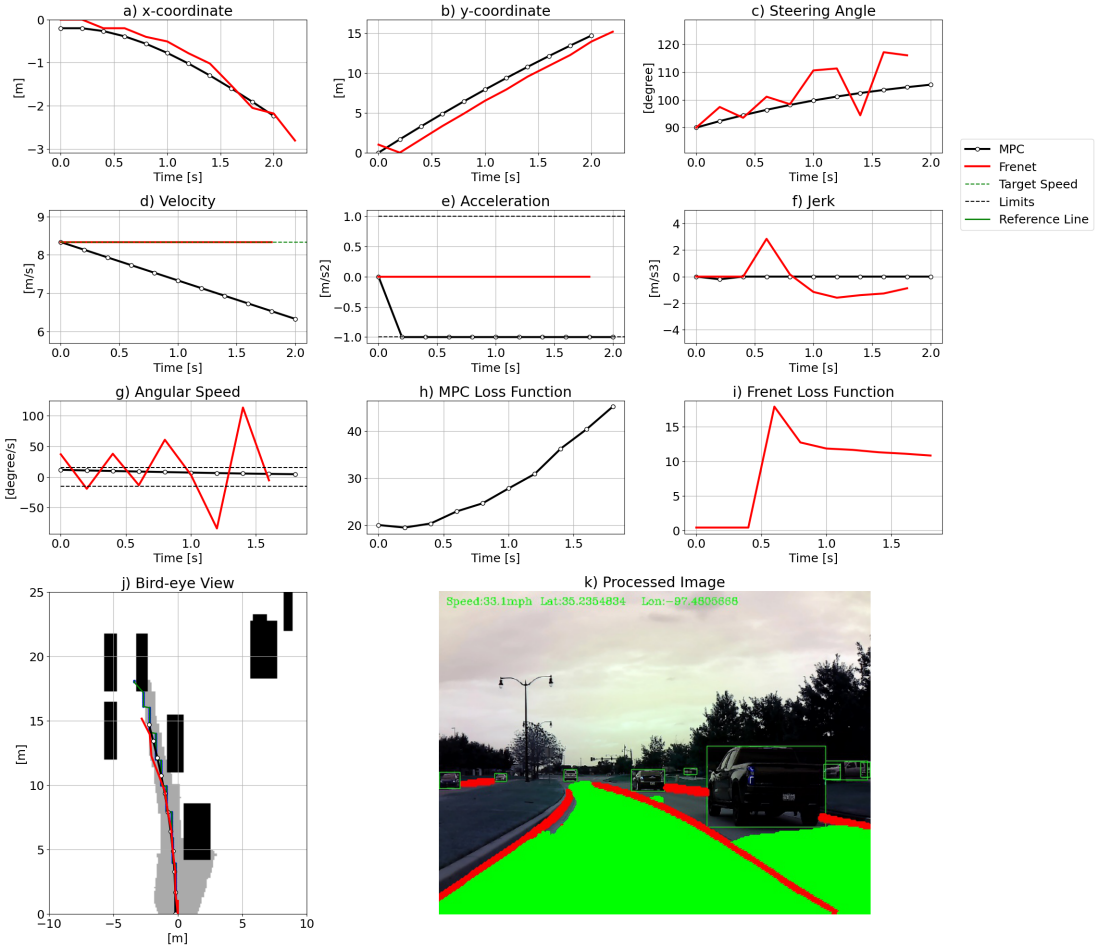


Figure 2.7: The results of scenario 2

On the other hand, in Fig. 2.8, the sampling-based approach seems to better follow the reference line with no jerk as represented in Fig. 2.8, subplots (j) and (f), respectively. The reason for MPC's deviation from the reference line is the white car up ahead in the ego vehicle's lane (see Fig. 2.8, subplot (k)). MPC predicts a possible collision and deviates from the reference line to avoid the collision and maximize the distance from others, as the path planning only covers a two-second period that is completely covered by MPC's prediction horizon. In this case, as presented in Fig. 2.8, subplot (c,g), the sampling-based strategy exhibits poor steering handling, leading to over 50 [degree/s] of angular speed. Analogous to the previous scenario, MPC decelerates to

avoid possible collisions (Fig. 2.8, subplots (d,e)). Overall, the performance of MPC and the sampling-based method in terms of tracking the reference line and avoiding collision is close. However, as mentioned before, the computation time of MPC and its handling of steering angle are considerably smoother than that of the sampling-based method.

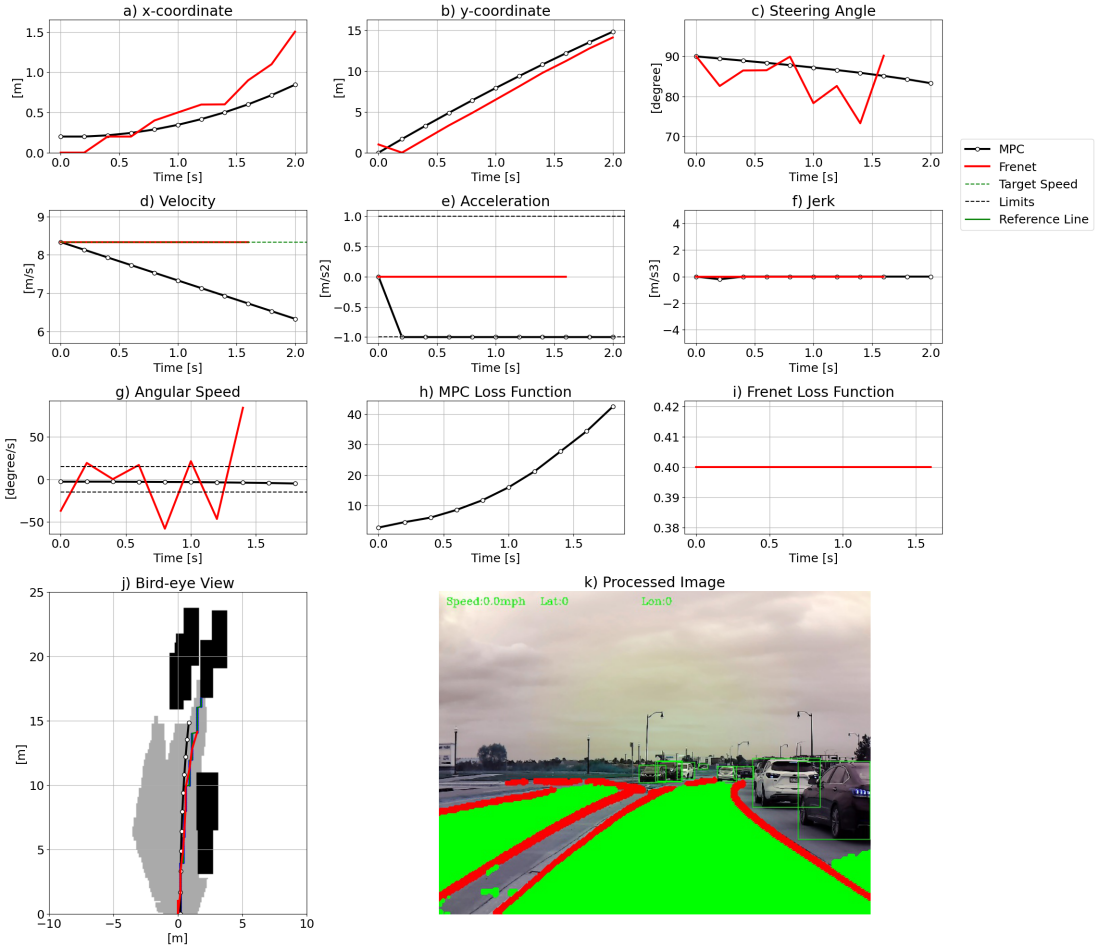


Figure 2.8: The results of scenario 3

2.9 Summary

This chapter presented an AV local path planning optimization method based on MPC. The proposed approach was evaluated using real-world data collected in Norman, Oklahoma, with our own Nissan Leaf, which was converted into an AV by equipping it with the necessary sensors and actuators. The MPC algorithm was benchmarked against a sampling-based method that utilizes Frenet frame coordinates. As shown in the results section, MPC outperforms the sampling-based approach in terms of path planning efficiency and accuracy.

Despite its advantages, MPC may not always meet the real-time computational requirements for AV local path planning. Its computation speed is heavily influenced by the prediction horizon, which must be carefully tuned according to the specific optimization problem. Thus, an MPC setup optimized for one scenario may not remain optimal if the objective function or constraints change. Additionally, MPC inherently performs redundant computations: only the first action of the optimized control sequence is executed, while the remaining actions are discarded. Similarly, sampling-based methods also incur redundancy by generating multiple paths at each step but executing only one.

The need for faster computation and reduced redundancy motivated the exploration of Lyapunov optimization (LO) for AV local path planning in the next chapter. Unlike MPC and sampling-based methods, LO employs a single-step optimization strategy, calculating only the actions that will actually be executed—eliminating unnecessary computations while maintaining optimality.

Chapter 3

A Lyapunov Optimization-Based Approach to Autonomous Vehicle Local Path Planning

This chapter defines the path planning problem and presents a solution approach using a sampling-based method illustrated in the Frenet frame, an MPC-based strategy, along with a description of the LO technique.

3.1 Environment

Fig. 2.2 illustrates an example of a car (the blue rectangle) on the road with a reference line to follow, where each point is denoted by (x, y) in the Cartesian system. Further, (x_r, y_r) indicates a point on the reference line closest to the vehicle's center of gravity (CG), while the heading angle and vehicle's velocity are represented by θ and v, respectively. The Euler approximation method is chosen for state equation discretization due to its widespread use and simplicity. However, it can result in coarse approximations when the sampling time is too large. To mitigate this limitation, this work reduces the sampling time to a level that strikes a balance between improved accuracy and acceptable computational cost for autonomous driving applications. The

dynamic equation of the vehicle can be formulated as the following,

$$z_{k+1} = z_k + f(z_k, c_k)\Delta t, \tag{3.1}$$

where subscript k is time step, Δt shows sampling time, c_k denotes input, and vehicle's state at k is $z_k = \begin{bmatrix} x_k & y_k & \theta_k & v_k \end{bmatrix}^T$. Moreover,

$$f(z_k, c_k) = \begin{bmatrix} v_k \cos\theta_k & v_k \sin\theta_k & \omega_k & a_k \end{bmatrix}^T,$$

where ω_k is angular velocity at k, a_k indicates vehicle longitudinal acceleration at k, and $c_k = \begin{bmatrix} \omega_k & a_k \end{bmatrix}^T$. Furthermore, environmental information, such as roadway waypoints, lane markings, and the positions of surrounding vehicles, is gathered through image processing using a real AV. (see Section **??**).

3.2 Path Planning Optimization Problem

The following nonlinear, constrained optimization problem is formulated to address the local path planning challenge, prioritizing safety and comfort for passengers while maintaining the desired speed.

$$\begin{aligned} \min \quad & \frac{1}{N}\mathbb{E}\left\{\sum_{k=1}^{N}\left(w_{gd}D_k^2 + w_{ym}r_k + w_v\left(v_k^d - v_k\right)^2 + \right.\right. \\ & \left.\left. w_a a_k^2 + w_y\omega_k^2 + w_j\left(a_k - a_{k-1}\right)^2\right)\right\} \end{aligned} \tag{3.2a}$$

subject to

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_k \end{bmatrix} + \begin{bmatrix} v_k \cos\theta_k \\ v_k \sin\theta_k \\ \omega_k \\ a_k \end{bmatrix} \Delta t, \quad \forall k, \tag{3.2b}$$

$$a^{\min} \leq a_k \leq a^{\max}, \quad \forall k, \tag{3.2c}$$

$$\omega^{\min} \leq \omega_k \leq \omega^{\max}, \quad \forall k, \tag{3.2d}$$

where N is the total number of time steps, $\mathbb{E}\{\cdot\}$ denotes expectation, w_{gd}, w_{ym}, w_v, w_a, w_y, and w_j represent weights, D_k expresses the Euclidian distance between ego vehicle's CG, (x_k, y_k), and (x_r, y_r) at time k. This term aims to minimize the reference position tracking error. v_k^d presents the desired speed at k. This term aims to minimize the speed tracking error. $a^{\min}$ and $a^{\max}$ are acceleration boundaries. $\omega^{\min}$ and $\omega^{\max}$ show angular velocity boundaries. These acceleration and angular velocity terms aim to minimize the control input. $(a_k - a_{k-1})^2$ is to minimize longitudinal jerk. The barrier function r_k is fashioned for collision avoidance as follows,

$$r_k = \sum_{i=1}^{N_o} \frac{1}{\eta_x l_{x,k}^i + \eta_y l_{y,k}^i}, \tag{3.3}$$

$$l_{x,k}^i = \begin{cases} x_1^i - x_2^e, & \text{if } x_2^e < x_1^i, \\ x_1^e - x_2^i, & \text{if } x_1^e > x_2^i, , \\ 0, & \text{O.W.}, \end{cases} \tag{3.4}$$

$$l^i_{y,k} = \begin{cases} y^i_1 - y^e_2, & \text{if } y^e_2 < y^i_1, \\ \infty, & \text{if } y^e_1 > y^i_2, \\ 0, & \text{O.W.}, \end{cases} \tag{3.5}$$

where i is the obstacle index, e represents the ego vehicle, N_o shows the total number of surrounding obstacles captured by image processing, η_x and η_y are weights, $l^i_{x,k}$ is the lateral distance of the ego vehicle from the obstacle at time k as displayed in Fig. 3.1, $l^i_{y,k}$ indicates the longitudinal distance from the object, x^i_1, x^i_2, y^i_1, and y^i_2 are used to define the corners of the i-th obstacle's bounding box, x^e_1, x^e_2, y^e_1, and y^e_2 locate the bounding box corners of the ego vehicle. The function r_k is defined such that if a collision occurs with object i, $l^i_{x,k} = l^i_{y,k} = 0$ and

$$\forall i : \lim_{\substack{l^i_{x,k} \to 0 \\ l^i_{y,k} \to 0}} r_k = \infty.$$

The MPC approach directly utilizes the optimization problem (3.2) for path planning, but this problem must be reformulated to apply the LO algorithm.

3.3 Reformulation

For each of the states in (3.2b), a virtual queue is constructed as the following,

$$Q^x_k = x_k - x^d_k, \tag{3.6a}$$

$$Q^y_k = y_k - y^d_k, \tag{3.6b}$$

$$Q^\theta_k = \theta_k - \theta^d_k, \tag{3.6c}$$

$$Q^v_k = v_k - v^d_k, \tag{3.6d}$$

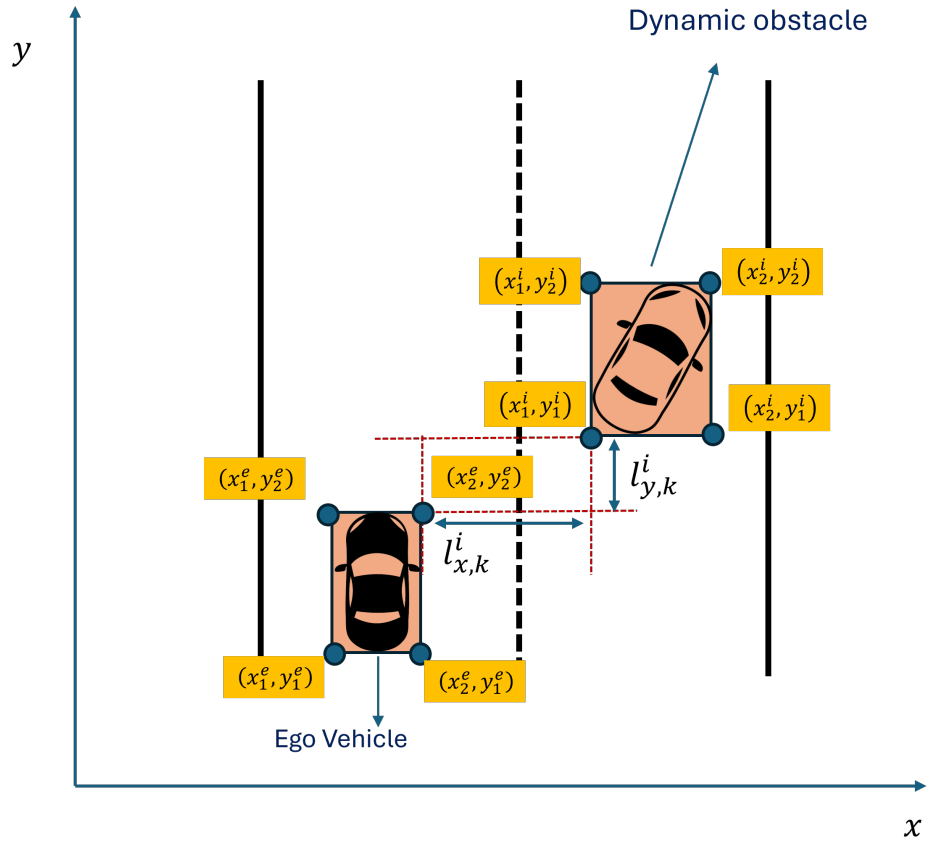


Figure 3.1: r_k components $l_{x,k}^i$ and $l_{y,k}^i$

where x_k^d, y_k^d, θ_k^d, and v_k^d are desired state values. In detail, (x_k^d, y_k^d) are coordinates of the nearest waypoint (obtained from the reference line), θ_k^d is the desired steering angle based on the relative position of the vehicle and the reference line, and v_k^d is the desired velocity. Define the queue length vector as $Q_k = \left[Q_k^x, Q_k^y, Q_k^\theta, Q_k^v\right]$. By combining (3.6) in (3.2b) we obtain the following

$$Q_{k+1}^x = Q_k^x + v_k \cos\theta_k \Delta t - x_{k+1}^d + x_k^d = Q_k^x + \xi_k^x, \tag{3.7a}$$

$$Q_{k+1}^y = Q_k^y + v_k \sin\theta_k \Delta t - y_{k+1}^d + y_k^d = Q_k^y + \xi_k^y, \tag{3.7b}$$

$$Q_{k+1}^\theta = Q_k^\theta + \omega_k \Delta t - \theta_{k+1}^d + \theta_k^d = Q_k^\theta + \xi_k^\theta, \tag{3.7c}$$

$$Q_{k+1}^v = Q_k^v + a_k \Delta t - v_{k+1}^d + v_k^d = Q_k^v + \xi_k^v, \tag{3.7d}$$

where $\xi_k^x = v_k \cos\theta_k \Delta t - x_{k+1}^d + x_k^d$, $\xi_k^y = v_k \sin\theta_k \Delta t - y_{k+1}^d + y_k^d$, $\xi_k^\theta = \omega_k \Delta t - \theta_{k+1}^d + \theta_k^d$, and $\xi_k^v = a_k \Delta t - v_{k+1}^d + v_k^d$.

Using the virtual queues in (3.6), problem (3.2) can be transformed into a queue stability problem (3.8), where (3.2b) is changed to the stability of the queues.

$$\begin{aligned} \min \quad & \frac{1}{N}\mathbb{E}\left\{\sum_{k=1}^{N}\left(w_{gd}D_k^2 + w_{ym}r_k + w_v\left(v_k^d - v_k\right)^2 + \right.\right. \\ & \left.\left. w_a a_k^2 + w_y\omega_k^2 + w_j\left(a_k - a_{k-1}\right)^2\right)\right\} \end{aligned} \tag{3.8a}$$

subject to

$$\begin{aligned} & (3.2c), (3.2d) \\ & \text{stability of virtual queues } Q_k. \end{aligned} \tag{3.8b}$$

Subsequently, the LO is applied to (3.8), which establishes an adaptive control policy for local path planning. Problem (3.8) is decomposed into N subproblems that are solved independently, while system stability is guaranteed.

3.4 Lyapunov Optimization Model

Define

$$\begin{aligned} u_k = {} & w_{gd}D_k^2 + w_{ym}r_k + w_v\left(v_k^d - v_k\right)^2 \\ & + w_a a_k^2 + w_y\omega_k^2 + w_j\left(a_k - a_{k-1}\right)^2. \end{aligned} \tag{3.9}$$

Next, the Lyapunov function is defined as follows,

$$L(Q_k) \triangleq \frac{\beta_1}{2}\left(Q_k^x\right)^2 + \frac{\beta_2}{2}\left(Q_k^y\right)^2 + \frac{\beta_3}{2}\left(Q_k^\theta\right)^2 + \frac{\beta_4}{2}\left(Q_k^v\right)^2, \tag{3.10}$$

where β_1, β_2, β_3, and β_4 are positive constants. The Lyapunov drift is

$$\Delta(Q_k) \triangleq \mathbb{E}\left\{L(Q_{k+1}) - L(Q_k)|Q_k\right\}. \tag{3.11}$$

Finally, the drift-plus-penalty term will be as,

$$\Delta(Q_k) + V\mathbb{E}\left\{u_k|Q_k\right\}. \tag{3.12}$$

Theorem 3.1. *At any time slot k, the Lyapunov drift-plus-penalty term has the following upper bound,*

$$\begin{aligned}\Delta(Q_k) + V\mathbb{E}\{u_k|Q_k\} &\leq B + \beta_1 Q_k^x \mathbb{E}\{\varrho_k^x|Q_k\} \\ &+ \beta_2 Q_k^y \mathbb{E}\{\varrho_k^y|Q_k\} + \beta_3 Q_k^\theta \mathbb{E}\{\varrho_k^\theta|Q_k\} + \beta_4 Q_k^v \mathbb{E}\{\varrho_k^v|Q_k\} \\ &+ V\mathbb{E}\{u_k|Q_k\},\end{aligned} \tag{3.13}$$

where B is a positive constant shown in (A.11). *In addition, ϱ_k^x, ϱ_k^y, ϱ_k^θ, and ϱ_k^v are defined in* (A.7).

Proof. See Appendix A. ■

In Theorem 3.2, we will demonstrate minimizing the upper bound of (3.13), instead of its left-hand side, for each time step k leads to a suboptimal solution which is close to the global optimal objective value of (3.2a), denoted as p^*. The convergence of the suboptimal solution to p^* is controlled by coefficient V.

Before diving into Theorem 3.2, an optimization problem is formulated to minimize the upper bound of (3.13) as follows,

$$\begin{aligned} \min \quad & Vu_k + \beta_1 Q_k^x \varrho_k^x + \beta_2 Q_k^y \varrho_k^y + \beta_3 Q_k^\theta \varrho_k^\theta + \beta_4 Q_k^v \varrho_k^v, \\ & \text{subject to} \qquad \text{(3.2c)-(3.2d)}, \end{aligned} \tag{3.14}$$

where a_k and ω_k are decision variables. Note that Q_k^x, Q_k^v, Q_k^θ, and Q_k^v are constants whose values are updated after solving (3.14) for time step k and before running (3.14) for time step $k+1$. Moreover, when (3.14) is being solved for $k+1$, the values of states at time step k are known. Therefore, according to (A.7), the values of ϱ_k^x and ϱ_k^y are known, making terms $\beta_1 Q_k^x \varrho_k^x$ and $\beta_2 Q_k^y \varrho_k^y$ constants. Hence, solving (3.14) in its current form won't lead to the minimization of Q_k^x and Q_k^y. In other words, by minimizing (3.14), the reference line is not tracked by the ego vehicle. The issue arises from the fact that x_k and y_k implicitly depend on a_k and ω_k. To resolve the issue, we propose using v_{k+1} and θ_{k+1} instead of v_k and θ_k in ϱ_k^x and ϱ_k^y as follows. The proposition is further verified with simulations in Section 3.6. Assume,

$$Q_k^x v_k \cos\theta_k \Delta t \approx Q_k^x v_{k+1} \cos\theta_{k+1} \Delta t, \tag{3.15a}$$

$$Q_k^y v_k \sin\theta_k \Delta t \approx Q_k^y v_{k+1} \sin\theta_{k+1} \Delta t. \tag{3.15b}$$

Utilizing (3.2b), v_{k+1} and θ_{k+1} are substituted as,

$$Q_k^x v_{k+1} \cos\theta_{k+1} \Delta t = Q_k^x \left(v_k + a_k \Delta t\right) \cos\left(\theta_k + \omega_k \Delta t\right) \Delta t, \tag{3.16a}$$

$$Q_k^y v_{k+1} \sin\theta_{k+1} \Delta t = Q_k^y \left(v_k + a_k \Delta t\right) \sin\left(\theta_k + \omega_k \Delta t\right) \Delta t. \tag{3.16b}$$

Finally, leveraging (A.7) and plugging (3.16) in the objective function of (3.14), we

Algorithm 1: Lyapunov Optimization

Data: V, N, w_{gd}, w_{ym}, w_v, w_a, w_y, w_j, x_0, y_0, θ_0, v_0, x_0^d, y_0^d, θ_0^d, v_0^d, ω_0, a_0, Q_1
Result: x_k, y_k, θ_k, v_k, ω_k, a_k, Q_k for $k \in \{1, \cdots, N\}$
for $k = 1, \cdots, N$ **do**
  Find a_k and ω_k by solving (3.17);
  Update states x_k, y_k, θ_k, and v_k via (3.2b);
  Update desired values x_{k+1}^d, y_{k+1}^d, θ_{k+1}^d, and v_{k+1}^d;
  Update Q_{k+1}^x, Q_{k+1}^y, Q_{k+1}^θ, and Q_{k+1}^v via (3.6)
end

obtain,

$$\begin{aligned} \min \quad & V u_k + \Lambda_k, \\ \text{subject to} \quad & \text{(3.2c)-(3.2d)}, \end{aligned} \tag{3.17}$$

where

$$\begin{aligned} \Lambda_k = & \beta_1 Q_k^x \left(v_k + a_k \Delta t\right) \cos\left(\theta_k + \omega_k \Delta t\right) \Delta t \\ & + \beta_2 Q_k^y \left(v_k + a_k \Delta t\right) \sin\left(\theta_k + \omega_k \Delta t\right) \Delta t + \beta_3 Q_k^\theta \omega_k \Delta t \\ & + \beta_4 Q_k^v a_k \Delta t. \end{aligned} \tag{3.18}$$

Problem (3.17) minimizes the upper bound of (3.13) for time step k. After each time step k, the queues should be updated before running problem (3.17) for the next time step $k+1$ as presented in Algorithm 1.

Theorem 3.2. *For all $N > 1$, the optimal solution of* (3.17)*, denoted by $\{\hat{a}_k, \hat{\omega}_k\}$, and the optimal solution of* (3.2)*, denoted by $\{a_k^*, \omega_k^*\}$, satisfy the following inequality,*

$$\frac{1}{N} \sum_{k=1}^{N} \mathbb{E}\left\{\hat{u}_k\right\} \leq \frac{B_1}{V} + p^* + \frac{1}{NV} \mathbb{E}\left\{L(\hat{Q}_1)\right\}, \tag{3.19}$$

where B_1 is a constant defined in (B.19)*, p^* is the optimal objective value of* (3.2a) *and $\hat{u}_k$ is obtained by* (3.9) *using $\{\hat{a}_k, \hat{\omega}_k\}$.*

Proof. See Appendix B. ■

Next, the problem is formulated for two baseline methods. Note that nonlinear problem (3.2) is directly tackled with a nonlinear MPC. Hence, the problem is only required to be reformulated for the sampling-based approach.

3.5 Sampling-based Method

This method employs the Frenet frame, which utilizes Frenet coordinates to describe the position of any arbitrary point along a reference path through longitudinal and lateral displacements, denoted as s and d, respectively. It is important to note that the coordinates s and d are always perpendicular to each other at every point along the reference line. The approach consists of two main steps: i) generating multiple candidate trajectories at each step, and ii) selecting the optimal path for the ego vehicle based on a cost function.

Using Frenet coordinates, we can decouple the lateral and longitudinal motion of the vehicle while simultaneously discretizing the driving state space to generate a set of target states. Based on the initial and target states, we create a series of trajectories for lateral and longitudinal motion, utilizing quintic and quartic polynomials, respectively. Additionally, a comprehensive cost function is formulated to evaluate comfort, safety, and deviation from the reference line. Furthermore, the variables s and d are represented by the following polynomials:

$$d(t) = c_{d_0} + c_{d_1}t + c_{d_2}t^2 + c_{d_3}t^3 + c_{d_4}t^4 + c_{d_5}t^5, \tag{3.20}$$

$$s(t) = c_{s_0} + c_{s_1}t + c_{s_2}t^2 + c_{s_3}t^3 + c_{s_4}t^4, \tag{3.21}$$

where t is time and c_{d_x} and c_{s_x} are coefficients that should be calculated. Each trajectory starts at t_s and ends at t_e under the assumption that t_s and t_e are known. Considering equations (3.20) and (3.21) and their time derivatives at the start and end points, the following set of equations is achieved,

$$\begin{bmatrix} d(t_s) \\ \dot{d}(t_s) \\ \ddot{d}(t_s) \\ d(t_e) \\ \dot{d}(t_e) \\ \ddot{d}(t_e) \end{bmatrix} = \begin{bmatrix} 1 & t_s & t_s^2 & t_s^3 & t_s^4 & t_s^5 \\ 0 & 1 & 2t_s & 3t_s^2 & 4t_s^3 & 5t_s^4 \\ 0 & 0 & 2 & 6t_s & 12t_s^2 & 20t_s^3 \\ 1 & t_e & t_e^2 & t_e^3 & t_e^4 & t_e^5 \\ 0 & 1 & 2t_e & 3t_e^2 & 4t_e^3 & 5t_e^4 \\ 0 & 0 & 2 & 6t_e & 12t_e^2 & 20t_e^3 \end{bmatrix} \begin{bmatrix} c_{d_0} \\ c_{d_1} \\ c_{d_2} \\ c_{d_3} \\ c_{d_4} \\ c_{d_5} \end{bmatrix}, \tag{3.22}$$

$$\begin{bmatrix} d(t_s) \\ \dot{d}(t_s) \\ \ddot{d}(t_s) \\ \dot{d}(t_e) \\ \ddot{d}(t_e) \end{bmatrix} = \begin{bmatrix} 1 & t_s & t_s^2 & t_s^3 & t_s^4 & t_s^5 \\ 0 & 1 & 2t_s & 3t_s^2 & 4t_s^3 & 5t_s^4 \\ 0 & 0 & 2 & 6t_s & 12t_s^2 & 20t_s^3 \\ 0 & 1 & 2t_e & 3t_e^2 & 4t_e^3 & 5t_e^4 \\ 0 & 0 & 2 & 6t_e & 12t_e^2 & 20t_e^3 \end{bmatrix} \begin{bmatrix} c_{s_0} \\ c_{s_1} \\ c_{s_2} \\ c_{s_3} \\ c_{s_4} \end{bmatrix}, \tag{3.23}$$

which are used to find coefficients c_{d_x} and c_{s_x}. To solve (3.22) and (3.23), assume $t_s = 0$ and $t_e = \tau$, where τ is a positive constant. Therefore, by plugging $t_s = 0$ into (3.22), we obtain $c_{d_0} = d(0)$, $c_{d_1} = \dot{d}(0)$, and $c_{d_2} = \ddot{d}(0)$. Consequently, the matrix equation (3.22) simplifies into three equations and three unknowns. Note that we

assume $\dot{d}(t_e) = \ddot{d}(t_e) = 0$ while $d(t_e)$ is selected such that the vehicle remains on the road. On the other hand, for (3.23) we have $c_{s_0} = s(0)$, $c_{s_1} = \dot{s}(0)$, and $c_{s_2} = \ddot{s}(0)$, which simplifies matrix equation (3.23) to a two-by-two one. Moreover, $s(t_s)$, $\dot{s}(t_s)$, and $\ddot{s}(t_s)$ are known from the previous step's calculation, and the candidate trajectories are generated such that $\ddot{s}(t_e) = 0$, and $\dot{s}(t_e)$ value is selected close to the desired speed.

Accordingly, at each step, several trajectories are generated based on the combination of $d(t_e)$ and $\dot{s}(t_e)$ values and the best one is selected according to the cost function in (3.24). The candidate trajectories are generated by changing the values of

$$
\begin{aligned}
d(t_e) &\in [d(t_s) - LW, d(t_s) + LW], \\
\dot{s}(t_e) &\in \left[v_k^d - 5/3.6, v_k^d + 5/3.6\right],
\end{aligned}
$$

where LW is lane width. Finally, the cost function to be minimized is defined as,

$$C = C^d + C^s, \tag{3.24a}$$

$$C^d = K_j \sum_{t=t_s}^{\tau} \dddot{d}(t)^2 + K_t \tau + K_d d(\tau)^2, \tag{3.24b}$$

$$C^s = K_j \sum_{t=t_s}^{\tau} \dddot{s}(t)^2 + K_t \tau + K_d \left(v_k^d - \dot{s}(t)\right)^2, \tag{3.24c}$$

where C^d and C^s indicate lateral and longitudinal costs, and K_j, K_t, and K_d are weights. The cost function aims to minimize both lateral and longitudinal jerks, reduce the lateral distance from the reference line, and ensure that the velocity matches the desired value.

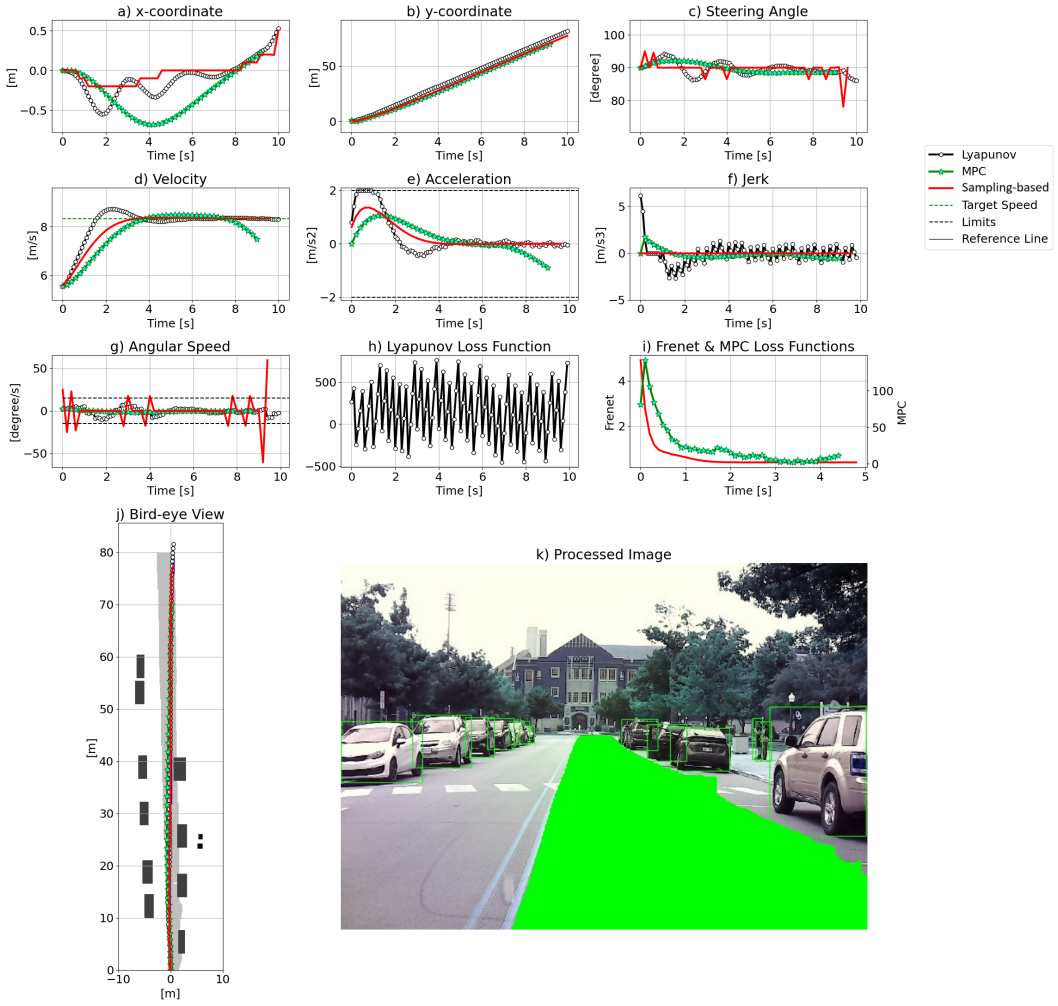


Figure 3.2: The results for scenario 1

3.6 Simulation and Results

This section presents the results of local path planning using the proposed LO approach, alongside two baseline methods for comparison. The primary advantage of the LO method lies in its substantially faster computation speed. This efficiency is particularly important, as the computation chip represents one of the most expensive components in an AV system. The experimental setup is identical to that described in Section 2.5. All computations were performed on a computer equipped with a 2.40 GHz Intel Core i7 CPU (20 cores) and 16 GB of RAM.

Fig. 3.2 displays the results of scenario 1, which is a straight road path planning in a narrow urban street where the lane is partially occupied by the vehicles parked next to the curb as Fig. 3.2k shows. The bird-eye-view image of the path ahead is generated and depicted in Fig. 3.2j. According to Fig. 3.2j, the paths generated by LO and the two baseline approaches are similar without significant disparity. MPC's path is closest to the left side solid lines, whereas the sampling-based method's path is closest to the right side where vehicles are parked, while the LO result falls in between the two baseline methods. Fig. 3.2a illustrates the x-coordinates of the three algorithms. While MPC smoothly steers left to avoid parked cars and then right to follow the reference line, both LO and the sampling-based method exhibit less smooth transitions compared to MPC. Fig. 3.2b demonstrates that the y-coordinates of all methods overlap and the three curves align well. On the other hand, Fig. 3.2c and Fig. 3.2g represent steering angle and angular velocity results, respectively. Sharp changes in the steering angle observed in the sampling-based method are caused by significant variations in angular speed. These fluctuations arise because the sampling-based approach does not impose limits on the angular speed input. In contrast, the MPC and LO methods produce smooth results for both steering angle and velocity. Fig. 3.2d

and Fig. 3.2e show velocity and acceleration, respectively. LO exploits maximum allowed acceleration to increase velocity to reach the desired value, whereas MPC and the sampling-based method show similar acceleration and velocity profiles. In the end, both the sampling-based algorithm and LO maintain the desired speed, while MPC experiences deceleration. This distinct behavior in MPC is attributed to its prediction horizon. As MPC anticipates the end of the road with no additional waypoints ahead, it proactively reduces speed. In terms of the jerk, shown in Fig. 3.2f, MPC and the sampling-based method perform better than LO. Nonetheless, LO's jerk is small and near zero. Lastly, subplots Fig. 3.2h and Fig. 3.2i present the underlying objective values of the three techniques throughout the trajectory. Note that different scales of the objective values stem from different objective functions used for each approach.

Fig. 3.3 shows the second scenario on a three-lane street with a curve to the left. Based on Fig. 3.3j, LO stays closer to the right edge of the lane, while MPC and the sampling-based approach maintain a larger lateral gap with the vehicle on the right (the sedan in front of the pickup truck in Fig. 3.3k). Note that the step-wise solid black line in Fig. 3.3j is the reference line. In Fig. 3.3a and Fig. 3.3b, LO closely follows MPC and the sampling-based method results. Furthermore, Fig. 3.3c and Fig. 3.3g demonstrate LO and MPC exhibit much smoother handling of steering compared to the sampling-based method. As seen in Fig. 3.3g, the sampling-based technique shows huge fluctuations. In addition, Fig. 3.3d and Fig. 3.3e show that all approaches attempt to approach the desired speed. However, MPC decelerates shortly after the beginning of scenario 2 as a result of predicting the ego vehicle approaches the end of the path. Finally, Fig. 3.3f displays the longitudinal jerk of the three approaches, where MPC has the smallest jerk and the sampling-based method has the largest one.

Fig. 3.4 presents the results for scenario 3, which is on a three-lane street with a curve to the right. As pictured in Fig. 3.4j, LO trajectory closely follows MPC's while

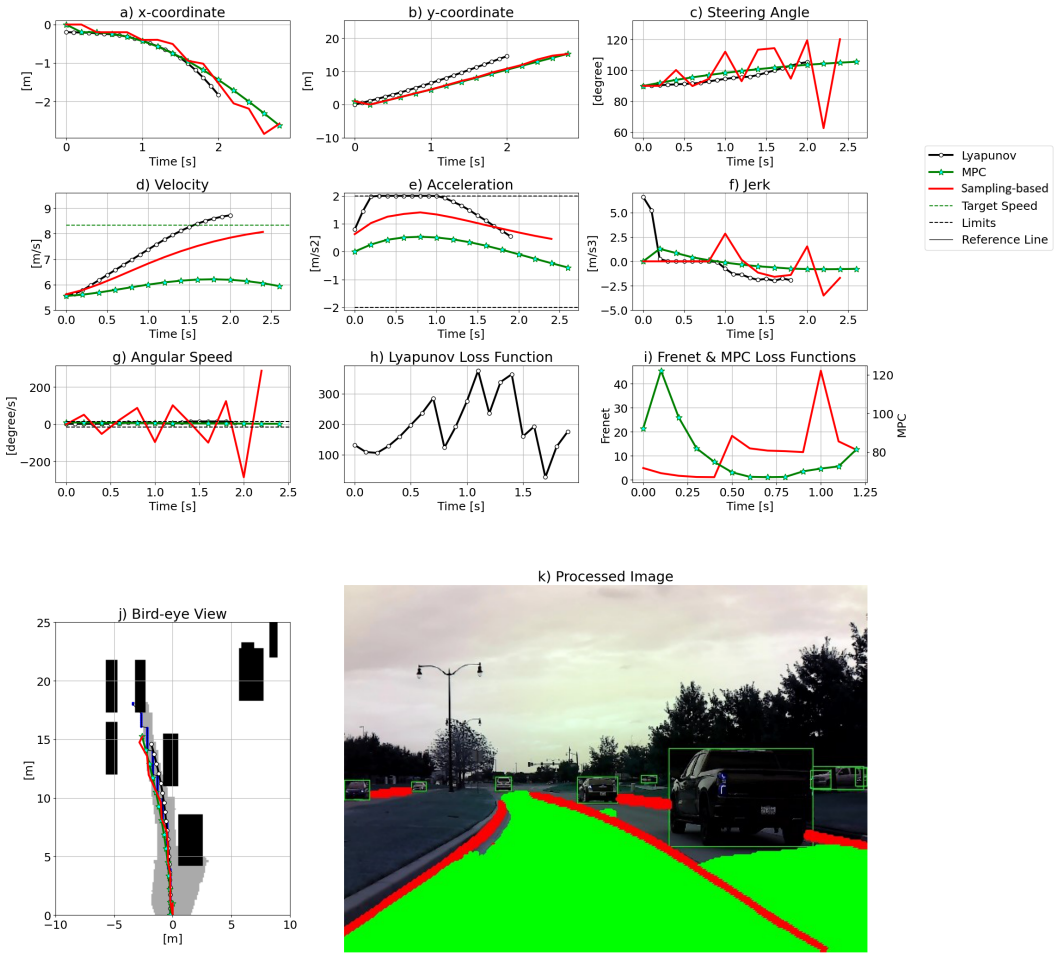


Figure 3.3: The results for scenario 2

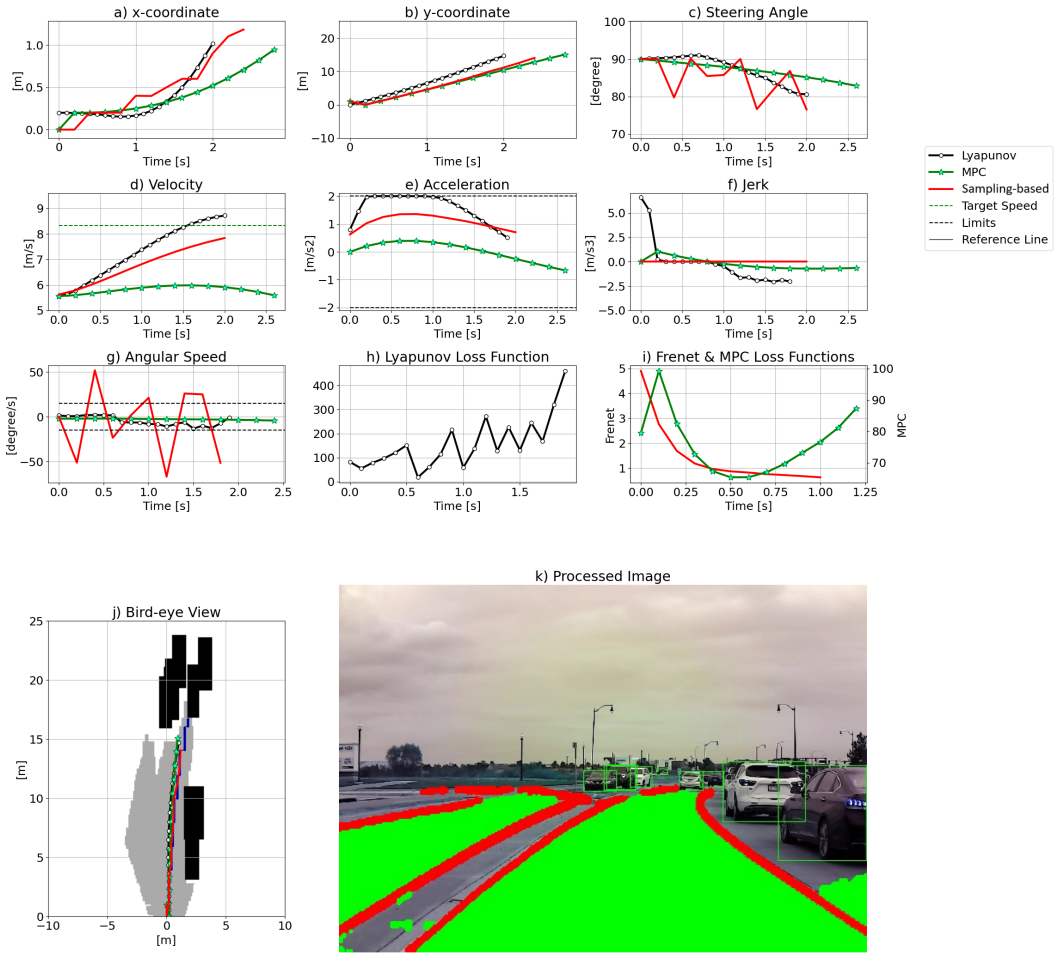


Figure 3.4: The results for scenario 3

the sampling-based method puts more effort into following the reference line, leading to non-smooth outcomes in subplots a, c, and g in Fig. 3.4. Further, as shown in Fig. 3.4d and Fig. 3.4e, MPC decelerates again to avoid passing the end point of the reference line. On the other hand, all approaches represent small jerks, as Fig. 3.4 shows.

The comparison of computation time, path smoothness, and safety among the three approaches are summarized in Table. 3.1. As shown in the table, to achieve a path generation task at a single time step, LO is at least faster than MPC by around 20 times and faster than the sampling-based method by 40 times. This computation time reduction by LO is significant. More importantly, it is achieved without safety and path smoothness compromise. The fast computation in LO, compared to MPC and the sampling-based approach, stems from the following. LO directly finds the optimal path, whereas the sampling-based method expends computational effort on analyzing multiple candidate trajectories. In comparison with MPC, LO only performs calculations for the current time step, while MPC runs the calculations for the next c steps, c being the length of the control horizon. But only one step out of c step calculations is executed, and then MPC repeats everything for steps 2 to $c + 1$, one of which will be executed next. In a sense, both MPC and the sampling-based methodology waste computation resources, generating paths that will not be realized. In contrast, all LO calculations are executed, making LO remarkably more efficient than MPC and the sampling-based method in terms of minimizing computation effort. Finally, in terms of path smoothness, LO and MPC attain a much smoother path compared to the sampling-based model. Besides, all approaches provide a safe trajectory as all paths are collision-free, and certain space margins are left against the obstacles.

Table 3.1: The computation time of one step (each step is Δt long), smoothness, and safety of the three approaches. The computation time is normalized.

Approaches	LO	MPC	sampling-based
Computation time	1	20	120
Path Smoothness	High	High	Low
Safety	High	High	High

3.7 Summary

This chapter introduced an LO-based strategy for AV local path planning and demonstrated its superiority over both MPC and a sampling-based approach. To the best of our knowledge, this is the first work to apply LO to the AV local path planning problem, and as such, we employed a relatively simple vehicle dynamic model for initial validation.

While the evaluation used real-world data collected from our own Nissan Leaf converted into an AV, the results presented here are simulation-based, utilizing the images captured by the vehicle's camera as it was manually driven around Norman, Oklahoma. This approach allowed us to test the LO strategy in a controlled setting while relying on authentic environmental data.

In the following chapter, we address these limitations by adopting a more sophisticated vehicle dynamic model and implementing the LO approach in a high-fidelity simulation environment using the CARLA simulator. This next step will reinforce the conclusions of the current chapter, demonstrating that the single-step LO framework is not only theoretically effective but also practically implementable for AV local path planning tasks under more realistic and complex driving conditions.

Chapter 4

Lyapunov Optimization-based Approach Validation in CARLA Simulator

In this chapter, the developed models are implemented in the high fidelity CARLA environment that accurately replicates the real-world conditions. An extended Kalman filter (EKF) is used to handle the discrepancies between our model and CARLA's model.

4.1 CARLA, ROS, and CARLA-ROS bridge

CARLA (Car Learning to Act) is an open-source simulator designed for autonomous driving research. It offers a rich and realistic virtual environment that includes urban layouts with roads, traffic signs, pedestrians, and vehicles, all modeled to mimic real-world conditions. The simulator incorporates physics-based vehicle dynamics, allowing for accurate representation of vehicle motion and interactions with the environment. In addition, it provides a wide range of sensors such as cameras, LiDAR (Light Detection and Ranging), radar, GNSS (Global Navigation Satellite System), and IMU (Inertial Measurement Unit), which are essential for perception and navigation tasks. Because testing on real vehicles is often costly and potentially unsafe in the early stages of

development, researchers rely on CARLA to experiment with perception, planning, and control algorithms in a safe, flexible, and reproducible manner.

ROS (Robot Operating System) is a middleware framework designed for building robotics software, providing developers with tools, libraries, and conventions to create robot applications more efficiently. It is widely used in autonomous driving, drones, manipulators, and research robots because it simplifies communication between components and enables the reuse of algorithms. The core functionality of ROS includes inter-module communication through topics, services, and actions, integration of reusable robotics software packages such as navigation and perception, and the execution of algorithms for robotics and autonomous vehicles. Within this ecosystem, CARLA provides a realistic virtual environment, while ROS serves as the robotics software framework. To connect these two systems, the CARLA–ROS bridge acts as a translator that enables seamless interaction by converting CARLA simulator data into ROS messages and translating ROS commands into CARLA controls, thereby ensuring effective communication between the simulation and robotics frameworks.

4.2 The Dynamic Model

We employed the following equations to represent the vehicle model:

$$m(\dot{v} - \nu r) = F_{xT} \tag{4.1}$$

$$m(\dot{\nu} + vr) = F_{yf} + F_{yr} \tag{4.2}$$

$$I_z \dot{r} = l_f F_{yf} - l_r F_{yr} \tag{4.3}$$

$$\dot{\theta} = r \tag{4.4}$$

$$\dot{X} = v\cos\theta - \nu\sin\theta \tag{4.5}$$

$$\dot{Y} = \nu\cos\theta + v\sin\theta \tag{4.6}$$

where v, ν, and r denote the longitudinal velocity, lateral velocity, and yaw rate of the vehicle at its center of gravity (CoG); X, Y, and θ are the longitudinal and lateral position and heading angle of the vehicle in the global coordinate; F_{yf} and F_{yr} are the total lateral forces of the front and rear tires; F_{xT} is the total longitudinal force of tires; m is the vehicle's mass; and I_z is the vehicle's moment of inertia around its vertical axis.

The vehicle is assumed to have a front steering system. A linear tire model is used for the lateral tire forces Gao et al. (2014):

$$F_{yf} = C_f \alpha_f = C_f \left(\delta - \frac{\nu + l_f r}{v} \right) \tag{4.7}$$

$$F_{yr} = C_r \alpha_r = C_r \left(-\frac{\nu - l_r r}{v} \right) \tag{4.8}$$

in which α_f and α_r are the sideslip angles of the front and rear tires, and δ is the steering angle. Moreover, C_f and C_r denote the cornering stiffness values of the front and rear tires, respectively, which are obtained similar to Gao et al. (2014).

Here are some road regulations on the minimum and maximum speed limits that the vehicle should not violate. Constraints are applied on the vehicle speed and tire forces to restrict their changes:

$$v_{\min} < v < v_{\max} \tag{4.9}$$

$$\left(\frac{F_{xT}}{F_{xT\text{-max}}} \right)^2 + \left(\frac{F_{y^*}}{F_{y^*\text{-max}}} \right)^2 < 1, \quad * = f, r \tag{4.10}$$

where $F_{xT\text{-max}}$ is the maximum total longitudinal tire force, and $F_{y^*\text{-max}}$ (for $* = f, r$) is the maximum front or rear lateral tire force. The parameters $u_{\min}$ and $u_{\max}$ represent

the minimum and maximum speed limits, respectively. In most cases, there is no minimum speed limit, so it is set to zero, and the desired speed is assigned to the maximum speed limit. It is notable that the constraints on the tire forces limit the sideslip angles to remain within intervals where the tires' lateral forces behave almost linearly Gao et al. (2014).

The first step in implementing the model in CARLA is to apply a calibration method for parameter identification. In this stage, the model parameters are determined by comparing the simulated outputs with the data obtained from open-loop experiments in CARLA. Some of the parameters are directly obtained from the CARLA documentation. Some of them are given as follows:

- Vehicle mass = 1847 (Kg)
- CoG to front axle = 1.16 (m)
- CoG to rear axle = 1.56 (m)
- Yaw moment of inertia = 1690 (Kgm^2)

However, both tire stiffnesses (C_f, C_r) vary widely depending on conditions such as load, tire type, and surface. The typical ranges found in literature are:

- $C_f \in$ [50,000 – 100,000] (N/rad)
- $C_r \in$ [60,000 – 120,000] (N/rad)

In order to identify these parameters, we define the following optimization problem:

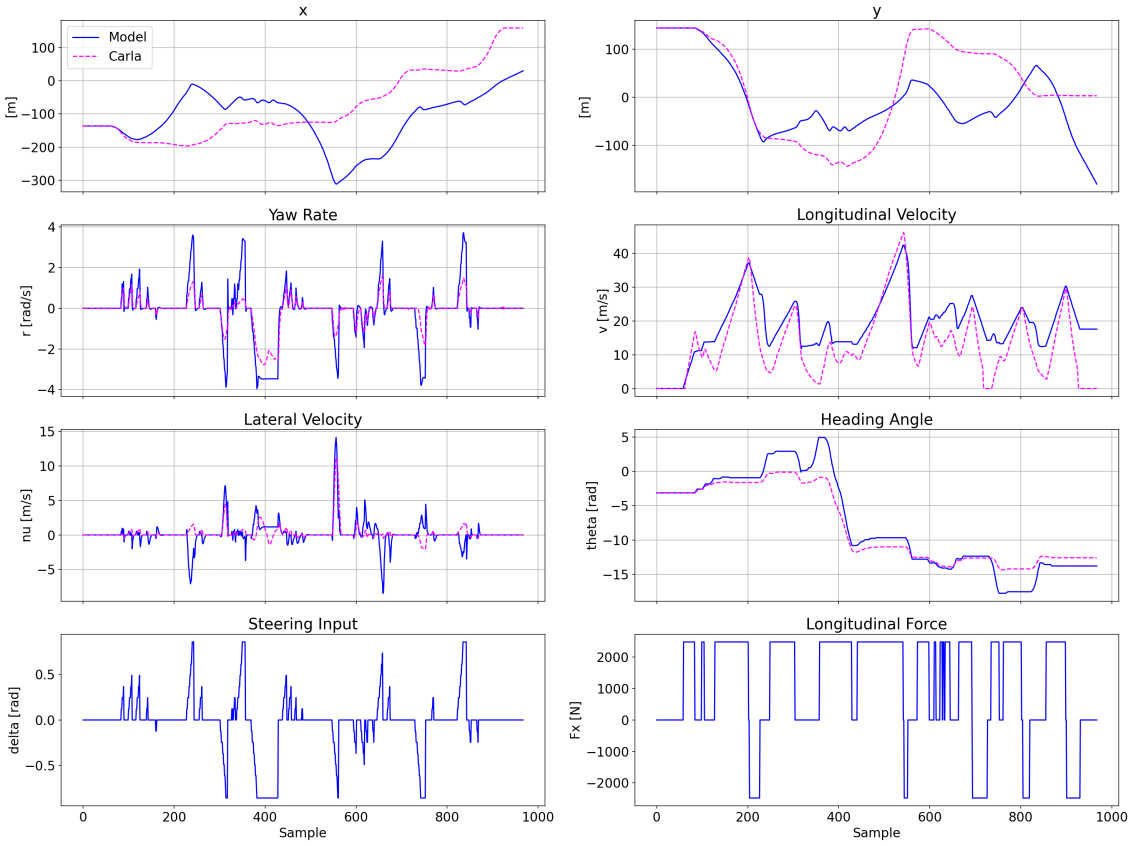


Figure 4.1: Model's output vs CARLA's data

$$\min \sum_{\text{data entries}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (v_i - \hat{v}_i)^2 + (r_i - \hat{r}_i)^2 + (\nu_i - \hat{\nu}_i)^2 + (\theta_i - \hat{\theta}_i)^2\right]$$

Subject to:

$$50,000 \leq C_f, C_r \leq 120,000 \tag{4.11}$$

This problem is solved by Particle Swarm Optimization (PSO). The optimal values are $C_f = 78,000$ and $C_r = 85,000$.

Figure 4.1 shows that while the model successfully captures the overall motion patterns observed in CARLA, notable discrepancies emerge across several states. The model consistently smooths out the sharp variations present in CARLA data, most

evident in the yaw rate, steering input, and longitudinal force, where CARLA demonstrates more abrupt and noisy behavior. In trajectory tracking, the model generally follows the same path but gradually drifts away from CARLA, with the deviations becoming more pronounced in the lateral direction. Similarly, in the velocity profiles, the model tends to underestimate or overshoot during rapid accelerations and decelerations, failing to reproduce the sudden fluctuations observed in CARLA. The lateral velocity and heading angle also appear less responsive, showing gradual drifts instead of the sharper variations captured by CARLA. Overall, these differences indicate that the model does not fully capture the high-frequency, discontinuous, or abrupt dynamics present in CARLA's data, producing instead a smoother but less accurate approximation of the vehicle's behavior.

4.3 Extended Kalman Filter

Since model (4.1)-(4.6) is smoother and cannot capture the abrupt, noisy variations of CARLA data, the EKF is applied. The EKF fuses the model predictions with the noisy CARLA-like measurements, correcting deviations and improving the accuracy of estimated states like yaw rate, velocity, and trajectory.

The standard Kalman Filter assumes the system dynamics and measurement models are linear, which makes it unsuitable for most real-world applications such as vehicle dynamics, robotics, or navigation, where the equations of motion are nonlinear. To address this, the EKF linearizes the nonlinear models around the current state estimate using a first-order Taylor series expansion (Jacobian matrices). This approach allows it to propagate uncertainties (covariances) through nonlinear models and still optimally fuse noisy measurements with predictions.

4.3.1 EKF Formulation

We assume the system evolves according to nonlinear state-space equations:

$$x_k = f(x_{k-1}, u_{k-1}) + w_{k-1}, \tag{4.12}$$

$$z_k = h(x_k) + v_k, \tag{4.13}$$

where $x_k \in \mathbb{R}^n$ denotes the state vector at time step k, u_k is the control input, and z_k is the measurement vector. The function $f(\cdot)$ represents the nonlinear state transition, while $h(\cdot)$ denotes the nonlinear measurement function. The process noise is modeled as $w_{k-1} \sim \mathcal{N}(0, Q_{k-1})$, and the measurement noise as $v_k \sim \mathcal{N}(0, R_k)$.

4.3.1.1 Initialization

Start with an initial guess:

$$\hat{x}_{0|0}, \quad P_{0|0},$$

where $\hat{x}_{0|0}$ is the initial state estimate and $P_{0|0}$ is the initial covariance (uncertainty in the estimate).

4.3.1.2 Prediction (Time Update)

We use the nonlinear system dynamics f to predict the next state:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}),$$

and propagate the covariance by linearizing f:

$$F_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1|k-1}, u_{k-1}},$$

$$P_{k|k-1} = F_{k-1} P_{k-1|k-1} F_{k-1}^T + Q_{k-1}.$$

4.3.1.3 Update (Measurement Correction)

When a measurement z_k arrives, we update:

Innovation (measurement residual):

$$y_k = z_k - h(\hat{x}_{k|k-1}),$$

Linearize the measurement model:

$$H_k = \left. \frac{\partial h}{\partial x} \right|_{\hat{x}_{k|k-1}},$$

Innovation covariance:

$$S_k = H_k P_{k|k-1} H_k^T + R_k,$$

Kalman gain:

$$K_k = P_{k|k-1} H_k^T S_k^{-1},$$

State update:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k y_k,$$

Covariance update:

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}.$$

4.4 The Lyapunov Model with Kalman Filter

Consider the following system derived from the model:

$$v_{k+1} = v_k + \left(\frac{F_{xk}}{m} + \nu_k r_k \right) dt, \tag{4.14}$$

$$\nu_{k+1} = \nu_k + \left(\frac{F_f + F_r}{m} - v_k r_k \right) dt, \tag{4.15}$$

$$\theta_{k+1} = \theta_k + r_k \, dt, \tag{4.16}$$

$$r_{k+1} = r_k + \left(\frac{l_f F_f - l_r F_r}{I_z} \right) dt, \tag{4.17}$$

$$x_{k+1} = x_k + v_k \cos(\theta_k) \, dt - \nu_k \sin(\theta_k) \, dt, \tag{4.18}$$

$$y_{k+1} = y_k + v_k \sin(\theta_k) \, dt + \nu_k \cos(\theta_k) \, dt. \tag{4.19}$$

where

$$x_k = [\, v_k, \ \nu_k, \ \theta_k, \ r_k, \ x_k, \ y_k \,]^T, \tag{4.20}$$

$$o_k = [\, v_k, \ \nu_k, \ \theta_k \,]^T. \tag{4.21}$$

are the state and output of the problem.

By writing the above system in vectorized form, we obtain the following represen-

tation:

$$\begin{bmatrix} v_{k+1} \\ \nu_{k+1} \\ \theta_{k+1} \\ r_{k+1} \\ x_{k+1} \\ y_{k+1} \end{bmatrix} = f(x_k, u_k) = \begin{bmatrix} v_k + \left(\frac{F_{xk}}{m} + \nu_k r_k\right)\Delta t \\ \nu_k + \left(\frac{F_f + F_r}{m} - v_k r_k\right)\Delta t \\ \theta_k + r_k \Delta t \\ r_k + \left(\frac{l_f F_f - l_r F_r}{I_z}\right)\Delta t \\ x_k + v_k \cos\theta_k \, \Delta t - \nu_k \sin\theta_k \, \Delta t \\ y_k + v_k \sin\theta_k \, \Delta t + \nu_k \cos\theta_k \, \Delta t \end{bmatrix}. \tag{4.22}$$

Jacubian matrices of the system are obtained as follows:

$$F = \begin{bmatrix} 1 & r_k\Delta t & 0 & \nu_k\Delta t & 0 & 0 \\ \left(\frac{C_f(\nu_k + l_f r_k)}{v_k^2} + \frac{C_r(\nu_k - l_r r_k)}{v_k^2}\right)\frac{\Delta t}{m} - r_k\Delta t & 1 - \frac{(C_f + C_r)}{m v_k}\Delta t & 0 & \left(\frac{C_r l_r - C_f l_f}{m v_k} - v_k\right)\Delta t & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ \left(\frac{C_f l_f(\nu_k + l_f r_k)}{v_k^2} - \frac{C_r l_r(\nu_k - l_r r_k)}{v_k^2}\right)\frac{\Delta t}{I_z} & \frac{(C_r l_r - C_f l_f)\Delta t}{I_z v_k} & 0 & 1 - \frac{\Delta t}{I_z v_k}(C_f l_f^2 + C_r l_r^2) & 0 & 0 \\ \cos\theta_k\Delta t & -\sin\theta_k\Delta t & -v_k\sin\theta_k\Delta t - \nu_k\cos\theta_k\Delta t & 0 & 1 & 0 \\ \sin\theta_k\Delta t & \cos\theta_k\Delta t & v_k\cos\theta_k\Delta t + \nu_k\sin\theta_k\Delta t & 0 & 0 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & r_k\,\Delta t & 0 & \nu_k\,\Delta t & 0 & 0 \\ \left(\frac{C_f(\nu_k + l_f r_k)}{v_k^2} + \frac{C_r(\nu_k - l_r r_k)}{v_k^2}\right)\frac{\Delta t}{m} - r_k\Delta t & 1 - \frac{(C_f + C_r)}{m v_k}\Delta t & 0 & \left(\frac{C_r l_r - C_f l_f}{m v_k} - v_k\right)\Delta t & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \end{bmatrix}$$

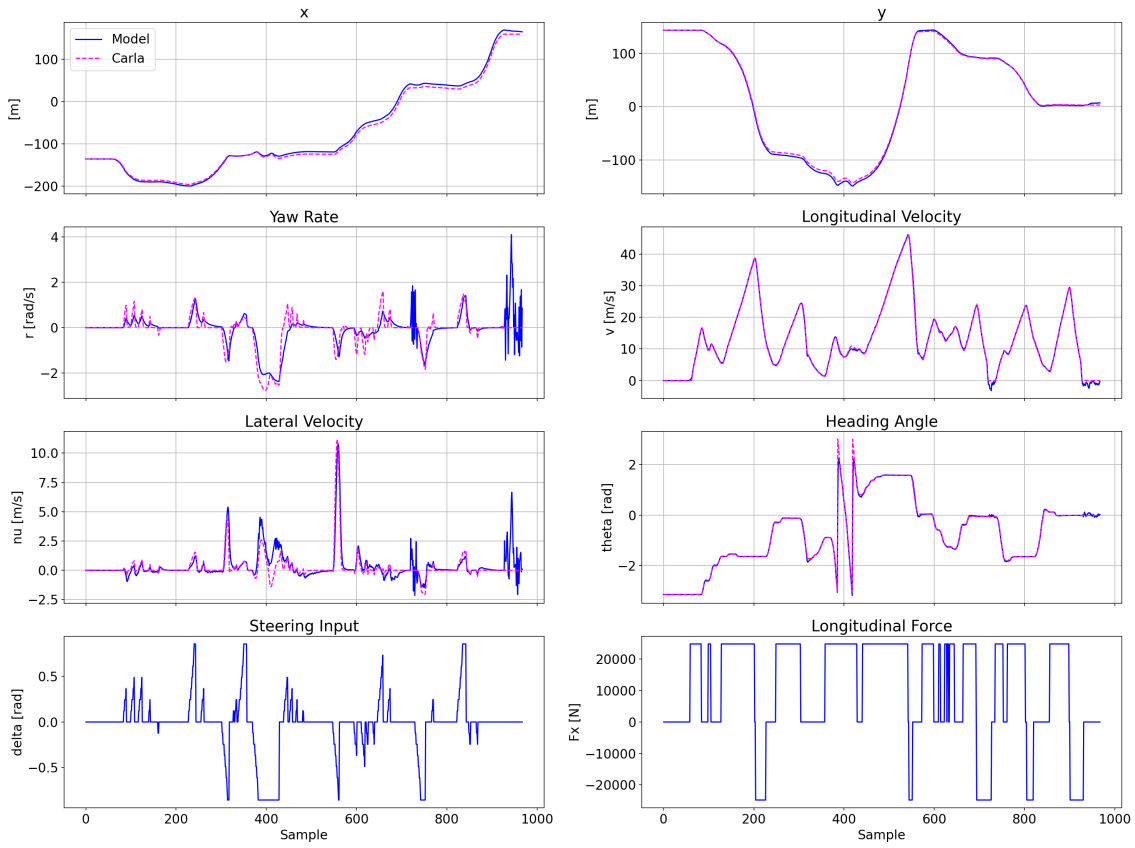


Figure 4.2: Model's output vs CARLA's data after applying EKF

Besides, if we select the covariance matrices Q_k and R_k, and the initial P_0 as,

$$Q_k = I_{6\times6}, \quad R_k = I_{3\times3}, \quad P_0 = I_{6\times6},$$

the model fitting result in Figure 4.2 will be obtained.

After applying the EKF, the system model shows a strong alignment with the CARLA data, demonstrating that it effectively captures the underlying dynamics. This close agreement confirms the model's accuracy and reliability, making it well-suited for successful implementation within the CARLA simulator.

4.4.1 Optimization Problem

The following nonlinear-constrained optimization problem is defined to solve the local path-planning problem while ensuring safety and maintaining the desired speed.

$$\min \qquad \frac{1}{N}\mathbb{E}\left\{\sum_{k=1}^{N}\left((v_k - v_d)^2 + (x_k - w_x)^2 + (y_k - w_y)^2 + 10^3\,(\delta_k - \delta_d)^2\right)\right\} \tag{4.23a}$$

subject to

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ v_{k+1} \\ \nu_{k+1} \\ r_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_k \\ \nu_k \\ r_k \end{bmatrix} + \begin{bmatrix} v_k\cos\theta_k - \nu_k\sin\theta_k \\ v_k\sin\theta_k + \nu_k\cos\theta_k \\ r_k \\ F_k^x/m + \nu_k r_k \\ (F_k^f + F_k^r)/m - v_k r_k \\ (l_f F_k^f - l_r F_k^r)/I_z \end{bmatrix} \Delta t, \quad \forall k, \tag{4.23b}$$

$$\left(\frac{F_k^x}{F_{\max}^x}\right)^2 + \left(\frac{F_k^f}{F_{\max}^f}\right)^2 < 1, \quad \forall k, \tag{4.23c}$$

$$\left(\frac{F_k^x}{F_{\max}^x}\right)^2 + \left(\frac{F_k^r}{F_{\max}^r}\right)^2 < 1, \quad \forall k, \tag{4.23d}$$

$$r^{\min} \le r_k \le r^{\max}, \quad \forall k, \tag{4.23e}$$

$$\nu^{\min} \le \nu_k \le \nu^{\max}, \quad \forall k, \tag{4.23f}$$

$$p_k^l < x_k < p_k^r, \quad \forall k, \tag{4.23g}$$

where N is the total number of time steps, $\mathbb{E}\{\cdot\}$ denotes expectation, v_d presents the desired speed, (x_k, y_k) is the position of the ego vehicle at time k, and (w_x, w_y) is the closest point on reference line to the ego vehicle. The second and third terms aim to minimize the reference position tracking error.

4.4.2 Finding p_k^r and p_k^l

Static obstacles are added to the boundaries of the road. To do so, 11 points are considered on the left (right) side of the obstacles on the right (left) side of the road, as shown in Fig. 4.3. These points can be easily obtained from the coordinates of obstacles' bounding box. Then, Spline interpolation is used to plot a curve connecting the 11 points. Finally, the points on this curve are represented by p_k^r and p_k^l. Subsequently, the Spline curve is rotated around the obstacle's center based on the road's curvature. Suppose by rotating the point (x, y) around (x_o, y_o), the new point (x', y') is obtained. We have

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = T(x_o, y_o)R(\theta)T(-x_o, -y_o) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{4.24}$$

where

$$T(x, y) = \begin{bmatrix} 1 & 0 & x \\ 0 & 1 & y \\ 0 & 0 & 1 \end{bmatrix},$$

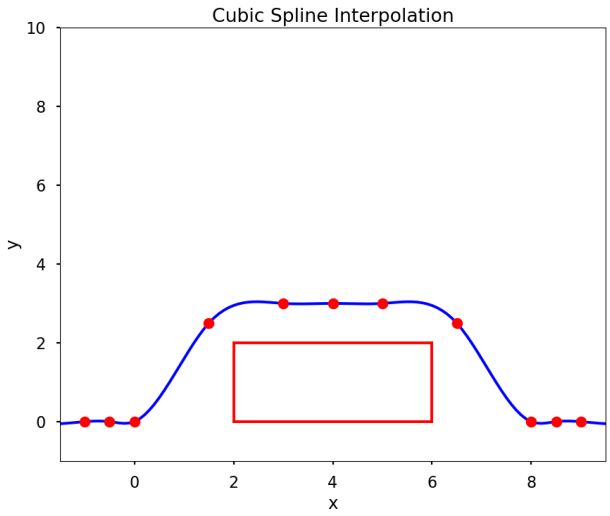


Figure 4.3: Adding the obstacle to the road boundary

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Note that the rotation angle θ is calculated using the ego vehicle's CoG and the obstacle's center (see Fig. 4.4). Lastly, Fig. 4.5 displays rotated Spline curves for the obstacles in a real scenario.

It is worth mentioning that the 11 points are selected in such a way that the obtained curve starts 15 meters before the obstacle and gradually changes the road boundary with a small slope. This setting helps the LO to smoothly follow the road boundary, avoiding infeasibility. If the road boundary curve is poorly shaped, it might be impossible for the LO to find a maneuver to respect the boundary. The reason is that the LO only solves the problem for one time step. We should make sure that this one time step includes enough information about the path ahead to help the AV

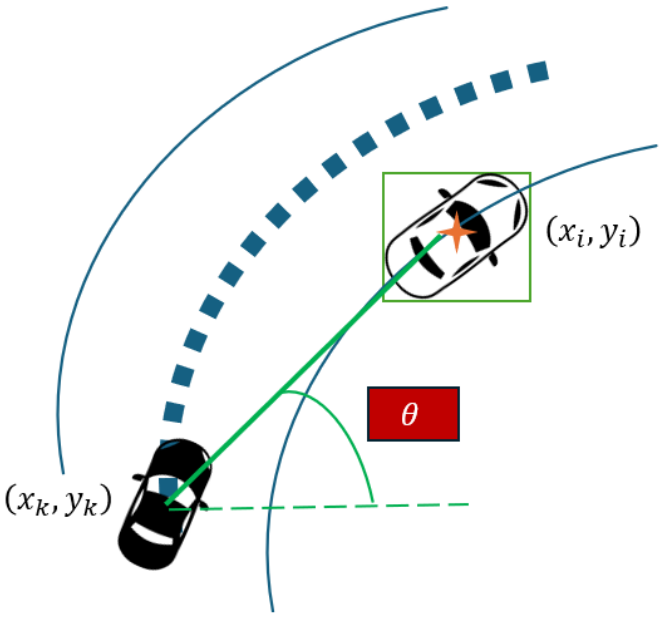


Figure 4.4: Finding the rotation angle

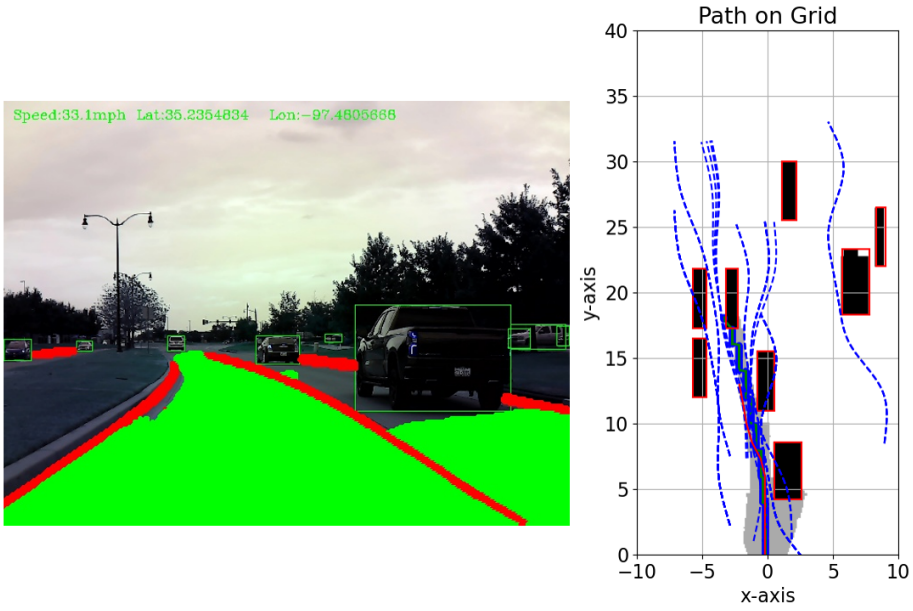


Figure 4.5: Rotated Spline curves. The blue, thick, stepwise curve is the reference line, while the pink curve is the optimal path found by LO. Note that the optimal path respects the bounds of p_k^r and p_k^l.

navigate through the obstacles safely without needing to exceed the limits of physical constraints. Additionally, although the Spline curve is plotted symmetrically before

and after each obstacle, the part after the obstacle is discarded in practice, seeing that we do not need to unnecessarily restrain AV maneuvers once the ego vehicle passes an obstacle.

4.4.3 Reformulation

A virtual queue is constructed for each state of (4.23b) as the following,

$$Q_k^x = x_k - x_k^d, \tag{4.25a}$$

$$Q_k^y = y_k - y_k^d, \tag{4.25b}$$

$$Q_k^\theta = \theta_k - \theta_k^d, \tag{4.25c}$$

$$Q_k^v = v_k - v_k^d, \tag{4.25d}$$

$$Q_k^\nu = \nu_k - \nu^d, \tag{4.25e}$$

$$Q_k^r = r_k - r^d, \tag{4.25f}$$

where x_k^d, y_k^d, θ_k^d, v_k^d, ν^d, and r^d are desired state values. In detail, (x_k^d, y_k^d) are coordinates of the nearest waypoint (obtained from the reference line), θ_k^d is the desired heading angle based on the relative position of the vehicle and the reference line, and v_k^d is the desired velocity. Define the queue length vector as $Q_k = \left[Q_k^x, Q_k^y, Q_k^\theta, Q_k^v, Q_k^\nu, Q_k^r\right]$. By putting (4.25) in (4.23b), we have

$$Q_{k+1}^x = Q_k^x + (v_k \cos\theta_k - \nu_k \sin\theta_k)\Delta t - x_{k+1}^d + x_k^d = Q_k^x + \xi_k^x, \tag{4.26a}$$

$$Q_{k+1}^y = Q_k^y + (v_k \sin\theta_k + \nu_k \cos\theta_k)\Delta t - y_{k+1}^d + y_k^d = Q_k^y + \xi_k^y, \tag{4.26b}$$

$$Q_{k+1}^\theta = Q_k^\theta + r_k \Delta t - \theta_{k+1}^d + \theta_k^d = Q_k^\theta + \xi_k^\theta, \tag{4.26c}$$

$$Q_{k+1}^v = Q_k^v + \left(\frac{F_k^x}{m} + \nu_k r_k\right)\Delta t - v_{k+1}^d + v_k^d = Q_k^v + \xi_k^v, \tag{4.26d}$$

$$Q_{k+1}^{\nu} = Q_k^{\nu} + \left(\frac{F_k^f + F_k^r}{m} - v_k r_k \right) \Delta t = Q_k^{\nu} + \xi_k^{\nu}, \tag{4.26e}$$

$$Q_{k+1}^{r} = Q_k^{r} + \frac{l_f F_k^f - l_r F_k^r}{I_z} \Delta t = Q_k^{r} + \xi_k^{r}, \tag{4.26f}$$

where

$$\begin{aligned}
\xi_k^x &= (v_k \cos\theta_k - \nu_k \sin\theta_k)\Delta t - x_{k+1}^d + x_k^d \\
\xi_k^y &= (v_k \sin\theta_k + \nu_k \cos\theta_k)\Delta t - y_{k+1}^d + y_k^d \\
\xi_k^\theta &= r_k \Delta t - \theta_{k+1}^d + \theta_k^d \\
\xi_k^v &= \left(F_k^x/m + \nu_k r_k\right) \Delta t - v_{k+1}^d + v_k^d \\
\xi_k^\nu &= \left((F_k^f + F_k^r)/m - v_k r_k\right) \Delta t \\
\xi_k^r &= \Delta t (l_f F_k^f - l_r F_k^r)/I_z
\end{aligned}$$

Using the virtual queues in (4.25), problem (4.23) can be transformed into a queue stability problem (4.27), where (4.23b) is changed to the stability of the queues.

$$\min \quad \frac{1}{N}\mathbb{E}\left\{ \sum_{k=1}^{N} \left(\Omega_1 (v_k - v_d)^2 + \Omega_2 (x_k - w_x)^2 + \Omega_3 (y_k - w_y)^2 + \Omega_4 (\delta_k - \delta_d)^2 \right) \right\} \tag{4.27a}$$

subject to

$$\begin{aligned}
&(4.23c) - (4.23g) \\
&\text{stability of virtual queues } Q_k.
\end{aligned} \tag{4.27b}$$

Subsequently, the LO is applied to (4.27), which establishes an adaptive control policy for local path planning. Problem (4.27) is decomposed into N subproblems that are

solved independently, while system stability is guaranteed.

4.4.4 Lyapunov Optimization Model

Define

$$u_k = \Omega_1(v_k - v_d)^2 + \Omega_2(x_k - w_x)^2 + \Omega_3(y_k - w_y)^2 + \Omega_4(\delta_k - \delta_d)^2. \tag{4.28}$$

Next, the Lyapunov function is defined as follows,

$$L(Q_k) \triangleq \frac{\beta_1}{2}\left(Q_k^x\right)^2 + \frac{\beta_2}{2}\left(Q_k^y\right)^2 + \frac{\beta_3}{2}\left(Q_k^\theta\right)^2 + \frac{\beta_4}{2}\left(Q_k^v\right)^2 + \frac{\beta_5}{2}\left(Q_k^\nu\right)^2 + \frac{\beta_6}{2}\left(Q_k^r\right)^2, \tag{4.29}$$

where $\beta_i,\ i \in \{1, 2, \cdots, 6\}$ are positive constants. The Lyapunov drift is

$$\Delta(Q_k) \triangleq \mathbb{E}\left\{L(Q_{k+1}) - L(Q_k)|Q_k\right\}. \tag{4.30}$$

Finally, the drift-plus-penalty term will be as,

$$\Delta(Q_k) + V\mathbb{E}\left\{u_k|Q_k\right\}. \tag{4.31}$$

Theorem 4.1. *At any time slot k, the Lyapunov drift-plus-penalty term has the following upper bound,*

$$\begin{aligned}\Delta(Q_k) + V\mathbb{E}\{u_k|Q_k\} \le B &+ \beta_1 Q_k^x \mathbb{E}\{\varrho_k^x|Q_k\} + \beta_2 Q_k^y \mathbb{E}\{\varrho_k^y|Q_k\} + \beta_3 Q_k^\theta \mathbb{E}\{\varrho_k^\theta|Q_k\} \\ &+ \beta_4 Q_k^v \mathbb{E}\{\varrho_k^v|Q_k\} + \beta_5 Q_k^\nu \mathbb{E}\{\varrho_k^\nu|Q_k\} + \beta_6 Q_k^r \mathbb{E}\{\varrho_k^r|Q_k\} + V\mathbb{E}\{u_k|Q_k\},\end{aligned} \tag{4.32}$$

where B is a positive constant shown in (C.11)*. In addition, ϱ_k^x, ϱ_k^y, ϱ_k^θ, ϱ_k^v, ϱ_k^ν, and ϱ_k^r are defined in* (C.7)*.*

Proof. See Appendix C. ■

In Theorem 4.2, we will demonstrate minimizing the upper bound of (4.32), instead of its left-hand side, for each time step k leads to a suboptimal solution which is close to the global optimal objective value of (4.23a), denoted as p^*. The convergence of the suboptimal solution to p^* is controlled by coefficient V.

Before diving into Theorem 4.2, an optimization problem is formulated to minimize the upper bound of (4.32) as follows,

$$\begin{aligned} \min \quad & V u_k + \beta_1 Q_k^x \varrho_k^x + \beta_2 Q_k^y \varrho_k^y + \beta_3 Q_k^\theta \varrho_k^\theta + \beta_4 Q_k^v \varrho_k^v \\ & + \beta_5 Q_k^\nu \varrho_k^\nu + \beta_6 Q_k^r \varrho_k^r, \\ \text{subject to} \quad & \text{(4.23c)-(4.23g)}, \end{aligned} \tag{4.33}$$

Note that Q_k^x, Q_k^v, Q_k^θ, Q_k^v, Q_k^ν, and Q_k^r are constants whose values are updated after solving (4.33) for time step k and before running (4.33) for time step $k+1$. Moreover, when (4.33) is being solved for $k+1$, the values of states at time step k are known. Therefore, according to (C.7), the values of ϱ_k^x and ϱ_k^y are known, making terms $\beta_1 Q_k^x \varrho_k^x$ and $\beta_2 Q_k^y \varrho_k^y$ constants. Hence, solving (4.33) in its current form won't lead to the minimization of Q_k^x and Q_k^y. In other words, by minimizing (4.33), the reference line is not tracked by the ego vehicle. The issue arises from the fact that x_k and y_k implicitly depend on a_k and ω_k. To resolve the issue, we propose using v_{k+1} and θ_{k+1} instead of v_k and θ_k in ϱ_k^x and ϱ_k^y as follows. The proposition is further verified with simulations in Section 4.5. Assume,

$$\begin{aligned} Q_k^x \varrho_k^x &= Q_k^x (v_k \cos\theta_k - \nu_k \sin\theta_k)\Delta t \approx \\ & \qquad Q_k^x (v_{k+1} \cos\theta_{k+1} - \nu_{k+1} \sin\theta_{k+1})\Delta t, \qquad \text{(4.34a)} \\ Q_k^y \varrho_k^y &= Q_k^y (v_k \sin\theta_k + \nu_k \cos\theta_k)\Delta t \approx \end{aligned}$$

$$Q_k^y(v_{k+1}\sin\theta_{k+1} + \nu_{k+1}\cos\theta_{k+1})\Delta t. \tag{4.34b}$$

Utilizing (4.23b), v_{k+1}, θ_{k+1}, and ν_{k+1} are substituted as,

$$\begin{aligned}
Q_k^x\varrho_k^x &= Q_k^x(v_k + (F_k^x/m + \nu_k r_k)\Delta t)\cos(\theta_k + r_k\Delta t)\Delta t \\
&- Q_k^x(\nu_k + ((F_k^f + F_k^r)/m - v_k r_k)\Delta t)\sin(\theta_k + r_k\Delta t)\Delta t, \qquad (4.35a) \\
Q_k^y\varrho_k^y &= Q_k^y(v_k + (F_k^x/m + \nu_k r_k)\Delta t)\sin(\theta_k + r_k\Delta t)\Delta t \\
&+ Q_k^y(\nu_k + ((F_k^f + F_k^r)/m - v_k r_k)\Delta t)\cos(\theta_k + r_k\Delta t)\Delta t \qquad (4.35b)
\end{aligned}$$

Finally, leveraging (C.7) and plugging (4.35) in the objective function of (4.33), we obtain,

$$\begin{aligned}
\min \quad & Vu_k + \Lambda_k\Delta t, \qquad (4.36) \\
\text{subject to} \quad & \text{(4.23c)-(4.23g)},
\end{aligned}$$

where u_k is defined in (4.28) and

$$\begin{aligned}
\Lambda_k &= \beta_1 Q_k^x\Big(v_k + (F_k^x/m + \nu_k r_k)\Delta t\Big)\cos(\theta_k + r_k\Delta t) \\
&- \beta_1 Q_k^x\Big(\nu_k + ((F_k^f + F_k^r)/m - v_k r_k)\Delta t\Big)\sin(\theta_k + r_k\Delta t) \\
&+ \beta_2 Q_k^y\Big(v_k + (F_k^x/m + \nu_k r_k)\Delta t\Big)\sin(\theta_k + r_k\Delta t) \\
&+ \beta_2 Q_k^y\Big(\nu_k + ((F_k^f + F_k^r)/m - v_k r_k)\Delta t\Big)\cos(\theta_k + r_k\Delta t) \\
&+ \beta_3 Q_k^\theta r_k + \beta_4 Q_k^v\Big(F_k^x/m + \nu_k r_k\Big) \\
&+ \beta_5 Q_k^\nu\Big((F_k^f + F_k^r)/m - v_k r_k\Big) \\
&+ \beta_6 Q_k^r\Big(l_f F_k^f - l_r F_k^r\Big)/I_z. \qquad (4.37)
\end{aligned}$$

Algorithm 2: Lyapunov Optimization

Data: V, N, x_0, y_0, θ_0, v_0, ν_0, r_0, x_0^d, y_0^d, θ_0^d, v_0^d, ν^d, r^d, F_0^x, δ_0, Q_1
Result: $\forall k$: x_k, y_k, θ_k, v_k, ν_k, r_k, F_k^x, δ_k, Q_k
for $k = 1, \cdots, N$ **do**
  Find $c_k = \begin{bmatrix} F_k^x & \delta_k \end{bmatrix}^T$ by solving (4.36);
  Update states x_k, y_k, θ_k, v_k, ν_k, and r_k via (4.23b);
  Update desired values x_{k+1}^d, y_{k+1}^d, θ_{k+1}^d, and v_{k+1}^d;
  Update Q_{k+1} via (4.25)
end

Problem (4.36) minimizes the upper bound of (4.32) for time step k. After each time step k, the queues should be updated before running problem (4.36) for the next time step $k+1$ as presented in Algorithm 2.

Theorem 4.2 comprises the following steps:

- A relaxed version of problem (4.23) is constructed in (D.2).
- It is shown the optimal objective of (D.2), denoted as $\tilde{p}$, is less than or equal to the optimal objective of (4.23), p^*, i.e., $\tilde{p} \leq p^*$.
- (D.2) satisfies the constraints of problem (4.36).
- The optimal solution of (4.36) is denoted by $\hat{c}_k$. Evaluating the objective of (4.36) at $\hat{c}_k$ yields $V\hat{u}_k + \hat{\Lambda}_k$.
- The objective of (4.36) evaluated at any values other than $\hat{c}_k$ is greater than or equal to $V\hat{u}_k + \hat{\Lambda}_k$.
- Evaluating the objective of (4.36) by the solution of (D.2) yields $V\hat{u}_k + \hat{\Lambda}_k \leq V\tilde{u}_k + \tilde{\Lambda}_k$.
- With the help of mathematical manipulations, using $\tilde{p} \leq p^*$ and $V\hat{u}_k + \hat{\Lambda}_k \leq V\tilde{u}_k + \tilde{\Lambda}_k$, we derive $\frac{1}{N}\sum_{k=1}^{N} \mathbb{E}\{\hat{u}_k\} \leq p^* + \frac{\text{some constants}}{V}$.

Theorem 4.2. *For all $N > 1$, the optimal solution of* (4.36)*, denoted by $\hat{c}_k$, and the optimal solution of* (4.23)*, denoted by c_k^*, satisfy the following inequality,*

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\left\{\hat{u}_k\right\} \leq \frac{B_1}{V} + p^* + \frac{1}{NV}\mathbb{E}\left\{L(\hat{Q}_1)\right\}, \tag{4.38}$$

where B_1 is a constant defined in (D.19)*, p^* is the optimal objective value of* (4.23a) *and $\hat{u}_k$ is obtained by* (4.28) *using $\hat{c}_k$.*

Proof. See Appendix D. ■

It is worth noting that the proposed LO approach is implemented through Algorithm 2.

4.5 Results

The developed model in this chapter is used to control a vehicle in high-fidelity CARLA simulator. The results are compared with a PID controller as well. Since all of the CARLA's vehicle information is not available in the form of the model parameters we use in this chapter, the discrepancies between the model and the CARLA simulator is handled by an EKF. CARLA has several towns to choose from. Here we selected town 5.

Fig. 4.6 shows the ego vehicle in CARLA. The global waypoints to follow are randomly generated and shown in blue circles, while the trajectory of the ego vehicle, that is the path traversed by the ego vehicle, is shown in red circles. Similarly, the next waypoint, out of all the blue waypoint in front of the ego vehicle, is also shown in red. This is the waypoint that the ego vehicle follows at this moment.

Fig. 4.7 illustrates the path to follow (in blue) and the path taken by the ego vehicle (in red). As shown in Fig. 4.7, the ego vehicle successfully follows the global path with

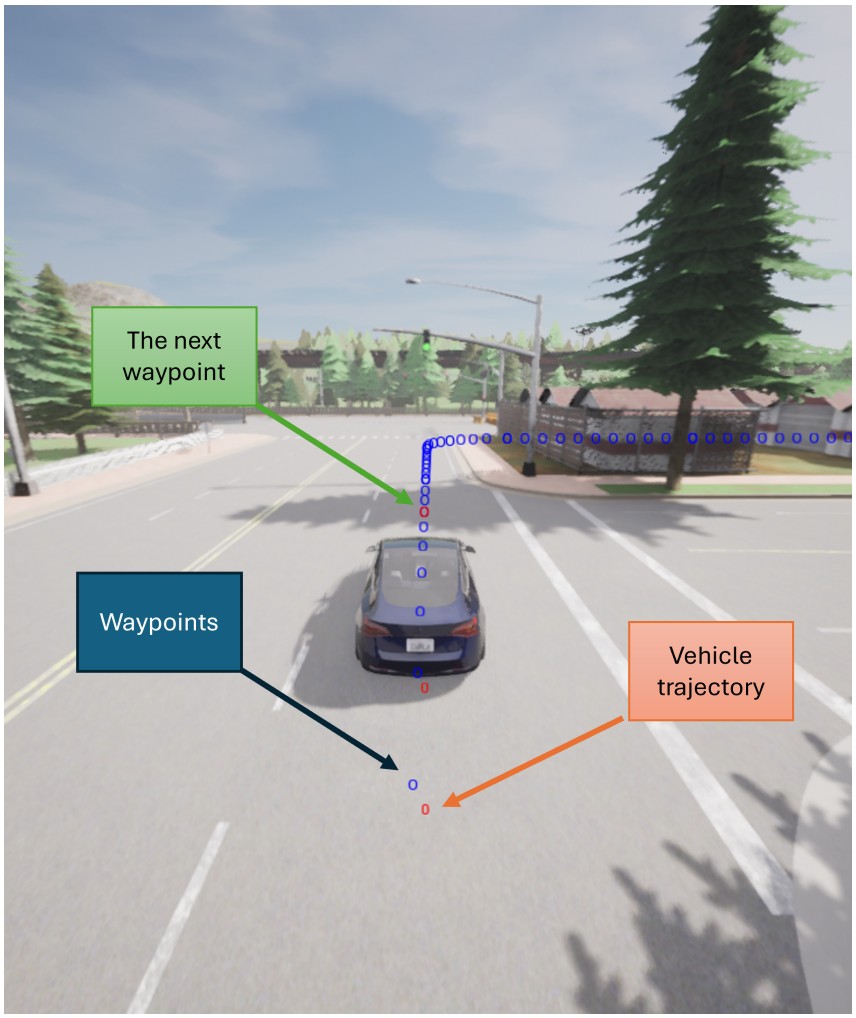


Figure 4.6: Ego vehicle in CARLA

minimal deviation. The vehicle exhibits slight deviations in turns, which is attributed to its effort to maintain the speed target.

In addition, the results of controlling the ego vehicle with a pair of PID controllers is plotted in Fig. 4.8, where the first PID controls the throttle (the longitudinal movement of the vehicle) and the second PID controls the steering (the lateral motion of the vehicle). Unlike the LO controller, in some parts of the trajectory, the PID controller exhibits an error in following the global path and follows it with a shift to the right. However, in other parts of the path, the deviation to the right disappears.



Figure 4.7: The LO local path planning result in CARLA's Town 5 environment.

Additionally, the left subplot in Fig. 4.9 displays the 2-dimensional view of the X-Y trajectory taken by the ego vehicle when controlled by LO or PID, while the subplot on the right compares the reference speed tracking of each controller. LO outperforms PID in both waypoint tracking and following the target speed. Both PID and LO exhibit somewhat similar patterns of speed drops throughout time which is due to the

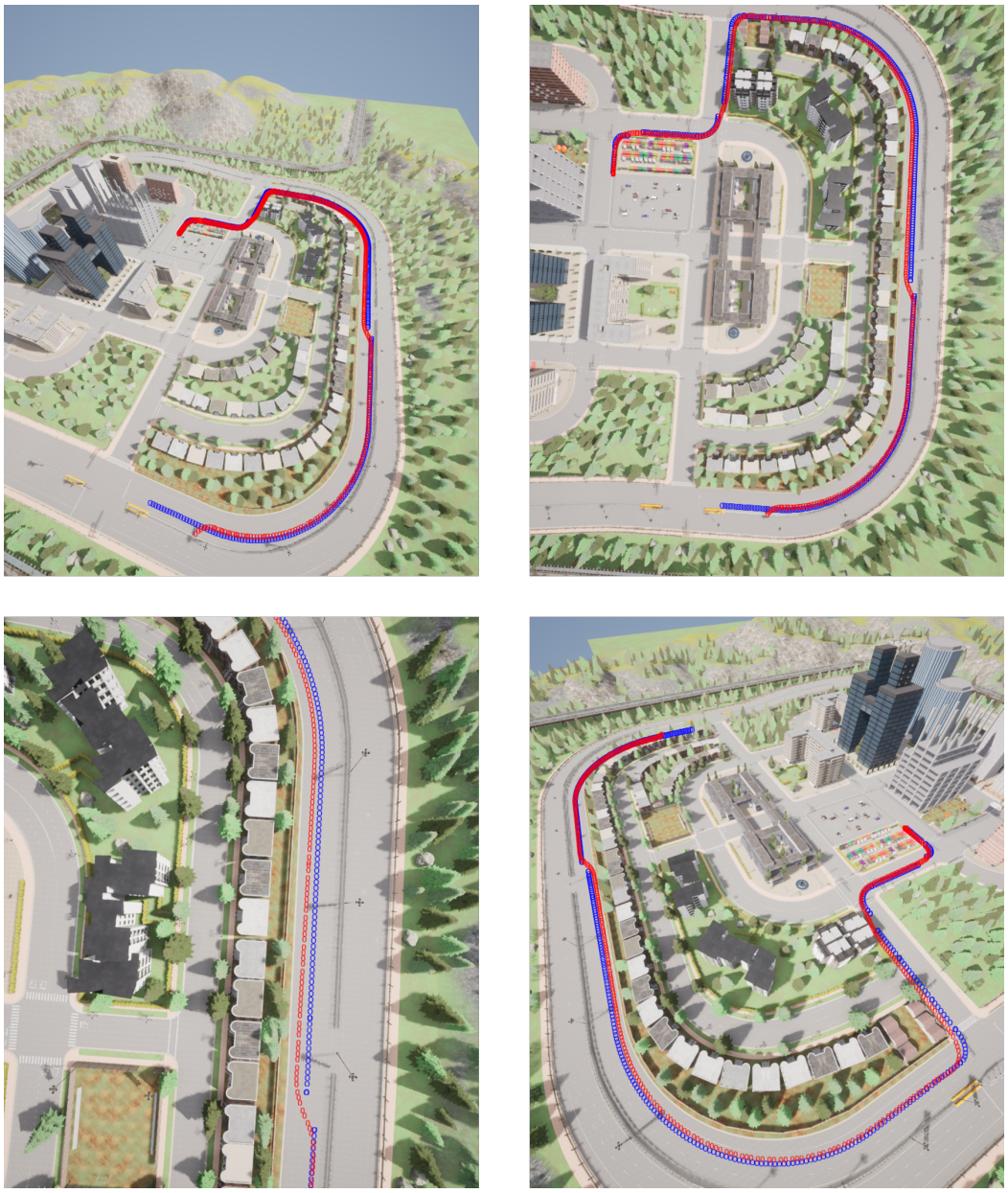


Figure 4.8: The PID local path planning result in CARLA's Town 5 environment.

turns on the road.

The LO controller offers significant advantages over the traditional PID controller for autonomous vehicle control. Unlike PID, which requires separate controllers for lateral and longitudinal motions, a single LO controller can simultaneously manage both, resulting in a more integrated and efficient control system. Furthermore, PID

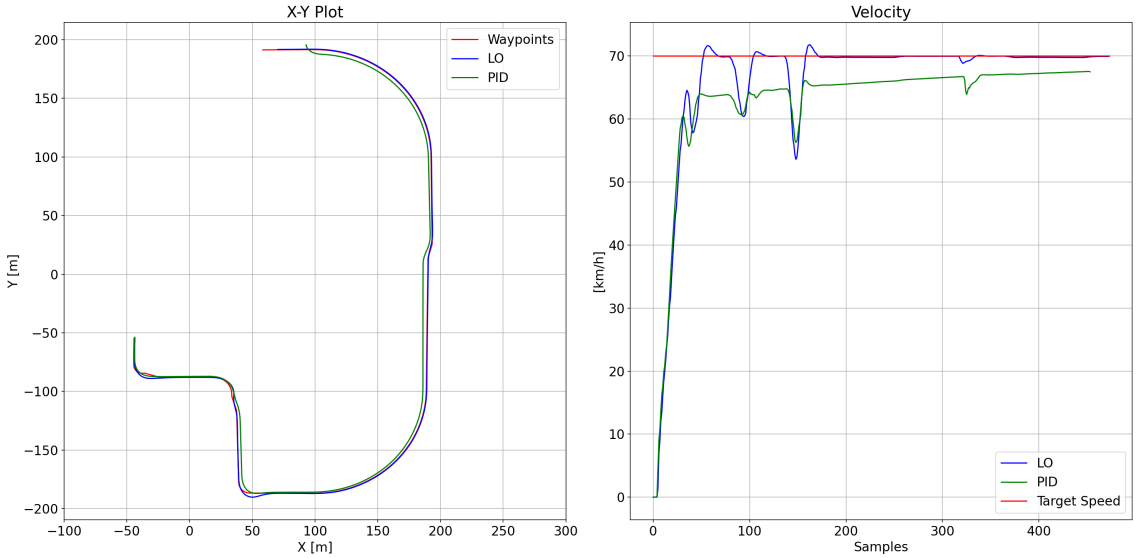


Figure 4.9: LO and PID local path planning and target speed tracking results.

controllers merely follow waypoints and target speeds without considering how the tracking is performed. In contrast, the LO controller can optimize the tracking process, for example, by minimizing control effort or fuel consumption while achieving both trajectory and speed objectives. Another key advantage of the LO controller is its ability to incorporate constraints and higher-level objectives, such as collision avoidance and adherence to road boundaries—capabilities that PID controllers inherently lack. This makes LO more suitable for complex, real-world driving scenarios where safety and regulation compliance are critical. Additionally, the LO approach decomposes the control problem into single-step subproblems, significantly reducing computational complexity. This decomposition results in exponential improvements in computation speed, which is crucial for real-time autonomous driving. Unlike Model Predictive Control (MPC), which solves an optimization problem over a multi-step prediction horizon at every time step, LO focuses on a single step, making it substantially faster. This computational efficiency, combined with its flexibility and optimization capabilities, makes the LO controller a highly suitable choice for autonomous vehicle

applications.

Chapter 5

Electric Vehicle Battery Thermal Management Under Extreme Fast Charging with Deep Reinforcement Learning

Electric Vehicles (EVs) and Autonomous Vehicles (AVs) are arguably two of the most prominent research topics in the transportation sector today. Both technologies play a pivotal role in reducing the sector's overall carbon footprint. Moreover, recent research trends are increasingly converging toward studies that integrate both domains—specifically, investigations where AVs are also EVs.

Accordingly, this chapter focuses on one of the critical safety challenges associated with EVs: thermal management during fast charging. As discussed in the Introduction section, there remains a significant gap in the literature on BTMS optimization under extreme fast charging conditions. Most existing studies primarily address BTMS design or focus on thermal management during vehicle operation (i.e., mostly during discharge at low C-rates). To address this gap, this chapter proposes a method to optimize an EV's BTMS under extreme fast charging conditions, aiming to minimize both the battery temperature and the power consumption required for cooling.

5.1 Thermal System Model

This section presents the dynamics of the BTMS. The general model of the BTMS under study is depicted in Fig. 5.1, where two loops are working together, involving the battery pack, radiator, reservoir, and chiller. As shown in Fig. 5.1, the chiller cools the battery-cooling liquid using refrigerant. This chapter focuses on controlling the battery coolant mass flow rate with deep RL while the refrigerant superheat control is executed through a PID controller. The RL agent and the PID controller configurations are shown in Fig. 5.1. In addition, since this chapter focuses on battery cooling during extreme fast charging, the battery is assumed to be charged via the constant current-constant voltage (CC-CV) algorithm, as CC-CV is widely used in practice and literature. Fig. 5.2, illustrates the CC-CV profile used in this chapter. Additionally, the indirect cooling system utilizes the non-flammable 1,1,1,2-Tetrafluoroethane (HFC-134a) coolant as the A/C system refrigerant and ethylene-glycol/water with proportions 50–50% as the battery cooling loop fluid.

The equation of battery heat generation based on the current I and internal resistance R_{int} is as follows:

$$q_{bat} = I^2 R_{int} \tag{5.1}$$

where the current is a function of time. Moreover, the battery pack temperature is computed by the following relation:

$$m_{bat} c_{bat} dT_{bat}/dt = q_{bat} + \eta_{bat} h_{bat,cool} A_{bat,cool} (T_{bat} - T_{bat,cool}) \tag{5.2}$$

where m_{bat} is the mass of the battery pack, c_{bat} is the battery specific heat capacity, η_{bat} is an efficiency factor that models the heat loss to the ambient by the battery cooling liquid, $h_{bat,cool}$ is the heat transfer coefficient between the battery and the cooling fluid,

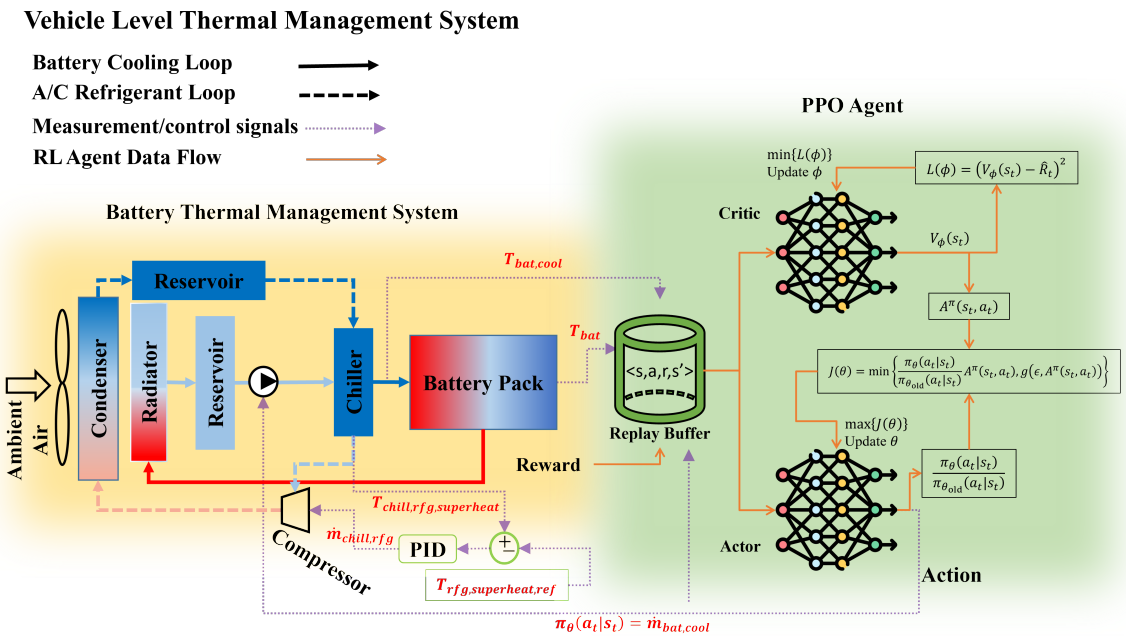


Figure 5.1: EV TMS configuration including A/C system (i.e., compressor, condenser, and reservoir), indirect battery cooling loop (i.e., chiller, radiator, and battery pack).

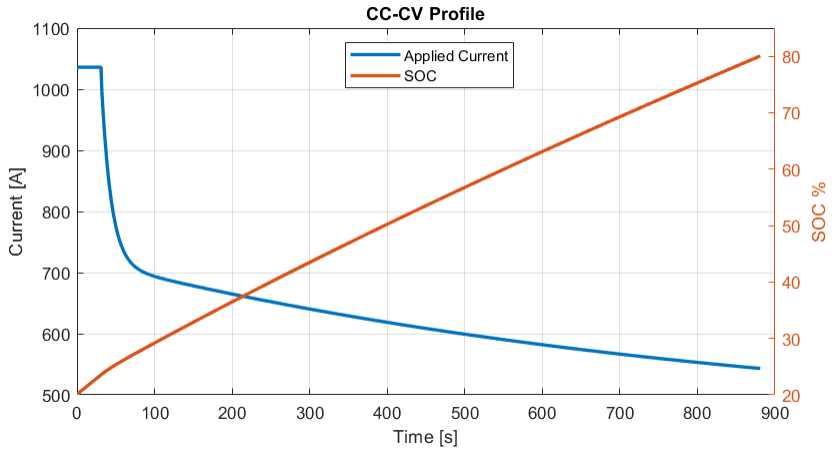


Figure 5.2: CC-CV profile used in this chapter

Parameter	Value	Parameter	Value
$m_{chill,cool}$	$1[Kg]$	c_{bat}	$1000[J/K/kg]$
$m_{chill,w}$	$1[Kg]$	$c_{cool,bat}$	$3323[J/K/kg]$
$m_{chill,rfg}$	$1[Kg]$	$A_{chill,cool,w}$	$0.75[m^2]$
$c_{chill,w}$	$871J/K/kg$	$h_{chill,rfg,w}$	$750[W/K/m^2]$
$A_{chill,rfg,w}$	$0.75m^2$	$\eta_{chiller}$	$90[\%]$
$m_{rad,air}$	$0.1[kg]$	η_{rad}	$90[\%]$
$m_{rad,cool}$	$1[kg]$	η_{bat}	$90[\%]$
$m_{rad,w}$	$2[kg]$	$c_{chill,cool}$	$3323[J/K/kg]$
$c_{chill,rfg}$	$0.75m^2$	$\eta_{chiller}$	$90[\%]$
$c_{rad,w}$	$0.75m^2$	$\eta_{chiller}$	$90[\%]$
$c_{chill,rfg}$	$3000[J/K/Kg]$	$c_{rad,cool}$	$3000[J/K/Kg]$
$c_{rad,air}$	$1006[J/K/kg]$	a_2	-0.07598
$A_{rad,cool,w}$	$2[m^2]$	a_3	-0.01733
$h_{rad,air,w}$	$300[W/K/m^2]$	a_4	0.01593
$A_{rad,air,w}$	$2[m^2]$	a_5	-0.001453
$h_{rad,cool,w}$	$500[W/K/m^2]$	a_6	-0.001061
$m_{bat,cool}$	$2[kg]$	b_1	1.644
$h_{bat,cool}$	$350[W/K/m^2]$	b_2	-0.2
$A_{bat,cool}$	$N_{cell} * 0.01 * 0.065$	b_3	-0.2246
$\Delta p_{cool,pump}$	$2.5[bar]$	b_4	0.06626
$\eta_{cool,pump}$	$70[\%]$	b_5	0.01856
m_{bat}	$400[kg]$	b_6	-0.00593
R_{int}	$0.01[Ohm]$	$V_{nominal}$	$3.8[V]$

Table 5.1: Thermal System Parameters

$A_{bat,cool}$ is the heat transfer area between the battery and the cooling liquid. Further, the battery cooling liquid temperature is calculated based on the heat introduced by the cooling fluid flow and battery heat transfer as follows:

$$m_{bat,cool}c_{bat,cool}dT_{bat,cool}/dt = \dot{m}_{bat,cool}c_{bat,cool}(T_{bat,cool,in} - T_{bat,cool,out}) + h_{bat,cool}A_{bat,cool}(T_{bat} - T_{bat,cool}) \tag{5.3}$$

where $\dot{m}_{bat,cool}c_{bat,cool}$ is the cooling fluid mass flow rate, $c_{bat,cool}$ is the cooling fluid's specific heat capacity, $T_{bat,cool,in}$ and $T_{bat,cool,out}$ are the cooling liquid temperature upon entering and exiting the battery pack. For the battery pack cooling loop, the pump power consumption is computed as follows:

$$P_{cool,pump} = \frac{\dot{m}_{chill,cool}\Delta p_{cool,pump}}{\rho_{cool,pump,in}\eta_{is,cool,pump}} \tag{5.4}$$

where $\Delta p_{cool,pump}$ is the pressure rise across the cooling fluid pump, $\rho_{cool,pump,in}$ is the density of cooling fluid at the pump inlet and $\eta_{is,cool,pump}$ is the cooling fluid pump efficiency. In the battery cooling fluid chiller, the temperature of the cooling liquid, the wall separating the cooling liquid and refrigerant, and refrigerant are calculated as follows, respectively:

$$m_{chill,cool}c_{chill,cool}dT_{chill,cool}/dt = h_{chill,cool,w}A_{chill,cool,w}(T_{chill,w} - T_{chill,cool}) + \dot{m}_{chill,cool}c_{chill,cool}(T_{chill,cool,in} - T_{chill,cool,out}) \tag{5.5}$$

$$m_{chill,w}c_{chill,w}dT_{chill,w}/dt = -h_{chill,cool,w}A_{chill,cool,w}(T_{chill,w} - T_{chill,cool}) + \eta_{chill}h_{chill,rfg,w}A_{chill,rfg,w}(T_{chill,w} - T_{chill,rfg}) \tag{5.6}$$

$$m_{chill,rfg} dh_{chill,rfg}/dt =$$

$$h_{chill,rfg,w} A_{chill,rfg,w}(T_{chill,w} - T_{chill,rfg}) + \dot{m}_{chill,rfg}(h_{chill,rfg,in} - h_{chill,rfg,out}) \quad (5.7)$$

In the radiator of the battery cooling loop, the temperature of cooling fluid, the wall separating the cooling fluid and ambient air, and ambient air are calculated as follows, respectively:

$$m_{rad,cool} c_{rad,cool} dT_{rad,cool}/dt =$$

$$h_{rad,cool,w} A_{rad,cool,w}(T_{rad,w} - T_{rad,cool}) + \dot{m}_{rad,cool} c_{rad,cool}(T_{rad,cool,in} - T_{rad,cool,out}) \quad (5.8)$$

$$m_{rad,w} c_{rad,w} dT_{rad,w}/dt =$$

$$- \eta_{rad} h_{rad.cool,w} A_{rad,cool,w}(T_{rad,w} - T_{rad,cool}) + h_{rad,air,w} A_{rad,air,w}(T_{rad,w} - T_{rad,air}) \quad (5.9)$$

$$m_{rad,air} c_{rad,air} dT_{rad,air}/dt =$$

$$h_{rad,air,w} A_{rad,air,w}(T_{rad,w} - T_{rad,air}) + \dot{m}_{rad,air} c_{rad,air}(T_{rad,air,in} - T_{rad,air,out}) \quad (5.10)$$

In the AC refrigerant loop, the refrigerant mass flow rate is calculated based on compressor volumetric efficiency $\eta_{comp,vol}$, speed N_{comp}, displacement V_{comp} and refrigerant density ρ_{rfg} at the compressor inlet as follows:

$$\dot{m}_{rfg} = \frac{60 \eta_{comp,vol} N_{comp} V_{comp}}{\rho_{rfg} 10^3} \quad (5.11)$$

where the volumetric efficiency is determined through the following empirical equation 5.5:

$$\eta_{comp,vol} = b_1 + b_2 \frac{p_{comp,out}}{p_{comp,in}} + (b_1 + b_2 \frac{p_{comp,out}}{p_{comp,in}}) N_{comp} + (b_1 + b_2 \frac{p_{comp,out}}{p_{comp,in}}) N_{comp}^2 \quad (5.12)$$

where the coefficients $b_1, b_2, ...$ are given in Table 5.1. The power consumption by the

compressor is calculated as follows:

$$P_{comp} = \frac{\dot{m}_{rfg}(p_{comp,out} - p_{comp,in})}{\rho_{rfg}\eta_{comp,is}} \tag{5.13}$$

where

$$\eta_{comp,is} = a_1 + a_2\frac{p_{comp,out}}{p_{comp,in}} + (a_1 + a_2\frac{p_{comp,out}}{p_{comp,in}})N_{comp} + (a_1 + a_2\frac{p_{comp,out}}{p_{comp,in}})N_{comp}^2 \tag{5.14}$$

For more details on thermal system modeling, please consult Xu and Arjmandzadeh (2023).

5.2 Electrical System Model

The charging current is characterized by the following second-order equivalent circuit model Abbasi and Zhang (2021), Lin et al. (2014):

$$\frac{dSOC}{dt} = \frac{I}{C_b} \tag{5.15}$$

$$C_1\frac{dV_1}{dt} = -\frac{V_1}{R_1} + I \tag{5.16}$$

$$C_2\frac{dV_2}{dt} = -\frac{V_2}{R_2} + I \tag{5.17}$$

$$V_t = OCV + V_1 + V_2 + R_{int}I \tag{5.18}$$

where V_1 and V_2 indicate first and second polarization voltages, respectively. Similarly, R_1 and R_2 denote first and second polarization resistances, respectively, and C_1 and C_2 are first and second polarization capacitances, respectively. Moreover, C_b shows the battery capacity, V_t is the battery terminal voltage, and SOC represents the battery state of charge (SOC). The procedure to determine the charging current based on the $CC - CV$ algorithm is as follows Tomaszewska et al. (2019). If V_t is smaller than the upper cut-off voltage, the algorithm is in the constant current stage, and I is fixed. Then, V_t is calculated through 5.15 - 5.18. Note that $I > 0$ means the battery is being charged. Fixing I to a positive value results in the increase of V_t. Once V_t reached the upper cut-off voltage, it is fixed at that value, i.e., the charging process

enters the constant voltage stage, and I is calculated using 5.15 - 5.18. When V_t is constant, I converges to zero in both charging and discharging.

5.3 Optimization Approaches

In this section, the details of control algorithms are elaborated, namely, RL, MPC, and PID. MPC and PID serve as baselines with which the proposed RL algorithm is compared.

5.3.1 Markov Decision Process (MDP)

An MDP is a mathematical framework for modeling consecutive decision-making problems in which the results are partly random and partly influenced by control actions. MDPs are efficient tools for solving optimization problems. The optimal solution will be obtained if an MDP is solved through dynamic programming. The MDP sequence flow is as follows. At first, the system is at state s. The agent, or the decision maker, takes action a, and this action causes the system to move to state s' and generate a reward $r(s, a, s^{'})$. The probability that action a would take the system from state s to s' and generate reward $r(s, a, s^{'})$ is $p(s^{'}, r|s, a)$. This probability is called transition probability as the system transitioned from state s to s'. Note that the transition probabilities should be known in order to solve an MDP via dynamic programming Fang et al. (2020). Gaining knowledge over transition probabilities is challenging as these probabilities are not available in many practical applications. Moreover, estimating the transition probabilities through approaches such as Monte Carlo will introduce inaccuracy and suboptimality to the results. Furthermore, dynamic programming is only feasible for the case of discrete action and state spaces. However, the action and state spaces are continuous in many engineering problems, such as the one we tackle in this

chapter. Consequently, RL is proposed to relax the need for transition probabilities and enable MDPs to be utilized for continuous action and state spaces.

5.3.2 Modeling Thermal System with MDP

The thermal management of an electric vehicle can be described as an agent who accomplishes its duty through sequential decisions on cooling fluid mass flow rate. In this regard, the TMS optimization is constructed in the form of an MDP, as clarified in the following.

5.3.2.1 Environment

The environment is the battery pack comprising N_{cell} number of cells along with the chiller, refrigerator, radiator, coolant pump, and compressor pump. The interaction of the RL agent with the environment comprises taking actions and observing the rewards and the subsequent states. Therefore, Eq. 5.1 - 5.14 are used to characterize the environment. The charging current generates joule heat, increasing the battery pack's temperature. In contrast, battery cooling fluid absorbs heat from the battery and reduces the battery temperature. The environment consists of the dynamics of battery pack temperature, battery cooling liquid temperature, the temperature of cooling liquid in the chiller, the temperature of the wall separating the cooling liquid and refrigerant, refrigerant enthalpy, the temperature of cooling fluid in the radiator, and the wall separating the cooling fluid. The RL agent takes continuous actions in terms of cooling fluid mass flow rate. The actions are within defined boundaries. Finally, the system or environment receives the actions, changes its states accordingly, and generates feedback in the form of a reward signal.

5.3.2.2 Observation Space

The observation or state space represents the variables that reflect the system's status. These variables must be measurable to be used in RL. Thus, this chapter considers battery pack temperature, $T_{bat}(t)$, and battery cooling liquid temperature, $T_{bat,cool}(t)$. It is worth mentioning that we considered more than two states, up to all seven states, that comprise the environment, but the results did not improve or became worse in different cases.

5.3.2.3 Action Space

The RL agent is trained to take action in such a way that the cumulative reward is maximized. In this chapter, the action is the coolant fluid mass flow rate, $\dot{m}_{bat,cool}(t)$. The action space is continuous, where $\dot{m}_{bat,cool}(t)$ can have any value between 0 and $0.1\frac{kg}{s}$, i.e., $\dot{m}_{bat,cool}(t) \in (0, 0.1]\frac{kg}{s}$. The upper bound of the mass flow rate is selected large enough so that the RL agent can explore different actions and properly cool the battery pack.

5.3.2.4 Reward

Reward signal indicates the desirability of transitioning from state s to $s^{'}$ by taking action a. In RL, the reward plays the role of the objective function of an optimization problem. In the present chapter, our goal is to control the battery temperature while minimizing the degradation and power consumption of TMS. Therefore, we define the reward as follows:

$$R(t) = -\xi_1 P_{cool,pump} - \xi_2 P_{comp} - \xi_3 C_{loss}(Ah) - \xi_4 R_{inc}(Ah) \tag{5.19}$$

where $\xi_i, i \in 1, 2, 3, 4$ is weighting factor, $C_{loss}(Ah)$ and $R_{inc}(Ah)$ are capacity loss and resistance rise of the battery pack, respectively. Capacity loss is achieved by Cordoba-

Parameter	Value	Parameter	Value
α_C	137	α_R	3.2053e+05
β_C	420	β_R	1.3674e+09
γ_C	9610	γ_R	3.6342e+03
b	0.34	C_R	5.45
c	3	d	0.9179
z	0.48	e	1.8277
SOC_0	0.25	SOC_0	0.25
E_{aC}	22406$[J/mol]$	CR_0	5
R_g	8.314$[J/K/mol]$	E_{aR}	51800$[J/mol]$
		R_u	8.314$[J/K/mol]$

Table 5.2: Parameters of Capacity Loss and Resistance Rise Equations Cordoba-Arenas et al. (2015)

Arenas et al. (2015):

$$C_{loss}(Ah) = a_C(SOC_{min}, Ratio).exp(\frac{-E_{a_C}}{R_g T})Ah^z \tag{5.20}$$

Moreover, a similar relation is used for resistance rise as follows Cordoba-Arenas et al. (2015):

$$R_{inc}(Ah) = a_R(SOC_{min}, Ratio, CR).exp(\frac{-E_{a_R}}{R_u})Ah \tag{5.21}$$

where $a_R(.)$ is defined by the following relation $a_R(.) = \alpha_R + \beta_R(SOC_{min} - SOC_0)^{C_R} + \gamma_R exp[d.(CR_0 - CR_{eq}) + e.(SOC_{min} - SOC_0)]$, and CR_{eq} is a function as:

$$CR_{eq} = \begin{cases} 0, & Ratio = 0 \\ CR, & Ratio > 0 \end{cases} \tag{5.22}$$

Lastly, the constants in 5.21 and α_R are presented in Table 5.2. Based on 5.19, the

reward function is non-positive, $R(t) \leq 0$, while the cumulative reward is negative.In batteries, the number of cycles is often computed based on total ampere-hour throughput $[Ah]$. This means the amount of saved energy in the battery during the useful age of the battery. The corresponding formula is given as follows:

$$Ah = \int_0^{\tau} \mid I(\tau) \mid \, d\tau \tag{5.23}$$

where $I(t)$ is the input current to the battery Zhao et al. (2023).

5.3.3 Proximal Policy Optimization

The deep RL solves the MDP problem, and the optimal solution is obtained by finding the optimal policy π^*. In MDP, policy $\pi : s \rightarrow a$ chooses action a when in state s. The optimal policy is achieved by training the underlying neural network. In this chapter, we selected the PPO algorithm as our deep RL methodology due to robust performance across a wide range of tasks Wang et al. (2023). One of PPO's key advantages is that it performs well even without intensive hyperparameter tuning, making it more accessible for users who prefer a streamlined setup. PPO includes entropy regularization, which encourages exploration during training and prevents the agent from becoming overly confident in specific actions too early. This mechanism helps PPO achieve a balanced trade-off between exploring new strategies and exploiting successful ones, often outperforming other RL algorithms. A distinguishing feature of PPO is its clipping mechanism, which restricts the policy's change at each update step. This limitation prevents excessive updates that could destabilize the training process, resulting in greater stability compared to other policy gradient methods like the vanilla policy gradient or Trust Region Policy Optimization (TRPO). Additionally, PPO's implementation is simpler than TRPO's, which requires solving complex constrained

optimization problems. Unlike Q-learning and DQN, PPO excels in handling continuous action spaces, whereas DQN often struggles in these environments. Compared to asynchronous methods like A3C and A2C, PPO provides more efficient and stable training, making it a preferred choice for many RL applications.

PPO maximizes the following cost function Schulman et al. (2017):

$$\theta_{k+1} argmax_{\theta} \hat{E}_t[L(\theta)] \tag{5.24}$$

$$L(\theta) = min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \tag{5.25}$$

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{old}}(a_t \mid s_t)} \tag{5.26}$$

where θ is the policy parameters vector, $\hat{E}_t[.]$ denotes expectation, $r_t(\theta)$ indicates the probability ratio between the updated policy outputs and the old policy outputs, $\hat{A}_t$ shows advantage function estimation, $clip(.)$ truncates $r_t(\theta)$ using ϵ parameter (for more detail, see Schulman et al. (2017)).

5.3.4 Model Predictive Control

In this subsection, the optimization problem is modeled to be solved with MPC, an optimal control method that optimizes a cost function under the constraints of the dynamics of the system. In fact, MPC solves an optimization problem to find the best control action that drives the predicted output to the reference. This optimization problem is solved over an interval named prediction horizon, but only the first step is executed. Next, the estimation and computation processes are repeated with a shifted horizon.

The main differences between MPC and other controllers are as follows: a) MPC can handle multi-input multi-output (MIMO) systems, b) MPC can handle constrained problems, which allows us to consider some bounds for actions and states, c) Since the

optimization problem is solved on a subinterval, MPC may reach a local optimal point, and d) It has a complex performance. All seven system states are considered in this MPC. In the following, the optimization problem defined for MPC is presented:

$$min \sum_{\tau=t}^{t+p} \{w_1 P^2_{cool,pump}(\tau) + w_2 P^2_{comp}(\tau) + w_3 C^2_{loss}(Ah) + w_4 R^2_{inc}(Ah)\} \tag{5.27}$$

$S.T.$

$(1) - (3), (5) - (13)$

$$253 \leq T_{bat}, T_{bat,cool}, T_{rad,cool}, T_{rad,w}, T_{chill,cool}, T_{chill,w} \leq 373 \tag{5.28}$$

$$10^5 \leq h_{chill,rfg} \leq 7 \times 10^5 \tag{5.29}$$

$$0 < \dot{m}_{bat,cool} \leq 0.1 \tag{5.30}$$

where p is the length of the prediction horizon, $w_i, i \in \{1, 2, 3, 4\}$ shows weighting factor, and constraints 5.28 - 5.30 set the states and input boundaries. The selection of weighting factors in 5.27 and 5.19 is by trial and error. The weighting factors in 5.27 are selected such that the results are reasonable. Then, the weighting factors in 5.19 are selected such that the battery temperature alteration is similar to that of 5.27 so that other features, such as power consumption and degradation, can be compared.

5.3.5 Proportional–Integral–Derivative (PID) Controller

PID is a popular controller that is mainly used in industrial control systems Chen et al. (2019). It is a type of control system that uses feedback to continuously adjust the output of a system to track a desired reference point. Due to the ability to provide stable and accurate control with relatively simple implementation, PID controllers are mainly used in various applications, like temperature control, flow control, and motor control. A PID controller consists of three principal components based on the difference between a set point (SP) and a measured process variable (PV).

The proportional component depends only on the difference between SP and PV. This difference is referred to as the Error term. Generally, increasing the proportional gain (Kp) will increase the speed of the control system response. However, an unreasonable amount of Kp might lead to instability as the oscillations will become larger and often out of control. The integral element (Ki) sums the error over time that increases gradually unless the error is zero, so the effect is to drive the Steady-State error to zero. Besides, the derivative term (Kd) adjusts the output based on the rate of change of the error. The main role of this component is to smooth oscillations. In fact, adjusting the overshoots quickens the global system's response.

Solving these problems finds flow rates in 5.3 and 5.12. The former is determined either by RL or MPC, while PID handles the latter. In order to implement RL, we use PPO. In the following section, extensive simulation results are presented.

5.4 Simulation and Results

This section first discusses the PPO hyperparameter selection and then analyzes the results of implementing three different controllers: RL, MPC, and PID. These methods are compared according to their efficiency and computation time. In addition, the setup for the three strategies is the same for fair comparison, including the CC-CV profile during extreme fast charging and the entire thermal system model. In all cases, a fresh battery is charged from 20% to 80% with identical charging C-rates under the ambient temperature of 25^oC. Furthermore, a battery pack comprising 66 series strings of 74 parallel cells is considered, and each cell's nominal capacity is 3.5 [Ah]. The simulations are executed on a machine with a 20-core 2.4 GHz CPU and 16 GB RAM.

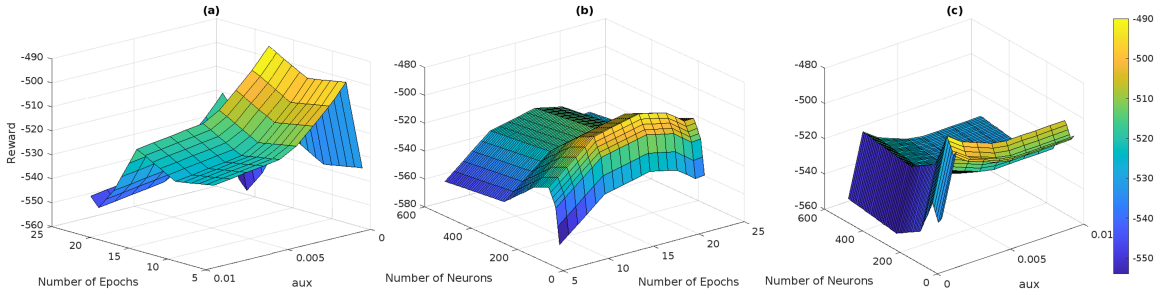


Figure 5.3: Sensitivity Analysis Results

5.4.1 PPO Hyperparameter Selection

Firstly, the hyperparameter selection of the proposed PPO approach is presented. Fig. 5.3 illustrates the selection of three hyperparameters, namely, the number of epochs, weights used in the reward function (aux), and the number of neurons in each layer of actor and critic neural networks. The chosen values for the number of epochs, the aux parameter, and the number of neurons are 15, $15e-4$, and 64, respectively. With the said selection, the reward peaks at -490 in all three subplots of Fig. 5.3. Furthermore, in each subplot, the change of reward with respect to two hyperparameters is illustrated. We cannot show reward change with respect to all three hyperparameters in one plot, as it requires a 4D plot. Fig. 5.3(a) portrays how the cumulative evaluation reward changes according to the number of epochs and aux parameter. Similarly, Fig. 5.3(b) and (c) depict reward alteration versus the number of neurons-number of epochs and the number of neurons-aux, respectively. The epoch values range from 5 to 20, the aux values vary between $5e-4$ and 0.01, and the number of neurons runs from 16 to 516. Note that aux in Fig. 5.3(a) and 2(c) is used in the weights in Eq. 5.19, $\xi_1 = \frac{w_1}{scale_1}, \xi_2 = \frac{w_2}{scale_2}, \xi_3 = \frac{w_3}{scale_3}$, and $\xi_4 = \frac{w_4}{scale_4}$, where $w_1 = aux, w_2 = aux, w_3 = 0.5 - aux$, and $w_4 = 1 - w_1 - w_2 - w_3$. The parameters $scale_i$, $i \in \{1, 2, 3, 4\}$ normalize the terms in the reward function to be between 0 and 1. In addition, $w_i, i \in 1, 2, 3, 4$ control the importance of each reward term. Each w_i is manipulated by an auxiliary parameter

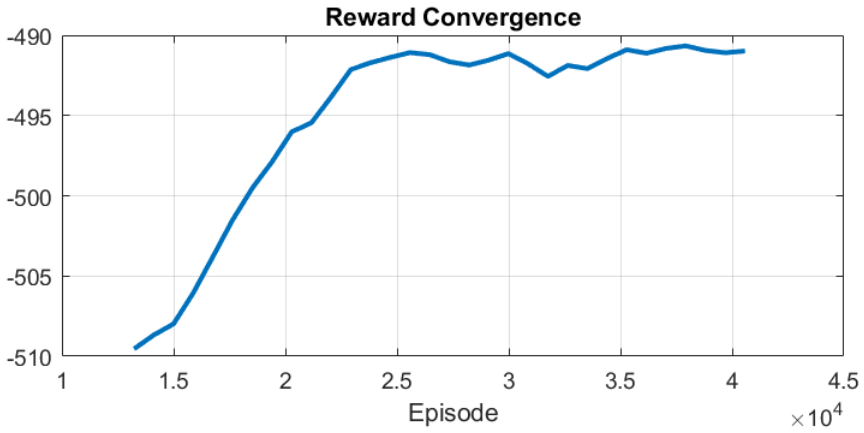


Figure 5.4: PPO reward convergence during train.

aux. For example, by setting $aux = 0.25$, we have $w_1 = w_2 = w_3 = w_4 = 0.25$.

5.4.2 Training The PPO Agent

BTMS aims to minimize capacity loss, resistance rise, compressor power consumption, and pump power consumption. PPO training required 5.73 minutes to complete, while evaluating the PPO took 1.74 [s]. Also, thermal management is performed when the battery is fast-charged utilizing the $CC - CV$ charging profile. In this regard, the battery is charged from 20% to 80% of SOC in 881 seconds, as shown in Fig. 5.2, where the CC stage lasts about 31 seconds. Moreover, the PPO agent is trained using the following weights in the reward function 5.19, $\xi_1 = 3.75 \times 10^{-4}$, $\xi_2 = 3 \times 10^{-7}$, $\xi_3 = 4.985$, and $\xi_4 = 498.5$. Fig. 5.4 shows the convergence and increase of the rewards throughout the training process in over 40k time steps. The rewards start at less than -510 and converge to around -490. It merits mentioning that the plotted reward curve is smoothed out by taking a moving average with a window of fifteen time steps from the original reward vector.

5.4.3 Results

The BTMS problem is also solved with MPC. Notably, MPC's computing time exceeds 85 [s], which is considerably higher than that of RL (the computation time is roughly 4,785% higher in the MPC case). In both MPC and PPO cases, the battery pack is fast charged from 20% to 80% in 881 [s] owing to the similar CC-CV charging profile used in both cases. Moreover, the MPC results are obtained using the following weights, $w_1 = 7.725 \times 10^{-5}, w_2 = 6.18 \times 10^{-7}, w_3 = 2.48455$, and $w_4 = 49.691$. Although w_1 and w_2 are much smaller than w_3 andw_4, setting them to zero, i.e., $w_1 = w_2 = 0$, causes $\dot{m}_{bat,cool}$ and $\dot{m}_{chill,cool}$ to immediately saturate at their upper bound of $0.1[\frac{kg}{s}]$. The reason lies in the fact that eliminating power consumption terms in cost function (27) leads to maximum effort for minimizing degradation, that is, maximum mass flow rate.

Next, we control the BTMS understudy with two PID controllers, one for $\dot{m}_{bat,cool}$ and one for $\dot{m}_{chill,cool}$. In this case, the problem is not an optimization problem but a tracking problem. The two PID controllers track two set points for battery temperature and superheat. Since PID does not optimize the battery temperature with the goal of minimizing degradation and power consumption, we select two different setpoints of 35^oC and 36.5^oC and run the program once for each setpoint. Note that 36.5^oC is chosen for comparison, as MPC and PPO approximately converge to this value. Fig. 5.5 represents the battery pack temperature rise when charged with the proposed approach as opposed to the baseline approaches. In the PPO case, the battery temperature converges to around 36.5^oC, topping at the second of 622, while the said temperature was 35^oC at the start of the charging process. However, there is a slight drop after the second of 623, which is attributed to the decreasing charging current due to the CV stage. The PPO curve closely follows the MPC one, while the two PID curves have a different trend. In addition, in the third case, a PID controller

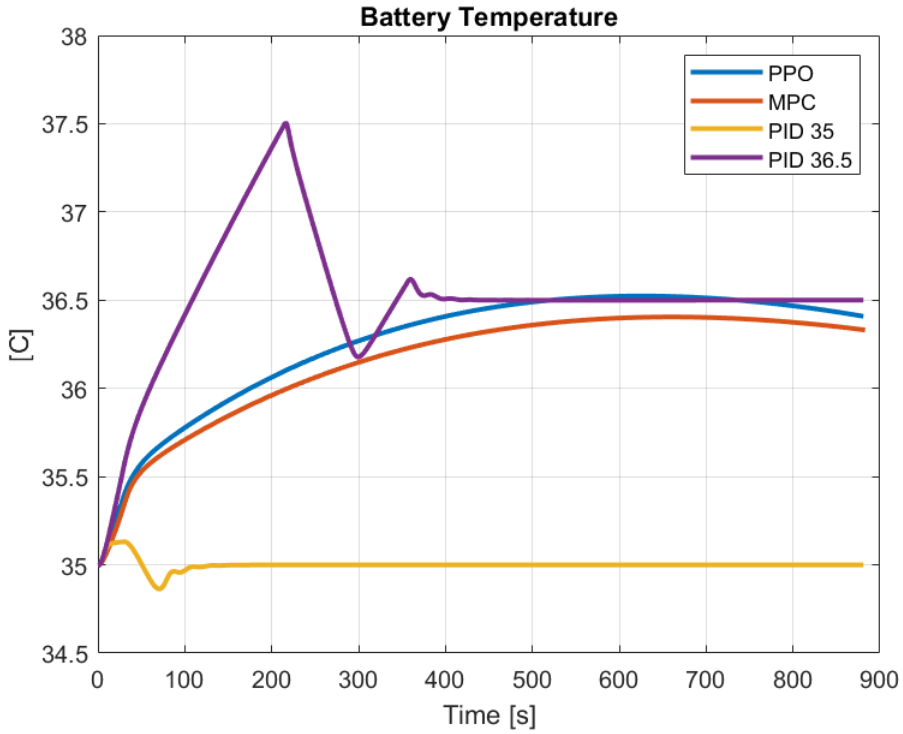


Figure 5.5: Battery Temperature Elevation.

handles the charging process with the goal of keeping the temperature at 35^oC. Since the battery pack is at 35^oC before the charging session, the PID controller should try to maintain the temperature as it is. However, after the start of fast charging, the battery temperature slightly increases to just above 35.1^oC followed by an undershoot, which leads to a steady state. Finally, the PID results are obtained when the battery temperature setpoint is 36.5℃ following the results of PPO and MPC. As seen in Fig. 5, the battery temperature fluctuates before settling at 36.5^oC for nearly 400 seconds and reaches the maximum value of 37.5^oC at the second 216.

Fig. 5.6 displays "Coolant Mass Flow Rate" and "Rfg. Mass Flow Rate" and their corresponding pump and compressor power consumption, respectively. The mass flow rates are the two control inputs of our optimization problem, i.e., cooling fluid mass flow rate, $\dot{m}_{bat,cool}$, and refrigerant mass flow rate, $\dot{m}_{chill,cool}$. PPO controls the former

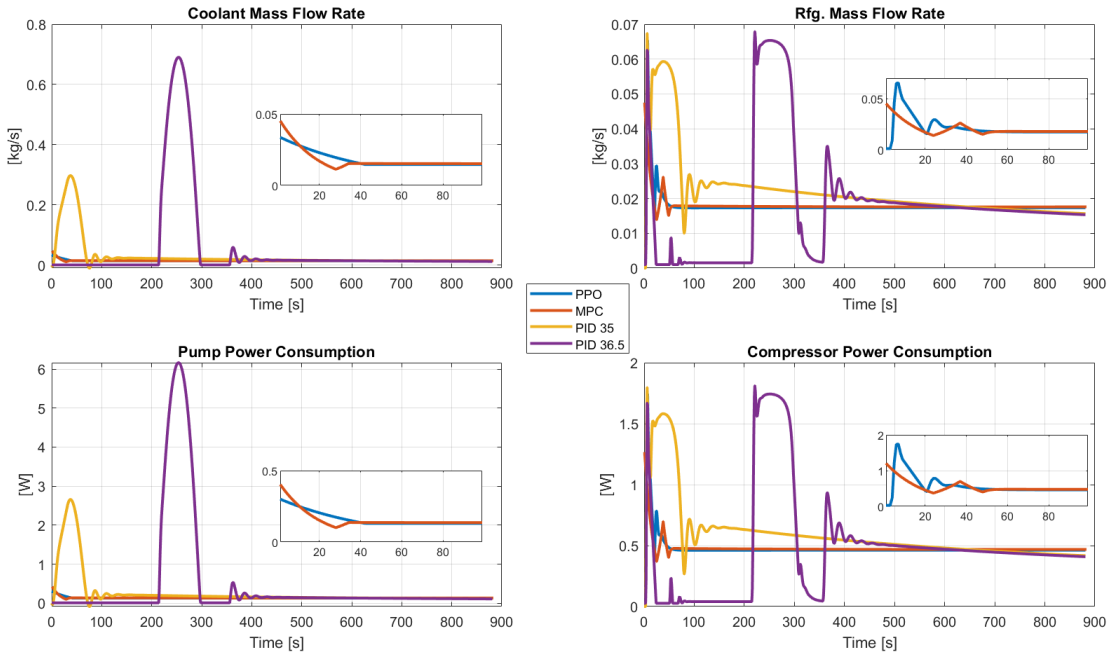


Figure 5.6: Coolant and refrigerant mass flow rates, as well as pump and compressor power consumption in all cases.

while the PID controller adjusts the latter. Note that the RL input influences the behavior of PID; that is, the shape and final converged value of the refrigerant mass flow rate change if those of the cooling fluid mass flow rate change. $\dot{m}_{bat,cool}$ starts with 0.0343 $[\frac{kg}{s}]$ and slowly drops to and settles at 0.0146 $[\frac{kg}{s}]$ after 43 seconds. The reason for the initial decreasing behavior of $\dot{m}_{bat,cool}$ is ascribed to the conflicting impact of the terms inside the reward function as well as the sluggish nature of heat transfer. Since the objective is to reduce battery degradation, $\dot{m}_{bat,cool}$ takes a high value at the beginning because of the high initial constant charging current. But $\dot{m}_{bat,cool}$ falls to mitigate compressor power consumption, allowing the pack's temperature to rise. However, the decline of $\dot{m}_{bat,cool}$ is not sharp due to the slow dynamics of the heat transfer. On the other hand, the initial value of $\dot{m}_{bat,cool}$ is 0.001 $[\frac{kg}{s}]$, which is the set lower-bound of mass flow rate. $\dot{m}_{bat,cool}$ curve fluctuations last about a minute before

settling at 0.017 $[\frac{kg}{s}]$, with a maximum of 0.065 $[\frac{kg}{s}]$ at the second 7. Furthermore, the compressor and pump power consumption linearly follow the functions of $\dot{m}_{chill,cool}$ and $\dot{m}_{bat,cool}$, respectively. These two plots are scaled for visualization purposes. The compressor power converges to 2.3 [kW], while the pump power settles at 5.2 [W].

Similarly, when MPC is employed, $\dot{m}_{bat,cool}$ is controlled by MPC and $\dot{m}_{chill,cool}$ is handled by PID. Note that the PID controller should be re-tuned when switching from RL to MPC. The overshoot of $\dot{m}_{chill,cool}$ occurs sooner at the second one in this case compared to PPO, leading to a lower overshoot of 23.27℃ in superheat temperature. $\dot{m}_{bat,cool}$ and $\dot{m}_{chill,cool}$ settle at 0.0152 and 0.0176 $[\frac{kg}{s}]$, marginally higher than the control inputs converged values in the RL case. Consequently, the battery temperature in the MPC case in Fig. 6 reaches a maximum of 36.40^oC as opposed to 36.52^oC in the PPO case. Further, compressor and pump power subplots indicate quick convergence after some oscillations. The compressor power converges to around 2.35 [kW], while pump power settles at 5.2 [W]. Compared to PPO, compressor power has a smaller peak but a higher converged value. In contrast, pump power sees greater overshoot and settles at a similar steady-state value.

In the first PID case, where the pack's temperature is kept at 35^oC, which is the initial temperature, the PID controller should try to maintain the temperature as it is. However, after the start of fast charging, the battery temperature slightly increases to just above 35.1^oC, and the corresponding mass flow rate and pump power rise to 0.3 $[\frac{kg}{s}]$ and 106.26 [W], respectively. Subsequently, the increase in $\dot{m}_{bat,cool}$ reduces the battery temperature to less than 34.9^oC, which causes the mass flow rate to drop to its minimum of 10^{-3} $[\frac{kg}{s}]$. Afterward, the battery temperature bounces back to and remains at 35^oC. This process takes 2 minutes from the start of charging to reaching the steady-state condition. Unlike MPC and PPO, where mass flow rates converged to a constant value, in the case of PID, the mass flow rates slowly drop after reaching the

steady state. This is because the PID controller tries to keep the battery temperature at 35^oC. Hence, as the charging current declines in the CV stage, the mass flow rates drop too. $\dot{m}_{bat,cool}$ reaches 0.023 $[\frac{kg}{s}]$ after two minutes and falls to ~ 0.013 by the end of the charging session. Moreover, the corresponding pump power descends from 7.6 [W] to 4.4 [W]. Similarly, $\dot{m}_{chill,cool}$ is 0.025 $[\frac{kg}{s}]$ at the second 140 and diminishes to 0.015 $[\frac{kg}{s}]$ at the end. In addition, compressor power reduces from 3.25 [kW] to 2.05 [kW].

In the second PID case, the temperature setpoint is 36.5^oC. The PID controller exhibits a large rise in $\dot{m}_{bat,cool}$ such that it goes above 0.6 $[\frac{kg}{s}]$, which violates the limit of 0.1 $[\frac{kg}{s}]$ we set for mass flow rates to a great extent. In the same sense, pump power consumption significantly increases for around 100 seconds, shooting up the average pump power consumption to 18.8 [W] compared to 5.2 [W] in the PPO case. The large jump of $\dot{m}_{bat,cool}$ between 214 [s] and 298 [s] occurs to reduce the battery temperature. At about the same period, $\dot{m}_{chill,cool}$ soars more than 0.06 $[\frac{kg}{s}]$ and remains there for about a minute, increasing the average compressor power consumption to 2.39 [kW]. However, this value is not much different than the PPO case, where the average compressor power is 2.37 [kW]. Fig. 5.7 represents the battery's internal resistance and capacity change, respectively. The internal resistance rise is infinitesimal over one charging cycle. On the other hand, the capacity fade seems considerable for one cycle. However, this is owing to the exponential nature of the capacity loss formula in 5.20. That is, as the battery ampere-hour throughput increases, the rate of capacity drop gradually decreases. Capacity drops and internal resistance rises in MPC and RL have imperceptible differences, with the former experiencing less aging, seeing that the battery temperature under MPC reaches the maximum of 36.40℃ as opposed to 36.52^oC in the PPO case. Likewise, for the PID case with 36.5℃ setpoint, the aging is similar to MPC and PPO cases. In contrast, in the PID with 35^oC setpoint, it is

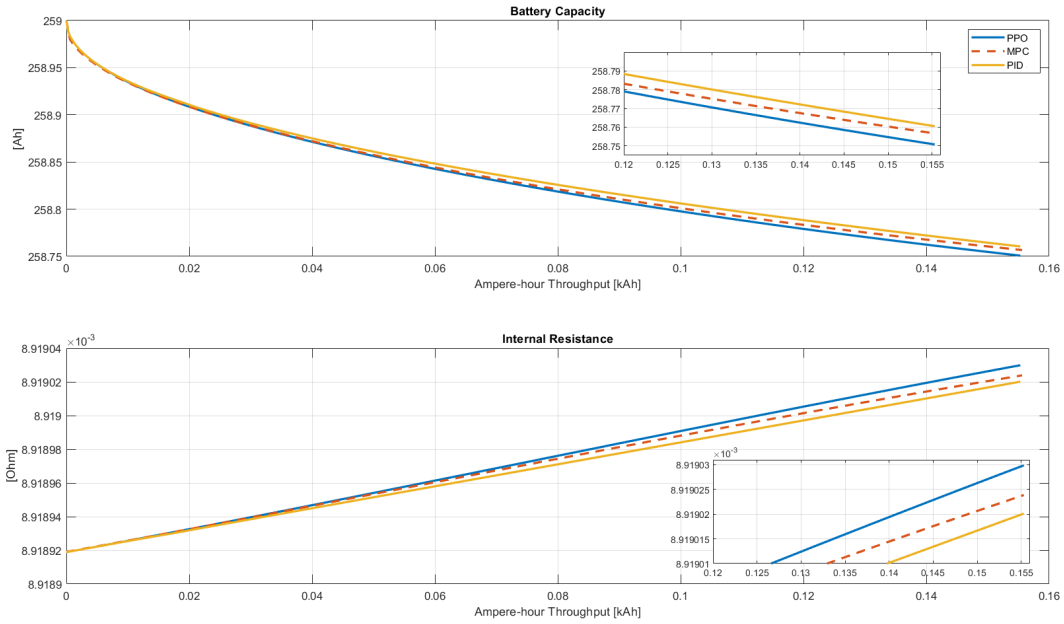


Figure 5.7: Fresh battery aging after one fast charging session in different cases

Metric	**PPO**	**MPC**	**PID1** (36.5^oC)	**PID2** (35^oC)
Model-free	Yes	No	Yes	Yes
Optimal	Yes	Yes	No (tracking)	No (tracking)
Computation time	1.74 [s]	85 [s]	2.5 [s]	2.39 [s]
Avg. comp. power cons.	2.37 [kW]	2.39 [KW]	2.39 [kW]	2.96 [kW]
Capacity loss	0.09266 [%]	0.09393 [%]	0.09266 [%]	0.08880 [%]
Int. resistance rise	0.001143 [%]	0.001172 [%]	0.001155 [%]	0.001042 [%]

Table 5.3: The results of PPO, MPC, and PIDs with battery temperature setpoints of 36.5℃ and 35℃.

discernable that the battery degradation is less severe compared to the other three cases at the cost of higher power consumption. The reason is that the battery temperature is kept at the lower value of 35^oC.

Table 5.3 and Table 5.4 provide a comparison between PPO and three other baseline methods. In Table 5.3, computation time, average compressor power consumption, battery capacity loss after the very first fast charging session, and internal resistance

Metric	**PPO**	**MPC**	**PID1** (36.5^oC)	**PID2** (35^oC)
Model-free	Yes	No	Yes	Yes
Optimal	Yes	Yes	No (tracking)	No (tracking)
Computation time	1.74 [s]	+4785 %	+43.68 %	+37.36 %
Avg. comp. power cons.	2.37 [kW]	+0.84 %	+0.84 %	+24.9 %
Capacity loss	0.09266 %	+1.37 %	0.0 %	-4.17 %
Int. resistance rise	0.001143 %	+2.54 %	+1.05 %	-8.84 %

Table 5.4: Percentage difference between the results of PPO and other methods

rise after the first charge are reported for PPO, MPC, and two PIDs that keep battery temperature at 36.5^oC and 35^oC, respectively. Table 5.4 presents the same metrics, where the values of MPC and PIDs are substituted with the percentage of how different they are from PPO in terms of a metric. For instance, in Table 3, the PPO computation time is 1.74 [s], while that of MPC is 85 [s]. Hence, Table 5.4 states that MPC computation time is $\frac{(85-1.74)}{1.74} \times 100 \approx 4785[\%]$ higher than that of PPO. Table 4 shows metrics of baseline approaches that are higher than PPO's in red color with a plus sign. On the contrary, metrics of baseline approaches that are smaller than PPO's are shown in green color with a minus sign. Overall, PPO outperforms MPC because PPO is much faster than MPC and also because MPC requires the system's model. Note that, in terms of battery aging and BTMS power consumption, the differences between PPO and MPC may seem minor. This is due to the accurate model used by MPC in the simulations. However, such an accurate model may not be available for MPC in real-world applications, which can deteriorate its performance.

Furthermore, in Table 5.4, PID2 outperforms PID1 in terms of both average pump power consumption and battery degradation. Having smaller battery degradation in the case of PID2 makes sense as it keeps the battery at a lower temperature. PID2's battery aging results are also better than those of PPO. However, why PID2 uses less

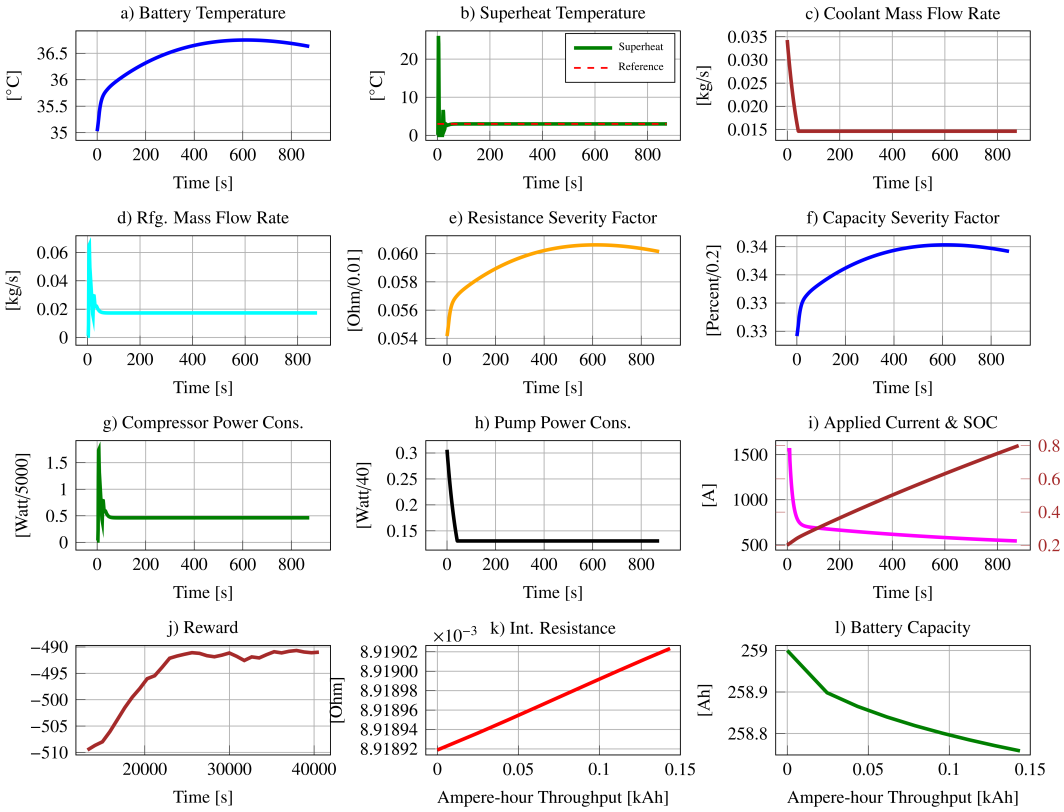


Figure 5.8: PPO results with 6C charging current

pump power consumption than PID1? This is attributed to the differences in tuning of PID1 and PID2. PID controllers should be manually tuned or adjusted via another technique, such as neural networks. Nonetheless, the BTMS power consumption in PID2 cases is significantly higher than in the PPO case.

5.4.4 Validation

In this subsection, the trained RL algorithm is validated by applying a 6C CC-CV charging current. Note that the training is performed utilizing 4C CC-CV. As shown in Fig. 5.8, the overall trend of subplots is close to previous PPO results with minor differences. For instance, the battery temperature is elevated to above 36.5℃, while in Fig. 5.5, the maximum temperature is limited to 36.5℃. In addition, Fig. 5.8(i)

represents the 6C current applied to the battery. Due to the high current value, the battery reaches its upper cut-off voltage shortly after the beginning of the charging process, ending the CC stage. Subsequently, the CV stage begins, and the charging current reduces accordingly. Finally, Fig. 5.8(k) illustrates the increase in the internal resistance, which is infinitesimally higher than the PPO's final internal resistance value in Fig. 5.7.

5.4.5 Model Error

The proposed PPO-based approach operates as a model-free technique, whereas MPC relies on a model-based framework. So far, the results have been derived under the assumption that MPC utilizes a perfectly accurate model; however, this assumption rarely holds in real-world scenarios. This subsection, therefore, explores the impact of model inaccuracies on the results obtained from MPC.

Figure 5.9 illustrates the impact of model error on MPC's performance, comparing it to the performance of PPO and baseline MPC. Three cases of model error are analyzed: in the first case, a 10% error is introduced in the battery cell's specific heat capacity (esscool), while the second and third cases introduce 1% and 5% errors in the chiller's specific heat capacity (chillccool), respectively. Among all cases, the 5% error in the chiller's specific heat capacity yields the poorest performance, followed by the 1% error in chiller capacity and the 10% error in battery cell capacity. Overall, Figure 9 highlights the robustness of the model-free PPO algorithm, underscoring its advantage in handling model inaccuracies effectively.

Moreover, Fig. 5.10 depicts the battery capacity drop, and internal resistance rise after 1000 fast charging cycles in three cases of the proposed PPO, MPC with a 10% model error in battery cell specific heat capacity and MPC with a 1% model error in the chiller's specific heat capacity. As seen, the MPC cases yield higher aging compared

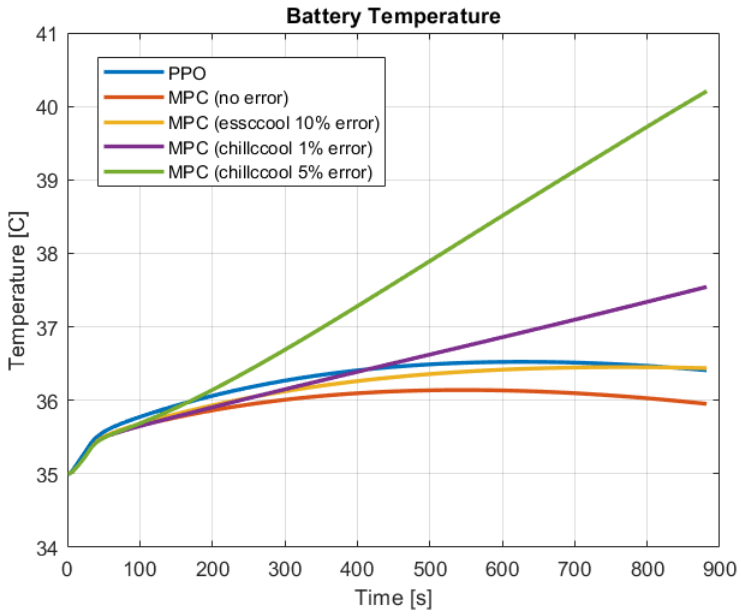


Figure 5.9: The impact of model error on MPC results

with the proposed model-free deep RL approach.

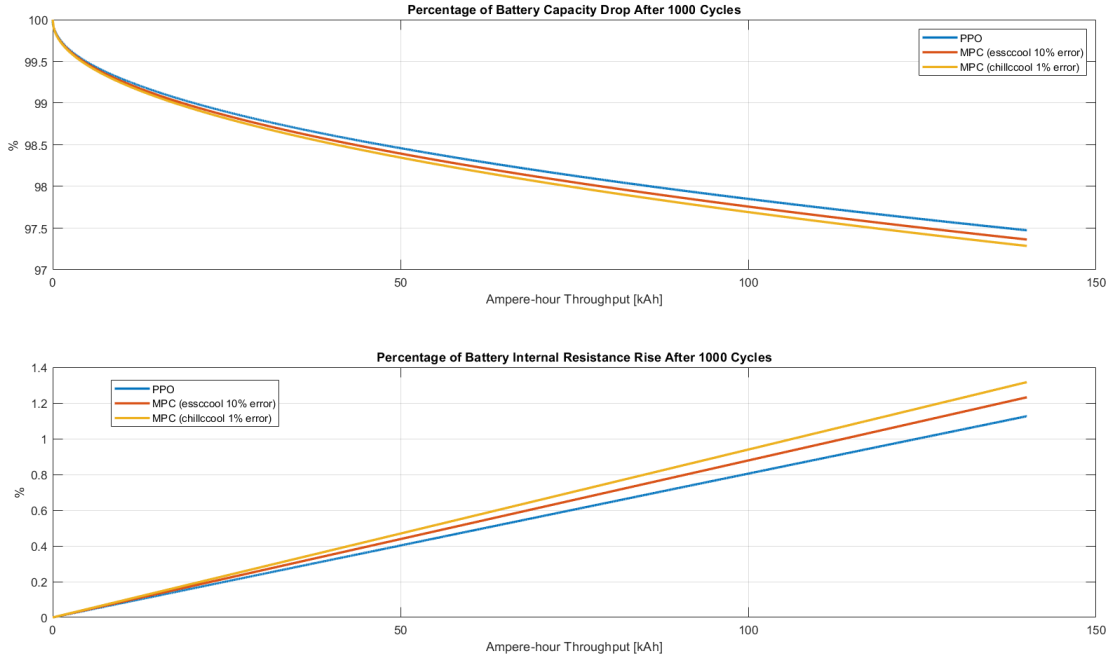


Figure 5.10: Battery aging after 1000 cycles in PPO, MPC with 10% and 1% model errors

Chapter 6

Conclusion

This dissertation addressed two major challenges in autonomous systems and energy management: local path planning for autonomous vehicles (AVs) and battery thermal management under extreme fast charging (XFC). Both problems were formulated as constrained optimization tasks and approached through analytical, simulation-based, and data-driven methodologies.

In the first part, the problem of local path planning was studied through several control paradigms, including Model Predictive Control (MPC), a sampling-based method, PID control, and a novel Lyapunov Optimization (LO) framework. A real-world experimental setup using a Nissan Leaf testbed equipped with lidar, radar, GPS, IMU, and cameras was established. Image data were processed using YOLOv5 and YOLOPv2 for perception and bird's-eye-view scene construction, enabling data-driven validation of the control models.

Comparative analyses demonstrated that while both the MPC and sampling-based methods could generate feasible and collision-free trajectories, MPC outperformed in terms of smoothness, comfort, and computational stability. The LO-based approach further advanced this performance by achieving comparable tracking accuracy with significantly lower computation time, making it a strong candidate for real-time path planning in dynamic environments. The CARLA-based validation confirmed the robustness of the proposed LO framework, where its integration with ROS and enhancement through Extended Kalman Filtering (EKF) and Particle Swarm Optimization

(PSO) parameter tuning ensured accurate and stable control in a realistic driving context.

The second part of the dissertation explored Battery Thermal Management System (BTMS) optimization during extreme fast charging using a model-free deep reinforcement learning (DRL) approach. A detailed BTMS model incorporating the battery pack, chiller, refrigerator, and radiator was developed, and degradation effects were modeled via capacity and power fade mechanisms. The Proximal Policy Optimization (PPO) algorithm was employed to regulate cooling fluid and refrigerant flow rates, minimizing both power consumption and battery degradation.

Simulation results revealed that PPO not only outperformed MPC and PID controllers in energy efficiency but also achieved up to 4785% faster computation while maintaining effective temperature regulation — highlighting its potential for real-time embedded implementation. Moreover, PPO's model-free nature makes it highly adaptable to system uncertainties, unlike MPC, which relies on precise modeling.

While the proposed frameworks successfully addressed their respective optimization challenges, several future research directions remain open. For the AV domain, integrating global path planning with the local LO-based controller and exploring hybrid MPC–RL strategies could further enhance decision-making robustness. For BTMS optimization, future work can expand the model to include dynamic driving scenarios, variable battery heat capacity, and advanced liquid cooling configurations such as ribbon or cold-plate systems. Additionally, coupling the BTMS with cabin air-conditioning control could yield a more comprehensive energy management architecture for electric vehicles.

In summary, this research contributes to the fields of autonomous vehicle control and energy optimization by developing computationally efficient, stable, and adaptive control frameworks validated through both simulation and real-world experiments.

These approaches pave the way for safer, more efficient, and more sustainable intelligent transportation systems.

Appendix A

Proof of Theorem 3.1

Proof. Exploiting (3.10), we have,

$$L(Q_{k+1}) - L(Q_k) = \frac{\beta_1}{2}\left(\left(Q^x_{k+1}\right)^2 - \left(Q^x_k\right)^2\right) + \frac{\beta_2}{2}\left(\left(Q^y_{k+1}\right)^2 - \left(Q^y_k\right)^2\right) + \frac{\beta_3}{2}\left(\left(Q^\theta_{k+1}\right)^2 - \left(Q^\theta_k\right)^2\right) + \frac{\beta_4}{2}\left(\left(Q^v_{k+1}\right)^2 - \left(Q^v_k\right)^2\right).$$

Using (3.6), we have,

$$\begin{aligned} L(Q_{k+1}) - L(Q_k) =& \frac{\beta_1}{2}\left(\left(x_{k+1} - x^d_{k+1}\right)^2 - \left(x_k - x^d_k\right)^2\right) + \\ & \frac{\beta_2}{2}\left(\left(y_{k+1} - y^d_{k+1}\right)^2 - \left(y_k - y^d_k\right)^2\right) + \\ & \frac{\beta_3}{2}\left(\left(\theta_{k+1} - \theta^d_{k+1}\right)^2 - \left(\theta_k - \theta^d_k\right)^2\right) + \\ & \frac{\beta_4}{2}\left(\left(v_{k+1} - v^d_{k+1}\right)^2 - \left(v_k - v^d_k\right)^2\right), \end{aligned}$$

which can be expressed as,

$$\begin{aligned} & L(Q_{k+1}) - L(Q_k) = \\ & \frac{\beta_1}{2}\left(x^2_{k+1} - x^2_k - 2x^d_{k+1}x_{k+1} + 2x^d_k x_k + \left(x^d_{k+1}\right)^2 - \left(x^d_k\right)^2\right) + \\ & \frac{\beta_2}{2}\left(y^2_{k+1} - y^2_k - 2y^d_{k+1}y_{k+1} + 2y^d_k y_k + \left(y^d_{k+1}\right)^2 - \left(y^d_k\right)^2\right) + \\ & \frac{\beta_3}{2}\left(\theta^2_{k+1} - \theta^2_k - 2\theta^d_{k+1}\theta_{k+1} + 2\theta^d_k\theta_k + \left(\theta^d_{k+1}\right)^2 - \left(\theta^d_k\right)^2\right) + \\ & \frac{\beta_4}{2}\left(v^2_{k+1} - v^2_k - 2v^d_{k+1}v_{k+1} + 2v^d_k v_k + \left(v^d_{k+1}\right)^2 - \left(v^d_k\right)^2\right). \end{aligned} \tag{A.1}$$

Define,

$$x^d_{k+1} \triangleq x^d_k + \gamma^x_k, \tag{A.2a}$$

$$y_{k+1}^d \triangleq y_k^d + \gamma_k^y, \tag{A.2b}$$

$$\theta_{k+1}^d \triangleq \theta_k^d + \gamma_k^\theta, \tag{A.2c}$$

$$v_{k+1}^d \triangleq v_k^d + \gamma_k^v. \tag{A.2d}$$

By plugging (A.2) into (A.1), we have,

$$\begin{aligned}
&L(Q_{k+1}) - L(Q_k) = \\
&\frac{\beta_1}{2}\left(x_{k+1}^2 - x_k^2 - 2x_k^d\left(x_{k+1} - x_k\right) - 2\gamma_k^x x_{k+1} + \Delta x_k^d\right) + \\
&\frac{\beta_2}{2}\left(y_{k+1}^2 - y_k^2 - 2y_k^d\left(y_{k+1} - y_k\right) - 2\gamma_k^y y_{k+1} + \Delta y_k^d\right) + \\
&\frac{\beta_3}{2}\left(\theta_{k+1}^2 - \theta_k^2 - 2\theta_k^d\left(\theta_{k+1} - \theta_k\right) - 2\gamma_k^\theta \theta_{k+1} + \Delta\theta_k^d\right) + \\
&\frac{\beta_4}{2}\left(v_{k+1}^2 - v_k^2 - 2v_k^d\left(v_{k+1} - v_k\right) - 2\gamma_k^v v_{k+1} + \Delta v_k^d\right),
\end{aligned} \tag{A.3}$$

where $\Delta x_k^d = \left(x_{k+1}^d\right)^2 - \left(x_k^d\right)^2$, $\Delta y_k^d = \left(y_{k+1}^d\right)^2 - \left(y_k^d\right)^2$, $\Delta\theta_k^d = \left(\theta_{k+1}^d\right)^2 - \left(\theta_k^d\right)^2$, and $\Delta v_k^d = \left(v_{k+1}^d\right)^2 - \left(v_k^d\right)^2$. Note that at each time step k, a limited number of waypoints are detected by the mounted camera on the ego vehicle. Assume the farthest waypoint that can be detected by the camera is at $(X^{\max}, Y^{\max})$, where $X^{\max}, Y^{\max} > 0$. Therefore, $-X^{\max} \leq x_k^d, \gamma_k^x \leq X^{\max}$ and $-Y^{\max} \leq y_k^d, \gamma_k^y \leq Y^{\max}$ (The vehicle is always at $(0,0)$). Consequently, $\Delta x_k^d \leq (X^{\max})^2$ and $\Delta y_k^d \leq (Y^{\max})^2$. Moreover, $-2\gamma_k^x x_{k+1} \leq 2\,(X^{\max})^2$ and $-2\gamma_k^y y_{k+1} \leq 2\,(Y^{\max})^2$. On the other hand, $-\pi/2 \leq \theta_k^d, \gamma_k^\theta \leq \pi/2$. Hence, $\Delta\theta_k^d \leq \pi^2/4$ and $-2\gamma_k^\theta\theta_{k+1} \leq \pi^2/2$. Additionally, vehicle speed is limited to $V^{\max}$. Thus, $-V^{\max} \leq v_k^d, \gamma_k^v \leq V^{\max}$, which yields $\Delta v_k^d \leq (V^{\max})^2$ and $-2\gamma_k^v v_{k+1} \leq 2\,(V^{\max})^2$. Finally, (A.3) can be rewritten as,

$$\begin{aligned}
&L(Q_{k+1}) - L(Q_k) \leq \\
&\frac{\beta_1}{2}\left(x_{k+1} - x_k\right)\left(x_{k+1} + x_k - 2x_k^d\right) + \zeta_x + \\
&\frac{\beta_2}{2}\left(y_{k+1} - y_k\right)\left(y_{k+1} + y_k - 2y_k^d\right) + \zeta_y + \\
&\frac{\beta_3}{2}\left(\theta_{k+1} - \theta_k\right)\left(\theta_{k+1} + \theta_k - 2\theta_k^d\right) + \zeta_\theta + \\
&\frac{\beta_4}{2}\left(v_{k+1} - v_k\right)\left(v_{k+1} + v_k - 2v_k^d\right) + \zeta_v,
\end{aligned} \tag{A.4}$$

where

$$\zeta_x = \frac{3\beta_1}{2}\left(X^{\max}\right)^2,$$
$$\zeta_y = \frac{3\beta_2}{2}\left(Y^{\max}\right)^2,$$
$$\zeta_\theta = \frac{\beta_3}{2}\left(\frac{\pi^2}{4} + \frac{\pi^2}{2}\right),$$
$$\zeta_v = \frac{3\beta_4}{2}\left(V^{\max}\right)^2.$$

Noting that from (3.6) and (3.7), we have,

$$x_{k+1} = x_k + \xi_k^x + x_{k+1}^d - x_k^d = x_k + \xi_k^x + \gamma_k^x, \tag{A.5a}$$
$$y_{k+1} = y_k + \xi_k^y + y_{k+1}^d - y_k^d = y_k + \xi_k^y + \gamma_k^y, \tag{A.5b}$$
$$\theta_{k+1} = \theta_k + \xi_k^\theta + \theta_{k+1}^d - \theta_k^d = \theta_k + \xi_k^\theta + \gamma_k^\theta, \tag{A.5c}$$
$$v_{k+1} = v_k + \xi_k^v + v_{k+1}^d - v_k^d = v_k + \xi_k^v + \gamma_k^v. \tag{A.5d}$$

Exploiting (A.5), (A.4) becomes,

$$\begin{aligned} &L(Q_{k+1}) - L(Q_k) \leq \\ &\frac{\beta_1}{2}\left(\xi_k^x + \gamma_k^x\right)\left(\xi_k^x + \gamma_k^x + 2x_k - 2x_k^d\right) + \zeta_x + \\ &\frac{\beta_2}{2}\left(\xi_k^y + \gamma_k^y\right)\left(\xi_k^y + \gamma_k^y + 2y_k - 2y_k^d\right) + \zeta_y + \\ &\frac{\beta_3}{2}\left(\xi_k^\theta + \gamma_k^\theta\right)\left(\xi_k^\theta + \gamma_k^\theta + 2\theta_k - 2\theta_k^d\right) + \zeta_\theta + \\ &\frac{\beta_4}{2}\left(\xi_k^v + \gamma_k^v\right)\left(\xi_k^v + \gamma_k^v + 2v_k - 2v_k^d\right) + \zeta_v. \end{aligned} \tag{A.6}$$

Define,

$$\varrho_k^x \triangleq \xi_k^x + \gamma_k^x = v_k \cos\theta_k \Delta t, \tag{A.7a}$$
$$\varrho_k^y \triangleq \xi_k^y + \gamma_k^y = v_k \sin\theta_k \Delta t, \tag{A.7b}$$
$$\varrho_k^\theta \triangleq \xi_k^\theta + \gamma_k^\theta = \omega_k \Delta t, \tag{A.7c}$$
$$\varrho_k^v \triangleq \xi_k^v + \gamma_k^v = a_k \Delta t. \tag{A.7d}$$

Substituting (A.7) into (A.6) and using (3.6), (A.6) becomes,

$$L(Q_{k+1}) - L(Q_k) \leq \frac{\beta_1}{2}\varrho_k^x\left(\varrho_k^x + 2Q_k^x\right) + \frac{\beta_2}{2}\varrho_k^y\left(\varrho_k^y + 2Q_k^y\right)$$

$$+\frac{\beta_3}{2}\varrho_k^\theta\left(\varrho_k^\theta+2Q_k^\theta\right)+\frac{\beta_4}{2}\varrho_k^v\left(\varrho_k^v+2Q_k^v\right)+\zeta, \tag{A.8}$$

where $\zeta=\zeta_x+\zeta_y+\zeta_\theta+\zeta_v$, Eq. (A.8) is equal to,

$$\begin{aligned}&L(Q_{k+1})-L(Q_k)\leq\\&\beta_1\left(\frac{1}{2}\left(\varrho_k^x\right)^2+Q_k^x\varrho_k^x\right)+\beta_2\left(\frac{1}{2}\left(\varrho_k^y\right)^2+Q_k^y\varrho_k^y\right)+\\&\beta_3\left(\frac{1}{2}\left(\varrho_k^\theta\right)^2+Q_k^\theta\varrho_k^\theta\right)+\beta_4\left(\frac{1}{2}\left(\varrho_k^v\right)^2+Q_k^v\varrho_k^v\right)+\zeta.\end{aligned} \tag{A.9}$$

Since the vehicle's speed is limited to $V^{\max}$, leveraging (A.7), we have $\left(\varrho_k^x\right)^2\leq\left(V^{\max}\Delta t\right)^2$ and $\left(\varrho_k^y\right)^2\leq\left(V^{\max}\Delta t\right)^2$, respectively. Moreover, according to (3.2d), we have $\left(\varrho_k^\theta\right)^2\leq\left(W\Delta t\right)^2$, $W=\max\{|\omega^{\max}|,|\omega^{\min}|\}$. Analogously, based on (3.2c), we have $\left(\varrho_k^v\right)^2\leq\left(A\Delta t\right)^2$, $A=\max\{|a^{\max}|,|a^{\min}|\}$. Accordingly,

$$\begin{aligned}&L(Q_{k+1})-L(Q_k)\leq\\&\quad B+\beta_1Q_k^x\varrho_k^x+\beta_2Q_k^y\varrho_k^y+\beta_3Q_k^\theta\varrho_k^\theta+\beta_4Q_k^v\varrho_k^v,\end{aligned} \tag{A.10}$$

where

$$B=\frac{\beta_1+\beta_2}{2}\left(V^{\max}\Delta t\right)^2+\frac{\beta_3}{2}\left(W\Delta t\right)^2+\frac{\beta_4}{2}\left(A\Delta t\right)^2+\zeta. \tag{A.11}$$

By taking conditional expectations from both hand sides with respect to Q_k, we have,

$$\begin{aligned}&\mathbb{E}\{L(Q_{k+1})-L(Q_k)|Q_k\}\leq B+\beta_1Q_k^x\mathbb{E}\{\varrho_k^x|Q_k\}+\\&\beta_2Q_k^y\mathbb{E}\{\varrho_k^y|Q_k\}+\beta_3Q_k^\theta\mathbb{E}\{\varrho_k^\theta|Q_k\}+\beta_4Q_k^v\mathbb{E}\{\varrho_k^v|Q_k\},\end{aligned}$$

Utilizing (3.11) and adding $V\mathbb{E}\{u_k|Q_k\}$ to both hand sides, we obtain (3.13),

$$\begin{aligned}&\Delta(Q_k)+V\mathbb{E}\{u_k|Q_k\}\leq B+\beta_1Q_k^x\mathbb{E}\{\varrho_k^x|Q_k\}\\&+\beta_2Q_k^y\mathbb{E}\{\varrho_k^y|Q_k\}+\beta_3Q_k^\theta\mathbb{E}\{\varrho_k^\theta|Q_k\}+\beta_4Q_k^v\mathbb{E}\{\varrho_k^v|Q_k\}\\&+V\mathbb{E}\{u_k|Q_k\}.\end{aligned}$$

■

Appendix B

Proof of Theorem 3.2

Theorem 3.2 comprises the following steps:

- A relaxed version of problem (3.2) is constructed in (B.2).
- It is shown the optimal objective of (B.2), denoted as $\tilde{p}$, is less than or equal to the optimal objective of (3.2), p^*, i.e., $\tilde{p} \leq p^*$.
- (B.2) satisfies the constraints of problem (3.17).
- The optimal solution of (3.17) is denoted by $\{\hat{a}_k, \hat{\omega}_k\}$. Evaluating the objective of (3.17) at $\{\hat{a}_k, \hat{\omega}_k\}$ yields $V\hat{u}_k + \hat{\Lambda}_k$.
- The objective of (3.17) evaluated at any values other than $\{\hat{a}_k, \hat{\omega}_k\}$ is greater than or equal to $V\hat{u}_k + \hat{\Lambda}_k$.
- Evaluating the objective of (3.17) by the solution of (B.2) yields $V\hat{u}_k + \hat{\Lambda}_k \leq V\tilde{u}_k + \tilde{\Lambda}_k$.
- With the help of mathematical manipulations, using $\tilde{p} \leq p^*$ and $V\hat{u}_k + \hat{\Lambda}_k \leq V\tilde{u}_k + \tilde{\Lambda}_k$, we derive $\frac{1}{N}\sum_{k=1}^{N} \mathbb{E}\{\hat{u}_k\} \leq p^* + \frac{\text{some constants}}{V}$.

Proof. In (3.2b), by taking expectations on $v_{k+1} = v_k + a_k \Delta t$ and summing over $\{1, 2, \cdots, N\}$, we have,

$$\mathbb{E}\{v_{N+1}\} - \mathbb{E}\{v_1\} = \sum_{k=1}^{N} \mathbb{E}\{a_k \Delta t\}, \tag{B.1}$$

where the left-hand side is obtained through the law of telescoping sums. All feasible solutions $\{a_k, \omega_k\}$ that satisfy constrains of problem (3.2) also satisfy (B.1). Define a new optimization problem by substituting $v_{k+1} = v_k + a_k \Delta t$ with (B.1),

$$\min \qquad \frac{1}{N} \sum_{k=1}^{N} \mathbb{E}\left\{u_k\right\}, \tag{B.2a}$$

subject to

$$(3.2c)\text{-}(3.2d),\ (B.1)$$

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} v_k \cos\theta_k \\ v_k \sin\theta_k \\ \omega_k \end{bmatrix} \Delta t, \quad \forall k. \tag{B.2b}$$

Since expectation of sums is sum of expectations, (B.2a) is the same as (3.2a). Furthermore, as all feasible solutions $\{a_k, \omega_k\}$ that satisfy constraints of problem (3.2) also satisfy constraints of (B.2), then problem (B.2) is a relaxed version of problem (3.2). Consequently, the optimal solution of (3.2) is a feasible solution of (B.2), and the optimal objective of (B.2) is less than or equal to the optimal objective of (3.2). Suppose $\tilde{p}$ represents the optimal objective value of (B.2) obtained from optimal solution $\{\tilde{a_k}, \tilde{\omega_k}\}$, therefore, $\tilde{p} \leq p^*$,

$$\tilde{p} = \frac{1}{N} \sum_{k=1}^{N} \mathbb{E}\left\{\tilde{u}_k\right\} \leq p^* = \frac{1}{N} \sum_{k=1}^{N} \mathbb{E}\left\{u_k^*\right\}, \tag{B.3}$$

where $\tilde{u}_k$ and u_k^* are obtained from the optimal solutions of (B.2) and (3.2) using (3.9), respectively. Additionally, since (3.17) minimizes the upper bound of drift-plus-penalty term, the solutions of (3.17) yield the minimum of $Vu_k + \Lambda_k$ as,

$$V\hat{u}_k + \hat{\Lambda}_k, \tag{B.4}$$

where $\hat{u}_k$ and $\hat{\Lambda}_k$ are evaluated at $\{\hat{a}_k, \hat{\omega}_k\}$. Evaluating $Vu_k + \Lambda_k$ at any other values satisfying the constraints of (3.17) yields a value greater than or equal to (B.4). The

solutions of the relaxed problem (B.2) satisfy constraints of (3.17), therefore,

$$V\hat{u}_k + \hat{\Lambda}_k \leq V\tilde{u}_k + \tilde{\Lambda}_k, \tag{B.5}$$

where $\tilde{u}_k$ and $\tilde{\Lambda}_k$ are evaluated at $\{\tilde{a}_k, \tilde{\omega}_k\}$. By taking expectations with respect to $\hat{Q}_t$ of (B.4),

$$V\mathbb{E}\left\{\hat{u}_k|\hat{Q}_k\right\} + \mathbb{E}\left\{\hat{\Lambda}_k|\hat{Q}_k\right\}, \tag{B.6}$$

where

$$\begin{aligned}\mathbb{E}\left\{\hat{\Lambda}_k|\hat{Q}_k\right\} = \beta_1\hat{Q}_k^x\mathbb{E}\left\{(\hat{v}_k + \hat{a}_k\Delta t)\cos\left(\hat{\theta}_k + \hat{\omega}_k\Delta t\right)\Delta t|\hat{Q}_k\right\} \\ + \beta_2\hat{Q}_k^y\mathbb{E}\left\{(\hat{v}_k + \hat{a}_k\Delta t)\sin\left(\hat{\theta}_k + \hat{\omega}_k\Delta t\right)\Delta t|\hat{Q}_k\right\} \\ + \beta_3\hat{Q}_k^\theta\mathbb{E}\left\{\hat{\omega}_k\Delta t|\hat{Q}_k\right\} + \beta_4\hat{Q}_k^v\mathbb{E}\left\{\hat{a}_k\Delta t|\hat{Q}_k\right\}.\end{aligned} \tag{B.7}$$

By substituting (B.6) into (3.13), under assumption (3.15), we have,

$$\begin{aligned}&\Delta(\hat{Q}_k) + V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\} \leq B + \beta_1\hat{Q}_k^x\mathbb{E}\{\hat{\varrho}_k^x|\hat{Q}_k\} \\ &+ \beta_2\hat{Q}_k^y\mathbb{E}\{\hat{\varrho}_k^y|\hat{Q}_k\} + \beta_3\hat{Q}_k^\theta\mathbb{E}\{\hat{\varrho}_k^\theta|\hat{Q}_k\} + \beta_4\hat{Q}_k^v\mathbb{E}\{\hat{\varrho}_k^v|\hat{Q}_k\} \\ &+ V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\}.\end{aligned} \tag{B.8}$$

By substituting $\hat{\varrho}_k^x$, $\hat{\varrho}_k^y$, $\hat{\varrho}_k^\theta$ and $\hat{\varrho}_k^v$ from (A.7) and using assumption (3.15), Eq. (B.8) can be equivalently expressed as,

$$\Delta(\hat{Q}_k) + V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\} \leq B + \mathbb{E}\left\{\hat{\Lambda}_k|\hat{Q}_k\right\} + V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\}. \tag{B.9}$$

By taking expectations from both hand sides, we have,

$$\mathbb{E}\left\{\Delta(\hat{Q}_k)\right\} + V\mathbb{E}\{\hat{u}_k\} \leq B + \mathbb{E}\left\{\hat{\Lambda}_k\right\} + V\mathbb{E}\{\hat{u}_k\}, \tag{B.10}$$

where the condition with respect to $\hat{Q}_k$ is removed according to the law of iterated expectations, and,

$$\begin{aligned}\mathbb{E}\left\{\hat{\Lambda}_k\right\} = \beta_1\mathbb{E}\left\{\hat{Q}_k^x\right\}\mathbb{E}\left\{(\hat{v}_k + \hat{a}_k\Delta t)\cos\left(\hat{\theta}_k + \hat{\omega}_k\Delta t\right)\Delta t\right\} \\ + \beta_2\mathbb{E}\left\{\hat{Q}_k^y\right\}\mathbb{E}\left\{(\hat{v}_k + \hat{a}_k\Delta t)\sin\left(\hat{\theta}_k + \hat{\omega}_k\Delta t\right)\Delta t\right\} \\ + \beta_3\mathbb{E}\left\{\hat{Q}_k^\theta\right\}\mathbb{E}\left\{\hat{\omega}_k\Delta t\right\} + \beta_4\mathbb{E}\left\{\hat{Q}_k^v\right\}\mathbb{E}\left\{\hat{a}_k\Delta t\right\}.\end{aligned} \tag{B.11}$$

Utilizing (B.5) and (B.10), we have,

$$\mathbb{E}\left\{\Delta(\hat{Q}_k)\right\} + V\mathbb{E}\{\hat{u}_k\} \leq B + \mathbb{E}\left\{\hat{\Lambda}_k\right\} + V\mathbb{E}\{\hat{u}_k\}$$
$$\leq B + \mathbb{E}\left\{\tilde{\Lambda}_k\right\} + V\mathbb{E}\{\tilde{u}_k\}. \tag{B.12}$$

Using (3.11) and summing over all time steps,

$$\mathbb{E}\left\{L(\hat{Q}_{N+1})\right\} - \mathbb{E}\left\{L(\hat{Q}_1)\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq$$
$$NB + \sum_{k=1}^{N}\mathbb{E}\left\{\tilde{\Lambda}_k\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\}. \tag{B.13}$$

As $L(Q_k)$ is a positive definite function, $\mathbb{E}\left\{L(\hat{Q}_{N+1})\right\}$ can be omitted,

$$-\mathbb{E}\left\{L(\hat{Q}_1)\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq$$
$$NB + \sum_{k=1}^{N}\mathbb{E}\left\{\tilde{\Lambda}_k\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\}. \tag{B.14}$$

By dividing both hand sides by NV and rearranging,

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq \frac{B}{V} +$$
$$\frac{1}{NV}\sum_{k=1}^{N}\mathbb{E}\left\{\tilde{\Lambda}_k\right\} + \frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\} + \frac{1}{NV}\mathbb{E}\left\{L(\hat{Q}_1)\right\}. \tag{B.15}$$

Based on the limits defined in order to construct inequality (A.4), the maximum value of Λ_k in (3.18) is

$$\Lambda_k \leq \Lambda^{\max},$$
$$\Lambda^{\max} = \left(\beta_1 X^{\max} + \beta_2 Y^{\max}\right)\left(V^{\max} + A\Delta t\right)\Delta t$$
$$+ \beta_3\frac{\pi}{2}W\Delta t + \beta_4 V^{\max}A\Delta t. \tag{B.16}$$

Therefore,

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq \frac{B}{V} +$$
$$\frac{\Lambda^{\max}}{V} + \frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\} + \frac{1}{NV}\mathbb{E}\left\{L(\hat{Q}_1)\right\}. \tag{B.17}$$

Using (B.3), we arrive at,

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq \frac{B_1}{V} + p^* + \frac{1}{NV}\mathbb{E}\left\{L(\hat{Q}_1)\right\}, \tag{B.18}$$

$$B_1 = B + \Lambda^{\max}. \tag{B.19}$$

If $V \to \infty$, the left-hand side of (B.18) converges to p^*. ■

Appendix C

Proof of Theorem 4.1

Proof. Exploiting (4.29), we have,

$$
\begin{aligned}
&L(Q_{k+1}) - L(Q_k) = \frac{\beta_1}{2}\left(\left(Q^x_{k+1}\right)^2 - \left(Q^x_k\right)^2\right) + \\
&\frac{\beta_2}{2}\left(\left(Q^y_{k+1}\right)^2 - \left(Q^y_k\right)^2\right) + \frac{\beta_3}{2}\left(\left(Q^\theta_{k+1}\right)^2 - \left(Q^\theta_k\right)^2\right) + \\
&\frac{\beta_4}{2}\left(\left(Q^v_{k+1}\right)^2 - \left(Q^v_k\right)^2\right) + \frac{\beta_5}{2}\left(\left(Q^\nu_{k+1}\right)^2 - \left(Q^\nu_k\right)^2\right) + \\
&\frac{\beta_6}{2}\left(\left(Q^r_{k+1}\right)^2 - \left(Q^r_k\right)^2\right).
\end{aligned}
$$

Using (4.25), we have,

$$
\begin{aligned}
L(Q_{k+1}) - L(Q_k) =& \frac{\beta_1}{2}\left(\left(x_{k+1} - x^d_{k+1}\right)^2 - \left(x_k - x^d_k\right)^2\right) + \\
&\frac{\beta_2}{2}\left(\left(y_{k+1} - y^d_{k+1}\right)^2 - \left(y_k - y^d_k\right)^2\right) + \\
&\frac{\beta_3}{2}\left(\left(\theta_{k+1} - \theta^d_{k+1}\right)^2 - \left(\theta_k - \theta^d_k\right)^2\right) + \\
&\frac{\beta_4}{2}\left(\left(v_{k+1} - v^d_{k+1}\right)^2 - \left(v_k - v^d_k\right)^2\right) + \\
&\frac{\beta_5}{2}\left(\left(\nu_{k+1} - \nu^d\right)^2 - \left(\nu_k - \nu^d\right)^2\right) + \\
&\frac{\beta_6}{2}\left(\left(r_{k+1} - r^d\right)^2 - \left(r_k - r^d\right)^2\right),
\end{aligned}
$$

which can be expressed as,

$$
\begin{aligned}
&L(Q_{k+1}) - L(Q_k) = \\
&\frac{\beta_1}{2}\left(x^2_{k+1} - x^2_k - 2x^d_{k+1}x_{k+1} + 2x^d_k x_k + \left(x^d_{k+1}\right)^2 - \left(x^d_k\right)^2\right) + \\
&\frac{\beta_2}{2}\left(y^2_{k+1} - y^2_k - 2y^d_{k+1}y_{k+1} + 2y^d_k y_k + \left(y^d_{k+1}\right)^2 - \left(y^d_k\right)^2\right) + \\
&\frac{\beta_3}{2}\left(\theta^2_{k+1} - \theta^2_k - 2\theta^d_{k+1}\theta_{k+1} + 2\theta^d_k\theta_k + \left(\theta^d_{k+1}\right)^2 - \left(\theta^d_k\right)^2\right) +
\end{aligned}
$$

$$\frac{\beta_4}{2}\left(v_{k+1}^2 - v_k^2 - 2v_{k+1}^d v_{k+1} + 2v_k^d v_k + \left(v_{k+1}^d\right)^2 - \left(v_k^d\right)^2\right)$$
$$\frac{\beta_5}{2}\left(\nu_{k+1}^2 - \nu_k^2 - 2\nu^d \nu_{k+1} + 2\nu^d \nu_k\right)$$
$$\frac{\beta_6}{2}\left(r_{k+1}^2 - r_k^2 - 2r^d r_{k+1} + 2r^d r_k\right). \tag{C.1}$$

Define,

$$x_{k+1}^d \triangleq x_k^d + \gamma_k^x, \tag{C.2a}$$
$$y_{k+1}^d \triangleq y_k^d + \gamma_k^y, \tag{C.2b}$$
$$\theta_{k+1}^d \triangleq \theta_k^d + \gamma_k^\theta, \tag{C.2c}$$
$$v_{k+1}^d \triangleq v_k^d + \gamma_k^v. \tag{C.2d}$$

By plugging (C.2) into (C.1), we have,

$$\begin{aligned}
&L(Q_{k+1}) - L(Q_k) = \\
&\frac{\beta_1}{2}\left(x_{k+1}^2 - x_k^2 - 2x_k^d\left(x_{k+1} - x_k\right) - 2\gamma_k^x x_{k+1} + \Delta x_k^d\right) + \\
&\frac{\beta_2}{2}\left(y_{k+1}^2 - y_k^2 - 2y_k^d\left(y_{k+1} - y_k\right) - 2\gamma_k^y y_{k+1} + \Delta y_k^d\right) + \\
&\frac{\beta_3}{2}\left(\theta_{k+1}^2 - \theta_k^2 - 2\theta_k^d\left(\theta_{k+1} - \theta_k\right) - 2\gamma_k^\theta \theta_{k+1} + \Delta\theta_k^d\right) + \\
&\frac{\beta_4}{2}\left(v_{k+1}^2 - v_k^2 - 2v_k^d\left(v_{k+1} - v_k\right) - 2\gamma_k^v v_{k+1} + \Delta v_k^d\right) \\
&\frac{\beta_5}{2}\xi_k^\nu\left(2Q_k^\nu + \xi_k^\nu\right) + \frac{\beta_6}{2}\xi_k^r\left(2Q_k^r + \xi_k^r\right),
\end{aligned} \tag{C.3}$$

where $\Delta x_k^d = \left(x_{k+1}^d\right)^2 - \left(x_k^d\right)^2$, $\Delta y_k^d = \left(y_{k+1}^d\right)^2 - \left(y_k^d\right)^2$, $\Delta\theta_k^d = \left(\theta_{k+1}^d\right)^2 - \left(\theta_k^d\right)^2$, and $\Delta v_k^d = \left(v_{k+1}^d\right)^2 - \left(v_k^d\right)^2$. Note that at each time step k, a limited number of waypoints are detected by the mounted camera on the ego vehicle. Assume the farthest waypoint that can be detected by the camera is at $(X^{\max}, Y^{\max})$, where $X^{\max}, Y^{\max} > 0$. Therefore, $-X^{\max} \le x_k^d, \gamma_k^x \le X^{\max}$ and $-Y^{\max} \le y_k^d, \gamma_k^y \le Y^{\max}$ (The vehicle is always at $(0,0)$). Consequently, $\Delta x_k^d \le \left(X^{\max}\right)^2$ and $\Delta y_k^d \le \left(Y^{\max}\right)^2$. Moreover, $-2\gamma_k^x x_{k+1} \le 2\left(X^{\max}\right)^2$ and $-2\gamma_k^y y_{k+1} \le 2\left(Y^{\max}\right)^2$. On the other hand, $-\pi/2 \le \theta_k^d, \gamma_k^\theta \le \pi/2$. Hence, $\Delta\theta_k^d \le \pi^2/4$ and $-2\gamma_k^\theta\theta_{k+1} \le \pi^2/2$. Additionally, vehicle speed is limited to $V^{\max}$. Thus, $-V^{\max} \le v_k^d, \gamma_k^v \le V^{\max}$, which yields $\Delta v_k^d \le \left(V^{\max}\right)^2$ and $-2\gamma_k^v v_{k+1} \le 2\left(V^{\max}\right)^2$.

Finally, (C.3) can be rewritten as,

$$
\begin{aligned}
&L(Q_{k+1}) - L(Q_k) \leq \\
&\frac{\beta_1}{2}\left(x_{k+1} - x_k\right)\left(x_{k+1} + x_k - 2x_k^d\right) + \zeta_x + \\
&\frac{\beta_2}{2}\left(y_{k+1} - y_k\right)\left(y_{k+1} + y_k - 2y_k^d\right) + \zeta_y + \\
&\frac{\beta_3}{2}\left(\theta_{k+1} - \theta_k\right)\left(\theta_{k+1} + \theta_k - 2\theta_k^d\right) + \zeta_\theta + \\
&\frac{\beta_4}{2}\left(v_{k+1} - v_k\right)\left(v_{k+1} + v_k - 2v_k^d\right) + \zeta_v + \\
&\frac{\beta_5}{2}\xi_k^\nu\left(2Q_k^\nu + \xi_k^\nu\right) + \frac{\beta_6}{2}\xi_k^r\left(2Q_k^r + \xi_k^r\right),
\end{aligned}
\tag{C.4}
$$

where

$$
\begin{aligned}
\zeta_x &= \frac{3\beta_1}{2}\left(X^{\max}\right)^2, \\
\zeta_y &= \frac{3\beta_2}{2}\left(Y^{\max}\right)^2, \\
\zeta_\theta &= \frac{\beta_3}{2}\left(\frac{\pi^2}{4} + \frac{\pi^2}{2}\right), \\
\zeta_v &= \frac{3\beta_4}{2}\left(V^{\max}\right)^2.
\end{aligned}
$$

Noting that from (4.25) and (4.26), we have,

$$x_{k+1} = x_k + \xi_k^x + x_{k+1}^d - x_k^d = x_k + \xi_k^x + \gamma_k^x, \tag{C.5a}$$

$$y_{k+1} = y_k + \xi_k^y + y_{k+1}^d - y_k^d = y_k + \xi_k^y + \gamma_k^y, \tag{C.5b}$$

$$\theta_{k+1} = \theta_k + \xi_k^\theta + \theta_{k+1}^d - \theta_k^d = \theta_k + \xi_k^\theta + \gamma_k^\theta, \tag{C.5c}$$

$$v_{k+1} = v_k + \xi_k^v + v_{k+1}^d - v_k^d = v_k + \xi_k^v + \gamma_k^v. \tag{C.5d}$$

Exploiting (C.5), (C.4) becomes,

$$
\begin{aligned}
&L(Q_{k+1}) - L(Q_k) \leq \\
&\frac{\beta_1}{2}\left(\xi_k^x + \gamma_k^x\right)\left(\xi_k^x + \gamma_k^x + 2x_k - 2x_k^d\right) + \zeta_x + \\
&\frac{\beta_2}{2}\left(\xi_k^y + \gamma_k^y\right)\left(\xi_k^y + \gamma_k^y + 2y_k - 2y_k^d\right) + \zeta_y + \\
&\frac{\beta_3}{2}\left(\xi_k^\theta + \gamma_k^\theta\right)\left(\xi_k^\theta + \gamma_k^\theta + 2\theta_k - 2\theta_k^d\right) + \zeta_\theta +
\end{aligned}
$$

$$\frac{\beta_4}{2}\left(\xi_k^v+\gamma_k^v\right)\left(\xi_k^v+\gamma_k^v+2v_k-2v_k^d\right)+\zeta_v+$$
$$\frac{\beta_5}{2}\xi_k^\nu\left(2Q_k^\nu+\xi_k^\nu\right)+\frac{\beta_6}{2}\xi_k^r\left(2Q_k^r+\xi_k^r\right). \tag{C.6}$$

Define,

$$\varrho_k^x \triangleq \xi_k^x+\gamma_k^x=(v_k\cos\theta_k-\nu_k\sin\theta_k)\Delta t, \tag{C.7a}$$
$$\varrho_k^y \triangleq \xi_k^y+\gamma_k^y=(v_k\sin\theta_k+\nu_k\cos\theta_k)\Delta t, \tag{C.7b}$$
$$\varrho_k^\theta \triangleq \xi_k^\theta+\gamma_k^\theta=r_k\Delta t, \tag{C.7c}$$
$$\varrho_k^v \triangleq \xi_k^v+\gamma_k^v=\left(\frac{F_k^x}{m}+\nu_k r_k\right)\Delta t, \tag{C.7d}$$
$$\varrho_k^\nu \triangleq \xi_k^\nu=\left(\frac{F_k^f+F_k^r}{m}-v_k r_k\right)\Delta t, \tag{C.7e}$$
$$\varrho_k^r \triangleq \xi_k^r=\frac{l_f F_k^f-l_r F_k^r}{I_z}\Delta t. \tag{C.7f}$$

Substituting (C.7) into (C.6) and using (4.25), (C.6) becomes,

$$L(Q_{k+1})-L(Q_k)\le\frac{\beta_1}{2}\varrho_k^x\left(\varrho_k^x+2Q_k^x\right)+\frac{\beta_2}{2}\varrho_k^y\left(\varrho_k^y+2Q_k^y\right)$$
$$+\frac{\beta_3}{2}\varrho_k^\theta\left(\varrho_k^\theta+2Q_k^\theta\right)+\frac{\beta_4}{2}\varrho_k^v\left(\varrho_k^v+2Q_k^v\right)+\frac{\beta_5}{2}\varrho_k^\nu\left(\varrho_k^\nu+2Q_k^\nu\right)$$
$$+\frac{\beta_6}{2}\varrho_k^r\left(\varrho_k^r+2Q_k^r\right)+\zeta, \tag{C.8}$$

where $\zeta=\zeta_x+\zeta_y+\zeta_\theta+\zeta_v$, Eq. (C.8) is equal to,

$$L(Q_{k+1})-L(Q_k)\le\zeta+$$
$$\beta_1\left(\frac{1}{2}\left(\varrho_k^x\right)^2+Q_k^x\varrho_k^x\right)+\beta_2\left(\frac{1}{2}\left(\varrho_k^y\right)^2+Q_k^y\varrho_k^y\right)+$$
$$\beta_3\left(\frac{1}{2}\left(\varrho_k^\theta\right)^2+Q_k^\theta\varrho_k^\theta\right)+\beta_4\left(\frac{1}{2}\left(\varrho_k^v\right)^2+Q_k^v\varrho_k^v\right)+$$
$$\beta_5\left(\frac{1}{2}\left(\varrho_k^\nu\right)^2+Q_k^\nu\varrho_k^\nu\right)+\beta_6\left(\frac{1}{2}\left(\varrho_k^r\right)^2+Q_k^r\varrho_k^r\right). \tag{C.9}$$

Since the vehicle's speed is limited to $V^{\max}$, leveraging (C.7), we have

$$\left(\varrho_k^x\right)^2\le\left(V^{\max}\Delta t\right)^2,$$
$$\left(\varrho_k^y\right)^2\le\left(V^{\max}\Delta t\right)^2.$$

respectively. Moreover, according to (4.23), we have

$$\begin{aligned}
\left(\varrho_k^{\theta}\right)^2 &\leq \left(r^{\max}\Delta t\right)^2, \\
(\varrho_k^{v})^2 &\leq \left((F_{\max}^{x}/m + \nu^{\max} r^{\max})\Delta t\right)^2, \\
(\varrho_k^{\nu})^2 &\leq \left(\mathfrak{A}\Delta t\right)^2, \\
\mathfrak{A} &= \max\left\{(F_{\max}^{f} + F_{\max}^{r})/m, V^{\max} r^{\max}\right\}, \\
(\varrho_k^{r})^2 &\leq \left(\mathfrak{B}\Delta t\right)^2, \\
\mathfrak{B} &= \max\left\{l_f F_{\max}^{f}/I_z, l_r F_{\max}^{r}/I_z\right\}.
\end{aligned}$$

Accordingly,

$$\begin{aligned}
L(Q_{k+1}) - L(Q_k) \leq B &+ \beta_1 Q_k^{x}\varrho_k^{x} + \beta_2 Q_k^{y}\varrho_k^{y} + \beta_3 Q_k^{\theta}\varrho_k^{\theta} \\
&+ \beta_4 Q_k^{v}\varrho_k^{v} + \beta_5 Q_k^{\nu}\varrho_k^{\nu} + \beta_6 Q_k^{r}\varrho_k^{r},
\end{aligned} \tag{C.10}$$

where

$$\begin{aligned}
B = \zeta &+ \frac{\beta_1 + \beta_2}{2}\left(V^{\max}\Delta t\right)^2 + \frac{\beta_3}{2}\left(r^{\max}\Delta t\right)^2 + \\
&\frac{\beta_4}{2}\left((F_{\max}^{x}/m + \nu^{\max} r^{\max})\Delta t\right)^2 + \frac{\beta_5}{2}\left(\mathfrak{A}\Delta t\right)^2 + \frac{\beta_6}{2}\left(\mathfrak{B}\Delta t\right)^2.
\end{aligned} \tag{C.11}$$

By taking conditional expectations from both hand sides with respect to Q_k, we have,

$$\begin{aligned}
&\mathbb{E}\{L(Q_{k+1}) - L(Q_k)|Q_k\} \leq B + \beta_1 Q_k^{x}\mathbb{E}\{\varrho_k^{x}|Q_k\} \\
&+ \beta_2 Q_k^{y}\mathbb{E}\{\varrho_k^{y}|Q_k\} + \beta_3 Q_k^{\theta}\mathbb{E}\{\varrho_k^{\theta}|Q_k\} + \beta_4 Q_k^{v}\mathbb{E}\{\varrho_k^{v}|Q_k\} \\
&+ \beta_5 Q_k^{\nu}\mathbb{E}\{\varrho_k^{\nu}|Q_k\} + \beta_6 Q_k^{r}\mathbb{E}\{\varrho_k^{r}|Q_k\},
\end{aligned}$$

Utilizing (4.30) and adding $V\mathbb{E}\{u_k|Q_k\}$ to both hand sides, we obtain (4.32),

$$\begin{aligned}
&\Delta(Q_k) + V\mathbb{E}\{u_k|Q_k\} \leq B + \beta_1 Q_k^{x}\mathbb{E}\{\varrho_k^{x}|Q_k\} \\
&+ \beta_2 Q_k^{y}\mathbb{E}\{\varrho_k^{y}|Q_k\} + \beta_3 Q_k^{\theta}\mathbb{E}\{\varrho_k^{\theta}|Q_k\} + \beta_4 Q_k^{v}\mathbb{E}\{\varrho_k^{v}|Q_k\} \\
&+ \beta_5 Q_k^{\nu}\mathbb{E}\{\varrho_k^{\nu}|Q_k\} + \beta_6 Q_k^{r}\mathbb{E}\{\varrho_k^{r}|Q_k\} + V\mathbb{E}\{u_k|Q_k\}.
\end{aligned}$$

■

Appendix D

Proof of Theorem 4.2

Proof. In (4.23b), by taking expectations on $\theta_{k+1} = \theta_k + r_k \Delta t$ and summing over $\{1, 2, \cdots, N\}$, we have,

$$\mathbb{E}\{\theta_{N+1}\} - \mathbb{E}\{\theta_1\} = \sum_{k=1}^{N} \mathbb{E}\{r_k \Delta t\}, \tag{D.1}$$

where the left-hand side is obtained through the law of telescoping sums. All feasible solutions c_k that satisfy constrains of problem (4.23) also satisfy (D.1). Define a new optimization problem by substituting $\theta_{k+1} = \theta_k + r_k \Delta t$ with (D.1),

$$\min \qquad \frac{1}{N} \sum_{k=1}^{N} \mathbb{E}\{u_k\}, \tag{D.2a}$$

subject to

$$\text{(4.23c)-(4.23g), (D.1)}$$

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ v_{k+1} \\ \nu_{k+1} \\ r_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ v_k \\ \nu_k \\ r_k \end{bmatrix} + \begin{bmatrix} v_k \cos\theta_k - \nu_k \sin\theta_k \\ v_k \sin\theta_k + \nu_k \cos\theta_k \\ F_k^x/m + \nu_k r_k \\ (F_k^f + F_k^r)/m - v_k r_k \\ (l_f F_k^f - l_r F_k^r)/I_z \end{bmatrix} \Delta t, \quad \forall k. \tag{D.2b}$$

Since expectation of sums is sum of expectations, (D.2a) is the same as (4.23a). Furthermore, as all feasible solutions c_k that satisfy constraints of problem (4.23) also

satisfy constraints of (D.2), then problem (D.2) is a relaxed version of problem (4.23). Consequently, the optimal solution of (4.23) is a feasible solution of (D.2), and the optimal objective of (D.2) is less than or equal to the optimal objective of (4.23). Suppose $\tilde{p}$ represents the optimal objective value of (D.2) obtained from optimal solution $\tilde{c}_k$, therefore, $\tilde{p} \leq p^*$,

$$\tilde{p} = \frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\left\{\tilde{u}_k\right\} \leq p^* = \frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\left\{u_k^*\right\}, \tag{D.3}$$

where $\tilde{u}_k$ and u_k^* are obtained from the optimal solutions of (D.2) and (4.23) using (4.28), respectively. Additionally, since (4.36) minimizes the upper bound of drift-plus-penalty term, the solutions of (4.36) yield the minimum of $Vu_k + \Lambda_k$ as,

$$V\hat{u}_k + \hat{\Lambda}_k, \tag{D.4}$$

where $\hat{u}_k$ and $\hat{\Lambda}_k$ are evaluated at $\hat{c}_k$. Evaluating $Vu_k+\Lambda_k$ at any other values satisfying the constraints of (4.36) yields a value greater than or equal to (D.4). The solutions of the relaxed problem (D.2) satisfy constraints of (4.36), therefore,

$$V\hat{u}_k + \hat{\Lambda}_k \leq V\tilde{u}_k + \tilde{\Lambda}_k, \tag{D.5}$$

where $\tilde{u}_k$ and $\tilde{\Lambda}_k$ are evaluated at $\tilde{c}_k$. By taking expectations with respect to $\hat{Q}_t$ of (D.4),

$$V\mathbb{E}\left\{\hat{u}_k|\hat{Q}_k\right\} + \mathbb{E}\left\{\hat{\Lambda}_k|\hat{Q}_k\right\}, \tag{D.6}$$

where

$$\begin{aligned}
&\mathbb{E}\left\{\hat{\Lambda}_k|\hat{Q}_k\right\} = \beta_1\hat{Q}_k^x\mathbb{E}\left\{(\hat{v}_k + (\hat{F}_k^x/m + \hat{\nu}_k\hat{r}_k)\Delta t)\cos(\hat{\theta}_k + \hat{r}_k\Delta t)|\hat{Q}_k\right\} - \\
&\beta_1\hat{Q}_k^x\mathbb{E}\left\{(\hat{\nu}_k + ((\hat{F}_k^f + \hat{F}_k^r)/m - \hat{v}_k\hat{r}_k)\Delta t)\sin(\hat{\theta}_k + \hat{r}_k\Delta t)|\hat{Q}_k\right\} \\
&+ \beta_2\hat{Q}_k^y\mathbb{E}\left\{(\hat{v}_k + (\hat{F}_k^x/m + \hat{\nu}_k\hat{r}_k)\Delta t)\sin(\hat{\theta}_k + \hat{r}_k\Delta t)|\hat{Q}_k\right\} + \\
&\beta_2\hat{Q}_k^y\mathbb{E}\left\{(\hat{\nu}_k + ((\hat{F}_k^f + \hat{F}_k^r)/m - \hat{v}_k\hat{r}_k)\Delta t)\cos(\hat{\theta}_k + \hat{r}_k\Delta t)|\hat{Q}_k\right\} \\
&+ \beta_3\hat{Q}_k^\theta\mathbb{E}\left\{\hat{r}_k|\hat{Q}_k\right\} + \beta_4\hat{Q}_k^v\mathbb{E}\left\{(\hat{F}_k^x/m + \hat{\nu}_k\hat{r}_k)|\hat{Q}_k\right\} + \\
&\beta_5\hat{Q}_k^\nu\mathbb{E}\left\{((\hat{F}_k^f + \hat{F}_k^r)/m - \hat{v}_k\hat{r}_k)|\hat{Q}_k\right\} + \beta_6\hat{Q}_k^r\mathbb{E}\left\{(l_f\hat{F}_k^f - l_r\hat{F}_k^r)/I_z|\hat{Q}_k\right\}.
\end{aligned} \tag{D.7}$$

By substituting (D.6) into (4.32), under assumption (4.34), we have,

$$\begin{aligned}\Delta(\hat{Q}_k) + V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\} \leq B + \beta_1\hat{Q}_k^x\mathbb{E}\{\hat{\varrho}_k^x|\hat{Q}_k\} \\ + \beta_2\hat{Q}_k^y\mathbb{E}\{\hat{\varrho}_k^y|\hat{Q}_k\} + \beta_3\hat{Q}_k^\theta\mathbb{E}\{\hat{\varrho}_k^\theta|\hat{Q}_k\} + \beta_4\hat{Q}_k^v\mathbb{E}\{\hat{\varrho}_k^v|\hat{Q}_k\} \\ + \beta_5\hat{Q}_k^\nu\mathbb{E}\{\hat{\varrho}_k^\nu|\hat{Q}_k\} + \beta_6\hat{Q}_k^r\mathbb{E}\{\hat{\varrho}_k^r|\hat{Q}_k\} + V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\}. \end{aligned} \tag{D.8}$$

By substituting $\hat{\varrho}_k^x$, $\hat{\varrho}_k^y$, $\hat{\varrho}_k^\theta$, $\hat{\varrho}_k^v$, $\hat{\varrho}_k^\nu$ and $\hat{\varrho}_k^r$ from (C.7) and using assumption (4.34), Eq. (D.8) can be equivalently expressed as,

$$\Delta(\hat{Q}_k) + V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\} \leq B + \mathbb{E}\left\{\hat{\Lambda}_k|\hat{Q}_k\right\} + V\mathbb{E}\{\hat{u}_k|\hat{Q}_k\}. \tag{D.9}$$

By taking expectations from both hand sides, we have,

$$\mathbb{E}\left\{\Delta(\hat{Q}_k)\right\} + V\mathbb{E}\{\hat{u}_k\} \leq B + \mathbb{E}\left\{\hat{\Lambda}_k\right\} + V\mathbb{E}\{\hat{u}_k\}, \tag{D.10}$$

where the condition with respect to $\hat{Q}_k$ is removed according to the law of iterated expectations, and,

$$\begin{aligned}&\mathbb{E}\left\{\hat{\Lambda}_k\right\} = \beta_1\mathbb{E}\left\{\hat{Q}_k^x\right\}\mathbb{E}\left\{(\hat{v}_k + (\hat{F}_k^x/m + \hat{\nu}_k\hat{r}_k)\Delta t)\cos(\hat{\theta}_k + \hat{r}_k\Delta t)\right\} - \\ &\beta_1\mathbb{E}\left\{\hat{Q}_k^x\right\}\mathbb{E}\left\{(\hat{\nu}_k + ((\hat{F}_k^f + \hat{F}_k^r)/m - \hat{v}_k\hat{r}_k)\Delta t)\sin(\hat{\theta}_k + \hat{r}_k\Delta t)\right\} \\ &+ \beta_2\mathbb{E}\left\{\hat{Q}_k^y\right\}\mathbb{E}\left\{(\hat{v}_k + (\hat{F}_k^x/m + \hat{\nu}_k\hat{r}_k)\Delta t)\sin(\hat{\theta}_k + \hat{r}_k\Delta t)\right\} + \\ &\beta_2\mathbb{E}\left\{\hat{Q}_k^y\right\}\mathbb{E}\left\{(\hat{\nu}_k + ((\hat{F}_k^f + \hat{F}_k^r)/m - \hat{v}_k\hat{r}_k)\Delta t)\cos(\hat{\theta}_k + \hat{r}_k\Delta t)\right\} \\ &+ \beta_3\mathbb{E}\left\{\hat{Q}_k^\theta\right\}\mathbb{E}\left\{\hat{r}_k\right\} + \beta_4\mathbb{E}\left\{\hat{Q}_k^v\right\}\mathbb{E}\left\{(\hat{F}_k^x/m + \hat{\nu}_k\hat{r}_k)\right\} + \\ &\beta_5\mathbb{E}\left\{\hat{Q}_k^\nu\right\}\mathbb{E}\left\{((\hat{F}_k^f + \hat{F}_k^r)/m - \hat{v}_k\hat{r}_k)\right\} + \beta_6\mathbb{E}\left\{\hat{Q}_k^r\right\}\mathbb{E}\left\{(l_f\hat{F}_k^f - l_r\hat{F}_k^r)/I_z\right\}. \end{aligned} \tag{D.11}$$

Utilizing (D.5) and (D.10), we have,

$$\mathbb{E}\left\{\Delta(\hat{Q}_k)\right\} + V\mathbb{E}\{\hat{u}_k\} \leq B + \mathbb{E}\left\{\hat{\Lambda}_k\right\} + V\mathbb{E}\{\hat{u}_k\} \leq B + \mathbb{E}\left\{\tilde{\Lambda}_k\right\} + V\mathbb{E}\{\tilde{u}_k\}. \tag{D.12}$$

Using (4.30) and summing over all time steps,

$$\mathbb{E}\left\{L(\hat{Q}_{N+1})\right\} - \mathbb{E}\left\{L(\hat{Q}_1)\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq NB + \sum_{k=1}^{N}\mathbb{E}\left\{\tilde{\Lambda}_k\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\}. \tag{D.13}$$

As $L(Q_k)$ is a positive definite function, $\mathbb{E}\left\{L(\hat{Q}_{N+1})\right\}$ can be omitted,

$$-\mathbb{E}\left\{L(\hat{Q}_1)\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq NB + \sum_{k=1}^{N}\mathbb{E}\left\{\tilde{\Lambda}_k\right\} + V\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\}. \tag{D.14}$$

By dividing both hand sides by NV and rearranging,

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq \frac{B}{V} + \frac{1}{NV}\sum_{k=1}^{N}\mathbb{E}\left\{\tilde{\Lambda}_k\right\} + \frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\} + \frac{1}{NV}\mathbb{E}\left\{L(\hat{Q}_1)\right\}. \tag{D.15}$$

Based on the limits defined in order to construct inequality (C.4), the maximum value of Λ_k in (4.37) is

$$\begin{aligned}
\Lambda_k &\leq \Lambda^{\max},\\
\Lambda^{\max} &= \beta_1 X^{\max}\left(V^{\max} + (F^x_{\max}/m + \nu^{\max}r^{\max})\Delta t\right)\\
&\quad + \beta_1 X^{\max}\left(\nu^{\max} + ((F^f_{\max} + F^r_{\max})/m)\Delta t\right)\\
&\quad + \beta_2 Y^{\max}\left(V^{\max} + (F^x_{\max}/m + \nu^{\max}r^{\max})\Delta t\right)\\
&\quad + \beta_2 Y^{\max}\left(\nu^{\max} + ((F^f_{\max} + F^r_{\max})/m)\Delta t\right)\\
&\quad + \beta_3 r^{\max}\pi/2 + \beta_4 V^{\max}\left(F^x_{\max}/m + \nu^{\max}r^{\max}\right)\\
&\quad + \beta_5\nu^{\max}(F^f_{\max} + F^r_{\max})/m + \beta_6 r^{\max}l_f F^f_{\max}/I_z.
\end{aligned} \tag{D.16}$$

Therefore,

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq \frac{B}{V} + \frac{\Lambda^{\max}}{V} + \frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\tilde{u}_k\} + \frac{1}{NV}\mathbb{E}\left\{L(\hat{Q}_1)\right\}. \tag{D.17}$$

Using (D.3), we arrive at,

$$\frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\{\hat{u}_k\} \leq \frac{B_1}{V} + p^* + \frac{1}{NV}\mathbb{E}\left\{L(\hat{Q}_1)\right\}, \tag{D.18}$$

$$B_1 = B + \Lambda^{\max}. \tag{D.19}$$

If $V \to \infty$, the left-hand side of (D.18) converges to p^*. ■

Reference List

Abbasi, M. H., and J. Zhang, 2021: Joint optimization of electric vehicle fast charging and dc fast charging station. *2021 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, IEEE, 1–6.

Abbasi, M. H., J. Zhang, and V. Krovi, 2022: A lyapunov optimization approach to the quality of service for electric vehicle fast charging stations. *2022 IEEE Vehicle Power and Propulsion Conference (VPPC)*, IEEE, 1–6.

Afzal, A., and M. Ramis, 2020: Multi-objective optimization of thermal performance in battery system using genetic and particle swarm algorithm combined with fuzzy logics. *Journal of Energy Storage*, **32**, 101 815.

Akbarzadeh, M., and Coauthors, 2021: A novel liquid cooling plate concept for thermal management of lithium-ion batteries in electric vehicles. *Energy Conversion and Management*, **231**, 113 862.

Ammour, M., R. Orjuela, and M. Basset, 2022: A mpc combined decision making and trajectory planning for autonomous vehicle collision avoidance. **23 (12)**, 24 805–24 817.

Audi, 2021: Battery and thermal management, audi media center. https://www.audimediacenter.com/en/emotive-design-and-revolutionary-technologythe-audi-e-tron-gt-quattro-and-the-audi-rs-e-tron-gt-13655/batteryand-thermal-management-13784.

Bao, H., J. H. Park, and J. Cao, 2016: Synchronization of fractional-order complex-valued neural networks with time delay. *Neural Networks*, **81**, 16–28.

Bower, G., 2022: Tesla model s plaid battery: Clever new advancements discovered. https://insideevs.com/news/566047/tesla-models-clever-battery-advancements/.

Buehler, M., K. Iagnemma, and S. Singh, 2009: *The DARPA urban challenge: autonomous vehicles in city traffic*, Vol. 56. springer.

Chen, J., E. Jiaqiang, S. Kang, X. Zhao, H. Zhu, Y. Deng, Q. Peng, and Z. Zhang, 2019: Modeling and characterization of the mass transfer and thermal mechanics of the power lithium manganate battery under charging process. *Energy*, **187**, 115 924.

Chen, S., N. Bao, A. Garg, X. Peng, and L. Gao, 2021: A fast charging–cooling

coupled scheduling method for a liquid cooling-based thermal management system for lithium-ion batteries. *Engineering*, **7 (8)**, 1165–1176.

Cheng, L., C. Liu, and B. Yan, 2014: Improved hierarchical a-star algorithm for optimal parking path planning of the large parking lot. *2014 IEEE international conference on information and automation (ICIA)*, IEEE, 695–698.

Choi, W., J. W. Kim, C. Ahn, and J. Gim, 2022: Reinforcement learning-based controller for thermal management system of electric vehicles. *2022 IEEE Vehicle Power and Propulsion Conference (VPPC)*, IEEE, 1–5.

Clark, L., 2023: Remote exploration with robotic networks: Queue-aware autonomy and collaborative localization. Ph.D. thesis, University of Southern California.

Claussmann, L., M. Revilloud, D. Gruyer, and S. Glaser, 2019: A review of motion planning for highway autonomous driving. **21 (5)**, 1826–1848.

CNN, 2021: Here's what's in the bipartisan infrastructure package. https://www.cnn.com/2021/07/28/politics/infrastructure-bill-explained/index.html.

Cordoba-Arenas, A., S. Onori, Y. Guezennec, and G. Rizzoni, 2015: Capacity and power fade cycle-life model for plug-in hybrid electric vehicle lithium-ion battery cells containing blended spinel and layered-oxide positive electrodes. *Journal of Power Sources*, **278**, 473–483.

Deng, Y., C. Feng, E. Jiaqiang, H. Zhu, J. Chen, M. Wen, and H. Yin, 2018: Effects of different coolants and cooling strategies on the cooling performance of the power lithium ion battery system: A review. *Applied Thermal Engineering*, **142**, 10–29.

Dinh, K., 2021: Nerd's eye view: Mercedes amg m139 engine with e-turbo and liquid immersion battery cooling. https://motoiq.com/nerds-eye-view-mercedes-amg-m139-engine-with-e-turbo-and-liquid-immersion-battery-cooling/2/.

Fan, Y., D. Zhan, X. Tan, P. Lyu, and J. Rao, 2021: Optimization of cooling strategies for an electric vehicle in high-temperature environment. *Applied Thermal Engineering*, **195**, 117 088.

Fan, Z., Y. Fu, H. Liang, R. Gao, and S. Liu, 2023: A module-level charging optimization method of lithium-ion battery considering temperature gradient effect of liquid cooling and charging time. *Energy*, **265**, 126 331.

Fang, C., H. Lu, Y. Hong, S. Liu, and J. Chang, 2020: Dynamic pricing for electric vehicle extreme fast charging. *IEEE Transactions on Intelligent Transportation*

Systems, **22 (1)**, 531–541.

Gao, Y., A. Gray, H. E. Tseng, and F. Borrelli, 2014: A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles. *Vehicle System Dynamics*, **52 (6)**, 802–823.

Han, J., Z. Ju, X. Chen, M. Yang, H. Zhang, and R. Huai, 2023: Secure operations of connected and autonomous vehicles.

Hannan, M., Y. Young, M. M. Hoque, P. J. Ker, and M. N. Uddin, 2019: Lithium ion battery thermal management system using optimized fuzzy controller. *2019 IEEE industry applications society annual meeting*, IEEE, 1–9.

He, L., Z. Gu, Y. Zhang, H. Jing, and P. Li, 2023: Review on thermal management of lithium-ion batteries for electric vehicles: advances, challenges, and outlook. *Energy & Fuels*, **37 (7)**, 4835–4857.

Ho, J. S., B. C. Tan, T. C. Lau, and N. Khan, 2023: Public acceptance towards emerging autonomous vehicle technology: A bibliometric research. *Sustainability*, **15 (2)**, 1566.

House, W., 2021: White house us. president biden's bipartisan infrastructure law. https://www.whitehouse.gov/build/.

Huang, G., P. Zhao, and G. Zhang, 2022: Real-time battery thermal management for electric vehicles based on deep reinforcement learning. *IEEE Internet of Things Journal*, **9 (15)**, 14 060–14 072.

Huang, J., Z. He, Y. Arakawa, and B. Dawton, 2023a: Trajectory planning in frenet frame via multi-objective optimization.

Huang, Z., H. Liu, J. Wu, and C. Lv, 2023b: Conditional predictive behavior planning with inverse reinforcement learning for human-like autonomous driving.

Huang, Z., J. Wu, and C. Lv, 2021: Driving behavior modeling using naturalistic human driving data with inverse reinforcement learning. **23 (8)**, 10 239–10 251.

Iberraken, D., L. Adouane, and D. Denis, 2019: Reliable risk management for autonomous vehicles based on sequential bayesian decision networks and dynamic inter-vehicular assessment. *2019 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2344–2351.

Lattarulo, R., and J. Pérez Rastelli, 2021: A hybrid planning approach based on mpc

and parametric curves for overtaking maneuvers. *Sensors*, **21 (2)**, 595.

Li, Z., and W. A. Xu, 2019: Path decision modelling for passengers in the urban rail transit hub under the guidance of traffic signs. *J. Ambient Intell. Humaniz. Comput.*, **10**, 365–372.

Lin, X., H. E. Perez, S. Mohan, J. B. Siegel, A. G. Stefanopoulou, Y. Ding, and M. P. Castanier, 2014: A lumped-parameter electro-thermal model for cylindrical batteries. *Journal of Power Sources*, **257**, 1–11.

Liu, C., S. Lee, S. Varnhagen, and H. E. Tseng, 2017: Path planning for autonomous vehicles using model predictive control. *2017 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 174–179.

Lopez-Sanz, J., C. Ocampo-Martinez, J. Alvarez-Florez, M. Moreno-Eguilaz, R. Ruiz-Mansilla, J. Kalmus, M. Gräeber, and G. Lux, 2016: Nonlinear model predictive control for thermal management in plug-in hybrid electric vehicles. *IEEE Transactions on Vehicular Technology*, **66 (5)**, 3632–3644.

Michelbacher, C. J., and Coauthors, 2017: Enabling fast charging: A technology gap assessment. Tech. rep., Idaho National Lab.(INL), Idaho Falls, ID (United States).

Ming, Y., Y. Li, Z. Zhang, and W. Yan, 2021: A survey of path planning algorithms for autonomous vehicles. *SAE international journal of commercial vehicles*, **14 (02-14-01-0007)**, 97–109.

Mo, X., and C. Lv, 2023: Predictive neural motion planner for autonomous driving using graph networks. **8 (2)**, 1983–1993.

Morris, C., 2018: Comparing the bolt and model 3 battery packs. https://chargedevs.com/newswire/comparingthe-bolt-and-model-3-battery-packs/.

Munro, 2021: Mach-e: Battery tray and battery cell features. https://www.youtube.com/watch?v=4vRjo0gaG1g.

Neely, M., 2010: *Stochastic network optimization with application to communication and queueing systems*. Springer Nature.

Patel, J. R., and M. K. Rathod, 2020: Recent developments in the passive and hybrid thermal management techniques of lithium-ion batteries. *Journal of power sources*, **480**, 228 820.

Reiter, R., A. Nurkanović, J. Frey, and M. Diehl, 2023: Frenet-cartesian model repre-

sentations for automotive obstacle avoidance within nonlinear mpc. *European Journal of Control*, 100847.

Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, 2017: Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shan, S., and Coauthors, 2023: Numerical investigation of a compact and lightweight thermal management system with axially mounted cooling tubes for cylindrical lithium-ion battery module. *Energy*, **274**, 127 410.

Su, S., W. Li, Y. Li, A. Garg, L. Gao, and Q. Zhou, 2021: Multi-objective design optimization of battery thermal management system for electric vehicles. *Applied Thermal Engineering*, **196**, 117 235.

Sundararaju, K., P. Priyanka, T. Revanth, K. Niviss, and S. Sabarish, 2023: Optimizing the thermal operation of lithium-ion battery packs using cell to cell variation. *2023 7th International Conference on Trends in Electronics and Informatics (ICOEI)*, IEEE, 176–180.

Tan, X., P. Lyu, Y. Fan, J. Rao, and K. Ouyang, 2021: Numerical investigation of the direct liquid cooling of a fast-charging lithium-ion battery pack in hydrofluoroether. *Applied Thermal Engineering*, **196**, 117 279.

Teng, S., and Coauthors, 2023: Motion planning for autonomous driving: The state of the art and future perspectives.

Tomaszewska, A., and Coauthors, 2019: Lithium-ion battery fast charging: A review. *ETransportation*, **1**, 100 011.

Udhayakumar, K., F. A. Rihan, K. Janani, and R. Rakkiyappan, 2022: Novel finite and fixed-time stability theorems for fractional-order impulsive discontinuous systems and their application to multi-agent systems. *Results in Control and Optimization*, **9**, 100 173.

Wan, S., J. Lu, P. Fan, and K. B. Letaief, 2019: Toward big data processing in iot: Path planning and resource management of uav base stations in mobile-edge computing system. **7 (7)**, 5995–6009.

Wang, H., Y. Ye, J. Zhang, and B. Xu, 2023: A comparative study of 13 deep reinforcement learning based energy management methods for a hybrid electric vehicle. *Energy*, **266**, 126 497.

Wang, M., H. Shan, R. Lu, R. Zhang, X. Shen, and F. Bai, 2014: Real-time path

planning based on hybrid-vanet-enhanced transportation system. **64 (5)**, 1664–1678.

Wang, X., H. Yu, and J. Liu, 2022a: Research on local path planning algorithm based on frenet coordinate system. *2022 4th International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, IEEE, 249–252.

Wang, Z., J. Zhan, C. Duan, X. Guan, P. Lu, and K. Yang, 2022b: A review of vehicle detection techniques for intelligent vehicles.

Wei, X., X. Han, B. Zhu, J. Li, and F. Yang, 2021: A double-layer collaborative apportionment method for personalized and balanced routing. *Peer-to-Peer Networking and Applications*, **14 (5)**, 3349–3359.

Werling, M., J. Ziegler, S. Kammel, and S. Thrun, 2010: Optimal trajectory generation for dynamic street scenarios in a frenet frame. *2010 IEEE international conference on robotics and automation*, IEEE, 987–993.

Wu, W., S. Wang, W. Wu, K. Chen, S. Hong, and Y. Lai, 2019: A critical review of battery thermal performance and liquid based battery thermal management. *Energy conversion and management*, **182**, 262–281.

Xia, Z., X. Ma, D. Qiu, G. Bu, Y. Xia, B. Zhao, Z. Lin, and Y. Shi, 2018: A design of battery thermal management system based on fuzzy control. *2018 International Conference on Computer Modeling, Simulation and Algorithm (CMSA 2018)*, Atlantis Press, 92–94.

Xing, X., B. Zhao, C. Han, D. Ren, and H. Xia, 2022: Vehicle motion planning with joint cartesian-frenét mpc. **7 (4)**, 10 738–10 745.

Xu, B., and Z. Arjmandzadeh, 2023: Parametric study on thermal management system for the range of full (tesla model s)/compact-size (tesla model 3) electric vehicles. *Energy Conversion and Management*, **278**, 116 753.

Xu, Z., J. Xu, Z. Guo, H. Wang, Z. Sun, and X. Mei, 2022: Design and optimization of a novel microchannel battery thermal management system based on digital twin. *Energies*, **15 (4)**, 1421.

Yan, W., M.-J. Li, N. Mei, C.-Y. Qu, Y. Wang, and L.-P. Liu, 2023: Control strategy research of electric vehicle thermal management system based on mga-svr algorithm. *Measurement and Control*, **56 (5-6)**, 1026–1036.

Zhang, Z., L. Zheng, Y. Li, P. Zeng, and Y. Liang, 2021: Structured road-oriented motion planning and tracking framework for active collision avoidance of autonomous

vehicles. *Science China Technological Sciences*, **64 (11)**, 2427–2440.

Zhao, Y., D. Dan, S. Zheng, M. Wei, and Y. Xie, 2023: A two-stage eco-cooling control strategy for electric vehicle thermal management system considering multi-source information fusion. *Energy*, **267**, 126 606.

Zuo, Z., X. Yang, Z. Li, Y. Wang, Q. Han, L. Wang, and X. Luo, 2020: Mpc-based cooperative control strategy of path planning and trajectory tracking for intelligent vehicles. **6 (3)**, 513–522.