Learning Agent-based Model Predictive Controllers for Holistic Vehicle Control

by

Jiaming Zhong

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2025

© Jiaming Zhong 2025

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Member:	Walter Lucia Associate Professor, Concordia Institute for Information Systems Engineering, Concordia University
Supervisor(s):	Amir Khajepour Professor, Department of Mechanical and Mechatronics Engineering, University of Waterloo
	Yash Vardhan Pant Assistant Professor, Department of Electrical and Computer Engineering, University of Waterloo
Internal Member:	William Melek Professor, Department of Mechanical and Mechatronics Engineering, University of Waterloo
Internal-External Member:	Nasser Lashgarian Azad Professor, Department of Systems Design Engineering, University of Waterloo
	Gennaro Notomista Assistant Professor, Department of Electrical and Computer Engineering, University of Waterloo

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Holistic vehicle control (HVC) is an advanced, integrated approach that optimizes a vehicle's mobility, stability, and safety by coordinating all available control systems. As the automotive industry progresses towards electrification and increasing intelligence, functional integration has emerged as a dominant trend in vehicle control systems. This evolution necessitates the simultaneous coordination of multiple controllers to achieve diverse objectives. The growing demand for flexibility and reliability in automotive systems has given rise to a "plug-and-play" paradigm in control system design. This approach, while beneficial, poses significant challenges for traditional "all-in-one" integrated control methods, such as integrated model predictive control (MPC).

Distributed control schemes have demonstrated greater scalability and robustness compared to integrated schemes, especially in applications such as vehicle control systems, where managing complex, multi-agent dynamics is important. A recently proposed prominent approach within this framework is agent-based model predictive control (AMPC), where controllers are treated as interactive agents and are coordinated together to achieve a common objective iteratively, taking advantage of the distributed control structure. However, in practice, two critical pain points arise:

a) *Uncertain contributions from unknown controllers*: The optimal control performance from the AMPC highly depends on the prediction accuracy, which requires all agents or their contributions to be accurately known. This requirement is often too idealistic for practical implementation. For example, when a third party develops a "black-box" controller, its underlying algorithm is unknown, and there is no specified interface to know its contribution to vehicle dynamics. This lack of information is likely to cause a significant error in predicting vehicle behaviour, leading to unexpected or even harmful control results.

b) *Limitations on controller-oriented decomposition*: Decomposing from the objective's perspective is a more practical and ideal approach in the development of function-oriented or feature-oriented automotive control systems. However, AMPC cannot decompose coupled objectives with shared agents because it is designed only to decompose the integrated system from the controller's perspective, not from the objectives. The objective in AMPC is usually implemented as a weighted sum of multiple goals, which greatly limits the flexibility of control system design.

This thesis is hence motivated to overcome these two pain points through **practical solutions using data-driven machine-learning techniques and flexible distributed schemes for multi-agent-multi-objective (MAMO) control systems.**

For the first pain point, this thesis proposes a practical hybrid control scheme: learning agent-based MPC (L-AMPC). This scheme combines the model-based AMPC approach with data-driven learning methods to improve the control performance for multi-agent systems. The Gaussian process regression (GPR) enhanced by an online data management strategy serves as the learning core to predict unknown contributions along the prediction horizon, completing the system model in the MPC for more accurate control. Meanwhile, a stochastic framework is formulated to guarantee control safety and feasibility using soft chance constraints based on the prediction variance. The proposed hybrid control scheme is efficient for real-time implementation and is flexible to any control agent topology.

For the second pain point, this thesis proposes a distributed control scheme: multi-objective AMPC (MO-AMPC). This scheme adapts the alternating direction method of multipliers (ADMM) into a general control strategy that achieves global optimization while decoupling objectives. Three formulations that can maintain convergence while addressing control regularization and inequality constraints are systematically developed. The convergence and computational efficiency of the proposed methods are verified and compared on two vehicle control scenarios with multi-objective configurations.

Furthermore, this thesis proposes a data-driven distributed control scheme based on the MO-AMPC: learning multi-objective AMPC (L-MO-AMPC). To accelerate the converging process of MO-AMPC, a learning-based initialization method for iterations is proposed. The proposed scheme is compared with the MO-AMPC scheme through a path-tracking simulation using various controllers. The results show that the L-MO-AMPC scheme achieves similar control performance while significantly reducing computational time.

All proposed controllers (L-AMPC, MO-AMPC and L-MO-AMPC) are verified by real-time simulations respectively. In addition, the effectiveness and performance of L-AMPC and MO-AMPC are also verified in real vehicle experiments. The results from simulations and experiments could be concluded as follows:

- Compared to AMPC, the proposed L-AMPC can achieve higher tracking performance in well-learned scenarios with the learning capability and always guarantee constraint satisfaction even in less-learned scenarios.
- To decompose the MAMO system from the objective's perspective, the proposed MO-AMPC achieves the same global optimum as the corresponding integrated MPC with greater flexibility and can potentially reduce the computational cost.
- Compared to model-based MO-AMPC, the proposed L-MO-AMPC has significantly higher computational efficiency while still retaining the property of converging to the global optimum, making it well-suited for real-time implementation.

Acknowledgements

Completing my PhD has been a richly rewarding journey, one that brought not just knowledge but also kindness and support from many quarters. This thesis represents not just my own efforts but the collective spirit of all those I've had the privilege to work alongside. The insights and support from my peers and mentors have been invaluable. Thank you to everyone who contributed to this journey, for both the rigorous debates and the enduring friendships formed along the way.

First and foremost, I would like to thank my supervisors, Prof. Amir Khajepour and Prof. Yash Vardhan Pant, for their indispensable roles in my doctoral journey. Prof. Khajepour, who first saw potential in me and welcomed me into the world of academic research, has been nothing short of a guiding light, illuminating the path of my scholarly pursuits with wisdom and encouragement. Prof. Pant, with his innovative thinking and fresh academic perspectives, has continually inspired me to explore new vistas and challenge the conventional boundaries of our field. Together, they have not only shaped my academic capabilities but also profoundly influenced my approach to research.

I extend my profound gratitude to Prof. Mohammad Pirani for his invaluable advice on my research, which has significantly enriched the depth and quality of my work. I am particularly indebted to Prof. Jay Wang for his tremendous support throughout my career. I also appreciate my committee members, Prof. Walter Lucia, Prof. William Melek, Prof. Nasser Lashgarian Azad, and Prof. Gennaro Notomista, for their time and insights in reviewing my thesis. I am eternally grateful for their suggestions, comments, inspiration, and encouragement.

Completing this research would not have been possible without the support of the remarkable fellows at the University of Waterloo. I sincerely appreciate the indispensable help from the technicians, Jeff Graansma, Michael Duthie, Aaron Sherratt, and Ruslan Liang, whose expertise and assistance were crucial in carrying out experimental work and ensuring the smooth operation of the Mechatronic Vehicle Systems (MVS) lab. In addition, I would like to thank the post-doctoral fellows Dr. Chen Sun, Dr. Yukun Lu, Dr. Yaodong Cui, Dr. Zejian Deng, Dr. Reza Valiollahi Mehrizi, Dr. Keqi Shu, Dr. Chao Yu, and Dr. Ladan Khoshnevisan, for their timely assistance. I am also grateful for the guidance of Shucheng Huang, Frank Yang, Minghao Ning, Pouya Panahandeh, Dr. Ruihe Zhang, Dr. Mehdi Zabihi and Alireza Esfandbod. Throughout my PhD study, I am profoundly grateful to my labmates Jiaming Guo, Changye Ma, Steven Tuer, Leo Zheng, and Tony Zhao. I would also like to thank my friends from Control, Learning and Logic (CL2) Lab, Prajwal Thakur and Leroy D'Souza, for their help and advice. I would also like to thank visiting researchers Tao Huang and Yang Zhang for their help in my study and life.

Special thanks to LoopX members Dr. Chao Yu, Dr. Yaodong Cui, Jinwei Zhang, Shucheng Huang, Teddy Xu, Hamlin Zheng, and Zhenan Liu. Although not a part of my thesis, I learned a lot from every one of them. Reflecting on the unforgettable days spent working alongside my incredible team on various projects, I am deeply moved by the memories of our collective pursuit of innovation for our startup. Each member of our team has imparted invaluable experiences and exemplary traits that have enriched my life immensely. They are not just my colleagues but my dear friends and role models. I am endlessly grateful for their influence and cherish the journey we have shared.

I would also like to acknowledge the financial support of the Automotive Partnership, Canada and Ontario Research Fund, and General Motors.

Finally, I would like to thank my parents, Haiyan Liu and Junmin Zhong, whose immense hard work, selfless dedication, and utmost sacrifice gave me the opportunity to pursue the journey of life. You are my life coaches, guiding lights, therapists, and the greatest comedians throughout everything.

Jiaming Zhong

Waterloo, Canada

October, 2025

Dedication

To my love, Yueyao Zhang, who makes me whole.

Table of Contents

Examining Committee ii

Author's Declaration iii

Abstract iv

Acknowledgements vi

Dedication viii

List of Figures xiv

List of Tables xxi

List of Abbreviations xxii

List of Symbols xxvi

1 Introduction 1

1.1 Motivation and Objectives 1

1.2 Contributions 3

1.3 Thesis Outline 5

2 Literature Review **8**
2.1 Holistic Vehicle Control . . . 8
2.1.1 Vehicle Stability Control . . . 8
2.1.2 Autonomous Vehicle Motion Control . . . 10
2.2 Model Predictive Control for Vehicle Control . . . 11
2.2.1 Linear, Nonlinear and Hybrid Model Predictive Control . . . 12
2.2.2 Robust and Stochastic Model Predictive Control . . . 13
2.3 Distributed Model Predictive Control . . . 15
2.4 Learning-Based Model Predictive Control . . . 16
2.5 Summary and Discussion . . . 18

3 Background **20**
3.1 Multi-Agent-Multi-Objective Control System . . . 20
3.2 Agent-Based Model Predictive Control . . . 22
3.3 Alternating Direction Method of Multipliers . . . 25
3.4 Gaussian Process Regression . . . 28
3.5 Chance Constraint . . . 31
3.6 Summary . . . 31

4 Learning Agent-Based Model Predictive Control **32**
4.1 Overview . . . 32
4.1.1 System with Unknown Agent . . . 32
4.1.2 Control Scheme Overview . . . 33
4.2 Data-Driven Learning Process . . . 35
4.2.1 Data Preparation . . . 35
4.2.2 Data Management . . . 37
4.2.3 Learning Process of Gaussian Process Regression . . . 39
4.3 Model-Based Soft Stochastic Control Scheme . . . 41

4.3.1 Contribution Compensation and Uncertainty Propagation 41
4.3.2 Soft Stochastic Control Scheme 43
4.4 Example: Yaw Rate Tracking 45
4.4.1 Agent Configuration 45
4.4.2 Desired Vehicle Response and Constraints 47
4.4.3 Model Selection and Training 48
4.4.4 Learning Process 52
4.4.5 Controller Design 53
4.5 Simulation Results 56
4.5.1 Less-Learned Scenario: Safety Guarantee 57
4.5.2 Well-Learned Scenario: Performance Improvement 57
4.6 Summary 63

5 Multi-Objective Agent-Based Model Predictive Control 66
5.1 Overview 66
5.1.1 Integrated Model Predictive Control 66
5.1.2 Control Scheme Overview 68
5.2 Formulations 70
5.2.1 Formulation 1: Fully Distributed 71
5.2.2 Formulation 2: Virtual Center Node 73
5.2.3 Formulation 3: Local Explicit 74
5.3 Convergence and Parameter Selection 77
5.4 Example: Cruise Control 78
5.4.1 Agent Configuration and Controller Design 78
5.4.2 Simulation Results 80
5.5 Example: Holistic Vehicle Stability Control 83
5.5.1 Agent Configuration and Controller Design 83
5.5.2 Simulation Results 85
5.6 Summary 89

6 Learning Multi-Objective Agent-Based Model Predictive Control **91**
6.1 Overview . . . 91
6.2 Algorithm Design . . . 92
6.2.1 Negotiation between Objectives . . . 94
6.2.2 Learning and Post-Processing . . . 95
6.3 Example: Autonomous Path-Tracking Control . . . 95
6.4 Simulation Results . . . 98
6.4.1 Convergence . . . 98
6.4.2 Improvements by Learning . . . 99
6.4.3 Limited Computational Resource . . . 101
6.5 Summary . . . 104

7 Experimental Study **105**
7.1 Experiment Setup . . . 105
7.2 Yaw Stability Control using L-AMPC . . . 106
7.2.1 Test 1: SineWaves . . . 107
7.2.2 Test 2: DLCs . . . 109
7.2.3 Multi-step GPR Prediction . . . 111
7.3 Cruise Control using MO-AMPC . . . 113
7.3.1 Test 1: Aggressive Lane Change . . . 113
7.3.2 Test 2: Normal Cruise Driving . . . 116
7.4 Holistic Stability Control using MO-AMPC . . . 117
7.4.1 Test 1: Straight Launch and Brake . . . 118
7.4.2 Test 2: Double Lane Change in Acceleration . . . 122
7.5 Summary and Discussion . . . 124

8 Conclusion and Future Work **126**
8.1 Conclusions . . . 126
8.2 Limitations . . . 130
8.3 Future Work . . . 131

Letter of Copyright Permission **134**

References **135**

APPENDICES **148**

A Non-parametric Kernel-based Regression **149**

A.1 Formulation of Kernel-based Regression 149

A.2 Model Selection for Regression 151

A.3 GPR Training Process: Hyper-parameter Optimization 152

B Convergence Discussion of MO-AMPC **156**

B.1 Proof of Convergence 156

B.2 Optimal Parameter Selection 158

C Parameters of the Chevrolet Equinox Electric Vehicle **160**

List of Figures

1.1 Road map of the research 5

3.1 Example of a MAMO control system with five agents and two objectives. The arrow from $A_{(i)}$ to O_j represents $G_{j(i)} = 1$. For objectives: $S_1 = \{1, 2, 3, 4\}$, $S_2 = \{3, 4, 5\}$. $U_1 = [U_{(1)}^T, U_{(2)}^T, U_{(3)}^T, U_{(4)}^T]^T$, $U_2 = [U_{(3)}^T, U_{(4)}^T, U_{(5)}^T]^T$. For agents: $S_{(1)} = \{1\}$, $S_{(2)} = \{1\}$, $S_{(3)} = \{1, 2\}$, $S_{(4)} = \{1, 2\}$, $S_{(5)} = \{2\}$. . 21

3.2 Example of a multi-agent control system with three agents and one common objective. All the agents contribute to the overall net forces and moments which affect the common objective. In this example, $U = U_1 = [U_{(1)}^T, U_{(2)}^T, U_{(3)}^T, U_{(4)}^T]^T$. 22

3.3 The overall structure of the AMPC. Each agent has its local optimization module. The common objective is evenly decomposed into each local cost function. Local optimizations exchange their contributions through an information bus. The AMPC will output the optimal control action as long as the consensus check is passed. 24

4.1 Example of a multi-agent control system with different types of agents. All the agents contribute to the overall net forces and moments which affect the common objective. In this example, agent $A_{(1)}$ and $A_{(2)}$ are controllable agents controlled by L-AMPC controllers. Agent $A_{(3)}$ is a "white-box" agent. Agent $A_{(4)}$ is a "black-box" agent. 34

4.2 The overall structure of the L-AMPC. The $\Delta\hat{y}_{CG(\mathrm{UA})} \in \mathbb{R}^{N_p \times 1}$ is the unknown contribution estimated as the output of the data point vector along the horizon from Equation 4.2. The $x_{norm} \in \mathbb{R}^{n_d \times 1}$ is the normalized data input where n_d is the number of selected model variables based on the off-line dataset described in Section 4.2.1. D_{off}, D_{on}, and D_{sub} denote the off-line, on-line, and the selected (sub-)datasets, respectively. $\boldsymbol{\theta}$ is the hyper-parameters of the GP model that trained on the off-line dataset. $\hat{y}_* \in \mathbb{R}^{N_p}$ and $\Sigma_* \in \mathbb{R}^1$ are the GPR prediction mean values along the horizon and the variance, respectively. 35

4.3 Data management strategies. (a)-(c) are data collection strategies: (a) Add the new point; (b) Replace the farthest point with the new point; (C) Replace the nearest point with the new point. (d)-(e) are subset selection strategies: (d) Select points in the bounding box; (e) Select points in the bounding sphere; (f) Select the nearest K neighbour points. 38

4.4 Schematic diagram of data collection for multi-step prediction. 40

4.5 Different prediction methods along the horizon. 41

4.6 PMSE results for model selection for the yaw stability tracking example using L-AMPC. 49

4.7 R^2_{adj} and AIC results for model selection for the yaw stability tracking example. 50

4.8 PMSE comparison of GPR predictions using different training methods. . . 51

4.9 Results of GPR prediction and data collection in L-AMPC. 52

4.10 Comparison of yaw rate prediction and tracking performance. 54

4.11 Diagram of the shrinking of state constraints and control constraints due to the propagation of learning uncertainty. Green is the original constraint. Black or gray is the pruned constraint at each step in the horizon. Red is the distribution of uncertainty of the state or constraint propagated at each step in the horizon. 55

4.12 Simulation results of L-AMPC for a DLC in the less-learned scenario. . . . 58

4.13 Optimization results of L-AMPC in the less-learned DLC at T=5.75 and T=10.80. 59

4.14 Overall simulation results of L-AMPC in Scenario A1.: Maneuvers change. 61

4.15 Overall simulation results of L-AMPC in Scenario A2.: Friction change. . . 62

4.16 Overall simulation results of L-AMPC in Scenario B.: Two controllable agents. 63
4.17 Detailed simulation results of L-AMPC in Scenario B. 64
4.18 Overall simulation results of L-AMPC in Scenario C.: Two black-box agents. 65
4.19 Detailed simulation results of L-AMPC in Scenario C. 65

5.1 Example of decomposing a MAMO control system (dashed line box) from the objective's (feature's) perspective. In this example, agent $A_{(3)}$ and $A_{(4)}$ are shared agents. After the system is decomposed, each local objective will be solved on its own agent group and exchange information with others (solid gray boxes). 68
5.2 Example of the MAMO control system for cruise control. The two objectives share the FDB agent. The STR agent is the external input on both objectives, so its contributions are drawn as dashed arrows. Thus, $S_{\mathrm{YRT}} = \{\mathrm{RTV}, \mathrm{FDB}\}$, $S_{\mathrm{LST}} = \{\mathrm{DTR}, \mathrm{FDB}\}$. For agents: $S_{(\mathrm{DTR})} = \{\mathrm{LST}\}$, $S_{(\mathrm{FDB})} = \{\mathrm{YRT}, \mathrm{LST}\}$, $S_{(\mathrm{RTV})} = \{\mathrm{YRT}\}$. 79
5.3 Simulation results of the vehicle response and control actions in the cruise control example using MO-AMPC. Figures (a) and (b) show the longitudinal speed and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC. 81
5.4 Comparison results between three formulations: Form 1 (*Fully Distributed*), Form 2 (*Virtual Center Node*, and Form 3 (*Local Explicit*). The suboptimality is the absolute error of the torque control actions between the MO-AMPC and the integrated MPC. The formulation of MO-AMPC is said to be more computationally efficient than the integrated MPC if the time consumption ratio is less than 1. 83
5.5 Example of the MAMO control system for holistic vehicle stability control. The two objectives share the RTV agent. The STR and DTR agents are the external input on each objective, respectively, so their contributions are drawn as dashed arrows. Thus, $S_{\mathrm{VSC}} = \{\mathrm{RTV}, \mathrm{FDB}\}$, $S_{\mathrm{WSC}} = \{\mathrm{TCR}, \mathrm{RTV}\}$. For agents: $S_{(\mathrm{FDB})} = \{\mathrm{VSC}\}$, $S_{(\mathrm{RTV})} = \{\mathrm{VSC}, \mathrm{WSC}\}$, $S_{(\mathrm{TCR})} = \{\mathrm{WSC}\}$. 84

5.6 Simulation results of the vehicle response and control actions in the holistic stability control example using MO-AMPC, conducting a DLC in acceleration on a high friction surface. Figures (a) and (b) show the wheel speed control and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC. 86

5.7 Simulation results of the vehicle response and control actions in the holistic stability control example using MO-AMPC, conducting a DLC in acceleration on a low friction surface. Figures (a) and (b) show the wheel speed control and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC. 88

5.8 Simulation results of the vehicle response and control actions in the holistic stability control example using MO-AMPC, conducting a straight acceleration on a split-mu surface. Figures (a) and (b) show the wheel speed control and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC. 89

6.1 Example of a MAMO control system with three agents and two objectives. Figure (a) is the structure of the system where agent $A_{(2)}$ is the shared agent. Figure (b) shows the converging process of control action $U_{(2)}$ with $n_{u(2)} = 2$. The blue and green solid lines represent the iterative trajectory of $U_{(2)}$ from MO-AMPC, respectively. The dashed lines represent the iterative trajectory from L-MO-AMPC. 92

6.2 The overall structure of the L-MO-AMPC. Each objective has its local optimization module, initialized by a learning prediction module after a reliability authentication. 93

6.3 Vehicle and path tracking model. 96

6.4 Example of the MAMO control system for path-tracking. The two objectives share the FDB agent. For objective: $S_{\text{LST}} = \{\text{DTR}, \text{FDB}\}$, $S_{\text{LPT}} = \{\text{STR}, \text{RTV}, \text{FDB}\}$. For agents: $S_{(\text{DTR})} = \{\text{LST}\}$, $S_{(\text{STR})} = \{\text{LPT}\}$, $S_{(\text{FDB})} = \{\text{LST}, \text{LPT}\}$, $S_{(\text{RTV})} = \{\text{LPT}\}$. 97

6.5 Real-time simulation of the combined path-tracking on a racing track. (a) Global reference path. (b) Reference speed profile. (c) Visualization in CarSim. (d) Control results. (c) and (d) are at station $\approx 1900m$ and marked as red dots in (a) and (b). A comparison of the converging processes at this position is shown in Section 6.4.2. 99

6.6 Comparison of control actions between the proposed L-MO-AMPC scheme and the integrated scheme. 100

6.7 Data collection and learning process. A comparison shows the improvement in computational efficiency. 101

6.8 Detailed converging processes of the shared FDB agent at the station $\approx 1900m$ on the racing track, which marked as a red dot in Figure 6.5. 102

6.9 Comparison results of three laps under limited maximum iterations. The speed tracking was improved in the last two laps. 103

6.10 Six slices of the comparison results under limited maximum iterations. . . . 103

7.1 Test vehicle with independent wheel control actuators. Distributed controllers, such as torque-vectoring and differential braking, are coordinated as connected agents to achieve multiple objectives. 106

7.2 Experiment of L-AMPC: learning process of yaw stability control in Sinewaves. 107

7.3 Experiment of L-AMPC: Comparison of yaw stability control in Sinewaves. 108

7.4 Experiment of L-AMPC: MSE of yaw stability control in Sinewaves. 109

7.5 Experiment of L-AMPC: Comparison of yaw stability control in DLCs. . . 110

7.6 Experiment of L-AMPC: MSE of yaw stability control in Sinewaves. 111

7.7 Experiment of L-AMPC: Multi-step GPR predictions. 112

7.8 Experiment of MO-AMPC: MAMO configuration for test of cruise control. 113

7.9 Experiment of MO-AMPC: Comparison of the vehicle response for the aggressive lane change in the example of cruise control. The over-steering was significantly suppressed when the objective of YRT was enabled. The yaw stability was improved. 114

7.10 Experiment of MO-AMPC: Wheel torque commands from the MO-AMPC during the aggressive lane change in cruise control. The *Local Explicit* formulation with optimal parameters was adopted. All control agents have cooperated: drive torque of rear-wheel-driving (DTR) generated the driving and braking torque on the rear wheels; rear torque vectoring (RTV) generated differential torque on the rear wheels; front differential braking (FDB) generated independent braking torques on the front wheels. 115

7.11 Experiment of MO-AMPC: Results of the convergence and computational efficiency during the aggressive lane change in cruise control: (a) torque suboptimality (b) number of iterations (c) time consumption ratio. "Whole": the whole process; "Focus": the period marked in Figure 7.9 where both objectives require significant control. 115

7.12 Experiment of MO-AMPC: Testing ground of the cruise control in normal cruise driving. 116

7.13 Experiment of MO-AMPC: Results of the cruise control in normal cruise driving. (a) speed tracking results. (b) yaw rate tracking results. (c) number of iterations. (d) comparison of time consumption between the MO-AMPC and the integrated MPC. 117

7.14 Experiment of MO-AMPC: MAMO configuration for test of holistic vehicle stability control. 118

7.15 Experiment of MO-AMPC: Comparison of the vehicle response for straight launch and braking in the example of holistic stability control. The oversteering was significantly suppressed, and no obvious wheel slip occurred during the second test. The holistic stability was improved when both VSC and WSC were enabled. 119

7.16 Experiment of MO-AMPC: Torque commands from the MO-AMPC using the *Local Explicit* formulation with optimal parameters for straight launch and braking in the example of holistic stability control. All control agents cooperated: FDB generated independent braking torques on the front wheels; the drive's torque was first reduced by TCR and then adjusted by RTV on the rear wheels. 120

7.17 Experiment of MO-AMPC: comparison of the control action between the proposed MO-AMPC and the integrated MPC for straight launch and braking in the example of holistic stability control. Red lines: control action from the integrated MPC; green lines: control action from the proposed MO-AMPC. 121

7.18 Experiment of MO-AMPC: Results of the convergence and computational efficiency for straight launch and braking in the example of holistic stability control. "Focus" represents the specific period marked in Figure 7.15 where both objectives require significant control. 122

7.19 Experiment of MO-AMPC: Comparison of the vehicle response for DLC acceleration in the example of holistic stability control. The over-steering was significantly suppressed, and no obvious wheel slip occurred during the second test. The holistic stability was improved when both VSC and WSC were enabled. 123

7.20 Experiment of MO-AMPC: Results of the convergence and computational efficiency for DLC acceleration in the example of holistic stability control. "Focus" represents the specific period marked in Figure 7.19 where both objectives require significant control. 124

C.1 Test vehicle: the Chevrolet Equinox electric vehicle. 160

C.2 Equinox CarSim model verification under a DLC maneuver. 161

List of Tables

4.1 Agent Type and Definition 33
4.2 Agent Configuration for Yaw Stability Control Example 46
4.3 Hyper-Parameter Training Results for GPR in L-AMPC 51
4.4 Comparison of Time Consumption of Modules 53
4.5 Controller Settings of L-AMPC for Yaw Stability Control 56
4.6 Simulation Configurations using L-AMPC in Well-Learned Scenarios 60

5.1 Formulation Comparison of MO-AMPC 77

6.1 Quantitative Convergence Accuracy of L-MO-AMPC 99

7.1 Experimental Configurations using L-AMPC in Well-Learned Scenarios . . 106
7.2 MSE improvement of yaw stability control in Sinewaves. 110
7.3 MSE improvement of yaw stability control in DLCs. 111
7.4 Experimental Configurations using MO-AMPC in Cruise Control 113
7.5 Experimental Configurations using MO-AMPC in Holistic Stability Control 118

C.1 Specification of the Test Vehicle 161

List of Abbreviations

fl the front-left wheel 46

fl the rear-left wheel 46

fr the front-right wheel 46

rr the rear-right wheel 46

ADMM alternating direction method of multipliers 3, 4, 6, 20, 25–28, 31, 66, 68, 69, 77, 89, 128

AFS active front steering 60, 62

AIC Akaike information criterion 29, 36, 49, 50, 151, 152

AMPC agent-based model predictive control 2, 3, 5, 6, 15, 16, 18–20, 22–25, 31–34, 41, 53, 56, 57, 60, 61, 64, 66, 109, 126–128, 130

AS active steering 9, 12

AV autonomous vehicle 10, 11

BIC Bayesian information criterion 29, 36, 151, 152

CBF control barrier function 131

CG center of gravity 10, 22, 23, 33, 35, 36, 41, 42, 45, 67, 96, 127

CP conformal prediction 131

CV cross-validation 154

DB differential braking 9, 12

DLC double lane change 6, 49–53, 57, 60, 61, 80, 85, 87, 109, 111, 113, 122, 161

DMPC distributed model predictive control 3, 5, 8, 15, 18

DT drive torques 45, 46, 109

DTR drive torque of rear-wheel-driving xix, 78, 83, 97, 115

FDB front differential braking xix, 78, 87, 97, 100, 115, 120, 129

FDT front differential torque 45–48, 50, 57, 62, 109

GCV generalized cross-validation 151

GNSS global navigation satellite system 105, 160

GP Gaussian process 17, 18, 29, 39, 44, 64, 92

GPR Gaussian process regression 3, 6, 14, 17, 18, 20, 29–31, 33, 34, 36, 38, 39, 41–43, 49–53, 56, 57, 60–64, 94, 95, 98, 99, 104, 106, 111, 112, 127, 131, 154

HMPC hybrid model predictive control 13

HVC holistic vehicle control 1, 3–5, 8, 9, 15, 18, 32, 66, 91, 126

IMU inertial measurement unit 105, 160

KA known agent 33

KNN k-nearest neighbors 37–39, 51, 95, 107

L-AMPC learning agent-based model predictive control 3–6, 32, 33, 36, 37, 39, 43–45, 50, 53, 56, 57, 60–64, 105, 106, 109–111, 124, 126–131

L-MO-AMPC learning multi-objective agent-based model predictive control 4–6, 91, 92, 94, 95, 98, 100, 104, 129, 132, 133

LLM large language model 131

LMPC learning-based model predictive control 2, 5, 8, 16–19, 44

LOOCV leave-one-out cross-validation 151, 152

LPT lateral path tracking 97

LQR linear quadratic regulators 11

LST longitudinal speed tracking 78, 97, 113

MAMO multi-agent-multi-objective 2–6, 16, 20, 21, 31, 66, 68, 70, 89–92, 113, 116, 118, 126, 128, 132

MO-AMPC multi-objective agent-based model predictive control 4–7, 31, 66, 68–70, 78, 80–82, 85, 87–94, 100, 104, 105, 113–117, 121–125, 128–130, 132

MPC model predictive control 1, 2, 4–8, 10–19, 22, 23, 25, 31, 33, 34, 36, 39, 43, 44, 64, 66, 67, 77, 81, 82, 85, 87–91, 98, 100, 114, 116, 117, 121–123, 125, 127–129, 131

MSE mean squared error 109, 111

NC normal constrain 57

NMPC nonlinear model predictive control 12–14, 17, 22, 67

PCC Pearson correlation coefficient 37, 49

PI proportional integral 46

PMSE prediction mean squared error 29, 36, 49–51, 151

PWA piece-wise-affine 13

QP quadratic programming 22, 78, 130, 131

RDT rear differential torque 45–47, 57, 109, 110

RTV rear torque vectoring xix, 78, 88, 97, 115, 120

RWD rear-wheel-drive 78

SCC softened chance constrain 44, 57

SLC single lane change 113

SMC sliding mode control 11

SoD subset of data 38

STR driver or autonomous steering 45, 46, 78, 83, 97, 109

SURE Stein's Unbiased Risk Estimator 151

TCR traction control rear 83, 120

TV torque vectoring 9

UA unknown agent 33

VAE variational autoencoder 131

VSC vehicle stability control 83

WSC wheel stability control 83

YRT yaw rate tracking 78, 114

List of Symbols

A_{CGd} Discretized and linearized system matrix on the states at the vehicle CG 23

$A_{d(i)}$ Discretized and linearized system matrix on the states for the i-th agent 23

A_d Discretized and linearized system matrix on the overall states 67

$\tilde{A}_{e(i)}$ Uncertainty dynamics using feedback stabilization 43

$A_{in,j}$ Coefficient matrix of the linear inequality constraints of the control action from agents contributing to objective O_j 71

A_{in} Coefficient matrix of the linear inequality constraints of the overall control action U 67

$A_{(i)}$ The i-th agent in a multi-agent system 20

B_{CGd} Discretized and linearized system matrix on control inputs at the vehicle CG 23

$B_{d(i)}$ Discretized and linearized system matrix on control inputs for the i-th agent 23

$B_{d,j}$ Discretized and linearized system matrix on control inputs from the agents contributing to objective O_j 71

$B_{in,j}$ Coefficient matrix of the linear inequality constraints of the control action from agents not contributing to objective O_j 71

$C_{\alpha f}$ Cornering stiffness for front tires 47

$C_{\alpha r}$ Cornering stiffness for rear tires 47

$C^{\dagger}$ Pseudo-inverse of the observation matrix C 36

C_j Observation matrix of the j-th objective 67

χ_{CG} Constraint set of the overall states on the vehicle CG 23

$\chi_{(i)}$ Constraint set of the system states for the i-th agent 23

C The overall observation matrix 23

$\mathcal{D}_{N_d}$ Dataset with N_d data points 29

D_p Preview distance from the closest point on the reference path 96

$\mathcal{E}_{v(i)}^{(k)}$ Slack polygons on the nominal control action from agent i in L-AMPC with SCC 44

$\mathcal{E}_{z}^{(k)}$ Slack polygons on the nominal states in L-AMPC with SCC 44

F_{xt} Total longitudinal forces 96

F_{yCG} Net lateral force at the vehicle CG 45

F_y Lateral tire force 46

$G_{j(i)}$ The Boolean variable that indicating weather the agent $A_{(i)}$ contributes to (or has influence on) the objective O_j or not 20

I_z Vehicle yaw inertia 45

$J_{(i)}$ Cost function of the i-th agent in AMPC 24

J_j Cost function of the j-th objective in MO-AMPC or L-MO-AMPC 26

J The overall cost function 23

$K_F F$ Feedforward gain in a feedforward controller 47

K_I Integral gain in a PI controller 47

K_P Proportional gain in a PI controller 47

$K_{(i)}$ Feedback matrix for the uncertainty dynamics of agent i 42

K_{us} Under-steering coefficient 47

K_y The Gramian matrix in GPR 30

$K(\cdot,\cdot)$ Covariance (kernel) matrix in GPR 30

L_b Wheelbase 47

L_ρ The augmented Lagrangian in ADMM 26

M_{zCG} Net yaw moments at the vehicle CG 45

M_{zt} Total active yaw moment 96

N_A The number of total agents 20

N_O The number of total objectives 20

N_d Number of points in the dataset 29

N_p Number of time steps in the prediction horizon 21

O_j The j-th objective in a multi-objective system 20

$\Omega_{(i)}$ Constraint set of the control inputs from the i-th agent 23

Ω Constraint set of the overall control inputs from all agents 67

P_j Projection matrix from Z to Z_j 69

Q_{max} Maximum capacity of the wheel motor 48

Q_{min} Minimum capacity of the wheel motor 48

Q Torque 46

$R_{u(i)}$ Semi-positive definite diagonal matrix of penalty weights for the control action from the i-th agent 23

R_{uj} Semi-positive definite diagonal matrix of the average penalty weight for the control action from agents contributing to objective O_j 71

$R_{v(i)}$ Heavy penalties on slack polygons for control action from agent i in L-AMPC with SCC 44

R_w Wheel radius 46

R_{xj} Positive definite diagonal matrix of penalty weights for the j-th tracking error 67

R_x Positive definite diagonal matrix of penalty weights for the tracking error 23

R_z Heavy penalties on slack polygons for nominal states in L-AMPC with SCC 44

S_A The set of all agents 20

S_O The set of all objectives 20

$S_{(i)}$ Contribution set for $A_{(i)}$: the index set of objectives that the agent $A_{(i)}$ contributes to 21

S_j Agent group for O_j: the index set of agents that contribute to the objective O_j 21

$S_{x,des}$ Desired longitudinal slip ratio 84

Σ_* Predicted variance from GPR 30

T_s Sample time 45

T_w Track width 46

$U_{CG(i)}$ The corresponding net forces and moments at the vehicle center of gravity from the i-th agent 22

U_{CG} The overall net forces and moments at the vehicle center of gravity 22

$U_{(i)}$ The control input from the i-th agent 21

U_j The control inputs that contribute to the objective O_j 21

U The overall control inputs from all agents 21

W_{CGd} Discretized and linearized additive term at the vehicle CG 23

$W_{d(i)}$ Discretized and linearized additive term for the i-th agent 23

W_d The overall discretized and linearized additive term 67

$X_{CG,ref}$ The reference states on the vehicle CG 23

X_{CG} System states at the vehicle CG 23

$X_{(i)}$ System states related to the i-th agent 23

X_{ref} The reference vehicle states 67

X Overall system states 67

$Y_{CG(i)}$ Contribution at vehicle CG from the i-th agent 23, 33

$Z_{(c)}$ Local projection from Z for the c-th linear inequality constraints 76

Z_j Local projection from Z for objective O_j 68

Z Overall common global variable in ADMM 68

$a_{x,des}$ Desired longitudinal acceleration 96

$\alpha_{rear,max}$ Maximum allowable side slip angle at the rear axle 48

α Learning rate in AMPC 25

$b_{in,j}$ Constant vector of the linear inequality constraints of the control action from agents contributing to objective O_j 71

b_{in} Constant vector of the linear inequality constraints of the overall control action U 67

$\bar{R}_{uj}$ Semi-positive definite diagonal matrix of the average penalty weight for the control action from agents contributing to objective O_j 71

$\bar{U}_{(i)}$ The control sequence from the i-th agent 21

$\bar{U}_j$ The control sequence that contributes to the objective O_j along the prediction horizon 21

$\bar{U}$ The overall control sequence along the prediction horizon 21

$\bar{Z}_j$ Sequence of the local projection from Z for objective O_j along the prediction horizon 69

$\bar{Z}$ Overall common global variable along the prediction horizon 69

$\boldsymbol{x}$ Input matrix in the dataset 29

$\boldsymbol{y}$ Output vector or matrix in the dataset 29

c index that refers to the linear inequality constraints 76

δ_f Equivalent front wheel steering angle 47

δ Steering angle 46

$\Delta y_{CG(\mathbf{KA})}$ Contribution on the observation from all the known agents 36

$\Delta \hat{y}_{CG(\mathbf{UA})}$ Estimated contribution on the observation from unknown agents 36

$\Delta \hat{y}_{CG(\mathbf{UA}),prd}$ Predicted contribution on the observation from unknown agents from GPR 42

$\Delta y_{CG(\mathbf{UA})}$ Contribution on the observation from all the unknown agents 36

$\Delta y_{CG(\mathbf{total})}$ The overall contribution on the observation from all agents 36

$e_{CG}^{(k)}$ Unbiased disturbance terms with the uncertainty of error dynamics at the k-th step in horizon 42

ϵ_Y The threshold for stop criteria of the consensus check in the AMPC 25

ε_c The admissible probability of constraint violation 31

ϵ_l The threshold for learning reliability check in the L-MO-AMPC 94

e_r Yaw rate error between the desired value and measured value 46

e_{sub} suboptimality of control actions between distributed and integrated control schemes 82

e_u Longitudinal speed error between the actual speed and the desired speed 96

e_φ Heading error between the actual heading angle of the vehicle and the desired heading angle 96

e_y Lateral error from the vehicle CG to the preview point on the reference path 96

ϵ_r Convergence threshold of the primal residual norm in MO-AMPC for objective O_j 70

ϵ_w Convergence threshold of the dual residual norm in MO-AMPC for objective O_j 70

f_d Discrete system equation 67

g_a Gravity acceleration 47

$g_{(i)}$ Regularization function for the control action $U_{(i)}$ from the i-th agent 67

γ_j Local Lagrange multiplier for objective O_j 69

γ Lagrange multiplier in ADMM 26

g Regularization function for the overall control action U 67

h_j State penalty for objective O_j 67

$\hat{y}_*$ Predicted mean from GPR 30

i Index that refers to the agent $A_{(i)}$ 20

j Index that refers to the objective O_j 20

k Index that refers to the step in the prediction horizon 20

l_f Distances from the front axle to CG 46

l_r Distances from the rear axle to CG 46

$\lambda_{max}(\cdot)$ The maximum non-zero eigenvalues of the bracketed matrix $(\cdot)$ 78

$\lambda_{min}(\cdot)$ The minimum non-zero eigenvalues of the bracketed matrix $(\cdot)$ 78

$m_{(c)}$ number of objectives that the c-th constraint is involved 76

$m_{(i)}$ number of objectives that the agent $A_{(i)}$ is contributing through $U_{(i)}$ 71

μ Road friction 47

m Vehicle mass 45

n_{cj} Dimension of constraints related to U_j 71

n_c Dimension of inequality constraints of the overall control action U 67

n_d Dimension of the input vector 29

$n_{u(i)}$ Dimension of the control action from the i-th agent 21

n_{uj} Dimension of the control action that contributes to the objective O_j 21

n_u Dimension of the overall control actions from all agents 21

n_d Dimension of the overall system state 67

n_{zj} Dimension of the local projection from Z for objective O_j 69

n_z Dimension of the overall common global variable Z 69

$\omega_{CG,prd}$ Unbiased disturbance terms with the uncertainty from GPR prediction 43

ω_{des} Desired wheel speed 84

$\ominus$ Minkowski subtraction 43

$\oplus$ Minkowski sum 44

p Index that refers to the data point in the dataset 29

q Index that refers to the iteration in one time-step 20

R^2_{adj} Coefficient of determination 29

r_{des} Desired yaw rate 46

$r_j^{\langle q \rangle}$ Primal residual for objective O_j at the iteration q in MO-AMPC 70

r_{max} Maximum allowable yaw rate 48

r_{mes} Measured yaw rate 46

r_{time} Time consumption ratio of the distributed and integrated control schemes 82

ρ_j Local penalty parameter for objective O_j 69

ρ Penalty parameter in ADMM 26

r Vehicle yaw rate 45

$s_j^{\langle q \rangle}$ Dual residual for objective O_j at the iteration q in MO-AMPC 70

σ_{N_d} Noise standard deviation 39

σ_f Signal standard deviation 39

$\tilde{U}_j$ Control action from agents not contributing to objective O_j 72

t Index that refers to the time step 20

$v_{(i)}^{(k)}$ Nominal control action for agent i at the k-th step in horizon 42

$v_{x,des}$ Desired longitudinal speed 96

v_x Longitudinal vehicle speed 45

$v_{y,des}$ Desired lateral vehicle speed 47

v_y Lateral vehicle speed 45

$\varepsilon_{v(i)}^{(k)}$ Slack variables on nominal control constraints on agent i at k-th step 55

$\varepsilon_z^{(k)}$ Slack variables on nominal state constraints at k-th step 55

φ_{des} Desired heading angle 96

φ Actual heading angle of the vehicle 96

x_* Input of the test point for GPR 30

x_p Input vector of the p-th data point in the dataset 29

y_* Ground truth of the output of the test point for GPR 30

y_{CG} Observation variable(s) 35

y_p Output vector or matrix of the p-th data point in the dataset 29

$z_{CG}^{(k)}$ Nominal (noise-free) states on CG at the k-th step in horizon 42

z_p p-th quantile of Gaussian distribution 55

Chapter 1

Introduction

1.1 Motivation and Objectives

The automotive industry is undergoing a profound transformation driven by electrification and increasing intelligence. Electric vehicles have gained significant market share, accounting for 14% of global sales in 2022 [1]. This shift towards electrification, along with the proliferation of "X-by-wire" control systems, has paved the way for more extensive controller integration in vehicles. At the same time, the penetration rate for Level 2 autonomous vehicles in major markets is expected to exceed 20% by 2024 [2], indicating a strong demand for a larger variety of advanced assisted driving features like automatic emergency braking and adaptive cruise control in intelligent vehicles. In response to these trends, the concept of holistic vehicle control (HVC) has emerged as a promising approach. HVC represents an integrated control methodology that leverages all vehicle controllers to enhance overall mobility, stability, and safety. This approach is gaining attention as the automotive industry moves towards more advanced and interconnected technologies.

As an effective way to predict vehicle dynamics and to deal with a constrained optimal control problem, model predictive control (MPC) has the potential to satisfy diverse and even conflicting requirements of vehicles [3][4]. The integrated MPC in an "all-in-one" fashion naturally becomes the easiest and most common approach to coordinate multiple controllers to achieve various objectives simultaneously, where all objectives and agents are stacked and optimized together [5]. Nevertheless, as more agents and objectives are involved, the high-dimensional optimization problem in the integrated MPC will cause computational burdens and maintenance difficulties in practice. Besides, the need for flexibility and reliability has led to a "plug-and-play" approach in HVC design, which

presents challenges for traditional integrated MPC. Therefore, distributed control schemes, which can decouple complex vehicle control systems into distributed topology, become natural solutions for these challenges.

In recent studies, agent-based model predictive control (AMPC) has recently emerged as a distributed solution that treats controllers as agents, creating a collaborative framework among them to reach a common goal [6]. The main idea was that each control module is individually modelled as an interactive agent assigned with a local control task. Agents could be flexibly connected while emulating integrated MPC performance using an iterative consensus algorithm. However, two critical pain points arise in practice:

a. *Uncertain contributions from unknown controllers*: The optimal control results from the AMPC highly depend on the prediction accuracy, which requires all agents or their contributions to be accurately known. This requirement is often too idealistic for actual implementation. For example, when a third party develops a "black-box" controller, its underlying algorithm is unknown, and there is no specified interface to know its contribution to vehicle dynamics. This lack of information is likely to cause a significant error in predicting vehicle behaviour, leading to unexpected or even harmful control results.

b. *Limitations on controller-oriented decomposition*: As mentioned earlier, nowadays, vehicle control systems often involve multiple controllers to achieve multiple objectives simultaneously. We refer to such complex systems as the multi-agent-multi-objective (MAMO) systems, where each controller is treated as an interacting agent. In practice, decomposing from the perspective of control objectives is an ideal approach to developing function-oriented automotive control systems. However, AMPC can only decompose the MAMO system from the controller's perspective, not from the objectives. The objective in AMPC is usually implemented as a weighted sum of multiple goals. Hence, AMPC cannot manage distributed conflicting objectives when agents are coupled or interdependent, which greatly limits the system's flexibility.

Therefore, this research is motivated to overcome these two pain points through practical solutions using data-driven machine-learning techniques and flexible distributed schemes for MAMO control systems.

Firstly, thanks to increased computational capabilities in modern control systems, data-driven learning-based control techniques have gradually become possible. Notably, the learning-based model predictive control (LMPC) has become one of the main trends in

modern controller design [7]. This research is motivated to propose a hybrid scheme combining the data-based machine learning approaches with AMPC to solve the problem of any unknown agents existing in a multi-agent system. The learning module predicts unknown information from data, while the AMPC provides an optimal solution within safety boundaries.

Secondly, different from AMPC, which can only decompose the system from the perspective of the controller, some distributed model predictive control (DMPC) frameworks have been proposed to decompose the system from the perspective of the control objective. However, some DMPCs [8][9] adopt a hierarchical architecture to optimize each objective sequentially without any information exchange, which is easy to develop but difficult to obtain the same solution as the integrated one (the global optimum) when objectives are coupled. Therefore, this thesis also aims to systematically propose a more flexible distributed scheme tailored from the alternating direction method of multipliers (ADMM), in which objectives are distributed with a corresponding controller group. The global optimum should be achieved iteratively through information exchange between controller groups. The proposed scheme will have excellent flexibility since objectives, i.e., vehicle features, can be designed in a "plug-and-play" fashion.

Furthermore, for distributed control frameworks that require iterative convergence to reach consensus, the computational time consumption will increase with the number of iterations, which is a huge challenge for real-time control systems. Therefore, this study proposes to use data-driven machine-learning techniques to accelerate the converging process. By predicting the negotiation of different objectives on the coupled controller(s), initial states that are closer to the global optimum are learned to reduce the number of iterations needed, providing the possibility of improving computational efficiency.

1.2 Contributions

This study is motivated to explore more practical and efficient distributed control methods for MAMO systems in HVC. The contributions of this thesis can be summarized as follows:

- This thesis proposes a practical hybrid control scheme: learning agent-based model predictive control (L-AMPC). This scheme combines the model-based AMPC approach with data-driven learning methods to improve the holistic vehicle performance for multi-agent systems. The Gaussian process regression (GPR) enhanced by an online data management strategy serves as the learning core to predict unknown

contributions, which complete the system model in the MPC for more accurate control. Meanwhile, a stochastic framework is formulated to guarantee control safety and feasibility using soft chance constraints based on the prediction variance. Both simulations and experiments show that, with the learning capability, L-AMPC can achieve higher tracking performance in well-learned scenarios and always guarantee constraint satisfaction even in less-learned scenarios. Moreover, the proposed hybrid control scheme is efficient for real-time implementation and is flexible to any multi-agent topology.

- In order to decompose the MAMO system from the perspective of control objectives, this thesis proposes a practical distributed control scheme: multi-objective agent-based model predictive control (MO-AMPC). This scheme adapts the ADMM into a general control strategy that achieves global optimization while decoupling objectives. Three formulations that can maintain convergence while addressing control regularization and inequality constraints are systematically developed, and they are applied to complex vehicle control systems for the first time. The proposed method has been tested on two vehicle control scenarios with a MAMO topology. Different formulations are compared through simulations, and the most computationally efficient one was implemented on an electric vehicle for real-world evaluations. The results demonstrate that the proposed MO-AMPC achieves the same global optimum as integrated MPC with greater flexibility and reduced computational costs.

- This work further proposes a data-driven distributed control scheme based on the MO-AMPC: learning multi-objective agent-based model predictive control (L-MO-AMPC). To accelerate convergence, a learning-based method for initializing iterations is proposed. An authentication module is designed for learning reliability. The proposed scheme is compared with the integrated scheme through a combined path-tracking simulation for autonomous vehicles using various controllers. It is shown that the L-MO-AMPC scheme achieves similar control performance while significantly reducing computational time, making it well-suited for real-time implementation.

The control strategies presented in this thesis offer novel insights into practical HVC systems within MAMO frameworks. These approaches show significant potential for applications in distributed control and planning, particularly in the fields of vehicle systems, autonomous driving, and robotics.

1.3 Thesis Outline

The road map illustrated in Figure 1.1 summarizes the goal and structure of this work in brief. First, the recently proposed distributed control approach, AMPC, will be reviewed as a background in Section 3.2. Based on this, by integrating data-driven methods to learn the contribution of unknown agents, a hybrid control scheme, the L-AMPC, is proposed in Chapter 4. Next, inspired by the concept of AMPC that treats the controller as an agent, a distributed control method that can decompose complex MAMO control systems from the perspective of objectives, the MO-AMPC, is proposed in Chapter 5. Furthermore, leveraging learning-based methods to reduce the number of iterations for higher computational efficiency, the L-MO-AMPC is proposed in Chapter 6.

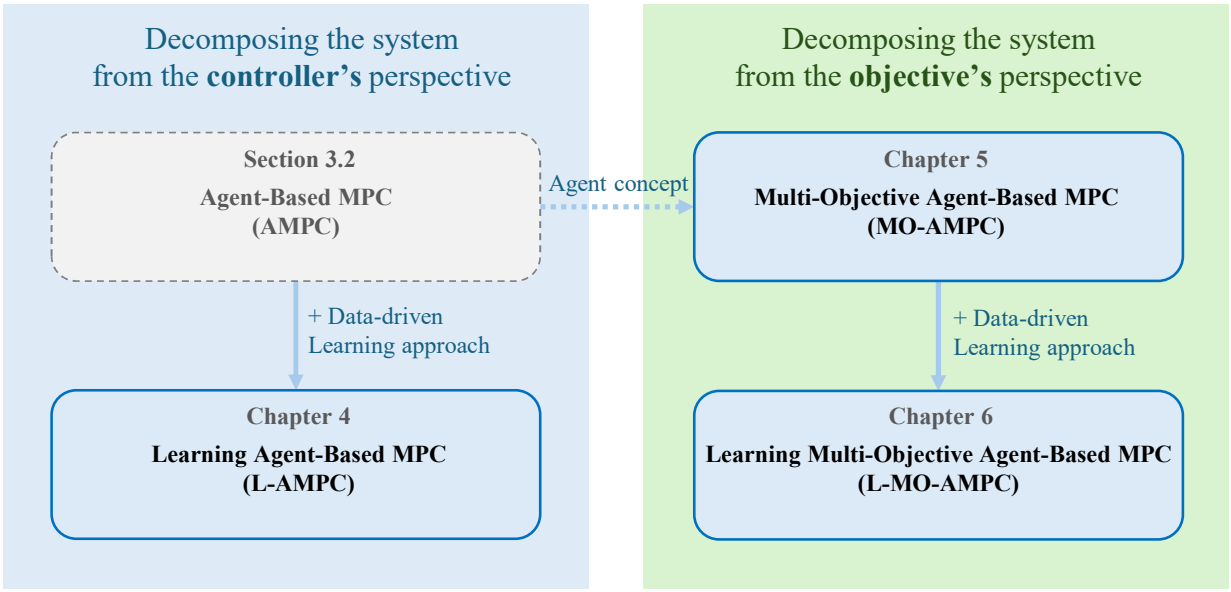


Figure 1.1: Road map of the research

The remaining chapters are organized as follows:

Chapter 2 Literature Review

This chapter presents a review of the background of the HVC. It starts with the concept of vehicle stability and motion control and proceeds with various MPC methods. Then, related research on DMPC is also discussed. In the last portion, the literature of LMPC for vehicle control is reviewed. In summary, it infers the significance and novelty of the proposed work.

Chapter 3 Background

This chapter presents the backgrounds needed for the thesis. First, the general topology of a MAMO control system is given with notations defined. Then, the basic distributed control scheme in this thesis, the AMPC, is formulated. Following that, the algorithm of ADMM is introduced. Lastly, the concept and important features of GPR and chance constraint are presented.

Chapter 4 Learning Agent-Based Model Predictive Control

In this chapter, the hybrid control scheme, the L-AMPC, is developed based on the AMPC. First, a high-level overview of the proposed scheme is presented. Second, the data-driven learning process is introduced. Third, the model-based soft stochastic control scheme using the learning results is formulated. Next, an example of yaw rate tracking is demonstrated with the design of the L-AMPC controller. Last, the developed controller is validated through several simulations.

Chapter 5 Multi-Objective Agent-Based Model Predictive Control

This chapter presents the proposed MO-AMPC scheme, which can decompose MAMO control systems from the objective perspective. It starts with the general introduction of the MAMO control systems with notations and the integrated MPC as a reference baseline. Following that, an overview of the proposed scheme is given. Next, three different formulations are presented. Then, the convergence and parameter selection of the last formulation are discussed. Furthermore, two examples of cruise control and holistic vehicle stability control are demonstrated with the design of the MO-AMPC controller. The convergence is validated and three formulations are compared through several simulations.

Chapter 6 Learning Multi-Objective Agent-Based Model Predictive Control

This chapter presents the proposed L-MO-AMPC scheme, which leverages data-driven learning methods to accelerate the convergence process. First, an overview of the proposed scheme is given. Then, the algorithm of the L-MO-AMPC is presented. Next, an example of path-tracking control is demonstrated. Furthermore, the convergence and the improvement of computational efficiency are validated by simulations. First,

Chapter 7 Experimental Study

This chapter presents the experimental study to validate the proposed control schemes. First, the yaw stability control using the L-AMPC is implemented in a real vehicle under double lane change (DLC) and the sine-wave maneuvers. The improved tracking performance using the learning techniques is validated. For the MO-AMPC, experiments of cruise control are conducted. moreover, an additional example of holistic stability control

using MO-AMPC is introduced, which is also implemented in two maneuvers. It is shown that the proposed MO-AMPC can converge to the same global optimum as integrated MPC while exhibiting superior computational efficiency.

Chapter 8 Conclusion and Future Work

This chapter concludes the work presented in the previous chapters and directs potential future work and directions.

Chapter 2

Literature Review

In this chapter, a review on HVC is presented, including the vehicle stability control and motion control in autonomous vehicles nowadays. Furthermore, the concepts of the MPC related to this thesis are discussed. Specifically, in Section 2.1, several position and velocity estimation methods are reviewed. Then, Section 2.2 discusses the different types of the MPC used for vehicle control. Following that, two variants of DMPC and LMPC are reviewed separately in Section 2.3 and 2.4. Finally, Section 2.5 summarizes the chapter and reveals the work in the next chapters.

2.1 Holistic Vehicle Control

2.1.1 Vehicle Stability Control

Stability control systems have been developed over the past few decades to assist drivers in keeping vehicles under control. The purpose of stability analysis is to find the conditions for the stability of a vehicle or to find regions inside which stability is inherently guaranteed [10]. The literature on vehicle stability control has two competing views on controller design. One leads to envelope control, and the other one is reference tracking.

Envelope vehicle control has received considerable attention from researchers. Stability analysis, particularly phase-plane analysis, is one of the most classic and effective methods and is extensively used in envelope control [11]. Different phase plane analysis methods were established by researchers, such as the yaw plane established by Sacks [12], the side slip angle plane established by Pacejka [13], the ratio of turning kinetic energy and forward

driving kinetic energy phase plane used to evaluate steering performance established by Konghui Guo [14]. The authors in [15] used the side slip angle and side slip angle rate as the state variables to analyze the vehicle stability region. However, the side slip angle and its rate are difficult to estimate in practice, especially in harsh maneuvers, which brings many difficulties to the practical stability analysis. The authors in [16] established the vehicle yaw stability criterion based on the side slip angle-yaw rate phase plane method with the nonlinear two-degree-of-freedom vehicle model and the Magic Formula of the nonlinear tire model. Different conditions of stability boundary plane database look-up tables were set up, and the effects of different conditions on the stability boundary's relevant design parameters were investigated. A similar stability analysis was done in [17][18][19]. In these studies, the stable and unstable regions of operation for each velocity and steering angle were obtained. However, all these envelope-based control methods cannot avoid the weakness of inaccurate nonlinear parameter estimation, which makes precise control difficult.

Reference tracking is a method in which the desired state trajectory is first determined, and then the system follows the expectation by correcting the control variables. The desired vehicle state is usually an easily measured or calculated state quantity, such as yaw rate. The authors in [20] showed that the yaw moment generated by differential drive torques significantly influences the yaw stability of the vehicle, especially in limit cornering. Moreover, the authors in [21] used sliding surface control to track vehicle side-slip angle and yaw rate reference values. The reference values were obtained using a linearized model to design the tracking controller. They also concluded that tracking the side-slip angle is much more complicated than the yaw rate. Similar stability analyses were done in [22][23][24]. In these studies, yaw rate tracking is the most commonly used vehicle stability control method and can be applied to various controllers.

HVC is a method for controlling a vehicle to improve vehicle dynamics by using any single or multiple configurable actuators. Controllers that affect vehicle stability can be of many kinds, such as differential braking (DB), active steering (AS) or torque vectoring (TV). Inconsistent control actions can be generated with individual modules, so the collaboration between the various controllers becomes particularly important. Among the noteworthy studies on HVC, it can be generally divided into centralized and decentralized (or distributed) schemes.

The authors in [25] introduced a re-configurable modeling technique to cover three-wheeled and four-wheeled vehicles with various actuator configurations and provided a centralized structure for omni-actuation systems to improve the overall stability of the vehicle. The work in [26] also proposed an integrated stability control strategy for tilting vehicles, which improved the vehicle handling performance while still guaranteeing its

lateral and roll stability. The authors in [27] developed a coordinated re-configurable vehicle dynamics control system through high-level control of generalized forces or moments. The work in [28] developed a hierarchical integrated controller for a distributed electric vehicle to improve driving safety, handling stability, ride comfort, and road tracking capabilities. The body control layer decouples complex problems to achieve multi-target independent tracking, while the lower actuator layer controls the corresponding actuator to achieve the optimal tire force output from the middle layer of tire force distribution.

A possible alternative is to equip distributed control systems instead. Chen et al. used this scheme early in vehicle control for vehicle stabilization [29]. This research described an analytical approach to control the tire forces (corner) based on the center of gravity (CG). All the control variables were calculated in real-time by minimizing a weighted cost function of the errors between actual and target CG forces and moments. Moreover, such a distributed control scheme has been applied to power systems to address computation and communication concerns [30]. For multi-vehicle connection and vehicle platooning studies [31][32][33][34], the distributed approach has also been adopted for joint decision-making among all connected vehicles. Although the concept of distributed control has been proposed for a long time [35][36][37], it only recently has been combined with the MPC method and applied in a multi-controller system in a single vehicle.

2.1.2 Autonomous Vehicle Motion Control

The design of the autonomous vehicle (AV) depends on sophisticated control systems that ensure safety, efficiency, and adaptability in various driving conditions. The evolution of AV technology hinges on the ability to develop reliable control systems that integrate both high-level path tracking and low-level motion or stability controls. Such holistic control systems are critical for ensuring safety, stability, and reliability in diverse driving scenarios.

High-level path tracking involves navigation and trajectory planning for autonomous vehicles. It focuses on choosing the best path for the vehicle, considering road shapes, traffic, and environmental factors. Path-tracking of autonomous driving is a comprehensive control that includes longitudinal speed tracking, lateral position tracking, and vehicle stability control. Many integrated MPC schemes proposed in the literature for path-tracking have excellent coordination of longitudinal and lateral objectives to achieve global optimization [38]. However, in practice, integrated schemes are not flexible enough in controller design as more electronic control systems that can affect vehicle motion are equipped. MPC is a popular method that can adjust the vehicle's trajectory by optimizing a cost function over a future prediction period. Research in [39] presented an MPC framework to dynamically adjust vehicle trajectories considering various constraints such as vehicle dynamics

and environmental interactions. This approach not only ensures robust path tracking but also facilitates the incorporation of low-level control actions by predicting the necessary adjustments in real time. Works in [40] also highlighted MPC's ability to manage vehicle dynamics and constraints, ensuring strong path tracking under uncertain conditions.

Low-level motion and stability control are vital for following the planned path accurately. This includes managing the vehicle's lateral and longitudinal movements through adjustments in steering, braking, and throttle. Authors in [41] explored various control techniques for AVs, discussing the use of linear quadratic regulators (LQR) and sliding mode control (SMC) for stable control. Research in [42] studied SMC's performance in challenging weather, noting its benefits in maintaining stability.

Combining high-level path tracking with low-level motion and stability controls is essential for holistic AV control. This ensures that strategic decisions from path-tracking algorithms translate into effective control actions. The integration of high-level path tracking with low-level motion and stability control is crucial for the development of reliable autonomous vehicle systems. By leveraging advanced control techniques and integrating diverse control objectives, researchers can significantly enhance the operational safety and efficiency of autonomous vehicles. The study in [43] described a control system that merged MPC for path planning with LQR for motion control, showing improved handling of difficult maneuvers. Authors in [44] proposed a robust control scheme that simultaneously addresses lane-keeping and wheel slip constraints. This integration ensures that the vehicle maintains its lane while preventing excessive wheel slip, highlighting the complex interplay between longitudinal and lateral dynamics.

2.2 Model Predictive Control for Vehicle Control

Among several vehicle control algorithms, the MPC, also known as receding horizon control, has recently received much attention from researchers. MPC has the advantage of model-based prediction of vehicle behavior and consideration of constraints. Therefore, MPC is an excellent choice for considering reference tracking and envelope control and producing the best control in real-time. There are many different types of MPC widely used in vehicle control, as follows.

2.2.1 Linear, Nonlinear and Hybrid Model Predictive Control

Linear MPC is based on a linear system model, which is simple enough to make the optimization problem tractable and solvable in real-time. However, the control effect of MPC is highly dependent on the accuracy of the prediction model. As vehicle dynamics are highly nonlinear due to nonlinear components, for example, tires, a simple linear model often cannot accurately describe its dynamic motion. A feasible method is using the linear time-varying MPC, in which the model can also change over the prediction horizon at each sampling time. In the study [45], a linear MPC was formulated using a linear two-degree-of-freedom model to improve the vehicle's handling performance. It presented an approach to vehicle stabilization that leverages these estimates to define state boundaries and utilizes a model predictive envelope controller to bind the vehicle motion within this stable region of the state space. Besides, the lateral Fiala tire model was used to estimate tire forces to make the prediction model more accurate. The authors in [46] proposed a linear MPC structure for vehicle lateral stability control using coordinated AS and DB. The controller was designed based on a linear bicycle model, and the side slip angle of the vehicle was enforced within an acceptable range using soft constraints on the lateral velocity to guarantee persistent feasibility. Also, to improve the model accuracy, the cornering stiffness of the front and rear axles in the linear model was estimated using measurements of the vehicle's lateral acceleration and yaw acceleration. Similar approaches can be found in studies such as [47][48][49]. Furthermore, the development of multi-parametric quadratic or linear programming [50][51][52][53], which allows the optimal control problem to be solved offline, has considerably increased the applicability of linear MPC in practice, albeit primarily to systems with a low number of states.

An alternative is to apply nonlinear model predictive control (NMPC), where the prediction is made using a nonlinear model. The main advantage of using a nonlinear prediction model is a more accurate prediction of the vehicle response in a broader range of operations. The global dynamics of a system can be described better with a nonlinear model, which is the most compelling reason for researchers to investigate NMPC [54]. The study in [39] compared linear MPC and NMPC by developing path-following controllers via combined DB and AS. Two controllers were designed to achieve the obstacle avoidance objective. One was based on a simple bicycle vehicle model, and the other was designed based on a ten-degree-of-freedom full-vehicle model. The results showed that the performance of the simplified linear model at high speed is sacrificed, while the NMPC controller is very time-consuming but could stabilize the vehicle both at low and high speeds. The same results could also be found in [55], where the NMPC with a more precise rolling model achieved better stability on low-friction roads and high-speed conditions. Similar research works

have been conducted in [56]. Although there are already some solvers for NMPC, such as qpOASES [57] and IPOPT [58], the enormous computational burden is still a barrier to real-time application, which makes NMPC less common in practical engineering than linear MPC.

One of the practical alternatives instead of NMPC is the hybrid model predictive control (HMPC), which is a trade-off between prediction precision and complexity. HMPC is an MPC algorithm that employs a piece-wise-affine (PWA) approximation of future system behavior. The state-input space forms polyhedral regions, and each region has an affine equation that defines the system dynamics [59]. The HMPC could be easily solved by the mixed-integer quadratic or linear programming solvers [60]. For example, the authors in [61] used the PWA system as a prediction model in an HMPC strategy to improve yaw stability control through active front steering and differential braking, assuming that the tire conditions (linear/saturated) would not change during the prediction horizon. The assessment of the controller's computational load and memory requirements indicates that it is capable of real-time execution while maintaining high performance. However, mixed-integer programming does not always scale very nicely. It is usually suitable for those MPC problems with a short horizon, few 'modes' of the hybrid system, and a small number of constraints.

2.2.2 Robust and Stochastic Model Predictive Control

In many control problems, decisions must be taken under uncertainty. Both disturbances in the model and noises from measurements may introduce uncertainties. Besides, model uncertainty could also be naturally introduced if data-driven models were combined with first-principles models to represent the system's behavior (one of the topics this research will focus on). There are usually two ways to solve the MPC problem in this case. The first is the robust MPC, which does not model the uncertainty (only assumes that it is bounded) and pessimistically considers the worst case. The other is the stochastic MPC, in which stochastic models provide additional information about uncertainty. In many control problems, decisions must be taken under uncertainty. Both disturbances in the model and noises from measurements may introduce uncertainties. Besides, model uncertainty could also be naturally introduced if data-driven models were combined with first-principles models to represent the system's behavior (one of the topics this research will focus on). There are usually two ways to solve the MPC problem in this case. The first is the robust MPC, which does not model the uncertainty (only assumes that it is bounded) and pessimistically considers the worst case. The other is the stochastic MPC, in which stochastic models provide additional information about uncertainty.

Robust MPC methods have been studied since the last century [62]. Generally, it can be divided into two categories for achieving robust stability: open-loop vs. closed-loop. The robust open-loop MPC is based on an open-loop prediction (min-max formulation) of the uncertainty. Its robust stability is not guaranteed through indirect analysis. Alternatively, if robust stability has to be ensured, the uncertainty must be assumed to be time-varying [63]. The better solution is the robust closed-loop MPC, which introduces a less conservative constraint formulation by including a feedback term. It guarantees that hard state constraints are met in the presence of input disturbances while attempting to reduce the effect of disturbances [62]. However, due to extensive computational cost, only a few robust MPC approaches have been applied in real-time implementation, such as multi-parametric programming and Tube-MPC, have been applied to the robust MPC [64][65].

The author in [66] developed a robust NMPC for yaw rate control through a front steer-by-wire controller. The vehicle model employed by the NMPC algorithm was obtained from previously collected input and output data using a nonlinear set membership identification methodology. Thus, the model uncertainty provided by the identification allows one to perform a theoretical robust stability analysis of the closed-loop system. The study in [67] developed a robust multiple-model adaptive control scheme for tracking tasks. Recursive-least-squares methods were proposed for uncertain nonlinear system identification, which could provide the uncertain variation with known bounds in tire longitudinal and lateral force capacity. It can also be found from these studies that how to obtain uncertain features (such as bounds) is one of the keys to designing robust MPC controllers.

Robust MPC approaches rely on bounded, deterministic descriptions of system uncertainties. In real-world systems, however, uncertainties are often considered probabilistic [68]. Instead, the stochastic MPC uses a stochastic dynamical model of the process to predict its possible future evolution and choose the optimal control actions. The fundamental challenge of stochastic MPC is predicting the time-varying probabilities of uncertainties. Therefore, some stochastic models, such as the Markov chain [69][70] or recent data-driven machine learning methods, such as GPR [71][72], are often combined with the stochastic MPC.

Based on that, stochastic uncertainty has two main effects on MPC: objective function and constraints. The cost function in stochastic MPC is either an expected value minimizing worst-case performance, such as the min-max function [73], or a kind of trade-off between the expectation and risk, such as the conditional value-at-risk function [74]. Regarding the constraints in stochastic MPC, chance-constrained programming was first introduced, where a violation of the constraints is accepted below a given threshold of probability [75][76]. Besides, under a particular Gaussian assumption or approximation,

the probabilistic constraints provide a convenient and natural point of entry for the estimate covariance. Thus, the stochastic optimal control could be easily converted to a deterministic optimization problem in a standard MPC form [77][72]. More recently, an alternative scheme called sampled (or scenario or random) convex programming was proposed to deal with stochastic MPC [78]. Because, in practice, discrete probability measures are more feasible than continuous probability distributions, and the uncertainty's realized values give rise to a scenario tree, which processes the possible uncertainty trajectories [79]. The sampled convex program is less conservative than the corresponding chance-constrained program and can accommodate unbounded uncertainty sets [80]. There have already been many applications of stochastic MPC in vehicle control [69][81][82], assisted driving [83][84], and multi-vehicle network control [77].

2.3 Distributed Model Predictive Control

For HVC on multi-agent systems, the most significant limitation of the centralized or integrated MPC scheme is the firm reliance on a specific actuator configuration developed for target platforms. This drawback limits their scalability and re-usability, especially with the increasing number of onboard controllers. Instead, a distributed control architecture inspired by the modular chassis design, which allows for great re-usability, was then developed in [85] for a modular urban vehicle. Early application of DMPC can be found in [86], where the interconnections between different local MPCs are assumed to be weak and are neglected. For modern DMPC, communication and information exchange in a network is needed. Unlike the non-cooperative DMPC, which usually leads to the Nash equilibrium as local MPCs are selfish, cooperative DMPC optimizes the same global cost function at each local controller [87]. Commonly used forms in wider applications of cooperative DMPC include sequential DMPC [88] and iterative DMPC [89]. However, achieving convergence to the global optimum as the corresponding integrated formulation is challenging. Some DMPCs [8][9] adopt a hierarchical architecture to optimize each objective sequentially without any information exchange, which is easy to develop but difficult to obtain the same solution as the integrated one (the global optimum) when objectives are coupled. Many DMPCs used in some formation control tasks of unmanned aerial vehicles [9] and vehicle platooning [90] cannot guarantee the global optimally.

The recently proposed AMPC is a type of cooperative DMPC based on a consensus mechanism and has been proven in vehicle control systems to converge to a centralized MPC. The main idea of AMPC is that each controller is individually modeled as an interactive agent assigned with a local control task. Agents could be connected in a flexi-

ble "plug-and-play" fashion [91] while emulating centralized MPC performance, which can meet the demands for flexibility while emulating centralized MPC performance under complete coordination assumptions. First, the Mechatronic Vehicle Systems Lab (MVSL) has proposed some initial solutions, such as the re-configurable integrated MPCs [25][92][93]. These controllers all required a centralized control unit to distribute the control efforts among all available actuators, and the topology of the vehicle must be fixed at the controller design stage for such centralized distribution schemes. Thus, the flexibility of such an approach still needed to be improved, and the resultant large-scale optimization problem could be less efficient for real-time execution. That is the earliest AMPC form, although it is not developed for production vehicles. However, this method is limited to the many complex multi-agent control systems in practice with a MAMO topology, where multiple objectives need to be decomposed, and there are couplings or shared agents between those distributed objectives.

This contribution was subsequently generalized as the AMPC scheme by extending its application from modular vehicles to conventional automobiles in the study [6]. The simulation results showed that the proposed agent-based cooperative controllers give no compromise to control performances as their integrated-designed counterparts. The AMPC scheme also features flexibility in handling vehicle configuration changes and non-cooperative agents. Later, this novel AMPC scheme was implemented on an all-wheel-drive test platform [94]. It was shown from experimental results that AMPC is more flexible in agent configuration challenges while maintaining the same high performances compared to its integrated counterparts.

2.4 Learning-Based Model Predictive Control

Recently, the combination of model-driven and data-driven design, especially the MPC and machine learning, has become one of the main trends in control development [95]. Thanks to increased sensing and computational capabilities in modern control systems, data-driven learning-based control techniques have gradually become possible. They show substantial advantages in mathematically modelling and intelligently achieving the goal [7]. On the other hand, a model-based control scheme could provide safe boundaries for learning, which is one of the most challenging problems in machine learning today. To this end, the integration of machine learning techniques into the MPC, robust MPC, and stochastic MPC framework are becoming a hot spot of research in automatic control areas.

The primary goal of the LMPC mainly focuses on learning model errors or uncertainties through data to compensate for the inability of traditional MPC prediction models

to describe nonlinear problems under the limitation of computational cost accurately. According to updating the model by learning from data, LMPC can be divided into offline and online learning.

Repetitive trajectory following is often used in offline learning to revise the model in previous studies. Authors in [96][97] proposed a learning-based NMPC scheme for an autonomous mobile robot to reduce path-tracking errors over repeated traverses along a reference path. The model disturbances were modeled as a Gaussian process (GP) function of the system state, input, and other relevant variables. The GP was updated based on experience collected during previous trials. The results showed that the learning-based NMPC controller could start from a generic prior vehicle model and subsequently learn to reduce path-tracking errors based on experience.

The studies in [98][99], as well as the results of the experiment in [100], proposed a robust LMPC scheme. The central insight of the proposed structure is that safety and performance can be decoupled by maintaining two models of the system. The first is an approximate model with bounds on its uncertainty, and the second is an updated model by statistical methods. The input variables used for learning were filtered and significantly reduced, improving the efficiency of real-time solving. Experimental results demonstrated the improved dynamic response that the LMPC provides and the robustness of the LMPC. Furthermore, authors later applied this scheme to the energy-efficient building heating, ventilation, and air conditioning control using LMPC [101]. The experiments showed that the LBMPC method saved an estimated 30%-70% energy.

There are various machine learning methods for learning model uncertainty from data, for instance, the neural networks [102][103], reinforcement learning [104][105], and GPR, which will be described in detail in the next chapter.

Another vital role of learning techniques in LMPC is combining them with the other elements in the optimization scheme, such as the cost functions, prediction horizon, and constraints, to achieve more robust and precise control performance. Learning the parameterized cost function could improve closed-loop performance. The study in [106] introduced a parameterized cost function optimized to compensate for the deviation between actual and linear predictive dynamics by combining the linear optimal control with Bayesian optimization, resulting in an improved controller with fewer evaluations. As for learning the prediction horizon, the authors in [107] presented a new approach to tune the parameters in MPC using machine learning techniques. In their study, data-driven Bayesian optimization was used to tune the prediction horizon in MPC. Numerical simulation for a pendulum on a cart showed that using this approach, the prediction horizon can be optimized from data, and satisfactory controllers can be designed without knowledge of the system dynamics.

In the above two applications, LMPC is very similar to some variants of traditional MPC, such as linear-parameter-varying MPC. However, the main advantage of LMPC is that it updates the models using statistical methods and provides robustness to poor model updates by using theory from robust or stochastic MPC. Thus, the combination of learning results with constraints in LMPC received more attention from researchers. The authors in [97] proposed a robust constraint learning-based nonlinear model predictive control method to track the path of off-road terrain. The robust constraint became less conservative by updating model uncertainty online. The result showed that the proposed method provides safe, conservative control during initial trials when model uncertainty is high and converges to high-performance, optimal control during later trials when model uncertainty is reduced.

The authors in [71] proposed a cautious MPC scheme using GPR, a principled way of formulating the chance-constrained LMPC by considering residual uncertainties learned by the GP model. This scheme was applied to the autonomous racing car and proved to improve performance. The computational approximations in this study rendered GP-based MPC formulation computationally feasible with low sampling times.

More recent studies have applied learning to distributed control systems. Authors in [108] proposed a distributed learning MPC for coupled linear systems, where each subsystem has a learning model that captures the coupling effect of the neighbours. Besides, work in [109] has discussed the recursive feasibility and robustness of learning-based control for distributed schemes. Inspired by all those fascinating works, this paper proposes a learning scheme for ADMM-based AMPC to improve iterative computational efficiency.

2.5 Summary and Discussion

In this chapter, some latest MPC schemes on HVC were reviewed. For modern multi-agent vehicle control systems, the DMPC scheme is a strong candidate for HVC because of its high robustness and scalability. The latest AMPC scheme showed substantial flexibility in engineering practice, handling agents' configuration challenges while maintaining the same high performance as its integrated counterparts. However, the excellent performance of AMPC requires all agents to be controllable or un-controllable, but at least their efforts are known.

This chapter also reviewed various MPC schemes under practical uncertainty. For a strong nonlinear system such as a vehicle, on the one hand, it is difficult to use a linear model to predict vehicle behavior accurately. On the other hand, using a nonlinear model

requires high computational costs. Besides, although the model uncertainty in the actual driving process can be described by some variants of MPC, such as robust MPC and stochastic MPC, the estimation of the probability distribution for the model uncertainty is challenging. All of these have made the LMPC a hot research topic in recent years, which can improve the accuracy of the prediction model based on data while ensuring the stability and robustness of the system using stochastic methods.

However, as far as the author of this study is aware, there has yet to be any research that combines LMPC with AMPC to solve the problem that the model of known agents in a multi-agent system cannot fully describe the vehicle dynamics. Thus, using data-driven approaches to improve the performance and robustness of model-based multi-agent MPC is an area with great potential.

Chapter 3

Background

This chapter presents the backgrounds of the following proposed schemes. In Section 3.1, the concept and notations of MAMO control systems are introduced. In Section 3.2, the recently proposed AMPC scheme is reviewed. In Section 3.3, the ADMM method is formulated. Then, GPR is introduced in Section 3.4. The concept of chance constraint is introduced lastly in Section 3.5.

3.1 Multi-Agent-Multi-Objective Control System

In this thesis, the term "agent" refers to a controller in a MAMO control system. Subscripts of index j and index in parentheses (i) are used for notations referring to the objective O_j and the agent $A_{(i)}$, respectively. Superscripts of the index in parentheses (k) and (t) are used for the prediction step in the horizon and the time step, respectively. The index in angle brackets $\langle q \rangle$ is used for iterations in one time step. The superscript $(k = 0)$ on any variable represents its current value. Unless stated otherwise, the step marker (k) and the iteration marker $\langle q \rangle$ are usually omitted from all variables in this paper. Superscripts without parentheses or angle brackets denote powers as commonly used.

The set $S_A = \{A_{(i)} | i = 1, \ldots, N_A\}$ and the set $S_O = \{O_j | j = 1, \ldots, N_O\}$ are defined to represent all agents and objectives in a MAMO control system, respectively. N_A and N_O are the total number of agents and objectives, respectively. A Boolean value $G_{j(i)} = 1$ is used to represent that the agent $A_{(i)}$ contributes to (or has influence on) the objective O_j. Otherwise, there is no relationship between $A_{(i)}$ and O_j. The value of $G_{j(i)}$ can usually be obtained by analyzing the system model, such as the partial differential equation of the objective on the agent. We further define:

- *Agent group for O_j*: $S_j = \{i | G_{j(i)} = 1\}$, the index set of agents that contribute to the objective O_j.
- *Contribution set for $A_{(i)}$*: $S_{(i)} = \{j | G_{j(i)} = 1\}$, the index set of objectives that the agent $A_{(i)}$ contributes to.

Each agent $A_{(i)}$ generates its own control actions denoted as a vector $U_{(i)} \in \mathbb{R}^{n_{u(i)}}$. Thus, the overall control inputs $U = [U_{(1)}^T, \ldots, U_{(I)}^T]^T \in \mathbb{R}^{n_u}$ is the concatenation of all agents' control actions with $n_u = \sum n_{u(i)}$. The vector of control actions that contribute to objective O_j is defined as $U_j = [U_{(i)}^T | i \in S_j]^T \in \mathbb{R}^{n_{uj}}$, where $n_{uj} = \sum_{i \in S_j} n_{u(i)}$. It is easy to notice that U_j is a projection from U on the projection O_j. Over-lines are used to represent sequences along the prediction horizon: $\bar{U} = [(U^{(k)})^T | k = 0, \ldots, N_p - 1]^T \in \mathbb{R}^{n_u \cdot N_p}$, $\bar{U}_{(i)} = [(U_{(i)}^{(k)})^T | k = 0, \ldots, N_p - 1]^T \in \mathbb{R}^{n_{u(i)} \cdot N_p}$ and $\bar{U}_j = [(U_j^{(k)})^T | k = 0, \ldots, N_p - 1]^T \in \mathbb{R}^{n_{uj} \cdot N_p}$, where N_p is the number of steps in the prediction horizon.

A MAMO control system could be easily decoupled into fully independent subsystems for each objective if there is no shared agent ($S_j \cap S_{j'} = \emptyset$, $\forall j \neq j'$). However, this is often too idealistic in practice. An example of this type of MAMO control system with two objectives and several agents is shown in Figure 3.1.

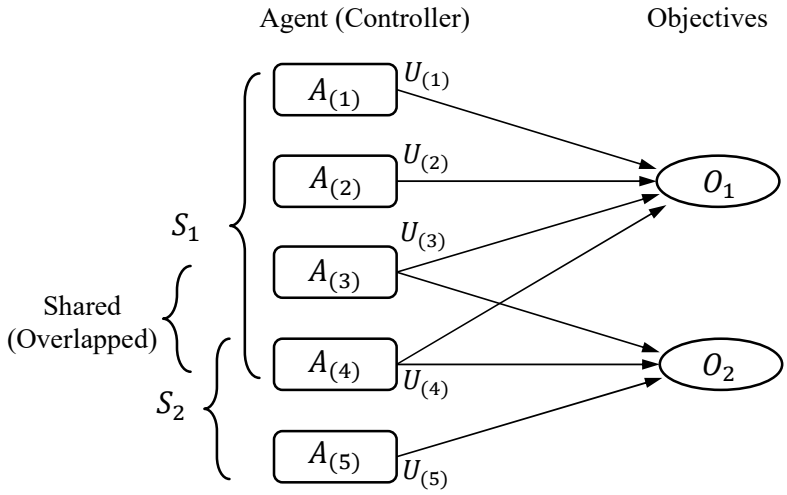


Figure 3.1: Example of a MAMO control system with five agents and two objectives. The arrow from $A_{(i)}$ to O_j represents $G_{j(i)} = 1$. For objectives: $S_1 = \{1, 2, 3, 4\}$, $S_2 = \{3, 4, 5\}$. $U_1 = [U_{(1)}^T, U_{(2)}^T, U_{(3)}^T, U_{(4)}^T]^T$, $U_2 = [U_{(3)}^T, U_{(4)}^T, U_{(5)}^T]^T$. For agents: $S_{(1)} = \{1\}$, $S_{(2)} = \{1\}$, $S_{(3)} = \{1, 2\}$, $S_{(4)} = \{1, 2\}$, $S_{(5)} = \{2\}$.

Besides, in this thesis, the following notations are used: The $n \times n$ identity matrix and the n-dimensional column vector with all elements taking the value 0 are denoted by I_n and $\mathbf{0}_n$ [1]. A symmetric positive definite (or positive semi-definite) matrix is denoted $A > 0$ (or $A \geq 0$, respectively). We define $\|x\|_2^2 = x^T x$ and $\|x\|_Q^2 = x^T Q x$ for positive semi-definite matrix Q.

3.2 Agent-Based Model Predictive Control

Suppose in a multi-agent control system, all agents contribute to the same objective. The contribution of each agent can be regarded as a two-stage approach [6]: the first stage models vehicle behaviour under the overall net forces and moments U_{CG} at the vehicle CG, and the second stage captures the actuator dynamics from control inputs $U_{(i)}$ to the corresponding $U_{CG(i)}$. An example of multi-agent systems with one common objective that uses AMPC is shown in Figure 3.2.

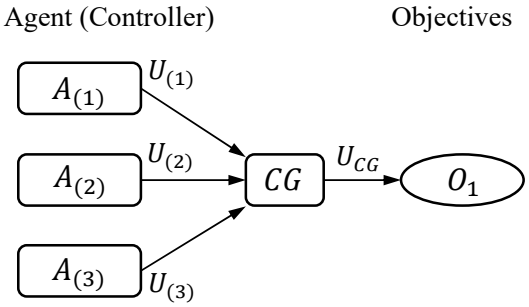


Figure 3.2: Example of a multi-agent control system with three agents and one common objective. All the agents contribute to the overall net forces and moments which affect the common objective. In this example, $U = U_1 = [U_{(1)}^T, U_{(2)}^T, U_{(3)}^T, U_{(4)}^T]^T$.

For many real-time control systems, such as vehicles, it is common to approximate NMPC as linear MPC to solve for higher computational efficiency. Therefore, this paper focuses on the optimization with quadratic cost functions, linear systems equations, and linear inequality constraints. Then, the conventional integrated control problem using the two-stage approach for the reference-tracking can be formulated as an quadratic

[1] sometimes the subscript n is omitted

programming (QP) optimization task as,

$$
\begin{aligned}
\min_{\bar{U}} \quad & J = \sum_{k=1}^{N_p} \Big(\| C(X_{CG}^{(k)} - X_{CG,ref}^{(k)}) \|_{R_x}^2 + \sum_{i=1}^{N_A} \| U_{(i)}^{(k-1)} \|_{R_{u(i)}}^2 \Big) \\
\text{s.t.} \quad & X_{CG}^{(k+1)} = A_{CGd} X_{CG}^{(k)} + B_{CGd} \sum_{i=1}^{N_A} U_{CG(i)}^{(k)} + W_{CGd}^{(k)}, \\
& U_{CG(i)}^{(k)} = A_{d(i)} X_{(i)}^{(k)} + B_{d(i)} U_{(i)}^{(k)} + W_{d(i)}^{(k)}, \\
& X_{CG}^{(k)} \in \chi_{CG}^{(k)}, X_{(i)}^{(k)} \in \chi_{(i)}^{(k)}, \\
& U_{(i)}^{(k)} \in \Omega_{(i)}^{(k)}
\end{aligned}
\tag{3.1}
$$

where J is the overall cost function, including the cost of the tracking error and the sum cost of the control input from each agent. X_{CG} and $X_{(i)}$ denote the vehicle stats at CG, and system states related to the i-th agent (i.e., the internal dynamic state of the actuator), respectively. A_{CGd}, B_{CGd}, and W_{CGd} are the discretized and linearized system matrices at the vehicle CG on the states, control inputs and the additive term, respectively. $A_{d(i)}$, $B_{d(i)}$, and $W_{d(i)}$ are the discretized and linearized system matrices for the i-th agent on the states, control inputs and the additive term, respectively. C is the observation matrix. $X_{CG,ref}$ denotes the reference states on the vehicle CG. Positive-definite R_x and semi-positive definite $R_{u(i)}$ are penalty weights for tracking error and the control action from the i-th agent, respectively. χ_{CG}, $\chi_{(i)}$, and $\Omega_{(i)}$ denote the constraint set for the overall states on the vehicle CG, the system states for the i-th agent, and the control inputs from the i-th agent, respectively. The total contributions and total penalties are summed up in the CG system equation and the cost function of integrated MPC, respectively. The integrated MPC in equation 3.1 serves as a baseline for the AMPC design as a distributed control scheme.

In AMPC, all agents are assumed to be fully connected through the control network. The overall scheme of the AMPC proposed in [6] is shown in Figure 3.3. The integrated MPC is decomposed from the agent's perspective. Each agent has a local optimization with its own agent dynamics, constraints, and local cost function. However, the cost function is not independent but is a part of the common objective. The main mechanism of the AMPC is to iteratively achieve consensus by sharing the contributions of each agent at the vehicle CG to the common objective. At the q-th iteration in one time-step, each i-th agent will receive the others' contributions $Y_{CG(i')}(i' \neq i)$ from the previous $q-1$-th iteration, solve the local optimization, and then send its updated contribution $Y_{CG(i)} = B_{CGd} U_{CG(i)}$ to the information exchange bus. The two-stage architecture facilitates information exchange between local optimizations based on a unified CG-level contribution definition.

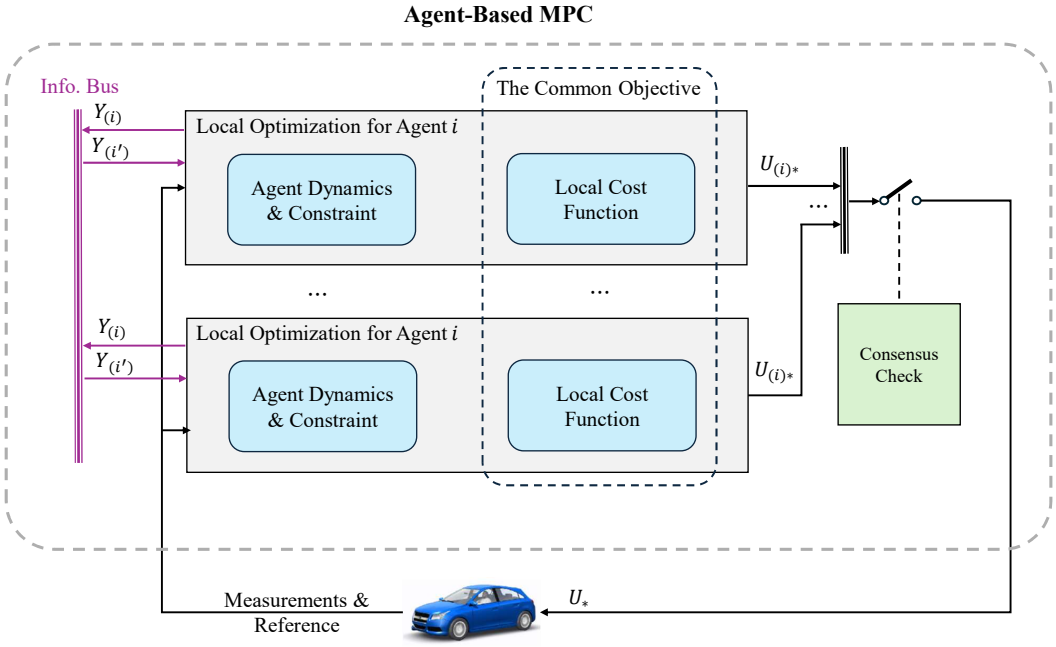


Figure 3.3: The overall structure of the AMPC. Each agent has its local optimization module. The common objective is evenly decomposed into each local cost function. Local optimizations exchange their contributions through an information bus. The AMPC will output the optimal control action as long as the consensus check is passed.

For each agent indexed by i, the local optimization in AMPC can be written as,

$$\begin{aligned}
\min_{\bar{U}_{(i)}} \quad & J_{(i)} = \sum_{k=1}^{N_p} \left(\left\| C(X_{CG}^{(k)} - X_{CG,ref}^{(k)}) \right\|^2_{\frac{R_x}{\sqrt{N_A}}} + \left\| U_{(i)}^{(k-1)} \right\|^2_{R_{u(i)}} \right) \\
\text{s.t.} \quad & X_{CG}^{(k+1)} = A_{CGd} X_{CG}^{(k)} + B_{CGd} U_{CG(i)}^{(k)} + W_{CGd}^{(k)} + \sum_{i' \neq i} Y_{CG(i')}^{(k)}, \\
& U_{CG(i)}^{(k)} = A_{d(i)} X_{(i)}^{(k)} + B_{d(i)} U_{(i)}^{(k)} + W_{d(i)}^{(k)}, \\
& X_{CG}^{(k)} \in \chi_{CG}^{(k)}, X_{(i)}^{(k)} \in \chi_{(i)}^{(k)}, \\
& U_{(i)}^{(k)} \in \Omega_{(i)}^{(k)}
\end{aligned} \tag{3.2}$$

where $\sum_{i' \neq i} Y_{CG(i')}^{(k)}$ is the sum of the contributions from all the other agents. In the local optimization in equation 3.2, the common tracking cost in equation 3.1 is evenly distributed in each local cost function $J_{(i)}$, making the sum of all the local cost functions is equal to

the global cost function in the integrated MPC.

In AMPC, iterations are required to reach an agreement among all agents about their share in the overall control efforts if there is more than one controllable agent in the control network. Within each time step, multiple iterations are allowed to reach a consensus. At each q-th iteration for the i-th agent, the receiving contributions from other agents are updated using a learning rate [85]:

$$Y_{CG(i')}^{\langle q \rangle} = \alpha Y_{CG(i')}^{\langle q \rangle} + (1-\alpha) Y_{CG(i')}^{\langle q-1 \rangle} \ , \ i' \neq i \tag{3.3}$$

where $\alpha \in [0, 1]$ denotes the learning rate. The change in the reported contribution from each agent is also monitored as an indicator of convergence. Once all agents agree on their share in the overall control distribution, the internal iteration stops, and the control command is sent to actuators. The stop criteria can be written as,

$$\| Y_{CG(i')}^{\langle q \rangle} - Y_{CG(i')}^{\langle q-1 \rangle} \|_2^2 < \epsilon_Y \ , \ \forall i \tag{3.4}$$

where ϵ_Y is the threshold for stop criteria.

By comparing equation 3.2 and equation 3.1, it can be shown that solutions determined by an agent-based approach also optimize the original integrated vehicle control problem as long as all agents are designed with fully aligned objective settings.

3.3 Alternating Direction Method of Multipliers

ADMM is a well-known method for solving optimization problems for multi-agent systems. It approximates the global optimum using iterative local optimizations on the augmented Lagrangian equation. Through the dual ascent method, it's proven to achieve residual convergence, objective convergence, and dual convergence [110], [111], [112]. The ADMM algorithm blends the techniques of dual decomposition and augmented Lagrangian methods. It is especially useful for solving large-scale optimization problems that can be decomposed into smaller sub-problems, making it particularly attractive for distributed computing environments. ADMM has seen widespread adoption in various fields, such as machine learning, signal processing, and control theory, owing to its versatility and scalability. Moreover, its extension to distributed settings, known as Consensus ADMM, has further broadened its applicability, especially in multi-agent systems where the goal is to optimize a global objective by solving local sub-problems and achieving consensus across agents.

In its most basic form, ADMM solves optimization problems of the form:

$$\min_{x,z} \; f(x) + g(z) \quad \text{subject to} \quad Ax + Bz = c, \tag{3.5}$$

where $f(x)$ and $g(z)$ are convex functions, and A and B are matrices that couple the variables x and z [2]. This formulation is particularly useful in cases where f and g have separable structures.

The augmented Lagrangian L_ρ for this problem is given by:

$$L_\rho(x, z, \gamma) = f(x) + g(z) + \gamma^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2, \tag{3.6}$$

where γ is the Lagrange multiplier and $\rho > 0$ is a positive penalty parameter. The algorithm iterates over updates for x, z, and γ, as follows:

$$x^{\langle q+1\rangle} = \arg\min_x \; L_\rho(x, z^{\langle q\rangle}, \gamma^{\langle q\rangle}), \tag{3.7}$$

$$z^{\langle q+1\rangle} = \arg\min_z \; L_\rho(x^{\langle q+1\rangle}, z, \gamma^{\langle q\rangle}), \tag{3.8}$$

$$\gamma^{\langle q+1\rangle} = \gamma^{\langle q\rangle} + \rho(Ax^{\langle q+1\rangle} + Bz^{\langle q+1\rangle} - c). \tag{3.9}$$

One of the key advantages of ADMM is its ability to decouple the objective functions f and g during optimization, making it amenable to parallel and distributed computation.

Consensus ADMM extends the basic ADMM algorithm to distributed settings, where multiple local optimizations collaboratively solve a global optimization problem by solving local sub-problems and exchanging information with their neighbours. The goal of consensus ADMM is to ensure that all local optimizations reach an agreement on the shared variables while solving their local optimization problems.

In a distributed consensus problem, each j-th optimization has a local objective function J_j, and the goal is to minimize the sum of these local objective functions while ensuring that all local optimization agrees on the value of x_j. This can be formulated as:

$$\min_x \sum_{j=1}^{N} J_j(x_j) \quad \text{subject to} \quad x_j - z_j = 0, \quad i = 1, \ldots, N. \tag{3.10}$$

[2] In this section (Section 3.3), functions f and g refer to general convex functions, variables x and z refer to general variables. A, B, and c also refer to general matrices or vectors. Therefore, their meanings may be different from those in other sections.

To achieve this global consensus problem, a common global variable z is introduced, and z_j is a projection (or a local copy) of z. The consensus constraint $x_j = z_j$ is enforced by the constraint. The augmented Lagrangian for this problem is given by:

$$L_\rho(x_j, z, \gamma_j) = \sum_{j=1}^{N} \left(J_j(x_j) + \lambda_j^T (x_j - z_j) + \frac{\rho}{2} \|x_j - z_j\|_2^2 \right). \tag{3.11}$$

Then, the local updates for the Consensus ADMM algorithm can be derived in an explicit form as follows:

$$x_j^{\langle q+1 \rangle} = \arg\min_{x_j} \; J_j(x_j) + \gamma_j^T (x_j - z_j^{\langle q \rangle}) + \frac{\rho}{2} \|x_j - z_j^{\langle q \rangle}\|_2^2, \tag{3.12}$$

$$z^{\langle q+1 \rangle} = \frac{1}{N} \sum_{j=1}^{N} \left(x_j^{\langle q+1 \rangle} + \frac{\gamma_j^{\langle q \rangle}}{\rho} \right), \tag{3.13}$$

$$\gamma_j^{\langle q+1 \rangle} = \gamma_j^{\langle q \rangle} + \rho (x_j^{\langle q+1 \rangle} - z_j^{\langle q+1 \rangle}). \tag{3.14}$$

In Consensus ADMM, the updates for x_j are performed locally by each agent, while the update for z requires communication between neighbouring agents. The update for the dual variables γ_j enforces the consensus constraint. The Consensus ADMM algorithm can be summarized as follows:

Algorithm 1 Consensus ADMM Algorithm

Input: Local objective functions $J_j(x_j)$, penalty parameter ρ, initial values for $x_j^{\langle 0 \rangle}$, $z^{\langle 0 \rangle}$, and $\gamma_j^{\langle 0 \rangle}$.
for each iteration q **do**
    for each local optimization i **do**
        Update $x_j^{\langle q+1 \rangle} \leftarrow \arg\min_{x_j} \; J_j(x_j) + \gamma_j^T (x_j - z_j^{\langle q \rangle}) + \frac{\rho}{2} \|x_j - z_j^{\langle q \rangle}\|_2^2$
    end for
    Update $z^{\langle q+1 \rangle} \leftarrow \frac{1}{N} \sum_{j=1}^{N} \left(x_j^{\langle q+1 \rangle} + \frac{\gamma_j^{\langle q \rangle}}{\rho} \right)$
    for each local optimization i **do**
        Update $\gamma_j^{\langle q+1 \rangle} \leftarrow \gamma_j^{\langle q \rangle} + \rho (x_j^{\langle q+1 \rangle} - z_j^{\langle q+1 \rangle})$
    end for
end for

The ADMM is a versatile optimization algorithm that has found widespread use in various fields, including machine learning, signal processing, and distributed control systems. Its ability to decompose large-scale problems into smaller, more manageable sub-problems makes it well-suited for distributed computing environments. Consensus ADMM, in particular, extends ADMM to distributed optimization problems, enabling multiple agents to collaborate on solving a global objective while maintaining consensus on shared variables. Recent advances in ADMM research, including improved convergence rates and inexact methods, continue to expand its applicability and efficiency.

3.4 Gaussian Process Regression

The field of machine learning has branched into several sub-fields dealing with different types of learning tasks. One of the distinctions to note is the difference between supervised, unsupervised, and semi-supervised learning (mainly under reinforcement learning) [113]. The primary objective of supervised learning is to use observed historical data to build a mathematical tool (a formula or an algorithm) to predict unobserved data, more statistically speaking, to optimize the prediction accuracy [114]. Another distinction is to be classified as either parametric learning or non-parametric learning. Parametric models have the advantage of often being faster to use but the disadvantage of making stronger assumptions about the nature of the data distributions. If the model is assumed based on a particular class of functions (e.g., linear functions) and the target function is not well-modelled by this class, then the predictions will be poor. However, non-parametric models are often computationally intractable for large datasets, but they are more flexible and could usually achieve better performance due to fewer assumptions about the data. Moreover, the form of the output or response variable could be another distinction. The output can, in principle, be anything. However, most methods assume that it is a categorical or nominal variable from some finite set or a real-valued scalar. When the output is categorical, the problem is known as classification or pattern recognition, and when it is real-valued, the problem is known as regression [115]. The focus of the learning method in this study is on supervised non-parametric regression.

The biggest challenge for non-parametric regression is the lack of an explicit form of the latent model. One approach to this problem is to define a generative model for the data, use the inferred latent representation and the model parameters as features, and then plug these features into standard methods—for example, deep learning methods [116]. The second approach is to give a prior probability to every possible function, where higher probabilities are given to functions considered more likely. Thus, the non-parametric

kernel-based regression problem introduced in Appendix A is considered in this study.

Let $\mathcal{D}_{N_d} = (\boldsymbol{x}, \boldsymbol{y})$ denotes the dataset with N_d data points (x_p, y_p) where $p = 1, \ldots, N_d$. $\boldsymbol{x} = [x_1, \ldots, x_{N_d}]^T \in \mathbb{R}^{N_d \times n_d}$ denotes the input matrix, where $x_p \in \mathbb{R}^{n_d \times 1}$ is the input vector of the data point and n_d is the dimension of the input vector. $\boldsymbol{y} = [y_1, \ldots, y_{N_d}]^T \in \mathbb{R}^{N_d \times 1}$ denotes the output vector, where $y_p \in \mathbb{R}^1$ is the output variable of the data point.

Assume the y_p is related to x_p through [117],

$$y_p = f(x_p) + \varepsilon_p. \tag{3.15}$$

where ε_p are additive noises and follow independent, identically distributed Gaussian distributions with zero mean and variances $\sigma^2_{N_d}$. As introduced in Appendix A.1, the estimated latent function $\hat{f}(x)$ could be formatted by the kernel trick, where kernels compute an inner product in a high-dimensional (implicit) feature space [115].

Also, the prediction $\hat{f}(x)$ could be regarded as a linear smooth function of outputs $\boldsymbol{y}$. Thus, the model selection criteria used in multiple linear regression also applies to this study. The commonly used standards can be roughly divided into three categories: (a) predictive risk function, e.g., prediction mean squared error (PMSE) and Stein's unbiased risk estimator; (b) coefficient of determination: R^2_{adj}; (c) likelihood-based information criteria: Mallows' C_p, Akaike information criterion (AIC), and Bayesian information criterion (BIC), as introduced in Appendix A.2.

GPR has become a popular machine learning tool for lots of applications in engineering problems [118]. It is defined in [119] that a GP is a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions. The Gaussian distribution is over vectors, whereas the GP is over functions such that a stochastic process governs the properties of functions. GP modelling differs from other "black-box" identification approaches in that it does not try to approximate the modelled system by fitting the parameters of the selected basis functions. Instead, it searches for the relationship among the measured data. The obtained research results indicated that GP models might be attractive in engineering practice for the following reasons [120]: (a) they are simple to use in system identification due to the relatively small number of design decisions; (b) it is possible to include various sorts of prior knowledge about the modelled process in the model; (c) they work well although the identification output data is noisy; (d) the model prediction contains a measure of confidence, which can be exploited in many ways for system identification, control system design and model-based fault detection.

GP governs the joint Gaussian distribution of outputs $\boldsymbol{y}$ and the prediction $\hat{y}_*$ for the

testing point (x_*, y_*), as follows:

$$\begin{bmatrix} \boldsymbol{y} \\ \hat{y}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I & K(\boldsymbol{x}, x_*) \\ K(x_*, \boldsymbol{x}) & K(x_*, x_*) \end{bmatrix} \right) \tag{3.16}$$

The predicted mean $\hat{y}_*$ and variance Σ_* could be expressed as,

$$\begin{aligned} \hat{y}_* &= K(x_*, \boldsymbol{x})[K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1} \boldsymbol{y} \\ \Sigma_* &= K(x_*, x_*) - K(x_*, \boldsymbol{x})[K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1} K(x_*, x_*) \end{aligned} \tag{3.17}$$

where $K(\cdot, \cdot)$ is the covariance (kernel) matrix whose elements are kernel functions of data pairs. The basic GPR algorithm in equation 3.17 has $\mathcal{O}({N_d}^3)$ time complexity and $\mathcal{O}({N_d}^2)$ memory complexity, where N_d is a size of the dataset N_d [118]. And the inverse process of the Gramian matrix $[K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1}$ is the most computationally expensive part in GPR.

As for Bayesian paradigms, the marginal likelihood function is usually used to train the hyper-parameters $\boldsymbol{\theta}$,

$$p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = \int p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w}) p(\boldsymbol{w} \mid \boldsymbol{\theta}) \, d\boldsymbol{w} \tag{3.18}$$

where $\boldsymbol{w}$ is the vector of parameters, $\boldsymbol{\theta}$ is the hyper-parameter, $p(\boldsymbol{w} \mid \boldsymbol{\theta})$ is the parameter prior, the $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{w})$ is the likelihood, and the $p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta})$ is the marginal likelihood.

As a Bayesian paradigm, GPR hyper-parameter training is usually conducted by maximizing the log marginal likelihood function:

$$\log p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = -\frac{1}{2} \boldsymbol{y}^\top [K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1} \boldsymbol{y} - \frac{1}{2} \log \left| K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I \right| - \frac{n}{2} \log 2\pi \tag{3.19}$$

The three terms of the marginal likelihood in equation 3.19 have readily interpretable roles: the only term involving the observed targets $(-\frac{1}{2} \boldsymbol{y}^\top [K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1} \boldsymbol{y})$ is the data-fit; the term $\frac{1}{2} \log \left| K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I \right|$ is the complexity penalty depending only on the covariance function and the inputs, and $-\frac{n}{2} \log 2\pi$ is a normalization constant. The (log-) marginal likelihood function considers both fitting accuracy and model complexity. So, over-fitting models would be penalized.

To set the hyper-parameters by maximizing the marginal likelihood, the partial derivatives of the marginal likelihood w.r.t. the hyper-parameters is a standard tool:

$$\frac{\partial}{\partial \theta_h} \log p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{2} \mathrm{tr} \left(\left(\alpha_y \alpha_y^T - K_y^{-1} \right) \frac{\partial K_y}{\partial \theta_h} \right) \tag{3.20}$$

where θ_h is the h-th hyper-parameter in $\boldsymbol{\theta}$. $K_y = K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I$ is the Gramian matrix and $\alpha_y = K_y^{-1} \boldsymbol{y}$.

3.5 Chance Constraint

The chance constraint is an important tool in stochastic MPC. For a system subject to a specific constraint $h(X, \delta) \leq 0$ where $X \in \mathbb{R}^N$ is the system variable and $\delta \in \mathbb{R}^d$ is the uncertain parameter [3], the chance constraint can be written as,

$$\Pr(h(X, \delta) \leq 0) \geq 1 - \varepsilon_c \tag{3.21}$$

where $\varepsilon_c \in [0, 1]$ quantifies the admissible probability of constraint violation. If uncertainty is unbounded, allowing some degree of violation is a necessary but insufficient condition for feasibility, which is also acceptable in practice.

3.6 Summary

This chapter presents the background materials used in this study. First, the concept and notations of MAMO control systems are introduced. Then, the recently proposed AMPC scheme is reviewed. The AMPC can decompose a multi-agent system with one common objective, which is a special topology of MAMO control systems. Next, the ADMM method is introduced, which will be tailored to construct the MO-AMPC in the following sections. Last, the basics of non-parametric kernel-based regression, which would be used as the learning method in this thesis, are introduced. Based on that, GPR formulations with the computational cost concern and the marginal likelihood function for hyper-parameter training are given. Finally, the concept of chance constraint is introduced.

[3] In this section (Section 3.5), functions h refer to general inequality constraints, variables X and parameter δ refer to general system variable and uncertain parameter, respectively. Therefore, their meanings may be different from those in other sections.

Chapter 4

Learning Agent-Based Model Predictive Control

This chapter presents the L-AMPC scheme for HVC. It is structured as follows: First, the vehicle model and agent configuration in this study are introduced. The overview design of the L-AMPC scheme is given in Section 4.1. Then, the data-driven learning process is introduced in Section 4.2, and the model-based soft stochastic control scheme is introduced in Section 4.3. Following an example of yaw stability control in Section 4.4, simulation results are demonstrated in Section 4.5.

4.1 Overview

As introduced in Section 1.3, the L-AMPC control scheme proposed in this chapter solves the first pain point by learning the unknown contributions in the multi-agent system in a data-driven way to make the prediction model more accurate.

4.1.1 System with Unknown Agent

The multi-agent system studied in this section is the same as the system used for AMPC in Section 3.2, where multiple controllers are regarded as agents contributing to the common objective. To systematically evaluate the contributions of different actuators, the two-stage approach is adopted in AMPC: the first stage models vehicle dynamical behaviour under

net forces and moments at vehicle CG, while the second stage captures the actuator dynamics from control inputs to the above-mentioned net forces and moments. Under the assumption from Figure 3.3 that agents will coordinate with their neighbours in the control network, each agent i should report its expected contribution term $Y_{CG(i)} = B_{CGd}U_{CG(i)}$ to other agents. However, this requirement is often too idealistic in practice. In real vehicle development, not all agents would be controllable or designed to report the contribution term $Y_{CG(i)}$ to other agents. That will eventually cause a significant error in vehicle behaviour prediction if directly apply the model in Equation 3.2, leading to unexpected or even harmful control results.

Thus, all agents could be classified into two types: known agent (KA) includes the controllable agent that is fully controlled by L-AMPC controller, and the "white-box" agent that is uncontrollable but will report the contribution to others. unknown agent (UA) is also called "black-box" agent, whose contribution is completely unknown.

Table 4.1: Agent Type and Definition

Type		Description
Known Agent (KA)	Controllable Agent	- It's fully controlled by L-AMPC.
	White-box Agent	- It's un-controllable. - It will report the contribution to others.
Unknown Agent (UA)	Black-box Agent	- It's un-controllable. - It will NOT report the contribution to others. - Its control police is unknown.

An example of multi-agent systems with different types of agents is shown in Figure 4.1. The goal of the L-AMPC is to learn the contribution of the "black-box" agent to complete the system model in the MPC for more accurate control.

4.1.2 Control Scheme Overview

The overall structure of the proposed L-AMPC scheme is shown in Figure 4.2. The learning process consists of data preparation, data management, and GPR learning. The predicted mean and variance from GPR will finally be used to reconstruct the stochastic AMPC.

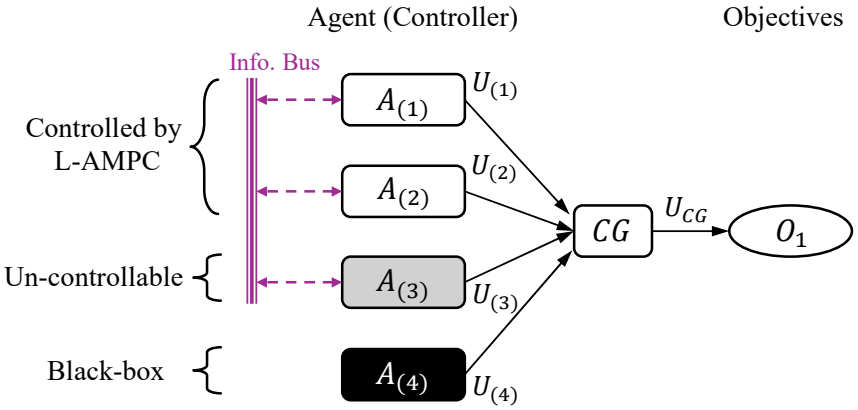


Figure 4.1: Example of a multi-agent control system with different types of agents. All the agents contribute to the overall net forces and moments which affect the common objective. In this example, agent $A_{(1)}$ and $A_{(2)}$ are controllable agents controlled by L-AMPC controllers. Agent $A_{(3)}$ is a "white-box" agent. Agent $A_{(4)}$ is a "black-box" agent.

- Data preparation first pairs the outputs and inputs at the previous step as a new data point for the dataset. Unknown contributions to vehicle dynamics will be estimated as the outputs. Inputs are measurements or vehicle states determined by model selection methods.
- Data management includes data collection and subset selection. The new paired data point will be conditionally collected into the dataset. A subset selection is required to accelerate the inference using a small number of data points from the dataset. The dataset could also be saved offline for model selection and hyper-parameter training.
- GPR learning includes offline hyper-parameter training and online inference. The distributions of unknown contributions along the horizon will be predicted with mean values and variance.
- The mean and variance of GPR predictions are used separately: The mean is used to complete the system model in MPC as an additive term to make it more accurate, as the total contribution will be compensated. Meanwhile, the soft chance constraints will be formulated based on the propagation of the prediction variance. The re-constructed AMPC will generate optimal and safety control commands for each controllable agent.

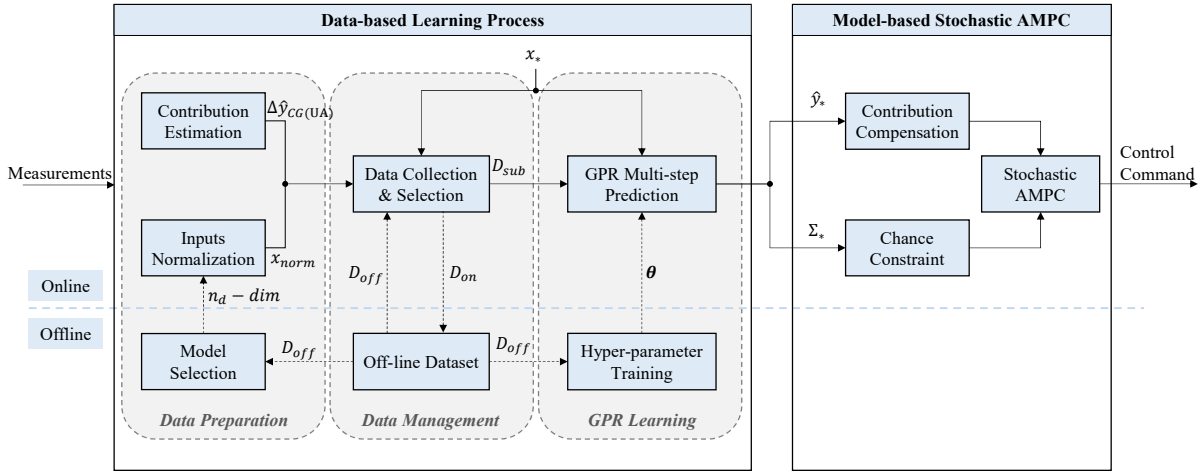


Figure 4.2: The overall structure of the L-AMPC. The $\Delta\hat{y}_{CG(\mathrm{UA})} \in \mathbb{R}^{N_p \times 1}$ is the unknown contribution estimated as the output of the data point vector along the horizon from Equation 4.2. The $x_{norm} \in \mathbb{R}^{n_d \times 1}$ is the normalized data input where n_d is the number of selected model variables based on the off-line dataset described in Section 4.2.1. D_{off}, D_{on}, and D_{sub} denote the off-line, on-line, and the selected (sub-)datasets, respectively. $\boldsymbol{\theta}$ is the hyper-parameters of the GP model that trained on the off-line dataset. $\hat{y}_* \in \mathbb{R}^{N_p}$ and $\Sigma_* \in \mathbb{R}^1$ are the GPR prediction mean values along the horizon and the variance, respectively.

4.2 Data-Driven Learning Process

4.2.1 Data Preparation

Estimation of unknown contribution

In many practical situations, the output of the black box controller cannot be accurately measured alone. For example, a typical torque sensor can only measure the torque of the motor's total output but cannot directly measure the value of the differential torque alone. In this case, the output variables in the supervise-learning dataset cannot be measured.

Instead, the contribution of unknown agents could be estimated from vehicle dynamics on the CG level regarding the type or inconvenient measurement of the unknown agents. The system observation is the combined results from known and unknown agents, which provides a hint for estimating the contribution of the unknown agent. Define $y_{CG} = CX_{CG}$ is the observation which could usually be measured or estimated. The system equation

from Equation 3.1 in the integrated MPC at time step (t) can be written as,

$$\begin{aligned} y_{CG}^{(t+1)} - CA_{CGd}C^{\dagger}y_{CG}^{(t)} &= CB_{CGd}\sum_{i=1}^{N_A} U_{CG(i)}^{(t)} + CW_{CGd}^{(t)} \\ &= CB_{CGd}\Big(U_{CG(\mathrm{KA})}^{(t)} + U_{CG(\mathrm{UA})}^{(t)}\Big) + CW_{CGd}^{(t)} \end{aligned} \tag{4.1}$$

where C is the observation matrix, and its pseudo-inverse is denoted as $C^{\dagger}$. $U_{CG(\mathrm{KA})}$ and $U_{CG(\mathrm{UA})}$ are the sum of the control effort at CG from the known agents and the unknown agents, respectively. Let $\Delta y_{CG(\mathrm{KA})} = CB_{CGd}U_{CG(\mathrm{KA})}^{(t)} + CW_{CGd}^{(t)}$, $\Delta y_{CG(\mathrm{UA})} = CB_{CGd}U_{CG(\mathrm{UA})}^{(t)}$, and $\Delta y_{CG(\mathrm{total})} = y_{CG}^{(t+1)} - CA_{CGd}C^{\dagger}y_{CG}^{(t)}$ demote the contribution on the observation in one time-step from the known agents, unknown agents, and all agents, respectively. The estimated unknown contribution $\Delta\hat{y}_{CG(\mathrm{UA})}$ can be derived as,

$$\begin{aligned} \Delta\hat{y}_{CG(\mathrm{UA})}^{(t)} =& \Delta y_{CG(\mathrm{total})}^{(t)} - \Delta y_{CG(\mathrm{KA})}^{(t)} \\ =& y_{CG}^{(t+1)} - CA_{CGd}C^{\dagger}y_{CG}^{(t)} - CW_{CGd}^{(t)} - CB_{CGd}U_{CG(\mathrm{KA})}^{(t)} \end{aligned} \tag{4.2}$$

This estimation makes the proposed L-AMPC apply to any type and number of unknown agents.

Model selection

Model selection is used to select the most n_d relevant input variable combinations x for the dataset. Since kernel-based regression, e.g., GPR, could be regarded as a linear smoother [121], the model selection criteria used in multiple linear regression also apply to this study. The commonly used criteria can be roughly divided into three categories: 1) predictive risk function, e.g., PMSE; 2) coefficient of determination R_{adj}^2; 3) likelihood-based information criteria, e.g., AIC and BIC. A dataset including possible input variables needs to be collected first, and then the method described above can be applied offline. The forward selection could find the smallest subset. It will start from the minimum subset and then add variables while comparing the above criteria for each selection to find the best one.

Data pre-processing

The ranges of the black-box agent's input variables may vary significantly due to their different physical meanings and units. Therefore, data scaling is needed as a pre-process. In general, data normalization and standardization are the two most used methods.

Normalization:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4.3}$$

where x_{min} and x_{max} are the minimum and maximum values in the dataset.

Standardization:

$$x_{std} = \frac{x - \mu}{\delta} \tag{4.4}$$

where μ and δ are the mean and standard deviation of the dataset.

Standardization usually works best when the variable follows a Gaussian distribution, while normalization has no requirement on the variable distribution and keeps the same distribution as the original data. Besides, normalization usually means to strictly scale a variable to a desired range, and it is useful when the data has varied scales and the algorithm does not make assumptions about the data distribution, such as k-nearest neighbors (KNN) and Pearson correlation coefficient (PCC). Thus, data normalization is used for pre-processing in this thesis.

4.2.2 Data Management

Data collection

The online process of data management, mainly the data collection and selection modules, ensures the continuous learning ability and high efficiency of L-AMPC.

A new data pair will be generated every moment during the controller's real-time operation. The input of the data pair is the normalized input variables from Equation 4.3 with selected dimension as described in Section 4.2.1. Therefore, when processing a large amount of repetitive and worthless data, the data collection module should manage the data density while preventing the data from being too aggregated or clustered. In this study, the minimum Euclidean distance is used to determine whether a new data point should be added or replaced with an original point. Three strategies shown in Figure 4.3 (a)-(c):

1. *Add the new input point*: the new point is far from the existing points in the dataset, and the dataset is not full. A 2-D example is shown in Figure 4.3 (a).

2. *Replace the farthest point*: the new point is far away from the existing points in the training dataset, and the dataset is full. A 2-D example is shown in Figure 4.3 (b).

3. *Replace the nearest point*: the new point is very close to it, and the output error is significant, which means the new point brings new information that needs updating. A 2-D example is shown in Figure 4.3 (c).

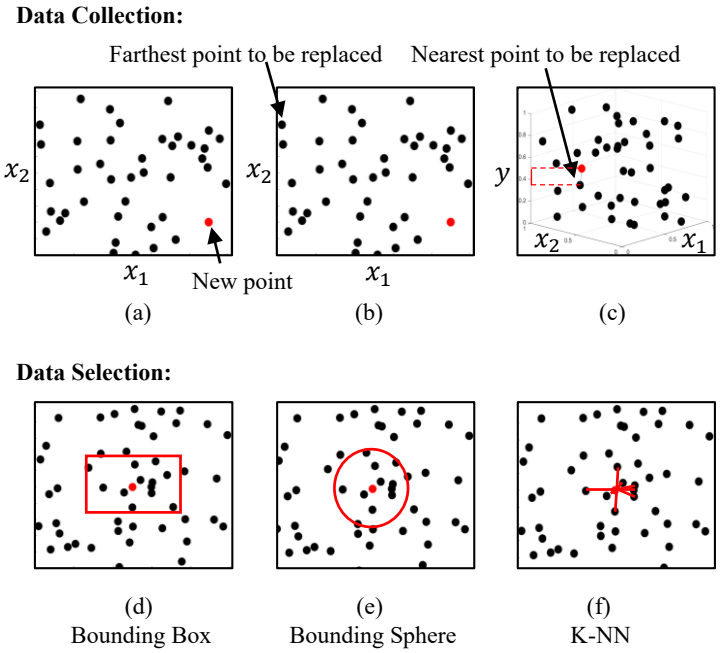


Figure 4.3: Data management strategies. (a)-(c) are data collection strategies: (a) Add the new point; (b) Replace the farthest point with the new point; (C) Replace the nearest point with the new point. (d)-(e) are subset selection strategies: (d) Select points in the bounding box; (e) Select points in the bounding sphere; (f) Select the nearest K neighbour points.

Subset selection

As introduced, the inverse process of the Gramian matrix K_y^{-1} is the most computationally expensive part of GPR. This study adopts three methods for the subset of data (SoD) selection: bounding box, bounding sphere, and KNN. The core idea is to make the data sparse to reduce the dimension of the Gramian matrix. The schematic diagrams are shown in Figure 4.3 (d)-(f). All three methods are based on Euclidean distance. The time complexity will reduce from $\mathcal{O}(N_D^3)$ to $\mathcal{O}(N_d^3)$ if only $N_d \leq N_D$ data points are selected.

4.2.3 Learning Process of Gaussian Process Regression

Hyper-parameter training

As introduced in Appendix A.3, compared to error-based loss functions, using the likelihood function as the optimization objective for hyper-parameter training is more suitable for Bayesian paradigms. Because the likelihood function considers both fitting accuracy and model complexity would penalize the over-fitting models. Thus, for GPR learning in the L-AMPC scheme, the log marginal likelihood function Equation 3.19 is used as the objective function, and its partial derivatives w.r.t. the hyper-parameters Equation 3.20 is used for solving the optimization problems based on gradient search method. The training process is processed offline based on an offline dataset D_{off}.

The Gaussian kernel used in this study is,

$$K(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{n_d}\frac{(x_i - x_i')^2}{l_i^2}\right) \tag{4.5}$$

where σ_f is the signal standard deviation. l_i are length scales for input variables. Therefore, there are a total of $n_d + 2$ hyper-parameters that need to be trained: n_d length scales, one signal standard deviation σ_f, and one noise standard deviation σ_{N_d} for the additive noises in Equation 3.15.

Multi-step GPR prediction

The prediction function in Equation 3.17 and the kernel function in Equation 4.5 is used to conduct the GPR inference process. The dataset is collected based on the estimated contribution of unknown agents shown and further filtered by KNN for computational acceleration.

In MPC, it is not just one step forward that needs to be predicted but the entire horizon. Conventional GP-MPCs use single-step GPR prediction with some assumptions (e.g., constant assumption or linear assumption) to predict all the steps along the horizon, but those methods fail to take full advantage of data-based machine-learning approaches. Therefore, some studies (e.g., in [122]) suggest a multi-step prediction method where GPR predicts all steps along the horizon without artificial assumptions. However, as shown in Figure 4.4, these traditional multi-step prediction methods are essentially iterations of multiple single-step predictions. Each step's prediction requires the previous step's prediction value, which means the input matrix $\boldsymbol{x}$ needs to be updated at each prediction

step along the horizon. Therefore, those methods are feasible but could be a computational disaster for real-time implementation, especially when the horizon is long. As introduced in Section 3.4, the inverse process of the Gramian matrix $[K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1}$ is the most computationally expensive part in GPR, with $\mathcal{O}({N_d}^3/6)$ computational complexity even if the Cholesky decomposition is used. Each one-step prediction needs to update the input matrix $\boldsymbol{x}$ for those traditional multi-step prediction methods. So, the inverse process is required each time, a computational disaster for real-time implementation.

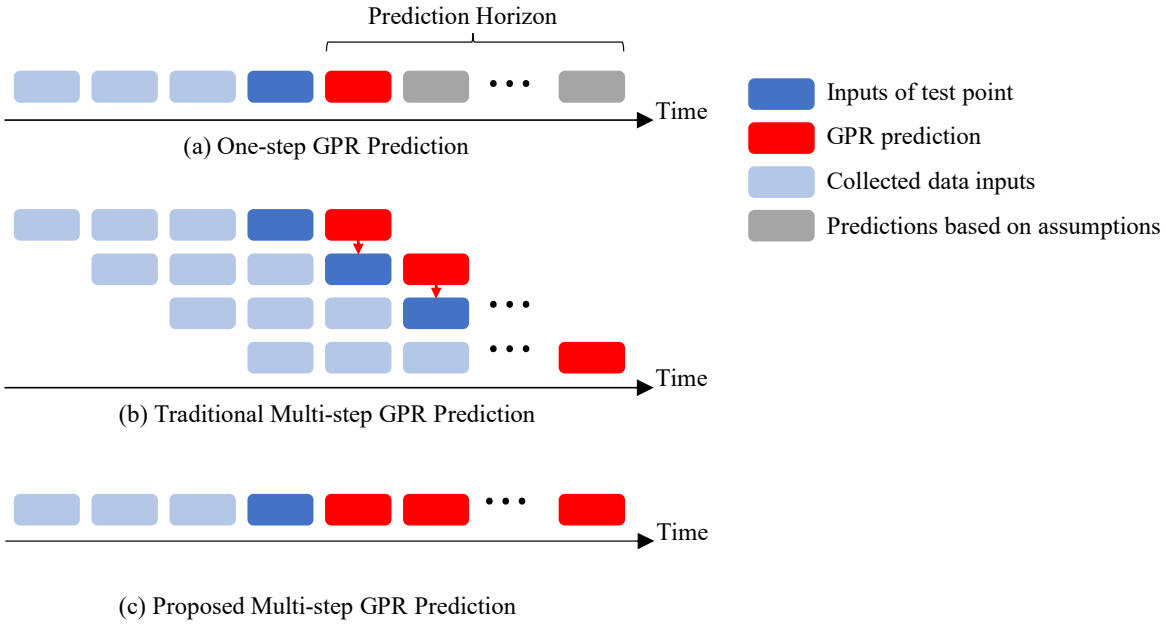


Figure 4.4: Schematic diagram of data collection for multi-step prediction.

This study proposes a novel and efficient multi-step prediction method. The main idea is, at each time step, to predict all steps along the horizon at once using a single input matrix $\boldsymbol{x}$, as shown in Figure 4.5. The key is the way of collecting the data. For example, at the current time step t with N_p step horizon, the collection of input matrix x_p starts from the time step $t - N_p$ and goes earlier. At the same time, the output data is a vector y_p with N_p elements. In this case, the input matrix $\boldsymbol{x}$ is fixed for predicting all the steps along the horizon. Thus, the inverse process of the Gramian matrix is only needed once at each time step. Besides, the GPR mean prediction can be calculated by simple matrix operations, where replace the vector $\boldsymbol{y}$ $(N_d \times 1)$ with the matrix $\boldsymbol{y}$ $(N_d \times N_p)$ in Equation 3.17. Thus, the predicted mean value $\hat{y}_*$ has the dimension of $N_d \times 1$.

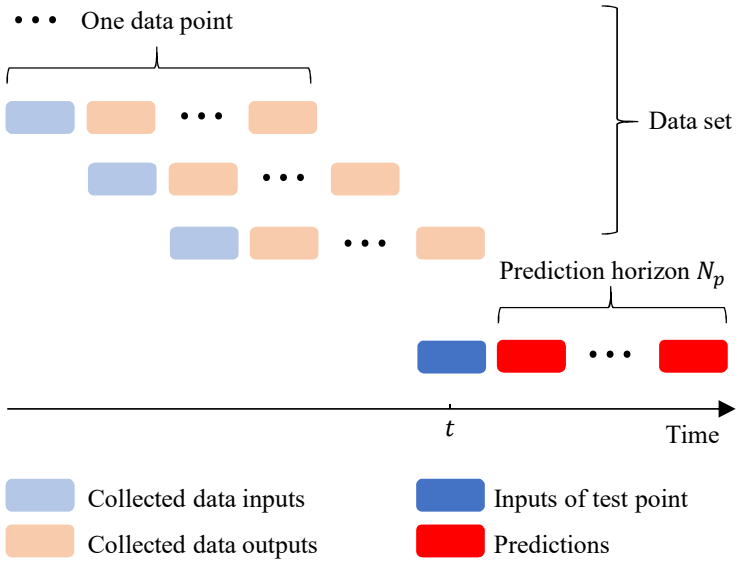


Figure 4.5: Different prediction methods along the horizon.

4.3 Model-Based Soft Stochastic Control Scheme

This section will discuss methods for applying GPR predictions of unknown agents' contribution to the AMPC scheme, especially ensuring constraint satisfaction under the prediction uncertainties.

4.3.1 Contribution Compensation and Uncertainty Propagation

There are generally two ways to use the GPR predictions to reconstruct the AMPC scheme. The first method is to subtract the unknown contribution predicted by GPR from the original desired tracking reference to formulate a new cost function. Then, the remaining reference from the subtraction is exactly the desired contribution of all the known agents. The second method is to simply complete the system model using the GPR prediction results while the desired reference remains unchanged. It is easy to prove that those two methods will have the exact same results on control police when solving the batch optimization problem. The latter is used in this thesis.

For each controllable agent i in the AMPC Equation 3.2, the system equation at CG

can be completed using the GPR prediction results as,

$$\begin{aligned} CX_{CG}^{(k+1)} &= CA_{CGd}X_{CG}^{(k)} + CB_{CGd}U_{CG(i)}^{(k)} + CW_{CGd}^{(k)} + C\sum_{i'\neq i} Y_{CG(i')}^{(k)} + \Delta\hat{y}_{CG(\mathrm{UA}),prd}^{(k)} \\ U_{CG(i)}^{(k)} &= A_{d(i)}X_{(i)}^{(k)} + B_{d(i)}U_{(i)}^{(k)} + W_{d(i)}^{(k)} \end{aligned} \tag{4.6}$$

where $\Delta\hat{y}_{CG(\mathrm{UA}),prd} \sim \mathcal{N}(\hat{y}_* \in \mathbb{R}^{N_p\times 1}, \Sigma_* \in \mathbb{R}^1)$ denotes the predicted unknown contributions along the horizon. $\Delta\hat{y}_{CG(\mathrm{UA}),prd}^{(k)} \sim \mathcal{N}(\hat{y}_*^{(k)} \in \mathbb{R}^1, \Sigma_* \in \mathbb{R}^1)$ denotes the predicted unknown contribution at the k-th step in the horizon. $\hat{y}_*$ and Σ_* are the predicted mean and variance from GPR in Equation 3.17, respectively. $\hat{y}_*^{(k)}$ is the k-th element in $\hat{y}_*$.

Since the $\Delta\hat{y}_{CG(\mathrm{UA}),prd}^{(k)}$ acts as an additional disturbance, the uncertainty from GPR predictions will propagate along the horizon through the system equation in (4.6). It is known that the uncertainty will increase along the horizon if the open-loop propagation is used, which is a risk of infeasibility if the uncertainty from GPR is significant (e.g., driving into a new scenario without previous experience.) or the horizon N_p is long. Therefore, the closed-loop propagation with a linear feedback assumption is considered, which can be written for the i-th agent as,

$$U_{(i)}^{(k)} = v_{(i)}^{(k)} + K_{(i)}(CX_{CG}^{(k)} - Cz_{CG}^{(k)}) = v_{(i)}^{(k)} + K_{(i)}e_{CG}^{(k)} \tag{4.7}$$

where $v_{(i)}^{(k)}$ is the nominal control action, $z_{CG}^{(k)}$ is the nominal (noise-free) states on CG, $e_{CG}^{(k)} \sim \mathcal{N}(0, \Sigma_e^{(k)})$ is the unbiased disturbance terms with the uncertainty of error dynamics between the actual observation $CX_{CG}^{(k)}$ and the nominal observation $Cz_{CG}^{(k)}$, where $\Sigma_e^{(k)}$ is the variance of the disturbance at step k. $K_{(i)}$ is the feedback matrix. Based on the reconstructed system Equation 4.6, the nominal states $z_{CG}^{(k)}$ and nominal control action $v_{(i)}^{(k)}$ follow the following dynamics:

$$\begin{aligned} Cz_{CG}^{(k+1)} =& CA_{CGd}z_{CG}^{(k)} + CB_{CGd}B_{d(i)}v_{CG(i)}^{(k)} + CW_{CGd}^{(k)} + C\sum_{i'\neq i} Y_{CG(i')}^{(k)} \\ &+ CB_{CGd}A_{d(i)}X_{(i)}^{(k)} + CB_{CGd}W_{d(i)}^{(k)} + \hat{y}_*^{(k)} \end{aligned} \tag{4.8}$$

Substituting Equation 4.7 into Equation 4.8 gives the uncertainty dynamics as,

$$\begin{aligned} e_{CG}^{(k+1)} &= \Big(CA_{CGd} + CB_{CGd}B_{d(i)}K_{(i)}\Big)e_{CG}^{(k)} + \omega_{CG,prd} \\ &= \tilde{A}_{e(i)}e_{CG}^{(k)} + \omega_{CG,prd} \end{aligned} \tag{4.9}$$

where $\omega_{CG,prd} \sim \mathcal{N}(0, \Sigma_*)$ are unbiased disturbance terms with the uncertainty from GPR prediction. Thus, the system dynamics on vehicle states $CX_{CG} = Cz_{CG} + e_{CG}$ could be regarded as the combination of the noise-free nominal dynamics of Cz_{CG} and the error dynamics e_{CG}. From Equation 4.9, It can be seen that the uncertainty of e_{CG} comes from two parts: GPR prediction on $\omega_{CG,prd}$ and inheritance from the previous step $\tilde{A}_{e(i)} e_{CG}^{(k)}$. Therefore, the variance of uncertainty in the horizon can be derived as,

$$\Sigma_e^{(k+1)} = \tilde{A}_{e(i)} \Sigma_e^{(k)} \tilde{A}_{e(i)}^T + \Sigma_*, \;\; k = 1, \ldots, N_p \tag{4.10}$$

Thus, by choosing feedback $K_{(i)}$ to stabilize the uncertainty dynamics $\tilde{A}_{e(i)}$, the increase of e_{CG} along the horizon will asymptotically close to zero.

The unbounded distribution at each step e_{CG}^k can be turned into a bounded polygon $\mathcal{E}_{CG}^{(k)}(\varepsilon_c)$ if manually set a confidence level $(1 - \varepsilon_c)$ in the quantile function, where ε_c is the admissible probability of constraint violation (e.g., 2.5%) in the chance constraint in (3.21). The uncertainty of the error dynamics will be further propagated to vehicle states and control actions. Thus, the constraints should be shrunken as,

$$\begin{aligned} z_{CG}^{(k)} &\in \chi_{CG}^{(k)} \ominus \mathcal{E}_{CG}^{(k)}(\varepsilon_c) \\ v_{(i)}^{(k)} &\in \Omega_{(i)}^{(k)} \ominus K_{(i)} \mathcal{E}_{CG}^{(k)}(\varepsilon_c) \end{aligned} \tag{4.11}$$

where $\ominus$ denotes the Minkowski subtraction.

4.3.2 Soft Stochastic Control Scheme

The stochastic L-AMPC is now an optimization on the noise-free nominal system of $v_{(i)}^{(k)}$ and $z_{CG}^{(k)}$, with shrunken constrain polygons at each time step. The optimal control law for the i-th agent should be,

$$U_{(i)*}^{(k)} = v_{(i)*}^{(k)} + K_{(i)}(CX_{CG}^{(k)} - Cz_{CG*}^{(k)}) \tag{4.12}$$

where $U_{(i)*}^{(k)}$ is the optimal control action as the final output from L-AMPC. $v_{(i)*}^{(k)}$ and $z_{CG*}^{(k)}$ are the optimal control action and optimal nominal states from solving the stochastic optimization, respectively. Theoretically, Equation 4.12 is intractable to solve because the state $X_{CG}^{(k)}$ is unknown in the horizon. However, in MPC context, only the first step needs to be applied as the control input, where $U_{(i)*}^{(0)} = v_{(i)*}^{(0)}$ because $X_{CG}^{(0)} = z_{CG*}^{(0)}$. Thus, we can still use the batch formulation to solve the optimization in real-time and only apply the $v_{(i)*}^{(0)}$ as the control action to the system.

Although the feedback assumption is applied, the feasibility of the optimization is still not guaranteed. Therefore, softened chance constrain (SCC) are further considered to guarantee the optimization feasibility. Defining $\mathcal{E}_z^{(k)}$ and $\mathcal{E}_{v(i)}^{(k)}$ are the slack polygons (all attributes of a polygon are either positive or non-negative values) for $z_{CG}^{(k)}$ and $v_{(i)}^{(k)}$ respectively, the local optimization for agent i in L-AMPC with SCC can be written as,

$$
\begin{aligned}
\min_{\substack{v_{(i)}^{(k-1)},\, z_{CG}^{(k)} \\ k=1,\dots,N_p}} \quad & J_{(i)} = \sum_{k=1}^{N_p} \begin{pmatrix} \left\| C(z_{CG}^{(k)} - X_{CG,ref}^{(k)}) \right\|^2_{\frac{R_x}{\sqrt{N_A}}} + \left\| v_{(i)}^{(k-1)} \right\|^2_{R_{u(i)}} \\ + \left\| \mathcal{E}_z^{(k)} \right\|^2_{R_z} + \left\| \mathcal{E}_{v(i)}^{(k)} \right\|^2_{R_{v(i)}} \end{pmatrix} \\
\text{s.t.} \quad & C z_{CG}^{(k+1)} = C A_{CGd} z_{CG}^{(k)} + C B_{CGd} U_{CG(i)}^{(k)} + C W_{CGd}^{(k)} \\
& \qquad\qquad + C \textstyle\sum_{i' \neq i} Y_{CG(i')}^{(k)} + \hat{y}_*^{(k)}, \\
& U_{CG(i)}^{(k)} = A_{d(i)} X_{(i)}^{(k)} + B_{d(i)} v_{(i)}^{(k)} + W_{d(i)}^{(k)}, \\
& z_{CG}^{(0)} = X_{CG}^{(0)}, \\
& z_{CG}^{(k)} \in \chi_{CG}^{(k)} \ominus \mathcal{E}_{CG}^{(k)}(\varepsilon_c) \oplus \mathcal{E}_z^{(k)}, \\
& v_{(i)}^{(k)} \in \Omega_{(i)}^{(k)} \ominus K_{(i)} \mathcal{E}_{CG}^{(k)}(\varepsilon_c) \oplus \mathcal{E}_{v,i}^{(k)}, \\
& X_{(i)}^{(k)} \in \chi_{(i)}^{(k)}
\end{aligned}
\tag{4.13}
$$

where R_z and $R_{v(i)}$ are heavy penalties on slack polygons. $\oplus$ denotes the Minkowski sum. The L-AMPC in (4.13) will always be feasible and as safe as possible based on the probabilistic prediction results.

The stability of data-driven control systems is still an ongoing research field, although it has been intensively studied recently. As an uncertain system, the stability of a LMPC usually consists of nominal stability and robust stability [123]. The former is defined as the asymptotic stability for the nominal control system without uncertainty, while the latter considers the bounded effect of the uncertainty on the system. For the nominal stability of a finite-horizon MPC optimization problem, the standard approach is to define a proper terminal cost as a control Lyapunov function with a terminal constraint [123][124]. For robust stability, some recent works [125][126] have proposed several learning-based controllers with data selection methods to achieve robust stability. However, these approaches are based on the assumption of bounded uncertainty, which is unsuitable for the GP-MPC. Moreover, they usually adopt iterative data management methods, which is another challenge for real-time implementations. We will extend this work and continue to study hybrid data/model-based control methods that can ensure stability in the future.

4.4 Example: Yaw Rate Tracking

4.4.1 Agent Configuration

For vehicles operating at a given speed (v_x), states are considered as the lateral (v_y) and yaw (r) motions. Considering the net lateral forces and yaw moments at the vehicle CG as F_{yCG} and M_{zCG}, the vehicle dynamics in the lateral and roll directions can be written as

$$\begin{aligned} \dot{v}_y &= \tfrac{1}{m} F_{yCG} - v_x r \\ \dot{r} &= \tfrac{1}{I_z} M_{zCG} \end{aligned} \tag{4.14}$$

where m denotes curb weight; I_z represents vehicle yaw inertia. Then, the system matrices in (3.2) for vehicle yaw stability control are:

$$\begin{aligned} X_{CG} = \begin{bmatrix} v_y \\ r \end{bmatrix}, U_{CG} = \begin{bmatrix} F_{yCG} \\ M_{zCG} \end{bmatrix}, W_{CG} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \\ A_{CG} = \begin{bmatrix} 0 & -v_x \\ 0 & 0 \end{bmatrix}, B_{CG} = \begin{bmatrix} 1/m & -v_x \\ 0 & 1/I_z \end{bmatrix}, C = \begin{bmatrix} 0 & 1 \end{bmatrix} \\ A_{CGd} = T_s A_{CG} + I, B_{CGd} = T_s B_{CG}, W_{CGd} = T_s W_{CG} \end{aligned} \tag{4.15}$$

where T_s is the sample time. It should be noted that the U_{CG} term is usually not direct control efforts. Different actuator models describe the mapping from control signals on each agent $U_{(i)}$ to their contributions on net force $U_{CG(i)}$

Four agents are considered in this example as in Table 4.2. The rear differential torque (RDT) is the controllable agent with the designed controller. The driver or autonomous steering (STR) and the drive torques (DT) agents are the white-box agents, which are uncontrollable, but control efforts are known. The front differential torque (FDT) agent is a black-box agent with unknown contributions to vehicle behaviours.

Agent RDT is the controllable agent using L-AMPC, and its contributions should be represented as ($i = \text{RDT}$),

$$\begin{aligned} Y_{CG(\text{RDT})} &= B_{CGd} U_{CG(\text{RDT})} \\ \textstyle\sum_{i' \neq \text{RDT}} Y_{CG(i')} &= B_{CGd} U_{CG(\text{STR})} + B_{CGd} U_{CG(\text{DT})} \end{aligned} \tag{4.16}$$

Table 4.2: Agent Configuration for Yaw Stability Control Example

Type	Agent	$U_{(i)}$
Controllable agent	Rear differential torque (RDT)	$Q_{\mathrm{RDT},rr} = -Q_{\mathrm{RDT},rl}$
White-box agent	Driver steering (STR)	$[F_{y,fl}, F_{y,fr}, F_{y,rl}, F_{y,rr}]^T$
White-box agent	Driver torque (DT)	$[Q_{\mathrm{DT},fl}, Q_{\mathrm{DT},fr}, Q_{\mathrm{DT},rl}, Q_{\mathrm{DT},rr}]^T$
Black-box agent	Front differential torque (FDT)	$Q_{\mathrm{FDT},fr} = -Q_{\mathrm{FDT},fl}$

Q denotes torque. F_y denotes lateral tire force. the front-left wheel (fl), the front-right wheel (fr), the rear-left wheel (fl), and the rear-right wheel (rr) denote the four wheels, respectively. In this example, agent index $i \in \{\mathrm{RDT}, \mathrm{STR}, \mathrm{DT}, \mathrm{FDT}\}$. Besides, here $N_A = 1$ because only one controllable agent is defined in this example (Section 4.5 will extend to more than one controllable agent).

where $U_{CG(\mathrm{RDT})}$, $U_{CG(\mathrm{STR})}$, and $U_{CG(\mathrm{DT})}$ denote the contribution from RDT, STR, and DT agents, respectively. Mathematical models of these agents are [1],

$$
\begin{aligned}
U_{CG(\mathrm{RDT})} &= \sum_{i\in\{rl,rr\}} \left[\frac{1}{R_{w,i}}, \frac{T_{w,i}}{R_{w,i}}\right]^T Q_{\mathrm{RDT},i} \\
U_{CG(\mathrm{DT})} &= \sum_{i\in\{fl,fr\}} \left[\frac{\sin\delta_i}{R_{w,i}}, \frac{T_{w,i}\cos\delta_i + a_i \sin\delta_i}{R_{w,i}}\right]^T Q_{\mathrm{DT},i} + \sum_{i\in\{rl,rr\}} \left[\frac{1}{R_{w,i}}, \frac{T_{wi}}{R_{w,i}}\right]^T Q_{\mathrm{DT},i} \\
U_{CG(\mathrm{STR})} &= \sum_{i\in\{fl,fr\}} [1, l_i]^T F_{y,i}, \quad l_i \triangleq \begin{cases} +l_f, \text{front} \\ -l_r, \text{rear} \end{cases}
\end{aligned}
\tag{4.17}
$$

where Q_{RDT} and Q_{DT} are the torques from RDT and DT agents, respectively. F_y is the lateral tire force from the estimation module. l_f and l_r are the distances from the front and rear axle to CG, respectively. T_w is the track width. R_w is the wheel radius. δ is the wheel steering angle.

As a ground truth for method verification, the model of the FDT agent is manually set in simulations and experiments ($Q_{\mathrm{FDT},fr} = -Q_{\mathrm{FDT},fl}$). A commonly used control architecture, namely a feedback control plus a feedforward control, is adopted in this study as the control model of FDT. The feedback control is realized by a proportional integral (PI) controller, whose input is the error between the measured and desired yaw rate ($e_r = r_{des} - r_{mes}$). The feed-forward control is proportional to the desired yaw angular

[1] Specially in Equation 4.17, index i refers to the wheel.

acceleration $\dot{r}_{des}$:

$$Q_{\text{FDT},fr} = K_P \cdot e_r + K_I \cdot \int_0^t e_r(\tau)d\tau + K_{FF} \cdot \dot{r}_{des} \tag{4.18}$$

where K_P, K_I, K_FF are the proportional, integral, and feed-forward gains, respectively. Thus, the ground truth contribution from the black-box FDT agent $U_{CG(\text{FDT})}$ can be summarized as,

$$U_{CG(\text{FDT})} = f(r_{mes}, U^{prev}_{CG(\text{FDT})}, r_{des}, r^{prev}_{des}) \tag{4.19}$$

where $U^{prev}_{CG(\text{FDT})}$ and r^{prev}_{des} are the contribution and the desired yaw rate at the previous time step, respectively. In other words, the set of variables in Equation 4.19 should be the ground truth set of input variables.

4.4.2 Desired Vehicle Response and Constraints

Vehicle stability control is usually formulated to track the desired yaw rate with the limitation of the maximum side slip angle at the rear axle and the maximum yaw rate. The desired yaw rate in this example is calculated using the linear bicycle model:

$$X_{CG,ref} = \begin{bmatrix} v_{y,des} \\ r_{des} \end{bmatrix} = \begin{bmatrix} 0 \\ \text{sign}(\delta_f) \min\left(|\frac{v_x}{L_b + K_{us}u^2}\delta_f|, \frac{\mu g_a}{v_x} \right) \end{bmatrix} \tag{4.20}$$

where $v_{y,des}$ is the desired lateral speed, which is set to be 0 since it's not actually been tracked. L_b is the wheelbase of the vehicle. μ is the road friction from the estimation module. δ_f is the equivalent front wheel turning angle. g_a is the gravity acceleration. K_{us} is the under-steering coefficient as,

$$K_{us} = \frac{2m(l_r C_{\alpha r} - l_f C_{\alpha f})}{L_b C_{af} C_{ar}} \tag{4.21}$$

where $C_{\alpha f}$ and $C_{\alpha r}$ are the cornering stiffness for each front and rear tire, respectively. The constraints, which include the actuator limitation of RDT and stability requirements, can

be written as,

$$
\begin{aligned}
\Omega_{(\mathrm{RDT})} \triangleq: \begin{bmatrix} I \\ -I \end{bmatrix} U_{(\mathrm{RDT})} \leq L \\
L = \begin{bmatrix} Q_{max} - \max\{Q_{\mathrm{DT},fr}, Q_{\mathrm{DT},fl}, 0\} \\ \min\{Q_{\mathrm{DT},fr}, Q_{\mathrm{DT},fl}, 0\} - Q_{min} \end{bmatrix} \\
\chi_{CG} \triangleq: F = \begin{bmatrix} 1/v_x & -b/v_x \\ -1/v_x & b/v_x \\ 0 & 1 \\ 0 & -1 \end{bmatrix} X_{CG} \leq G = \begin{bmatrix} \alpha_{rear,max} \\ \alpha_{rear,max} \\ r_{max} \\ r_{max} \end{bmatrix}
\end{aligned}
\tag{4.22}
$$

where Q_{max} and Q_{min} are the maximum and minimum capacity of the wheel motor. $\alpha_{rear,max} \geq 0$ and $r_{max} \geq 0$ are the maximum allowable side slip angle at the rear axle and yaw rate, respectively.

4.4.3 Model Selection and Training

Model selection is used to select the most relevant input variables for the dataset. The controller model of the black-box FDT agent has been built in Equation 4.18. Therefore, the variables that have the most significant impact on the output can first be obtained through mathematical analysis as a ground truth reference in Equation 4.19.

However, since the exact inputs of the black box agent are unknown, many potential variables can affect the output of the black box agent directly or indirectly. Therefore, some possible input variables need to be pre-set, and then the method described in Section 4.2.1 and Appendix A.2 should be used for model selection. If there is no information about the black-box agent at all, the possible inputs of this black-box FDT at the time step t could be yaw rate error ($e_r^{(t)}$), desired yaw rate ($r_{des}^{(t)}$), actual yaw rate ($r_{mes}^{(t)}$), steering angle ($\delta^{(t)}$), vehicle speed ($v_x^{(t)}$). However, if it is somehow known (or based on the experience) that the black-box agent is developed in an error-based feedback control form, then yaw rate errors ($e_r^{(t)}$, $e_r^{(t-1)}$, $e_r^{(t-2)}$) and the previous motor torque ($Q_{(\mathrm{FDT})}^{(t-1))}$) may be helpful. From the perspective of statistical learning, too many redundant variables will significantly increase computational cost and lead to solid co-linearity between variables, which will reduce the prediction accuracy. So, the task is to select the most miniature set of these variables that can best influence the output variables without knowing the black-box controller model. In

other words, the variable combination with the best statistical learning criteria is selected as the input variable for data collection.

Some studies use simple correlation analysis to select sets of input variables, for example, the PCC. However, PCC only analyzes the correlation between two datasets and does not care about the correspondence between them. Therefore, more advanced model selection methods mentioned in Section 4.2.1 are needed. The forward selection method is used to start with a single variable and increase the number of variables. In the simulation environment (which will be introduced in Section 4.5), the vehicle performs five consecutive DLC operations. As more data are collected and closer to the test point, the statistical indicators of GPR (PMSE, R^2_{adj}, and AIC) will gradually improve. Moreover, since these metrics put a harsher penalty for the dimension of the inputs, their values will reflect both prediction accuracy and model complexity. These statistical criteria of different variable selections are compared, and the results of the PMSE are shown in Figure 4.6.

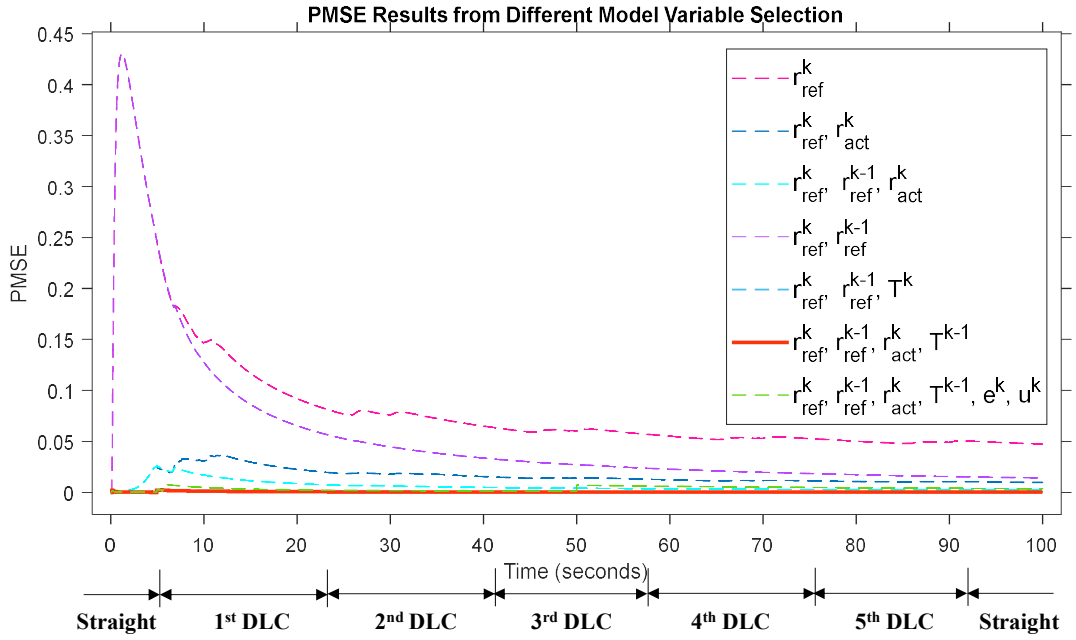


Figure 4.6: PMSE results for model selection for the yaw stability tracking example using L-AMPC.

PMSE (Equation A.8 $\sim$ A.10) represents the expected value of the squared difference between the predicted values and the observations. Smaller PMSE values indicate better variable set selection. As more data is collected and data is getting closer, the PMSE value becomes smaller and smaller. Moreover, It is easy to find that the red line, the result of

the variables set $\{r_{mes}, U^{prev}_{CG(\mathrm{FDT})}, r_{des}, r^{prev}_{des}\}$, is the best selection, which is consistent with the artificially set input variables (Equation 4.19) for the black-box FDT agent. Also, the model selection results using R_adj^2 (Equation A.11) and AIC (Equation A.12 and A.14) are shown as in Figure 4.7.

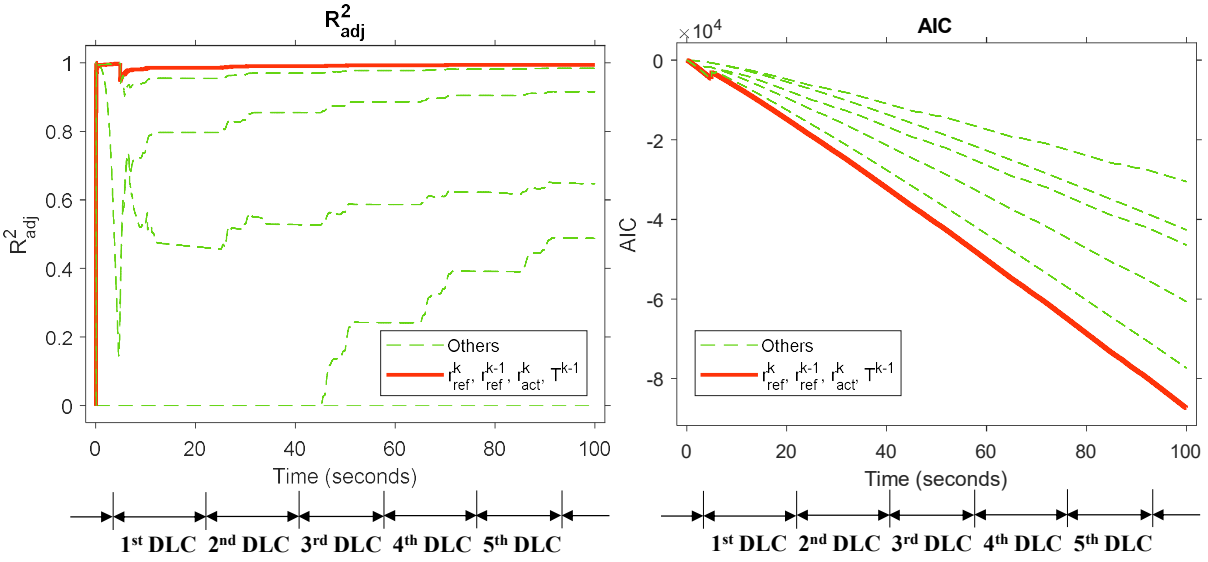


Figure 4.7: R^2_{adj} and AIC results for model selection for the yaw stability tracking example.

The above results prove that the ground-truth input set of the black-box FDT agent will always have the lowest PMSE, highest R^2_{adj}, and lowest AIC values. The model selection method used in this study can accurately identify the correct input variables in (4.19).

As introduced in Section 4.2.3, compared to error-based loss functions, using the likelihood function as the optimization objective for hyper-parameter training is more suitable for Bayesian paradigms. Because the likelihood function considers both fitting accuracy and model complexity would penalize the over-fitting models. Thus, for GPR learning in the L-AMPC scheme, the log marginal likelihood function Equation 3.19 is used as the objective function and its partial derivatives w.r.t. the hyper-parameters Equation 3.20 is used for solving the optimization problems based on gradient search method.

First, a dataset is collected for offline training. Several DLC maneuvers are conducted in high-fidelity simulations at various road friction ($\mu = 0.1 \sim 1.0$) and vehicle speeds ($v_x = 30\text{kph} \sim 90\text{kph}$). Finally, 2398 data points in total are collected. Based on the results of model selection, $n_d = 4$. Thus, six hyper-parameters in total need to be trained, including one signal standard deviation σ_f and one noise standard deviation σ_{N_d}. The

gradient-based stochastic search method is used to accelerate solving the optimization problem, and it takes 16.3 seconds to find the optimal hyper-parameters shown in Table 4.3.

Table 4.3: Hyper-Parameter Training Results for GPR in L-AMPC

Hyper-parameter	Length scales				Signal SD	Noise SD
	$l_1(U^{prev}_{CG(\mathrm{FDT})})$	$l_2(r_{des})$	$l_3(r^{prev}_{des})$	$l_4(r_{mes})$	σ_f	σ_{N-d}
Trained Value	5.2695	3.8677	2.0742	2.2555	2.2773	0.0412

Further, online training is also implemented. Online training only uses a subset selected according to KNN to improve training efficiency. So, the hyper-parameters will be updated in real time at each time step. Finally, the hyper-parameters from offline training, online training, as well as manual setting are all applied in the GPR inference process. The PMSE of the prediction results of three consecutive DLCs is shown in Figure 4.8.

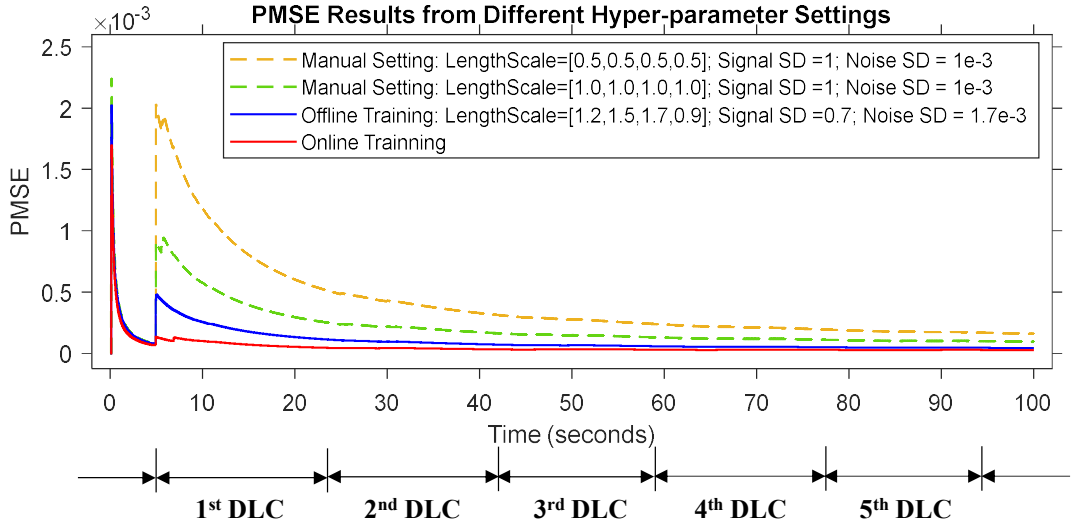


Figure 4.8: PMSE comparison of GPR predictions using different training methods.

As the figure shows, using hyper-parameters trained offline or online will significantly improve GPR prediction accuracy compared to using manually set hyper-parameters. Compared with using offline training hyper-parameters, using online training hyper-parameters for prediction has a smaller PMSE value in the initial stage, but as more data are collected in continuous working conditions, this advantage gradually decreases until almost zero.

In the simulation environment, online training takes around 9.174 (real/simulation time), which is not possible for real-time implementation. In conclusion, using offline training is the best choice.

4.4.4 Learning Process

Figure 4.9 shows the results of a data collection process. Four similar DLCs are conducted at a speed of about 50 kph while the data collection continues online from zero data points. The figure shows the GPR prediction results, prediction error, prediction covariance, and the total number of data points, respectively.

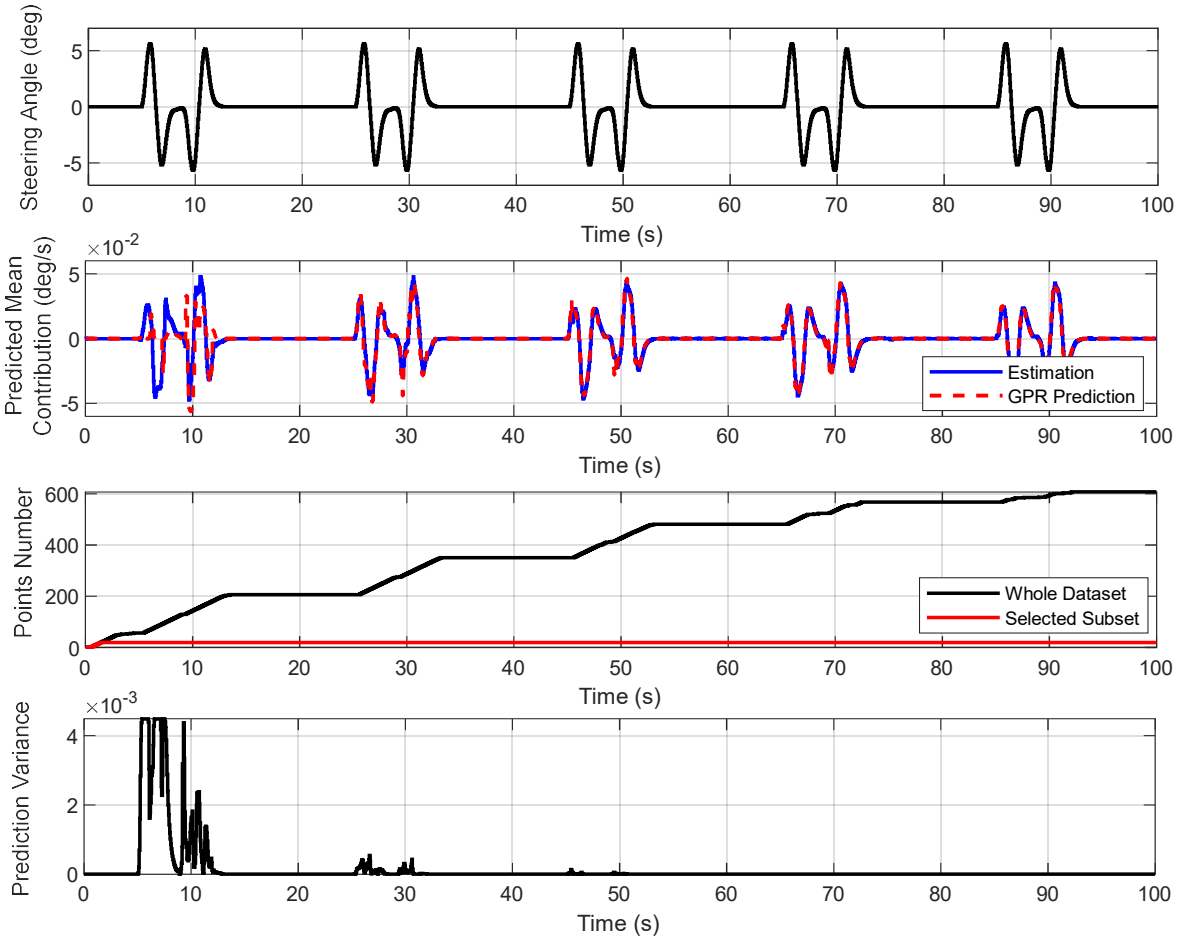


Figure 4.9: Results of GPR prediction and data collection in L-AMPC.

Table 4.4: Comparison of Time Consumption of Modules

Module	Prediction using the single-step GPR	Prediction using the proposed multi-step GPR
MPC Solving	0.25 ms	0.27 ms
Data Management	1.26 ms	1.44 ms
GPR Inference	0.38 ms	0.46 ms
Other	1.99 ms	1.90 ms

A detailed comparison of using different controllers in a well-learned DLC maneuver is shown in Figure 4.10. The dataset already has enough data collected from similar DLCs. The predictions of the compensated overall yaw rate are first compared. The predictions using the proposed multi-step GPR (red) were significantly closer to the actual measured yaw rate than single-step GPR with assumptions (blue and green), especially when the yaw rate changes fast. With more accurate predictions, the tracking performance of the proposed L-AMPC based on multi-step GPR is better than the traditional AMPC and those using the single-step GPR, especially when the yaw rate changes drastically.

Besides, a comparison of the computational time consumption between using the single-step GPR and multi-step GPR is given in Table 4.4, which is the average time consumption counted by the Simulink built-in toolkit in the simulation shown in Figure 4.10. As we can see, adopting the proposed multi-step GPR prediction only slightly increases the time consumption compared to single-step GPR prediction, mainly in data management and GPR prediction. However, the increase is not significant and, therefore, will not affect the overall real-time efficiency. In addition, the overall time consumption of all modules can fully meet the real-time requirements.

4.4.5 Controller Design

The uncertainty from GPR prediction will be propagated to each prediction step. Following a common approach that the state, control input, and disturbance are approximated as jointly Gaussian distributed at each step, affine transformations of the Gaussian distribution [71] are used for the propagation in (3.21) from error dynamics to system states and control actions. By setting an admissible probability of constraint violation ε_c, the

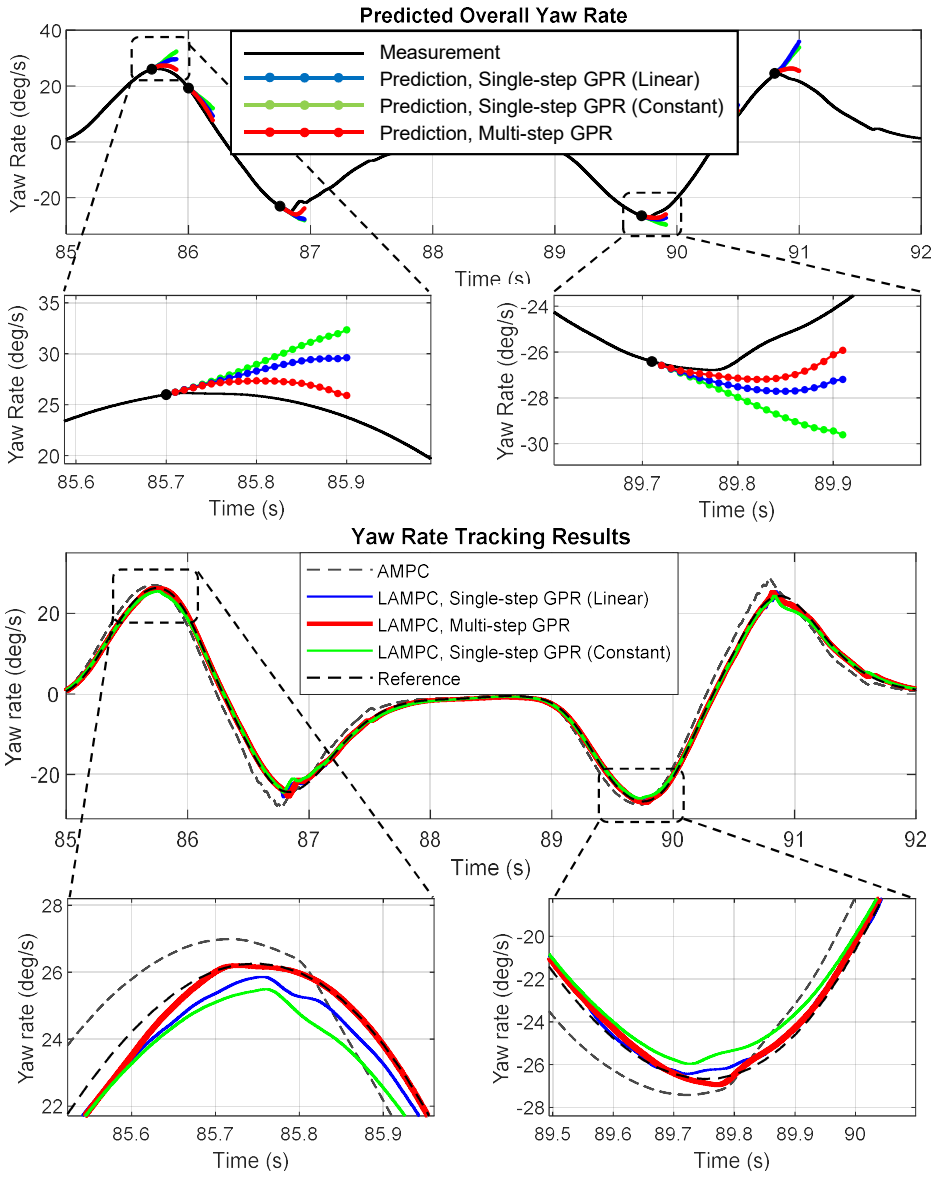


Figure 4.10: Comparison of yaw rate prediction and tracking performance.

shrunken biases at prediction step k on constraints in Equation 4.22 are,

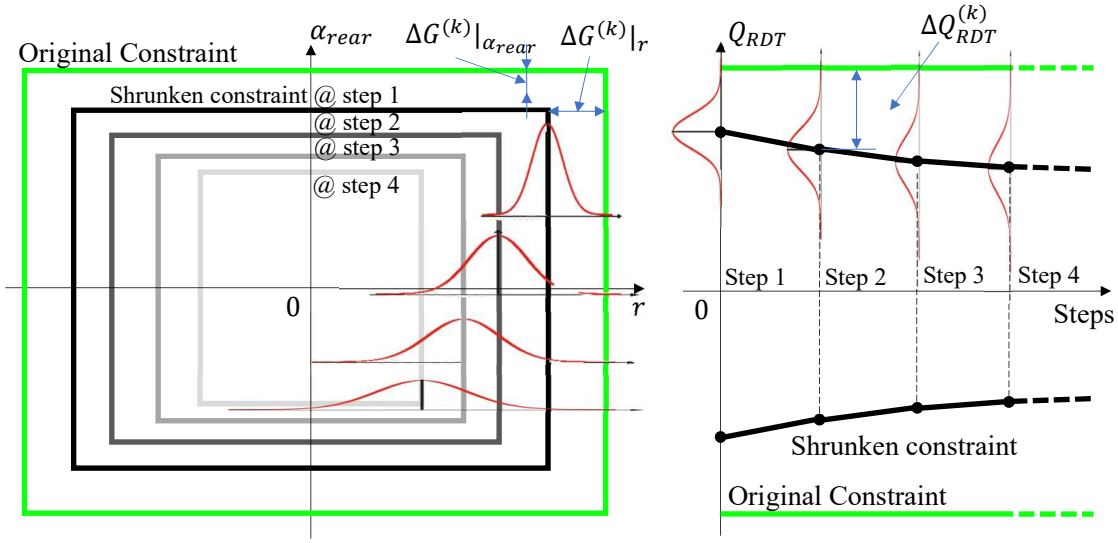
$$\begin{aligned} \Delta G^{(k)} &= z_p \sqrt{F \Sigma_e^{(k)} F^T} \\ \Delta Q_{(i)}^{(k)} &= z_p \sqrt{K_{(i)} \Sigma_e^{(k)} K_{(i)}^T}, i = \text{RDT} \end{aligned} \tag{4.23}$$

where $\Sigma_e^{(k)}$ is the variance of error dynamics in Equation 4.10 at step k. $K_{(i)}$ is the feedback matrix in Equation 4.7. z_p is the quantile of Gaussian distribution where $p = 1-\varepsilon_c$ indicates the proportion of the distribution that lies below the quantile.

The shrunken constraints are further relaxed as soft chance constraints to guarantee the feasibility:

$$\begin{aligned} \Omega^{(k)} \triangleq: \begin{bmatrix} I \\ -I \end{bmatrix} v_{(\mathrm{RDT})}^{(k-1)} \leq L - \begin{bmatrix} \Delta Q_{(\mathrm{RDT})}^{(k)} - \varepsilon_{v(\mathrm{RDT})}^{(k)} \\ \Delta Q_{(\mathrm{RDT})}^{(k)} - \varepsilon_{v(\mathrm{RDT})}^{(k)} \end{bmatrix} \leq L \\ \chi^j \triangleq: F z_{CG}^{(k)} \leq G^{(k)} - (\Delta G^{(k)} - \varepsilon_z^{(k)}) \end{aligned} \tag{4.24}$$

where $z_{CG}^{(k)}$ and $v_{(\mathrm{RDT})}^{(k-1)}$ are noise-free nominal vehicle state and control inputs. $\varepsilon_z^{(k)} \geq 0$ and $\varepsilon_{v(i)}^{(k)} \geq 0$ are slack variables on state constraints and control constraints, respectively, corresponding to the slack polygons $\mathcal{E}_z^{(k)}$ and $\mathcal{E}_{v(i)}^{(k)}$. A diagram of the shrunken constraints on the states and control actions is shown in Figure 4.11. The greater the uncertainty, the more the constraints shrink, and vice versa.



(a) Stability Constraints on Vehicle States

(b) Control Action Constraints

Figure 4.11: Diagram of the shrinking of state constraints and control constraints due to the propagation of learning uncertainty. Green is the original constraint. Black or gray is the pruned constraint at each step in the horizon. Red is the distribution of uncertainty of the state or constraint propagated at each step in the horizon.

4.5 Simulation Results

Based on the above example, simulation and experimental studies were conducted to demonstrate the features of the proposed L-AMPC control scheme. The controller is developed in Matlab/Simulink with parameters shown in Table 4.5. A high-fidelity calibrated CarSim model of this vehicle from a previous experimental study [127] is adopted for simulations, whose main parameters are specified in Appendix C. The controller runs at a frequency of 20 Hz.

Table 4.5: Controller Settings of L-AMPC for Yaw Stability Control

Symbol	Parameter Description	Value
N_p	Prediction horizon	10
R_x	Weight for yaw rate tracking	5×10^6
$R_{U(\mathrm{RDT})}$	Control weight on RDT agent	0.3
$R_z, R_{v(\mathrm{RDT})}$	Slack weight on soft constraints	1×10^{10}
Q_{max}	Maximum capacity of the wheel motor	1760 Nm
Q_{min}	Minimum capacity of the wheel motor	−1760 Nm
K_P	Proportional gain for FDT agent	100
K_I	Integral gain for FDT agent	0.1
K_{FF}	Feed-forward gain for FDT agent	0.5
$\alpha_{rear,max}$	Maximum side slip angle at the rear axle	10 deg
r_{max}	Maximum yaw rate	50 deg/s
ε_c	Admissible probability of violation	2.5%

The proposed method was first tested in the "less-learned" scenario, where there was not enough data with similar experience for GPR prediction, to verify the safety using the stochastic chance constraint. Then several tests in the "well-learned" scenario were conducted to demonstrate the improvement of the proposed L-AMPC compared to the conventional AMPC.

4.5.1 Less-Learned Scenario: Safety Guarantee

A DLC maneuver was operated at about 65 km/h on a high friction surface. Control agents were configured as in Table 4.2. The dataset of the controllable RDT agent started empty, meaning there is not enough experience for this DLC. Three controllers have been applied for the RDT agent:

a. AMPC controller;

b. L-AMPC controller with normal constrain (NC), i.e. the stability constraints without considering the learning uncertainties as described in (4.22);

c. L-AMPC controller with SCC as described in (4.24).

The simulation results of the L-AMPC with SCC are shown in Fig. 4.12, where the results of AMPC and L-AMPC with normal constraints are also compared. It is easy to notice that when the RDT agent was controlled by AMPC or L-AMPC with NC, the vehicle's rear slip angle and yaw rate exceeded the limits. Remarkably, the absolute value of the maximum side slip angle has reached 40 deg, which is very dangerous in reality. This violation is because the black-box FDT agent significantly impacts vehicle dynamics, but GPR cannot predict its contribution, shown as the significantly predicted error and significant variances. The RDT agent will, therefore, generate far more torque than is required. However, when the RDT agent was controlled by the proposed L-AMPC with SCC, the rear slip angle and the yaw rate were kept within constraints, ensuring the vehicle was stable and safe.

Details: Two typical optimization results are shown in Fig. 4.13. The prediction uncertainty is propagated as constraints shrink simultaneously on both state and control inputs. The original constraints (green solid lines) were first shrunken to hard chance constraints (blue dashed lines) and then relaxed as softened chance constraints (red dashed lines) at each step along the horizon. At time T=5.75, the state trajectory was constrained; At time T=10.80, the control input was constrained. The actual state trajectories were all guaranteed not to exceed the original constraint under the ε_c.

4.5.2 Well-Learned Scenario: Performance Improvement

This section investigates the control performance of the proposed L-AMPC to see if the learning results are reliable enough. A series of driving scenarios and agent configurations

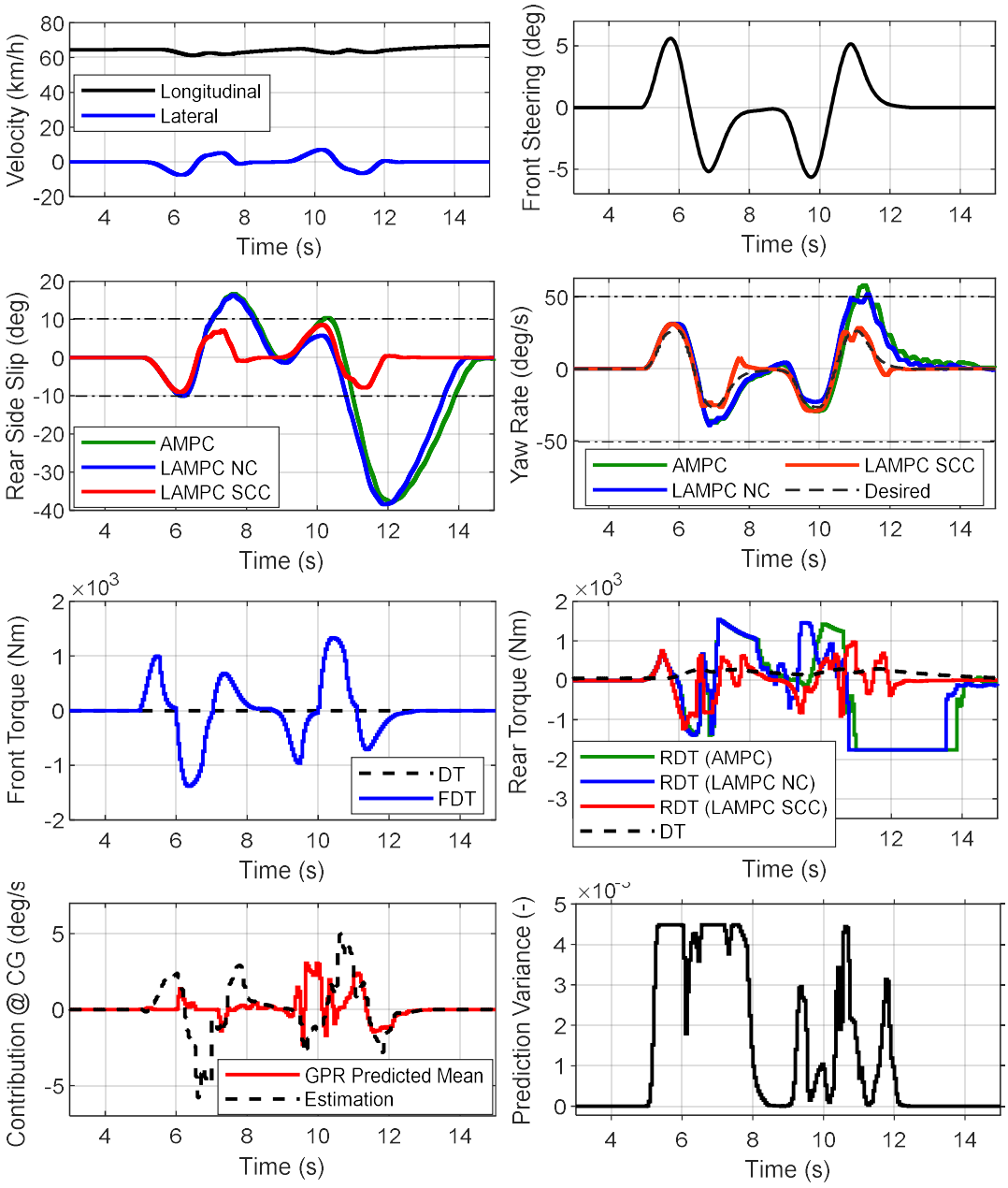


Figure 4.12: Simulation results of L-AMPC for a DLC in the less-learned scenario.

are conducted on the Equinox CarSim model in the Matlab/Simulink and CarSim co-

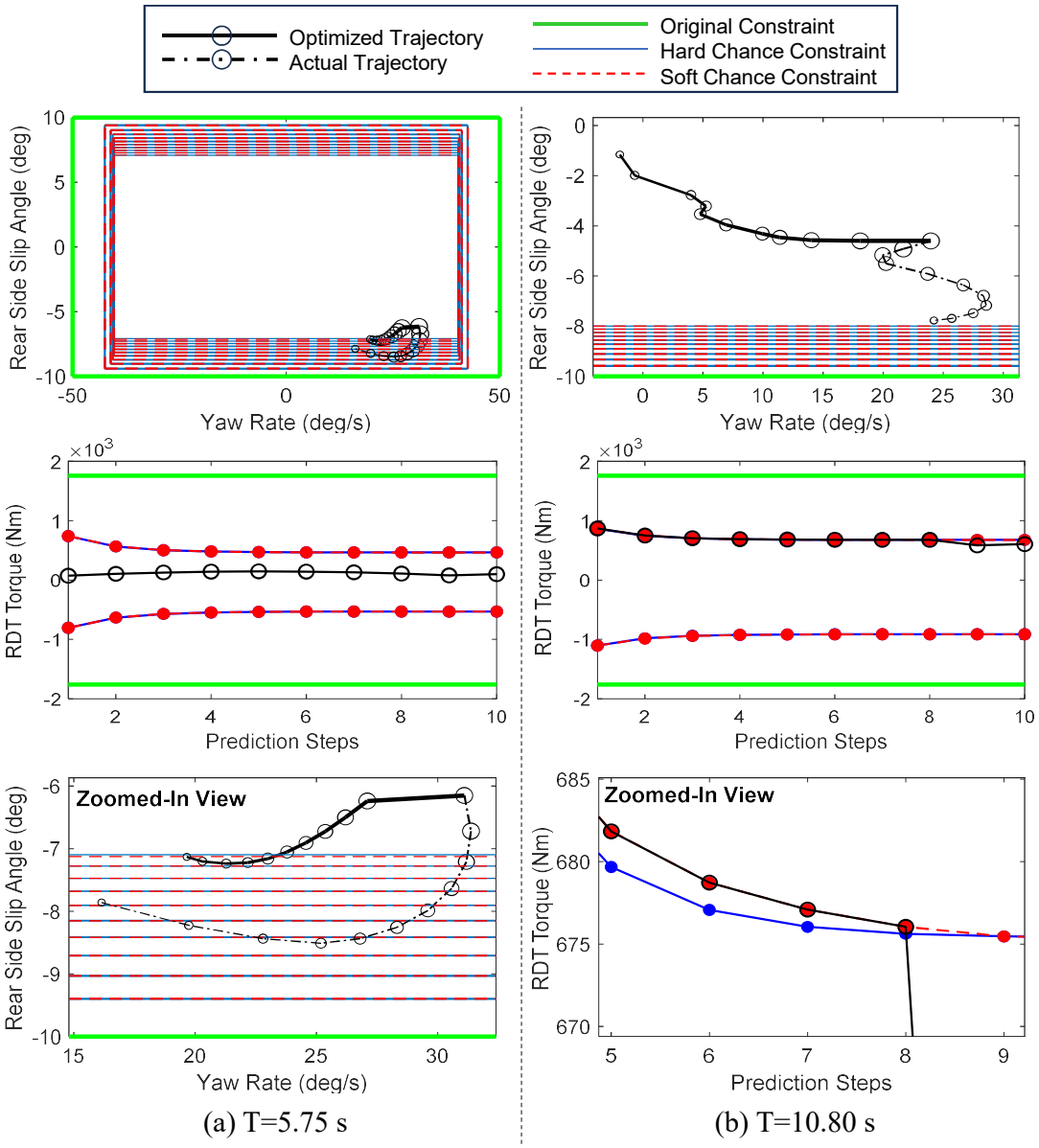


Figure 4.13: Optimization results of L-AMPC in the less-learned DLC at T=5.75 and T=10.80.

simulation environment to evaluate the performance of the developed controller, as shown in Table 4.6.

Table 4.6: Simulation Configurations using L-AMPC in Well-Learned Scenarios

Scenario		Agent configuration	Driving Maneuver	Vehicle Speed	Road Friction
A. Various Conditions	A1. Maneuver Change	Controllable: RDT White-box: STR, DT Black-box: FDT	DLC -> Sinewaves -> Fishhooks	45 kph	1.0
	A2. Friction Change	Controllable: RDT White-box: STR, DT Black-box: FDT	3 DLCs	45 kph	1.0 -> 0.6 @ 3rd DLC
B. Two Controllable Agents		Controllable: FDT, RDT White-box: DT Black-box: AFS	3 Fishhooks	60 kph	1.0
C. Two Black-Box Agents		Controllable: RDT White-box: DT Black-box: FDT, AFS	3 DLCs	60 kph	1.0

The active front steering (AFS) in this table is a controller that can generate an additional steering angle in addition to the driver's steering.

Scenario A. Different Driving Conditions

The simulation results of Scenario A1 are shown in Figure 4.14. In this test on a road with a coefficient of friction of 0.1, the vehicle maintains a speed of 45 kph through two DLCs, followed by several Sinewaves, and finally, three Fishhooks are implemented. The first plot shows the yaw rate comparison of traditional AMPC and proposed L-AMPC. Compared to traditional AMPC, the proposed L-AMPC with GPR predictions significantly improves the performance of yaw rate tracking, always closely following the desired yaw rate. The yaw rate error in the second plot also significantly reduces errors in various maneuvers. The third plot shows the authenticated results of the GPR prediction. As the plot shows, the predicted contribution from unknown agents is close to the ground truth value. When the maneuver changes, many new data points should be collected into the dataset, which makes the learning results quickly adapt to new maneuvers and quickly keep up with the ground truth.

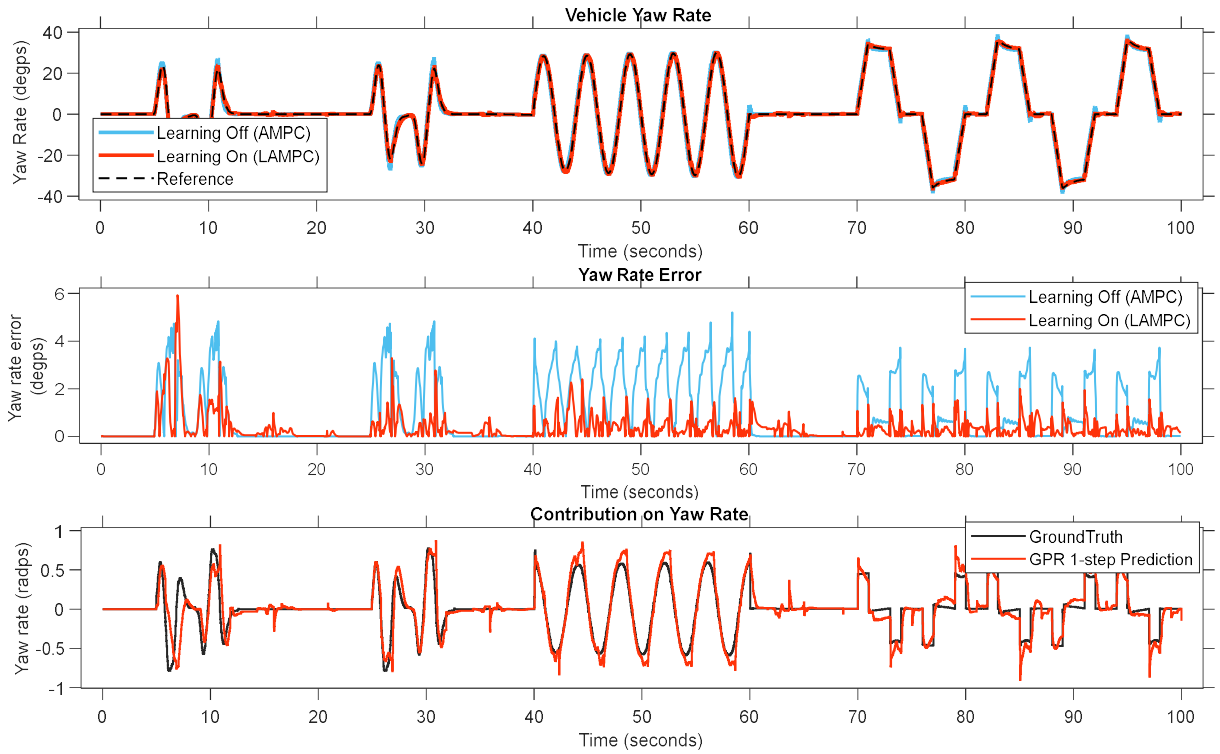


Figure 4.14: Overall simulation results of L-AMPC in Scenario A1.: Maneuvers change.

The simulation results of Scenario A2 are shown in Figure 4.15. This test is on the road with a changing coefficient of friction. The vehicle maintains a speed of 45 kph and drives through three DLCs with road friction changes from 1.0 to 0.6 after the second DLC but before the third DLC. The first plot shows the yaw rate comparison of traditional AMPC and proposed L-AMPC. It can be found from the first two DLCs that, compared to traditional AMPC, proposed L-AMPC with GPR predictions significantly improves the performance of yaw rate tracking, always closely following the desired yaw rate. Moreover, the vehicle controlled by AMPC starts to drift and even become unstable when the road friction decreases, while the vehicle controlled by L-AMPC still closely follows the desired yaw rate. The yaw rate error shown in the second plot also significantly improved using L-AMPC. The third plot shows the authenticated results of the GPR prediction.

The proposed L-AMPC scheme can significantly improve the yaw rate tracking performance with the help of GPR prediction. Moreover, the proposed learning method can quickly adapt to various maneuvers and driving conditions thanks to the online method of collecting and selecting data.

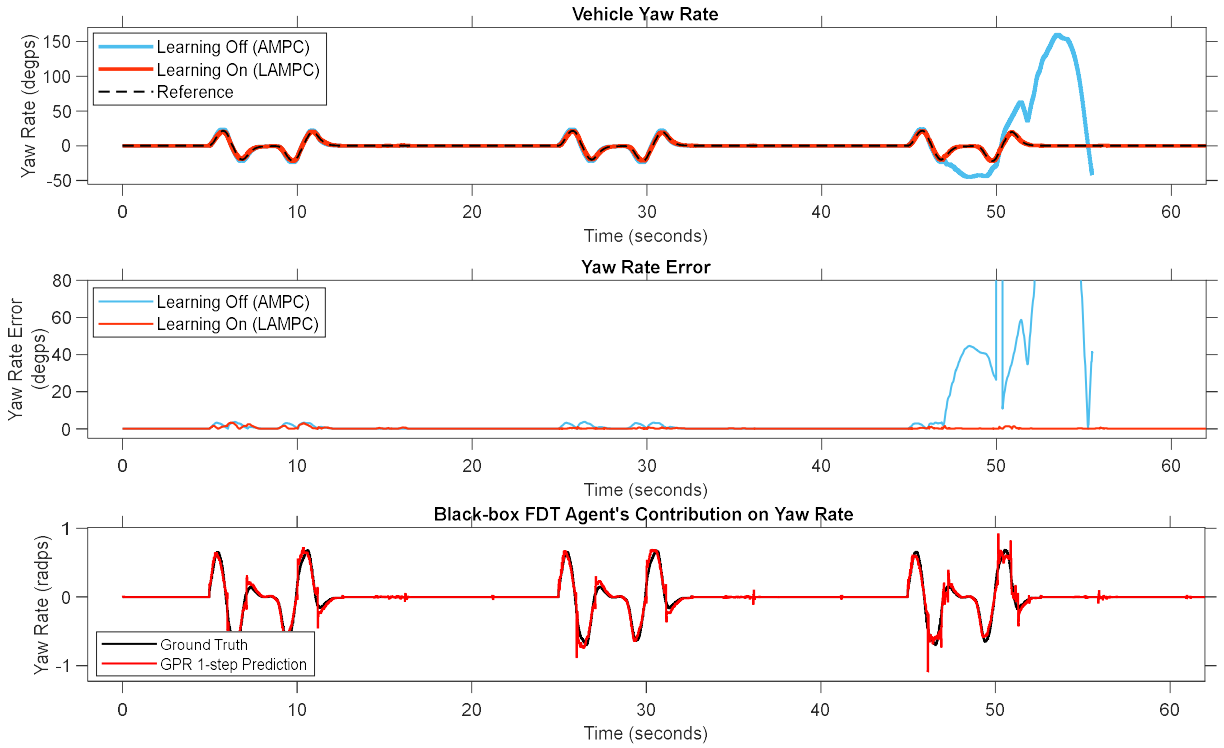


Figure 4.15: Overall simulation results of L-AMPC in Scenario A2.: Friction change.

Scenario B. Two Controllable Agents

The simulation results of Scenario B are shown in Figure 4.16. In this test, the front and rear differential torque agents are all controllable agents controlled by distributed L-AMPC, while an AFS works at the front axle as a black-box agent. The first plot shows the improvement using the proposed L-AMPC for each controllable agent, with a precise prediction from GPR shown in the third plot. The yaw rate error shown shows a huge error reduction in this test.

The detailed results of the consensus iterations between the two controllable agents during one of these fishhooks are shown in Figure 4.17.

Scenario C. Two Black-box Agents

The simulation results of Scenario B are shown in Figure 4.18, and details are shown in Figure 4.19. In this test, only the RDT agent is controllable, while the FDT and AFS agents

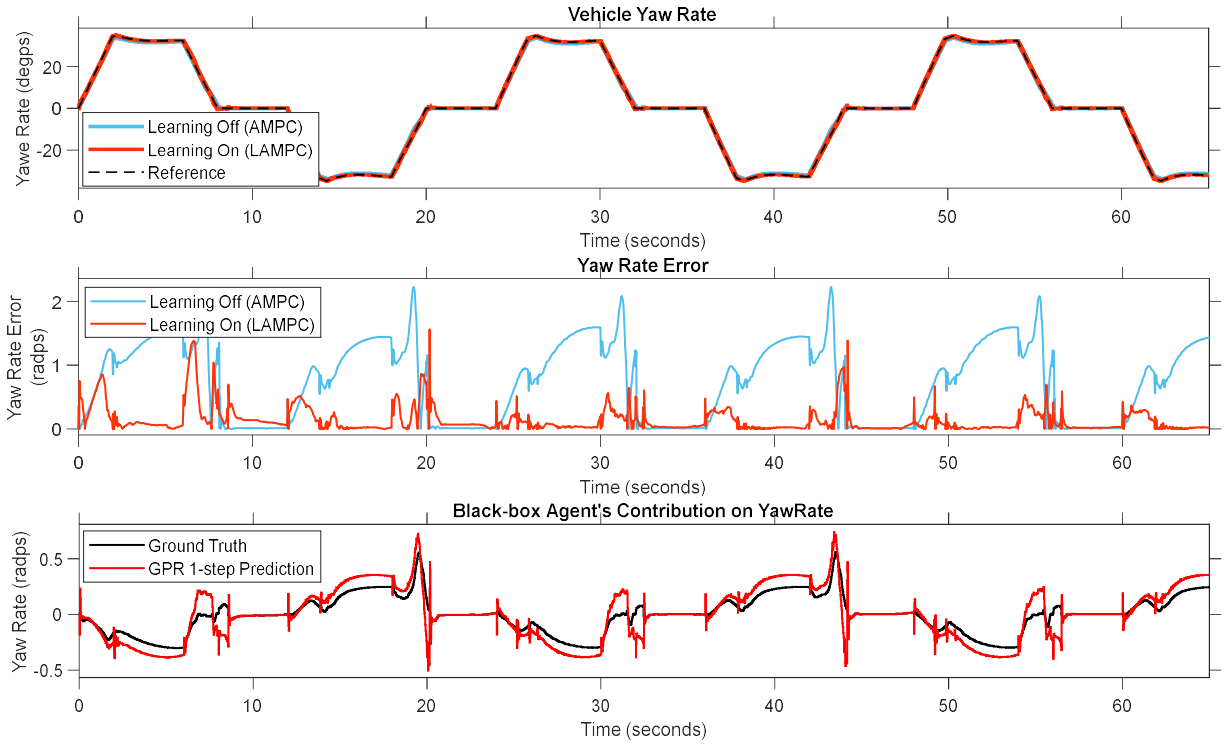


Figure 4.16: Overall simulation results of L-AMPC in Scenario B.: Two controllable agents.

are all black-box agents. The first plot also shows the improvement using the proposed L-AMPC for each controllable agent, with a precise prediction of the overall contribution from all the black-box agents shown in the third plot. The yaw rate error in the second plot also shows a significant error reduction in this test.

The detailed results are shown in Figure 4.19. From these results, the proposed L-AMPC scheme can be applied to any kind and number of unknown agents.

4.6 Summary

This chapter provided a complete introduction to the proposed L-AMPC scheme. First, the configuration of multiple agents was introduced, including the ground truth model of the black-box agent. Next, the complete pipeline of learning the black-box agent using GPR was introduced, including data preprocessing, model selection, hyper-parameter training, contribution estimation, data collection, data selection, uncertainty propagation, and soft

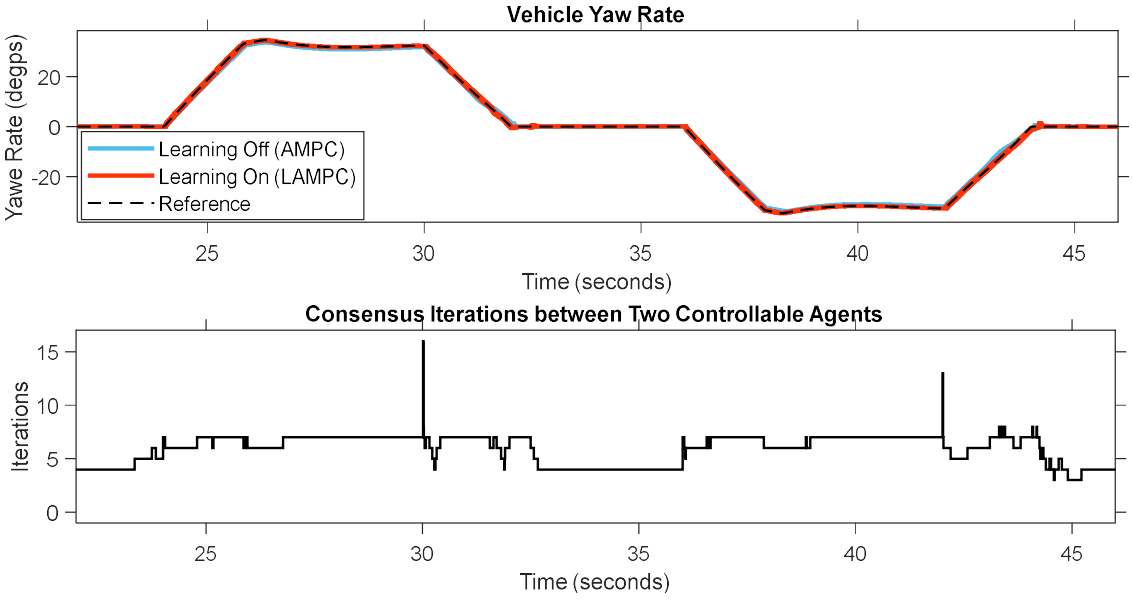


Figure 4.17: Detailed simulation results of L-AMPC in Scenario B.

stochastic optimization. Furthermore, methods of reconstructing the traditional AMPC scheme using GPR prediction results were introduced. The resultant L-AMPC scheme improved the tracking performance. Finally, an efficient multi-step GPR was proposed to improve prediction accuracy over the horizon. The L-AMPC scheme proposed in this chapter is not limited to any form or number of black-box agents. Moreover, it has higher prediction accuracy than traditional GP-MPC methods while maintaining the feasibility of real-time implementation. It has been proven by simulations that the proposed L-AMPC scheme has the following advantages:

1) *High tracking performance.* The tracking performance was significantly improved in a well-learned scenario based on reliable learning results.

2) *Guaranteed safety.* Stochastic chance constraint is used to guarantee the safety of constraint satisfaction. The feedback assumption in uncertainty propagation and soft constraint help to improve and ensure the optimization feasibility.

3) *Real-time efficiency.* This study adopts the high flexibility of GPR learning. Data density control and subset selection ensure data quality and real-time efficiency.

4) *Flexibility.* The proposed L-AMPC retains the flexibility of AMPC for agent configuration. This method can be applied to any kind or number of "black-box" agents.

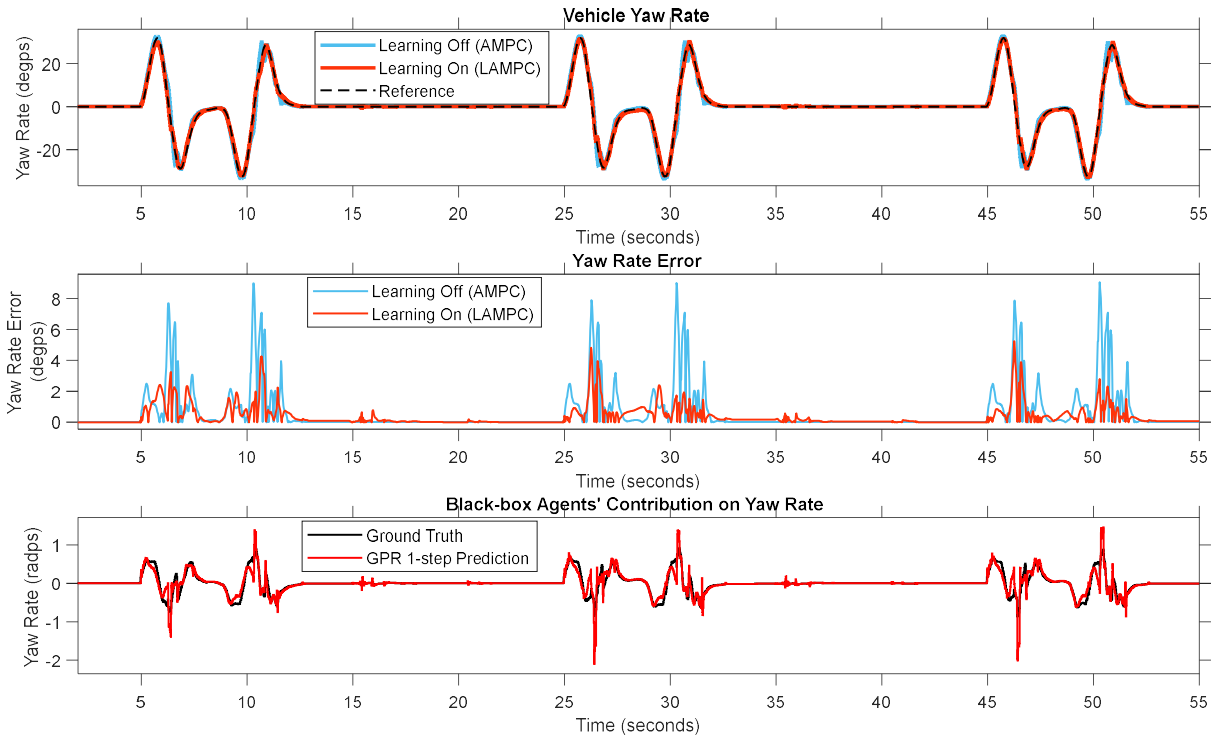


Figure 4.18: Overall simulation results of L-AMPC in Scenario C.: Two black-box agents.

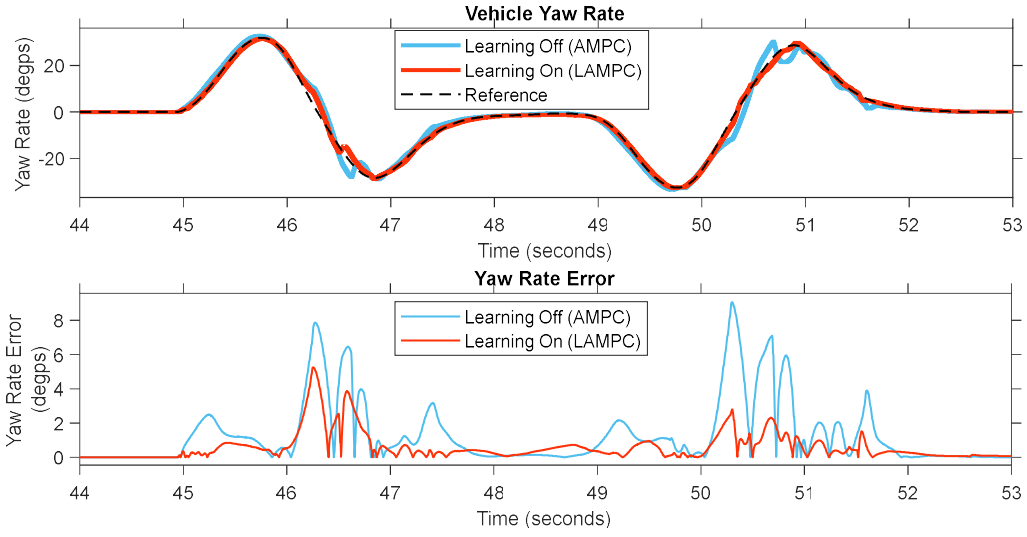


Figure 4.19: Detailed simulation results of L-AMPC in Scenario C.

Chapter 5

Multi-Objective Agent-Based Model Predictive Control

This chapter presents the MO-AMPC scheme for HVC. It is structured as follows: First, the integrated MPC for MAMO control systems are introduced as the baseline. The overview design of the MO-AMPC scheme is given in Section 5.1. Then, three formulations of MO-AMPC tailored from ADMM are introduced in Section 5.2, and the convergence and parameter selection are discussed in Section 5.3. Simulation results are demonstrated following two examples of cruise control and holistic vehicle stability control in Section 5.4 and in Section 5.5, respectively.

5.1 Overview

As introduced in Section 1.3, the MO-AMPC control scheme proposed in this chapter solves the second pain point by decomposing a MAMO control system from the objective (feature) perspective. MO-AMPC is a distributed control scheme with more flexibility to be designed in a "plug-and-play" fashion in practice than the AMPC scheme, which can only apply to one common objective.

5.1.1 Integrated Model Predictive Control

First, we establish the integrated MPC for the MAMO control systems as a baseline for distributed methods. For multi-objective optimization problems, the most common approach

is constructing an integrated cost function in the form of weighted summation. Thus, the discrete integrated MPC can be written as,

$$\begin{aligned} \min_{\bar{U}} \quad & J = \sum_{k=1}^{N_p} \sum_{j=1}^{N_O} h_j(X^{(k)}) + \sum_{k=1}^{N_p} g(U^{(k-1)}) \\ \text{s.t.} \quad & X^{(k+1)} = f_d(X^{(k)}, U^{(k)}), k \in \{0, \dots, N_p - 1\}, \\ & U^{(k)} \in \Omega^{(k)}, k \in \{0, \dots, N_p - 1\} \end{aligned} \tag{5.1}$$

where, $X \in \mathbb{R}^{n_d}$ is the system state and $X^{(0)}$ is the current state from measurement or estimation, N_p is the length of prediction horizon, Ω is the overall control constraint set, h_j is the state penalty for objective O_j, g is the regularization function for U, f_d is the discrete system equation.

For many real-time control systems, such as vehicles, it is common to approximate NMPC as linear MPC to solve for higher computational efficiency. Therefore, this chapter focuses on the optimization with quadratic cost functions, linear systems equations, and linear inequality constraints:

$$\begin{aligned} h_j(X^{(k)}) :=& \|C_j(X^{(k)} - X_{ref}^{(k)})\|_{R_{xj}}^2, \\ g(U^{(k)}) :=& \sum_{i=1}^{N_A} g_{(i)}(U_{(i)}^{(k)}) = \sum_{i=1}^{N_A} \|U_{(i)}^{(k)}\|_{R_{u(i)}}^2, \\ f_d :=& A_d X^{(k)} + \sum_{i=1}^{N_A} B_{d(i)} U_i^{(k)} + W_d^{(k)}, \\ \Omega^{(k)} :=& A_{in} U^{(k)} \leq b_{in}. \end{aligned} \tag{5.2}$$

where A_d, $B_{d(i)}$, and W_d are the system matrix, control matrix, and disturbance term, respectively [1]. C_j is the observation matrix for objective O_j. $g_{(i)}$ is the regularization function for the control action $U_{(i)}$ from the i-th agent. Positive-definite R_{xj} and semi-positive definite $R_{u(i)}$ are penalty weights for the j-th tracking error and the i-th control action, respectively. X_{ref} denotes the desired tracking reference. $A_{in} \in \mathbb{R}^{n_c \times n_u}$ and $b_{in} \in \mathbb{R}^{n_c \times 1}$ are the coefficient matrix and constant vector of the linear inequality constraints Ω, respectively, where n_c is the dimension of inequality constraints. Based on (5.2), the integrated MPC in (5.1) should be convex, and its solution is the global optimum.

[1] Different from in Equation 3.2, the integrated MPC in this chapter does not use a two-level structure. Therefore, all states are uniformly represented by X, and there is no subscript CG for the state, control quantity, and system matrix.

5.1.2 Control Scheme Overview

The basic idea of the proposed MO-AMPC is to tailor the ADMM to fit MAMO control systems with the shared agents ($S_j \cap S_{j'} = \emptyset$, $\forall j \neq j'$) as introduced in Section 3.1. An example of a MAMO control system is shown in Figure 5.1. The goal of the MO-AMPC is to decompose the overall system from the objective's (feature's) perspective to achieve a flexible "plug-and-play" fashion.

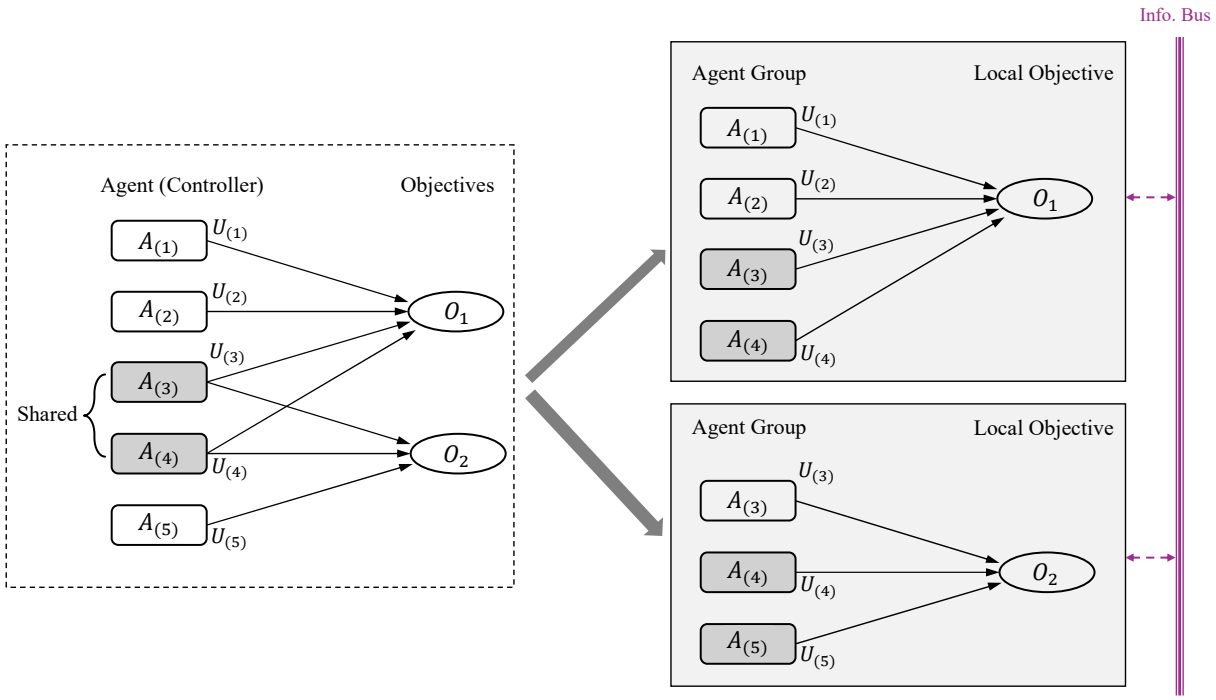


Figure 5.1: Example of decomposing a MAMO control system (dashed line box) from the objective's (feature's) perspective. In this example, agent $A_{(3)}$ and $A_{(4)}$ are shared agents. After the system is decomposed, each local objective will be solved on its own agent group and exchange information with others (solid gray boxes).

ADMM is a well-known method for solving optimization problems for multi-agent systems. It approximates the global optimum using iterative local optimizations on the augmented Lagrangian equation. Through the dual ascent method, it's proven to achieve residual convergence, objective convergence, and dual convergence [110], [111], [112]. In particular, the consensus formulation introduced in Section 3.3 was found to be fitted for distributed control systems. This thesis uses $Z \in \mathbb{R}^{n_z}$ to represent the common global variable in ADMM and $Z_j \in \mathbb{R}^{n_{zj}}$ to represent the local projection from Z for the objective O_j.

The values of dimensions n_z and n_{zj} may differ depending on the distributed formulation in Section 5.2. Similar to the notation of the control action U, over-lines are also used to represent sequences along the prediction horizon for Z: $\bar{Z} = [(Z^{(k)})^T | k = 0, \ldots, Z_p - 1]^T \in \mathbb{R}^{n_z \cdot N_p}$ and $\bar{Z}_j = [(Z_j^{(k)})^T | k = 0, \ldots, N_p - 1]^T \in \mathbb{R}^{n_{zj} \cdot N_p}$. Besides, we use $P_j \in \mathbb{R}^{n_{zj} \cdot N_p \times n_z \cdot N_p}$ as the projection matrix for $\bar{Z}_j = P_j \cdot \bar{Z}$.

Assume the cost function in Equation 5.1 can be broken down into local costs as $J_j(\bar{U}_j)$ and ignore the inequality constraints Ω, then the basic MO-AMPC based on the standard consensus ADMM problem can be written as,

$$\begin{aligned} \min_{\bar{U}_j} \quad & J = \sum_{j=1}^{N_O} J_j(\bar{U}_j) \\ \text{s.t.} \quad & \bar{U}_j - \bar{Z}_j = \mathbf{0}_{n_{zj} \cdot N_p} \end{aligned} \tag{5.3}$$

where J_j are the cost functions of local optimizations on $\bar{U}_j$. There is a one-to-one correspondence between every element in $\bar{U}$ and $\bar{Z}$, and so do $\bar{U}_j$ and $\bar{Z}_j$. Thus, in this basic optimization, $n_z = n_u$ and $n_{zj} = n_{uj}$. The constraint in (5.3) implies the requirement that all local variables in $\bar{U}$ shall converge to the same global variable $\bar{Z}$ in the end since $\bar{Z}_j$ are projections from $\bar{Z}$.

Similar to Equation 3.11, the augmented Lagrangian for the optimization in Equation 5.3 is constructed as,

$$\begin{aligned} & L_\rho(\bar{U}_1, \ldots, \bar{U}_{N_O}, \bar{Z}, \gamma_1, \ldots, \gamma_{N_O}) \\ & \quad = \sum_{j=1}^{N_O} \left[J_j(\bar{U}_j) + \gamma_j^T \bar{U}_j + \frac{\rho_j}{2} \|\bar{U}_j - \bar{Z}_j\|_2^2 \right] \end{aligned} \tag{5.4}$$

where $\gamma_j \in \mathbb{R}^{n_{zj} \cdot N_p}$ is the local Lagrange multiplier as the dual variable for objective O_j, and $\rho_j \in \mathbb{R} > 0$ is a local penalty parameter for objective O_j.

Algorithm 2 shows the basic MO-AMPC algorithm for Equation 5.3 based on the consensus ADMM. It is guaranteed by ADMM that if the problem in 5.3 has a global optimum, which is a saddle point $(\bar{U}, \bar{Z}, \gamma)$ [128]:

(i) The objective evaluated at the primal variables $(\bar{U}, \bar{Z})$ converges to its optimal value.

(ii) the primal residual $(\bar{U} - \bar{Z}_j)$ converge to zero.

(iii) the dual variable γ will converge to a saddle point.

Algorithm 2 Basic Algorithm of MO-AMPC

Input: local cost functions J_j and ρ_j
Local initialization $\gamma_j = \mathbf{0}_{n_{zj} \cdot N_p}$ and $\bar{Z}_j = \mathbf{0}_{n_{zj} \cdot N_p}$
Global initialization $\bar{Z} = \mathbf{0}_{n_z \cdot N_p}$
repeat
    Local Update on each objective: $\bar{U}_j = \arg\min_{\bar{U}_j} L_\rho$
    Broadcast $\bar{U}_j$
    Global Update on $\bar{Z} = \arg\min_{\bar{Z}} L_\rho$
    Project $\bar{Z}$ to update $\bar{Z}_j$
    Dual Update on each Lagrange multiplier:
$\gamma_j = \gamma_j + \rho_j(\bar{U}_j - \bar{Z}_j)$
until convergence achieved through Equation 5.5 for all objectives
Output: The first step of the control variable: $\bar{U}^{(0)}$

The convergence conditions of objective O_j are,

$$\begin{aligned} &\text{Primal: } \|r_j^{\langle q \rangle}\|_2^2 = \|\bar{U}_j^{\langle q \rangle} - \bar{Z}_j^{\langle q \rangle}\|_2^2 \leq \epsilon_r, \text{ AND} \\ &\text{Dual: } \|s_j^{\langle q \rangle}\|_2^2 = \|\rho(\bar{Z}_j^{\langle q \rangle} - \bar{Z}_j^{\langle q-1 \rangle})\|_2^2 \leq \epsilon_s \end{aligned} \tag{5.5}$$

where, $r_j^{\langle q \rangle}$ and $s_j^{\langle q \rangle}$ are the primal and dual residuals at the iteration q, respectively. ϵ_r and ϵ_w are the thresholds.

5.2 Formulations

However, the above basic algorithm neither explains how to decompose the loss function J in 5.1 into an equivalent distributed form as J_j in 5.3 nor does it contain any inequality constraints, which are crucial in establishing a generic distributed control scheme for practical MAMO control systems. The following requirements and goals are needed to establish a generic distributed scheme for practical MAMO control systems:

- *Requirement*: The distributed scheme should converge to the same global optimum as the integrated scheme while always satisfying constraints.
- *Goal*: Improvements in computational efficiency are expected compared to the integrated scheme.

Therefore, three feasible formulations that could meet the requirement are discussed in Section 5.2.1 - 5.2.3, where the key steps of local update, global update, and dual update are given.

5.2.1 Formulation 1: Fully Distributed

As the name describes, this formulation distributes every component related to the local variable U_j into the local optimization, the step-5 in Algorithm 2. Therefore, both the state penalty h_j and the control regularization $g_{(i)}$ are distributed into the local cost function J_j:

$$J_j(\bar{U}_j) = \sum_{k=1}^{N_p} \Big(h_j(X^{(k)}) + \frac{1}{m_{(i)}} \sum_{i \in S_j} g_{(i)}(U_{(i)}^{(k-1)}) \Big) \tag{5.6}$$

where $m_{(i)} = |S_{(i)}|$ is the number of objectives that the agent $A_{(i)}$ is contributing through $U_{(i)}$. So, $m_{(i)}$ should be greater than 1 if $A_{(i)}$ is a shared agent, otherwise $m_{(i)} = 1$. Therefore, $J = \sum J_j$ is satisfied. Besides, the inequality constraints on U_j should also be considered separately in each local optimization.

Local update

Thus, the optimization in Equation 5.2 for each local update in algorithm 2 can be written as,

$$\begin{aligned} \min_{\bar{U}_j} \quad & \sum_{k=1}^{N_p} \Big(\|C_j(X^{(k)} - X_{ref}^{(k)})\|_{R_{xj}}^2 \Big) + \|\bar{U}_j\|_{\bar{R}_{uj}}^2 \\ & \qquad + \gamma_j^T \bar{U}_j + \frac{\rho_j}{2} \|\bar{U}_j - \bar{Z}_j\|_2^2 \\ \text{s.t.} \quad & C_j X^{(k+1)} = C_j(A_d X^{(k)} + B_{d,j} U_j^{(k)} + W_d^{(k)}), \\ & U_j^{(k)} \in \Omega_j^{(k)} := A_{in,j} U_j^{(k)} + B_{in,j} \tilde{U}_j^{(k)} \leq b_{in,j} \end{aligned} \tag{5.7}$$

where $B_{d,j}$ is the discretized and linearized system matrix on control inputs from the agents contributing to objective O_j. R_{uj} is the semi-positive definite diagonal matrix of the average penalty weight for the control action from agents contributing to objective O_j, and $\bar{R}_{uj}$ is the corresponding sequence along the horizon. $A_{in,j} \in \mathbb{R}^{n_{cj} \times n_{uj}}$ and $b_{in,j} \in \mathbb{R}^{n_{cj}}$ are the coefficient matrix and constant vector of the linear inequality constraints of the control action from agents contributing to objective O_j, respectively, where n_{cj} is the dimension of constraints related to U_j. $B_{in,j} \in \mathbb{R}^{n_{cj} \times (n_u - n_{uj})}$ is the coefficient matrix of the

linear inequality constraints of the control action from agents not contributing to objective O_j. $\tilde{U}_j$ is the control actions from the agents that are not included in the agent group S_j and are generated from the previous iteration. Those variables can be written as,

$$\begin{aligned} \bar{R}_{uj} &= \text{diag}\Big(\underbrace{[R_{uj}, \ldots, R_{uj}]}_{N_p}\Big), R_{uj} = \text{diag}\Big(\Big[\frac{R_{(i)}}{m_{(i)}}\Big| i \in S_j\Big]\Big), \\ B_{d,j} &= [B_{d(i)} | i \in S_j], \tilde{U}_j^{(k)\langle q \rangle} = [(U_{(i)}^{(k)\langle q-1 \rangle})^T | i \notin S_j]^T. \end{aligned} \tag{5.8}$$

$A_{in,j}$, $B_{in,j}$, and $b_{in,j}$ are properly calculated from A_{in} and b_{in} to make the constraint in Equation 5.7 be equivalent to the constrain in Equation 5.2 on U_j.

Global update

After the local update, each control action $U_{(i)}^{(k)}$ in $\bar{U}$ (or the corresponding $Z_{(i)}^{(k)}$ in $\bar{Z}$) will receive $m_{(i)}$ distributed updates. The optimization on the global variable $\bar{Z}$ in algorithm 2 can be easily simplified as a global average for each i-th element as [110],

$$Z_{(i)}^{(k)} = \frac{1}{m_{(i)}} \sum_{j \in S_{(i)}} U_{j(i)}^{(k)} \tag{5.9}$$

where $U_{j(i)}^{(k)}$ is the control action for agent $A_{(i)}$ updated from the local optimization for O_j at the k-th step.

Dual update

Keep the same update on γ_j as Algorithm 2.

Formulation summary

In this formulation, the global and dual updates are simple and computationally efficient. However, the optimization in the local update is relatively complex because the regularization of the control action is considered in each local node.

5.2.2 Formulation 2: Virtual Center Node

In this formulation, the regularization of the control action is handled centrally, while only state penalty h_j is distributed in the local cost function J_j. It is called "virtual" because the centralized regularization can be computed on a physical center node or replicated on each local node. Hence, the integrated cost function in Equation 5.1 can be written as,

$$J = \bar{g}(\bar{Z}) + \sum_{j=1}^{N_O} f_j(\bar{U}_j) \tag{5.10}$$

where,

$$J_j(\bar{U}_j) = \sum_{k=1}^{N_p} h_j(X^{(k)}), \quad \bar{g}(\bar{Z}) = \sum_{k=1}^{N_p} g(Z^{(k-1)}). \tag{5.11}$$

The function $\bar{g}(\bar{Z})$ could handle the regularization for all the global control variables in a (virtual) center node. Then, the augmented Lagrangian is rewritten as,

$$\begin{aligned} &L_\rho(\bar{U}_1, \ldots, \bar{U}_{N_O}, \bar{Z}, \gamma_1, \ldots, \gamma_{N_O}) \\ &\qquad = \bar{g}(\bar{Z}) + \sum_{j=1}^{N_O} \Big[J_j(\bar{U}_j) + \gamma_j^T \bar{U}_j + \frac{\rho_j}{2} \|\bar{U}_j - \bar{Z}_j\|_2^2 \Big] \end{aligned} \tag{5.12}$$

Local update

Similar to the *Fully Distributed* formulation, inequality constraints should also be considered separately in each local optimization. Thus, the optimization for each local update in algorithm 2 can be written as,

$$\begin{aligned} \min_{\bar{U}^j} \quad & \sum_{k=1}^{N_p} \Big(\|C_j(X^{(k)} - X_{ref}^{(k)})\|_{R_{xj}}^2 \Big) + {\gamma_j}^T \bar{U}_j \\ & \qquad + \frac{\rho_j}{2} \|\bar{U}_j - \bar{Z}_j\|_2^2 \\ \text{s.t.} \quad & C_j X^{(k+1)} = C_j(A_d X^{(k)} + B_{d,j}^j U_j^{(k)} + W_d^{(k)}), \\ & U_j^{(k)} \in \Omega_j^{(k)} := A_{in,j} U_j^{(k)} + B_{in,j} \tilde{U}_j^{(k)} \leq b_{in,j} \end{aligned} \tag{5.13}$$

where $A_{in,j}$, $B_{in,j}$, $b_{in,j}$, and $\tilde{U}_j^{(k)}$ have the same definition as in Equation 5.7 and Equation 5.8.

Global update

The regularization function on control actions is handled directly on the global variable Z instead of the local variable U_j. Therefore, the global update in algorithm 2 can be derived as,

$$Z_{(i)}^{(k)} = \frac{\sum_{j \in S_{(i)}} (\gamma_{j(i)}^{(k)} + \rho_j \cdot U_{j(i)}^{(k)})}{2R_{u(i)} + \sum_{j \in S_{(i)}} \rho_j} \tag{5.14}$$

where $\gamma_{j(i)}^{(k)}$ is the $(k \cdot N_p + i)$-th element in γ_j. The regularization penalty $R_{u(i)}$ is shown as a part of the denominator: the $Z_{(i)}^{(k)}$ would be close to 0 if $R_{(i)}$ holds a very big value, while the $Z_{(i)}^{(k)}$ would be close to Equation 5.9 if $R_{(i)}$ holds a very small value since γ_j would be zeros after the first iteration.

Dual update

Keep the same update on γ_j as Algorithm 2.

Formulation summary

The global update using an explicit average expression in this formulation is still efficient, while the local update is a bit simpler than the *Fully Distributed* formulation.

5.2.3 Formulation 3: Local Explicit

The main advantage of this formulation is converting the inequality constraints into indicator functions showing in the cost functions, allowing local updates to be solved explicitly through simple matrix operations. To achieve this, the common global variables Z will no longer correspond to control actions U. Instead, inspired by [112], the global variable Z will be redefined as a slack variable to transform the inequality constraints into equality constraints. Hence, the integrated MPC in Equation 5.1 can be rewritten as,

$$\begin{aligned} \min_{\bar{U}} \quad & J = L_+(\bar{Z}) + \sum_{j=1}^{N_O} J_j(\bar{U}_j) \\ \text{s.t.} \quad & X^{(k+1)} = f_d(X^{(k)}, U^{(k)}), \\ & U^{(k)} \in \Omega^{(k)} := A_{in} U^{(k)} - b_{in} + Z^{(k)} = \mathbf{0}_{n_z} \end{aligned} \tag{5.15}$$

where $J_j(\bar{U}_j)$ has the same expression as the *Fully Distributed* formulation in Equation 5.6. L_+ is an indicator penalty function that puts an infinite penalty on negative components of the slack variable $\bar{Z}_j$, which makes it equivalent to the original optimization problem. It is worth noting that the dimension of Z in this formulation is no longer the same as U. Instead, the dimension of Z is the same as that of the constraint, that is, $n_z = n_c$ and $n_{zj} = n_{cj}$. Then, the augmented Lagrangian is rewritten as,

$$\begin{aligned} &L_\rho(\bar{U}_1, \ldots, \bar{U}_{N_O}, \bar{Z}, \gamma_1, \ldots, \gamma_{N_O}) \\ &\quad = L_+(\bar{Z}) + \sum_{j=1}^{N_O} \Big[J_j(\bar{U}_j) + \gamma_j^T \bar{U}_j + \frac{\rho_j}{2} \|\bar{U}_j - \bar{Z}_j\|_2^2 \Big] \end{aligned} \tag{5.16}$$

It is worth mentioning that, since the indicator function $L_+(\bar{Z})$ is not related to $\bar{U}_j$, the indicator function $L_+(\bar{Z})$ and the equality constraint $U^{(k)} \in \Omega^{(k)}$ can be considered in the global update instead of the local update, to simplify the optimization of each local update.

Local update

The j-th local update can be written as,

$$\begin{aligned} \min_{\bar{U}_j} \quad & \sum_{k=1}^{N_p} \Big(\|C_j(X^{(k)} - X_{ref}^{(k)})\|_{R_{xj}}^2 \Big) + \|\bar{U}_j\|_{\bar{R}_{uj}}^2 + \gamma_j^T \bar{U}_j + \frac{\rho_j}{2} \|\bar{U}_j - \bar{Z}_j\|_2^2 \\ \text{s.t.} \quad & C_j X^{(k+1)} = C_j (A_d X^{(k)} + B_{d,j} U_j^{(k)} + W_d^{(k)}) \end{aligned} \tag{5.17}$$

Therefore, the local update in (5.17) can be simply solved in an explicit expression as,

$$\bar{U}_j = -(H_j + \rho_j \bar{A}_{in,j}^T \bar{A}_{in,j})^{-1} \cdot \Big[E_j + \rho_j \bar{A}_{in,j}^T (\bar{Z}_j + \frac{1}{\rho^j} \gamma^j - \bar{b}_{in,j}) \Big] \tag{5.18}$$

where,

$$\begin{aligned} H_j &= 2\Big(Y_{u,j}^T \bar{Q}_j Y_{u,j} + \bar{R}_j \Big), \\ E_j &= 2 Y_{u,j}^T \bar{Q}_j \Big(Y_{x,j} X^{(0)} + Y_{w,j} \bar{W}_j - \bar{X}_{ref,j} \Big), \\ \bar{Q}_j &= \mathrm{diag}\Big([\underbrace{Q_j, \ldots, Q_j}_{N_p}] \Big), \end{aligned} \tag{5.19}$$

and,

$$
\begin{aligned}
\bar{X}_{ref,j} &= \left[\left(C_j X_{ref}^{(k)}\right)^T \mid k = 0, \ldots, N_p - 1 \right]^T, \\
\bar{W}_j &= \left[\left(C_j W_d^{(k)}\right)^T \mid k = 0, \ldots, N_p - 1 \right]^T, \\
Y_{u,j} &= \begin{bmatrix} C_j A_d^0 B_{d,j} & \cdots & \mathbf{0} \\ \vdots & \ddots & \mathbf{0} \\ C_j A_d^{N_p-1} B_{d,j} & \cdots & C_j A_d^0 B_{d,j} \end{bmatrix}, \\
Y_{x,j} &= \left[\left(C_j A_d^p\right)^T \mid p = 1, \ldots, N_p \right]^T, \\
Y_{w,j} &= \begin{bmatrix} C_j A_d^0 & \cdots & \mathbf{0} \\ \vdots & \ddots & \mathbf{0} \\ C_j A_d^{N_p-1} & \cdots & C_j A_d^0 \end{bmatrix}, \\
\bar{A}_{in,j} &= \operatorname{diag}\Big(\underbrace{[A_{in,j}, \ldots, A_{in,j}]}_{N_p} \Big), \\
\bar{b}_{in,j} &= \left[\left(b_{in,j} - B_{in,j} \tilde{U}_j^{(k)}\right)^T | k = 0, \ldots, N_p - 1 \right]^T.
\end{aligned}
\tag{5.20}
$$

Global update

In this formulation, shared agents lead to shared constraints. Let's define $c = \{1, \ldots, n_c\}$ to index the constraints of $U^{(k)} \in \Omega^{(k)}$. Then $Z_{(c)}$ will be associated with the c-th equation in Ω in Equation 5.15. We further use the expression $i \to c$ to represent that $U_{(i)}^{(k)}$ is linearly correlated to $Z_{(c)}^{(k)}$ through A_{in}, and use $m_{(c)}$ to denote the number of objectives that $Z_{(c)}^{(k)}$ is involved. Then we have:

$$
m_{(c)} = \left| \bigcup_{i \to c} S_{(i)} \right| \tag{5.21}
$$

A "max" operation can be used to handle indicator function $L_+(\bar{Z})$ and the equality constraint $\Omega^{(k)}$ for the global update in algorithm 2, which can be written as,

$$
Z_{(c)}^{(k)} = \max \left\{ 0, \frac{1}{m_{(c)}} \sum_{j=1}^{m_{(c)}} (-\bar{A}_{in,j} \bar{U}_j - \frac{1}{\rho^j} \gamma^j + \bar{b}_{in,j}) |_{(c)}^{(k)} \right\} \tag{5.22}
$$

where $(\cdot)|_{(c)}^{(k)}$ represent the $(k \cdot N_p + c)$-th element in $(\cdot)$.

Dual update

The update of the Lagrange multiplier in algorithm 2 can be updated as,

$$\gamma_j = \gamma_j + \rho_j(\bar{A}_{in,j}\bar{U}_j - \bar{b}_{in,j} + \bar{Z}_j) \tag{5.23}$$

Formulation summary

This formulation's biggest advantage is explicitly solving the local optimization for each objective. The satisfaction of constraints is achieved through the "max" operation of the virtual center node. Through iterative convergence, the final result can simultaneously meet the requirements of global optimum and constraint satisfaction. One requirement for this formulation is that each control variable $U_{(i)}$ should have an inequality constraint to be associated with at least one global variable $Z_{(c)}$. This requirement is reasonable in practical engineering because at least there is a min-max constraint for each actuator. The discussion about the convergence and the optimal parameter selection can be found in the next section.

5.3 Convergence and Parameter Selection

Table 5.1 summarizes the three forms proposed in Section 5.2. Computing locations (local or center node) of the penalty costs on tracking error, control regularization, and constraint satisfaction are listed.

Table 5.1: Formulation Comparison of MO-AMPC

Formulation	Tracking Penalty	Control Regularization	Constraint
Fully Distributed	Local	Local	Local
Virtual Center Node	Local	Center	Local
Local Explicit	Local (explicit)	Local (explicit)	Center

All three formulations strictly follow the ADMM framework, whose convergence has been proven in literature, such as [110]. The control action $\bar{U}$ generated from these distributed formulations will approximate the global optimal from the integrated MPC, as the residuals $\|r_j^{\langle q \rangle}\|_2^2$ and $\|s_j^{\langle q \rangle}\|_2^2$ converge to zero through iterations.

ρ_j is an important parameter in the proposed MO-AMPC because it affects the convergence speed. For the first two formulations, it is difficult to have a deterministic method to determine the value of ρ_j. So, it is usually selected based on experience in practice. However, for the last formulation, the guidance of choosing a proper ρ_j has been proposed in [112] on a similar distributed QP problem by optimizing the upper bound of the converging rate of $\|s_j^{\langle q \rangle}\|_2^2$. Similarly, the optimal selection of ρ_j for the proposed *Local Explicit* formulation in this paper can be written as,

$$\rho_j^* = \frac{1}{\sqrt{\lambda_{min}(\bar{A}_{in,j} H_j^{-1} \bar{A}_{in,j}^T) \lambda_{max}(\bar{A}_{in,j} H_j^{-1} \bar{A}_{in,j}^T)}} \tag{5.24}$$

where $\lambda_{min}(\cdot) > 0$ and $\lambda_{max}(\cdot) > 0$ are the minimum non-zero and maximum eigenvalues of the bracketed matrix $(\cdot)$, respectively.

The detailed discussion of convergence and parameter selection for the proposed *Local Explicit* formulation is given in Appendix B.

5.4 Example: Cruise Control

5.4.1 Agent Configuration and Controller Design

The longitudinal and lateral objectives should be met simultaneously in cruise control. The former is a task of longitudinal speed tracking (LST) (j = LST), while the latter is a stability control of yaw rate tracking (YRT) (j = YRT). In this example, we have four agents: the DTR (i = DTR) agent generates the driving torque on the rear axle of a rear-wheel-drive (RWD) vehicle; the RTV (i = RTV) agent generates the opposite torques on the left and right wheels of the rear axle; the FDB (i = FDB) agent generates individual braking torques on the front axle; the driver steering (i = STR) agent generates the front steering angle. In cruise control systems, the STR agent is uncontrollable for electronic controllers since the driver controls the steering, and its contribution is treated as an external disturbance in W_d in the system equation. The configuration is shown in Figure 5.2, where the agents' contributions are obtained by analyzing the following system equations. In this example, The control actions from each agent are considered as,

$$\begin{aligned}
&\text{DTR: } U_{(\text{DTR})} = Q_{\text{DTR}} \in [-2000, 2000](\text{Nm}) \\
&\text{RTV: } U_{(\text{RTV})} = Q_{\text{RTV}} \in [-500, 500](\text{Nm}) \\
&\text{FDB: } U_{(\text{FDB})} = [Q_{\text{FDB},fl}, Q_{\text{FDB},fr}]^T \in [-1000, 0](\text{Nm}) \\
&\text{STR: } U_{(\text{STR})} = [F_{y,fl}, F_{y,fr}, F_{y,rl}, F_{y,rr}]^T
\end{aligned} \tag{5.25}$$

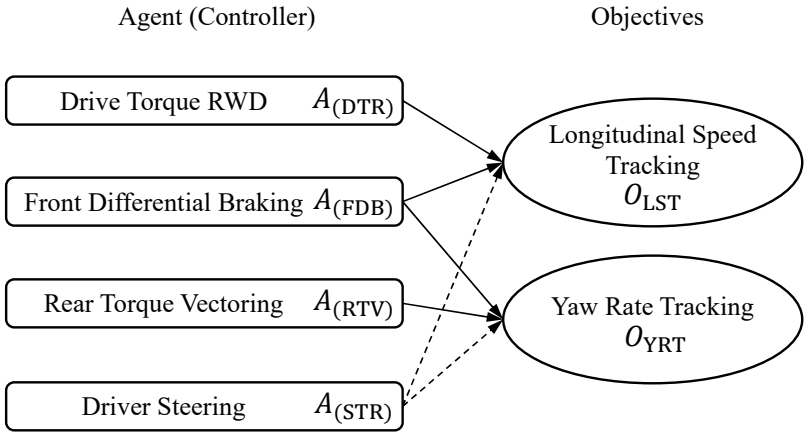


Figure 5.2: Example of the MAMO control system for cruise control. The two objectives share the FDB agent. The STR agent is the external input on both objectives, so its contributions are drawn as dashed arrows. Thus, $S_{\mathrm{YRT}} = \{\mathrm{RTV}, \mathrm{FDB}\}$, $S_{\mathrm{LST}} = \{\mathrm{DTR}, \mathrm{FDB}\}$. For agents: $S_{(\mathrm{DTR})} = \{\mathrm{LST}\}$, $S_{(\mathrm{FDB})} = \{\mathrm{YRT}, \mathrm{LST}\}$, $S_{(\mathrm{RTV})} = \{\mathrm{YRT}\}$.

where, Q_{DTR} is the total drive torque. Q_{RTV} is the differential torque applied on the right-rear wheel, while the opposite is applied on the left. $Q_{\mathrm{FDB},fl}$ and $Q_{\mathrm{FDB},fr}$ are the independent braking torque applied on the right-front and left-front wheels, respectively. F_y are the estimated lateral tire forces.

State X is considered as the longitudinal speed (v_x), lateral speed (v_y), and yaw rate (r). Then, for the objectives, we have:

$$\begin{gathered} X_{ref} = [v_{x,des}, 0, r_{des}], \\ r_{des} = \mathrm{sign}(\delta_f) \min\left(|\tfrac{v_x}{L_b + K_{us} v_x^2} \delta_f|, \tfrac{\mu g_a}{v_x} \right), \\ C_{\mathrm{YRT}} = [1, 0, 0],\ C_{\mathrm{LST}} = [0, 0, 1] \end{gathered} \tag{5.26}$$

where δ_f is the input of the driver's steering angle. μ is the road friction. g_a is the gravity acceleration. $v_{x,des}$ and r_{des} are the desired longitudinal speed from the upper-level speed planner and the desired yaw rate calculated from the single-track vehicle model and limited by the road friction, respectively.

The system matrices in Equation 5.2 for cruise control could be written as

$$
\begin{aligned}
A_d &= \begin{bmatrix} 1 & rT_s & v_yT_s \\ -rT_s & 1 & -v_xT_s \\ 0 & 0 & 1 \end{bmatrix}, \\
B_{d(\mathrm{DTR})} &= [T_s/(mR_w), 0, 0]^T, \\
B_{d(\mathrm{RTV})} &= [0, 0, T_sT_w/(I_zR_w)]^T, \\
B_{d(\mathrm{FDB})} &= T_s \cdot \begin{bmatrix} \cos\delta_{fl}/(mR_w) & \cos\delta_{fr}/(mR_w) \\ \sin\delta_{fl}/(mR_w) & \sin\delta_{fr}/(mR_w) \\ \dfrac{2l_f\sin\delta_{fl} - T_w\cos\delta_{fl}}{2I_zR_w} & \dfrac{2l_f\sin\delta_{fr} + T_w\cos\delta_{fr}}{2I_zR_w} \end{bmatrix}, \\
W_d &= B_{d,\mathrm{STR}}U_{\mathrm{STR}}, \\
B_{d(\mathrm{STR})} &= T_s \cdot \begin{bmatrix} -\frac{1}{m}\sin\delta_{fl} & -\frac{1}{m}\sin\delta_{fr} & 0 & 0 \\ \frac{1}{m}\cos\delta_{fl} & \frac{1}{m}\cos\delta_{fr} & \frac{1}{m} & \frac{1}{m} \\ \left(\begin{array}{l}\frac{l_f}{I_z}\cos\delta_{fl} \\ +\frac{T_w}{2I_z}\sin\delta_{fl}\end{array}\right) & \left(\begin{array}{l}\frac{l_f}{I_z}\cos\delta_{fr} \\ -\frac{T_w}{2I_z}\sin\delta_{fr}\end{array}\right) & -\frac{l_r}{I_z} & -\frac{l_r}{I_z} \end{bmatrix}.
\end{aligned}
\tag{5.27}
$$

where T_s is the sample time. δ_{fr} and δ_{fl} are right and left wheel steering angles at the front axle, respectively.

5.4.2 Simulation Results

Based on the above example, simulations were conducted to demonstrate the features of the proposed MO-AMPC control scheme. A high-fidelity calibrated CarSim model of this vehicle from a previous experimental study [127] is adopted for simulations, whose main parameters are specified in Appendix C. The controller runs at a frequency of 20 Hz.

A DLC during acceleration from 40 to 60 kph was conducted on a high friction ($\mu = 0.85$) surface. Agents and objectives were configured as in Figure 5.2. The desired path and longitudinal speed profile were pre-defined in both the controller and CarSim.

Comparison between the *Local Explicit* formulation and integrated MPC

The results of the vehicle response and control actions are shown in Figure 5.3. An aggressive DLC was conducted during the acceleration.

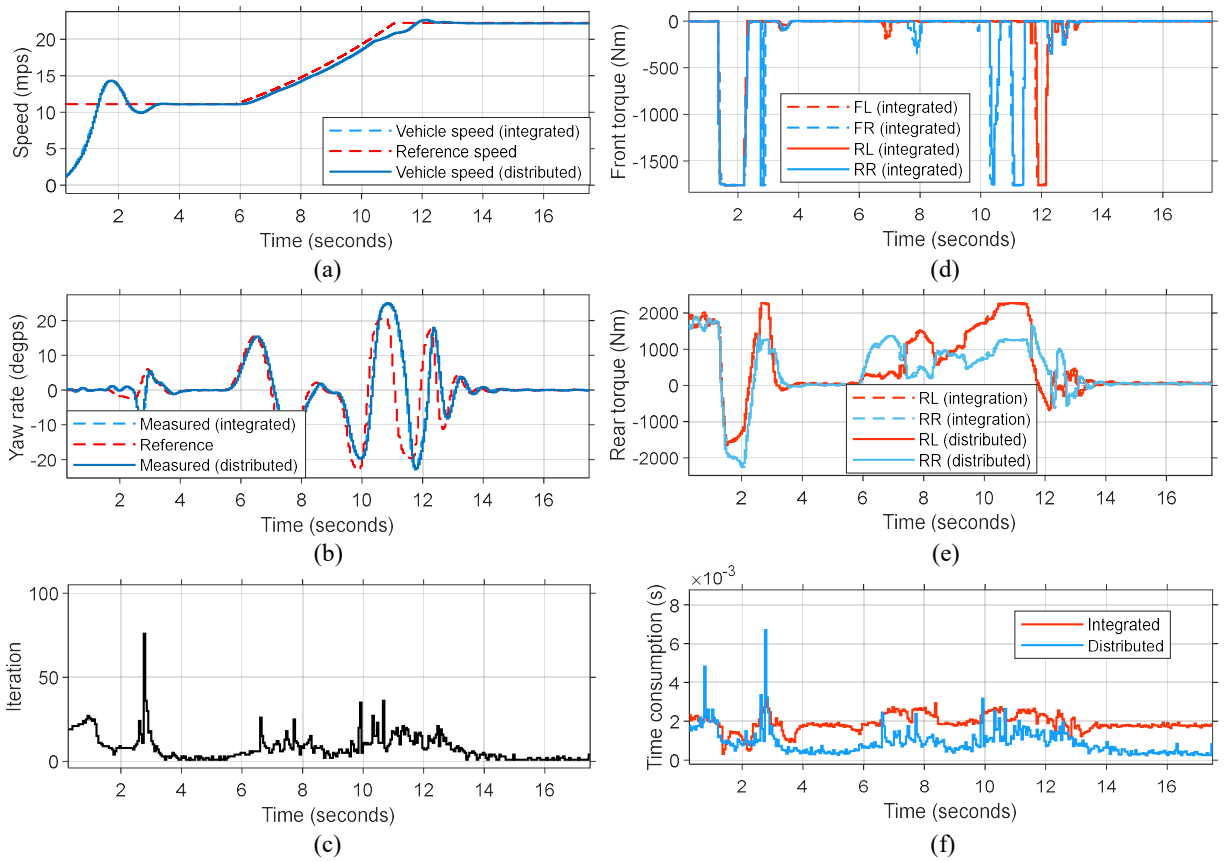


Figure 5.3: Simulation results of the vehicle response and control actions in the cruise control example using MO-AMPC. Figures (a) and (b) show the longitudinal speed and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC.

In comparison figures (a), (b), (d), and (e), the results from the MO-AMPC controller with *Local Explicit* formulation are shown as solid blue lines, while results from the integrated MPC are shown as dashed blue lines. For both two controllers, Figures (a) and (b) show good longitudinal speed and yaw rate tracking performance with small tracking errors. The vehicle speeds and yaw rates controlled by the two controllers are very close, which is because they generated almost the same control actions (torques in each wheel) as shown in Figures (d) and (e), showing the convergence of the proposed distributed method.

Besides, the significant improvement by using the MO-AMPC controller with *Local Explicit* formulation on computational efficiency can be seen in Figure (f) [2].

Comparison of three formulations

To better evaluate the convergence performance, we define the suboptimality as the absolute error between the control actions between the MO-AMPC and the integrated MPC:

$$e_{sub} = |U^{(0)}_{distributed} - U^{(0)}_{integrated}| \tag{5.28}$$

where $U^{(0)}_{distributed}$ and $U^{(0)}_{integrated}$ are the output control actions from the MO-AMPC and the integrated MPC, respectively. We also define r_{time} as the time consumption ratio of the distributed scheme and the integrated scheme:

$$r_{time} = \frac{t_{distributed}}{t_{integrated}} \tag{5.29}$$

where $t_{distributed}$ and $t_{integrated}$ are the time consumption using the distributed and integrated schemes, respectively.

Using different formulations, figure 5.4 compares the suboptimality and time consumption ratios. The parameter ρ_j is manually selected for the first two formulations, *Fully Distributed* and *Virtual Center Node*, to achieve the best efficiency, and it's calculated using Equation 5.24 for the *Local Explicit* formulation. As shown in the results, all three formulations can converge to the global optimum since the majority of the torque suboptimalities are less than 20 Nm (5 Nm for *Local Explicit*), which is acceptable and even can be ignored in practice. The *Local Explicit* formulation achieved the best computational efficiency and even ran faster than the integrated formulation with $r_{time} < 1$.

Theoretically, the advantages of ADMM-based optimization methods could only be dominant in high-dimensional and low-coupling systems ([111]). However, even for such a small multi-agent system (in terms of the value of N_A and N_O), the MO-AMPC in the *Local Explicit* formulation with the optimal parameter can still outperform the integrated MPC on the computational efficiency while converging to the global optimum. This should be attributed to its efficient explicit optimization in local updates and the optimal parameter selection. Therefore, all the rest examples in this chapter and the experiments in Section 7.3 and 7.4 will use the MO-AMPC with the *Local Explicit* formulation and optimal parameters as the distributed control method.

[2] In this thesis, the time consumption in all comparisons only counts the core processes of solving MPCs, excluding matrix initialization and post-processing.

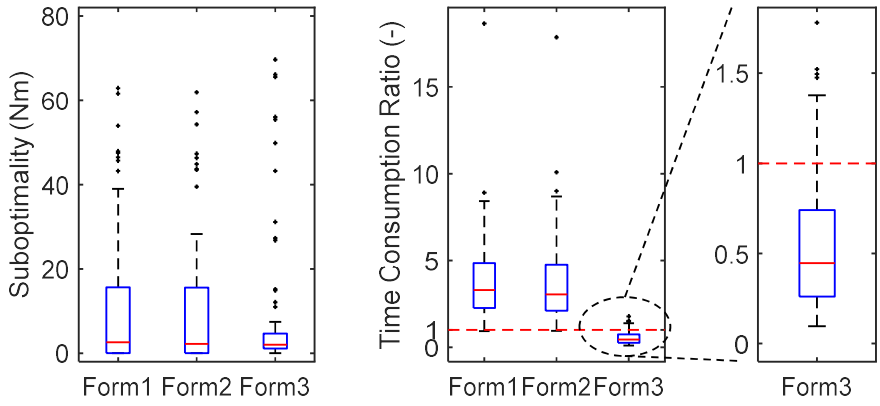


Figure 5.4: Comparison results between three formulations: Form 1 (*Fully Distributed*), Form 2 (*Virtual Center Node*, and Form 3 (*Local Explicit*). The suboptimality is the absolute error of the torque control actions between the MO-AMPC and the integrated MPC. The formulation of MO-AMPC is said to be more computationally efficient than the integrated MPC if the time consumption ratio is less than 1.

5.5 Example: Holistic Vehicle Stability Control

5.5.1 Agent Configuration and Controller Design

Holistic vehicle stability control is another challenging task that requires coordinating multiple controllers to meet various objectives. In this example, the objective of vehicle stability control (VSC) tracks the desired yaw rate, and the wheel stability control (WSC) tracks the desired wheel speed for maximum tire capability. This example has one more agent than the *Cruise Control*: the traction control rear (TCR) (i = TCR) agent will reduce the driving torque on the rear axle if the rear wheels start to slip. The control actions generated from agent TCR are considered as,

$$\text{TCR: } U_{(\text{TCR})} = Q_{\text{TCR}} \in [0, Q_{\text{DTR}}], \tag{5.30}$$

where, Q_{TCR} is the torque reduction in driving torque. In this example, the STR and DTR agents are uncontrollable, and their contributions are treated as external disturbances in objective VSC and WSC. The configuration is shown in Fig. 5.5.

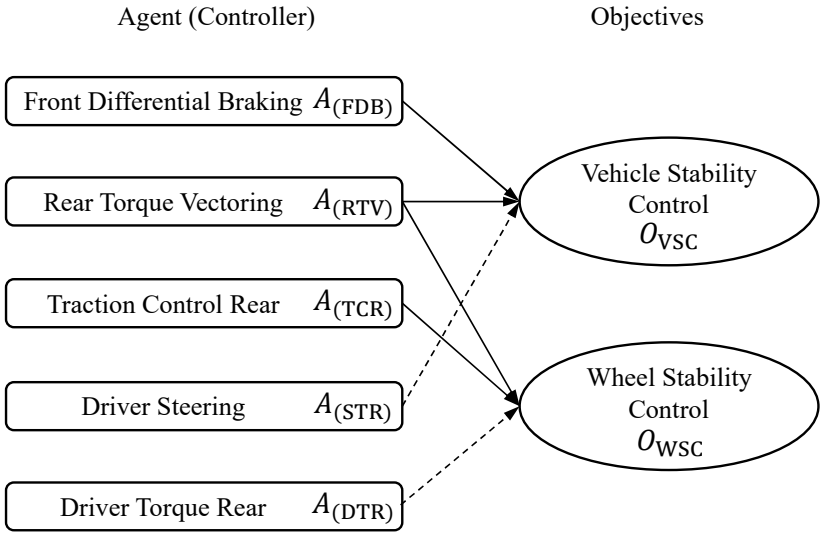


Figure 5.5: Example of the MAMO control system for holistic vehicle stability control. The two objectives share the RTV agent. The STR and DTR agents are the external input on each objective, respectively, so their contributions are drawn as dashed arrows. Thus, $S_{\text{VSC}} = \{\text{RTV}, \text{FDB}\}$, $S_{\text{WSC}} = \{\text{TCR}, \text{RTV}\}$. For agents: $S_{(\text{FDB})} = \{\text{VSC}\}$, $S_{(\text{RTV})} = \{\text{VSC}, \text{WSC}\}$, $S_{(\text{TCR})} = \{\text{WSC}\}$.

The objectives in this example can be written as:

$$\begin{aligned} X_{ref} &= [0, r_{des}, \omega_{des}, \omega_{des}], \\ \omega_{des} &= v_x(1 + S_{x,des}) \cdot [1, 1]^T \\ C_{\text{VSC}} = [0, 1, 0, 0],\ C_{\text{WSC}} &= \begin{bmatrix} 0, 0, 1, 0 \\ 0, 0, 0, 1 \end{bmatrix}, \end{aligned} \tag{5.31}$$

where δ_f and v_x are inputs of the driver's steering angle at front wheels and vehicle speed, respectively. The target yaw rate r_{des} is the same as in Equation 5.26. ω_{des} is the desired wheel speed. $S_{x,des}$ is the desired longitudinal slip ratio, which usually is 15%.

State X is considered as the lateral speed (v_y), yaw rate ($\dot{\phi}$), and speeds of rear wheels (ω_{RL}, ω_{RR}). Then, the system matrices in (5.2) for holistic vehicle control could be written

as,

$$
\begin{aligned}
A_d &= \begin{bmatrix} 1 & -v_x T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\
B_{d(\mathrm{TCR})} &= [0, 0, -T_s/(2I_w), -T_s/(2I_w)]^T, \\
B_{d(\mathrm{RTV})} &= [0, T_s T_w/R_w, -T_s/I_w, T_s/I_w]^T, \\
B_{d(\mathrm{FDB})} &= T_s \cdot \begin{bmatrix} \sin\delta_{fl}/mR_w & \sin\delta_{fr}/(mR_w) \\ \dfrac{2l_f \sin\delta_{fl} - T_w \cos\delta_{fl}}{2I_z R_w} & \dfrac{2l_f \sin\delta_{fr} + T_w \cos\delta_{fr}}{2I_z R_w} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \\
W_d &= B_{d(\mathrm{DTR})} U_{(\mathrm{DTR})} + B_{d(\mathrm{STR})} U_{(\mathrm{STR})}, \\
B_{d(\mathrm{DTR})} &= [0, 0, T_s/(2I_w), T_s/(2I_w)]^T, \\
B_{d(\mathrm{STR})} &= T_s \cdot \begin{bmatrix} \frac{1}{m}\cos\delta_{fl} & \frac{1}{m}\cos\delta_{fr} & \frac{1}{m} & \frac{1}{m} \\ \left(\begin{array}{l} \frac{l_f}{I_z}\cos\delta_{fl} \\ +\frac{T_w}{2I_z}\sin\delta_{fl} \end{array}\right) & \left(\begin{array}{l} \frac{l_f}{I_z}\cos\delta_{fr} \\ -\frac{T_w}{2I_z}\sin\delta_{fr} \end{array}\right) & -\frac{l_r}{I_z} & -\frac{l_r}{I_z} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.
\end{aligned}
\tag{5.32}
$$

5.5.2 Simulation Results

The same setting as the simulations for the cruise control is used in this example. A high-fidelity calibrated CarSim model of this vehicle from a previous experimental study [127] is adopted for simulations, whose main parameters are specified in Appendix C. The controller runs at a frequency of 20 Hz.

DLC in acceleration on a high friction surface

A DLC during acceleration was conducted on a high friction ($\mu = 0.85$) surface. Agents and objectives were configured as in Figure 5.5. The results of the vehicle response and control actions are shown in Figure 5.6.

In comparison figures (d) and (e), the results from the MO-AMPC controller with *Local Explicit* formulation are shown as solid lines, while results from the integrated MPC

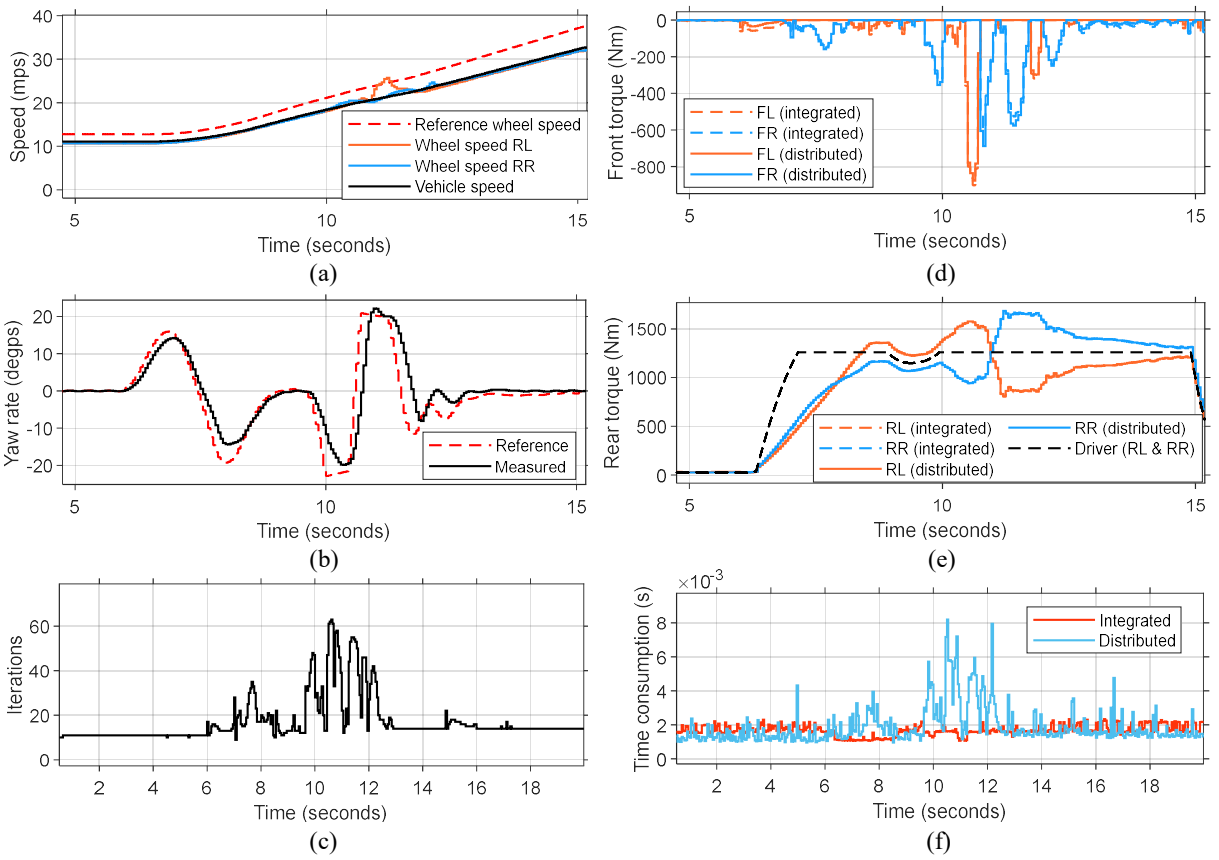


Figure 5.6: Simulation results of the vehicle response and control actions in the holistic stability control example using MO-AMPC, conducting a DLC in acceleration on a high friction surface. Figures (a) and (b) show the wheel speed control and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC.

are shown as dashed lines. During the whole process, there is no significant wheel speed exceeding the desired wheel speed, as shown in Figure (a), maintaining the wheel stability. Figure (b) shows good yaw rate tracking performance with small tracking errors. Different agents were enabled at different times, and their control actions can be seen in Figures (d) and (e), ensuring both the stability of the wheels and the vehicle. Besides, agents generated almost the same control actions (torques in each wheel) as shown in Figures

(d) and (e), showing the convergence of the proposed distributed method. Besides, similar computational cost by using the MO-AMPC controller with *Local Explicit* formulation and the integrated MPC can be seen in Figure (f).

DLC in acceleration on a low friction surface

A DLC during acceleration was also conducted on a low friction ($\mu = 0.3$) surface. Agents and objectives were configured as in Figure 5.5. The results of the vehicle response and control actions are shown in Figure 5.7.

In comparison figures (d) and (e), the results from the MO-AMPC controller with *Local Explicit* formulation are shown as solid lines, while results from the integrated MPC are shown as dashed lines. During the whole process, there is no significant wheel speed exceeding the desired wheel speed, as shown in Figure (a), maintaining the wheel stability. Figure (b) shows good yaw rate tracking performance with small tracking errors. Different agents were enabled at different times, and their control actions can be seen in Figures (d) and (e), ensuring both the stability of the wheels and the vehicle. Besides, agents generated almost the same control actions (torques in each wheel) as shown in Figures (d) and (e), showing the convergence of the proposed distributed method. Besides, a slightly lower computational cost by using the MO-AMPC controller with *Local Explicit* formulation compared to the integrated MPC can be seen in Figure (f).

Acceleration on a split-mu surface

A straight acceleration on a split-mu surface was conducted in the simulation, where the left wheels were driving on the low friction ($\mu = 0.3$) surface while the right wheels were driving on the high friction ($\mu = 0.85$) surface. The results of the vehicle response and control actions are shown in Figure 5.8.

In comparison figures (d) and (e), the results from the MO-AMPC controller with *Local Explicit* formulation are shown as solid lines, while results from the integrated MPC are shown as dashed lines. Different agents were enabled at different times, and their control actions can be seen in Figures (d) and (e). They ensured the stability of both the wheels and the vehicle. During the whole process, there was no significant wheel speed exceeding the desired wheel speed, as shown in Figure (a). The rear wheels followed the desired speed smoothly, maintaining wheel stability without losing any power, thanks to the significant torque reduction on the rear axle, shown in Figure (e). At the same time, Figure (b) shows good yaw rate tracking performance without losing yaw stability because the FDB and

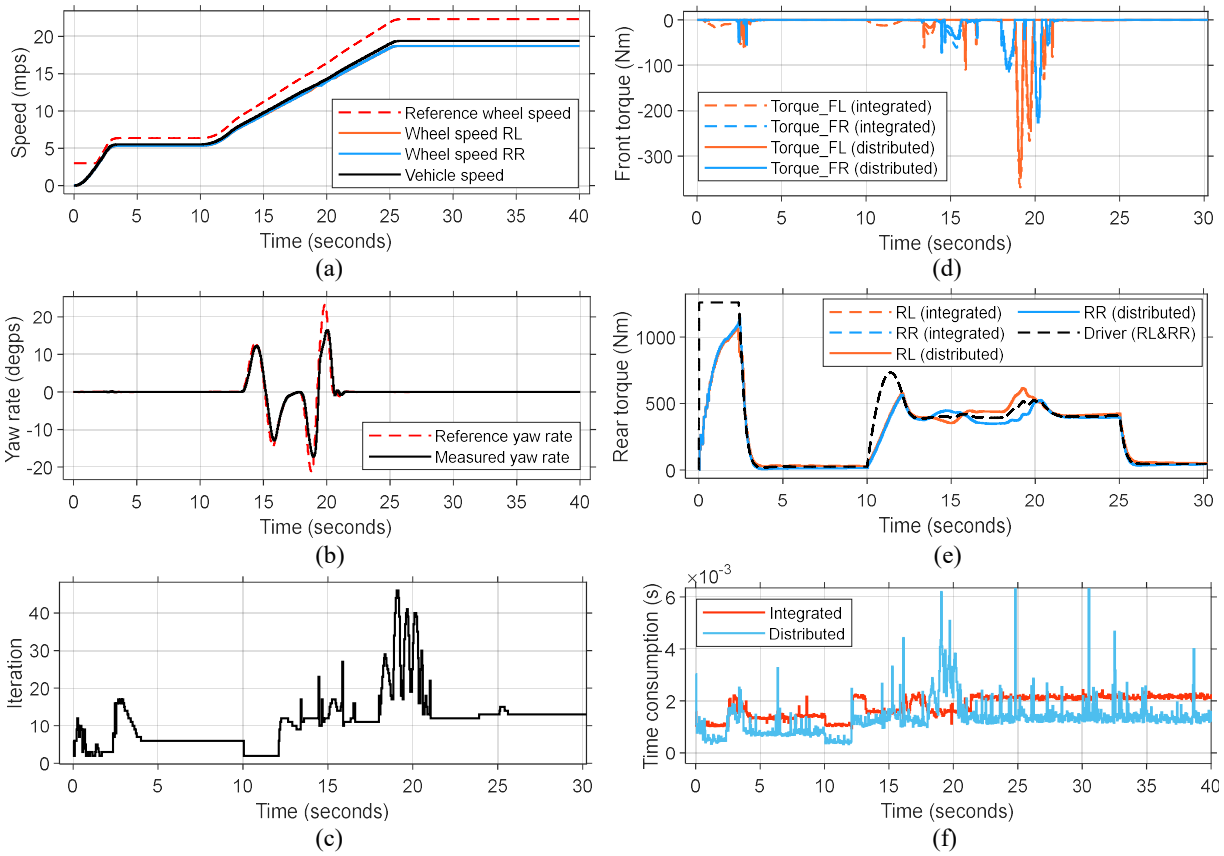


Figure 5.7: Simulation results of the vehicle response and control actions in the holistic stability control example using MO-AMPC, conducting a DLC in acceleration on a low friction surface. Figures (a) and (b) show the wheel speed control and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC.

RTV generate significantly different torques on the left and right side for additional yaw moment, as shown in Figure (b) and (e). Besides, agents generated almost the same control actions (torques in each wheel) as shown in Figures (d) and (e), showing the convergence of the proposed distributed method. Besides, a slightly higher computational cost by using the MO-AMPC controller with *Local Explicit* formulation compared to the integrated MPC can be seen in Figure (f).

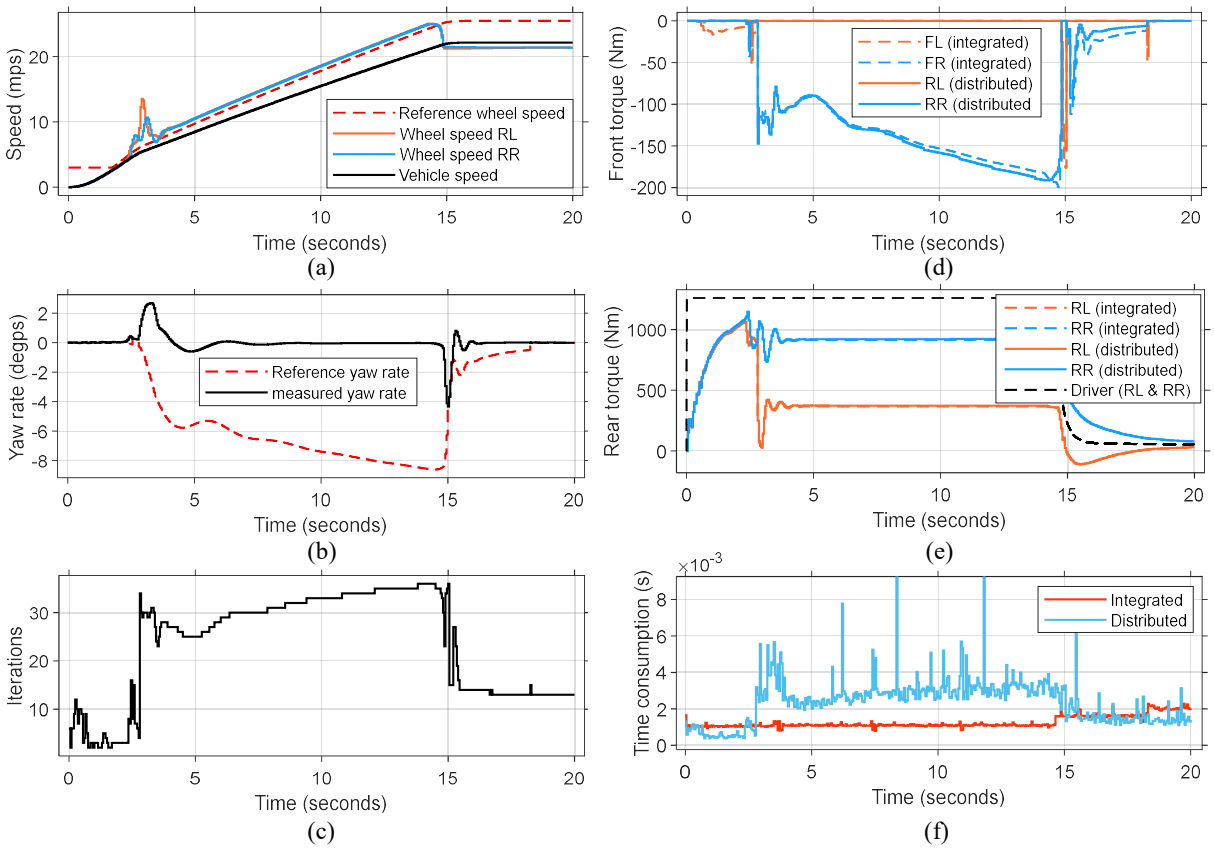


Figure 5.8: Simulation results of the vehicle response and control actions in the holistic stability control example using MO-AMPC, conducting a straight acceleration on a split-mu surface. Figures (a) and (b) show the wheel speed control and yaw rate tracking performance, respectively. Figures (d) and (e) show the final net torques on the front and rear wheels, respectively. Figure (c) shows the number of iterations needed for convergence at each time step. Figure (f) shows the comparison of time consumption between the *Local Explicit* formulation and the integrated MPC.

5.6 Summary

This chapter provided a complete introduction to the proposed MO-AMPC scheme for "plug-and-play" vehicle control. First, the integrated MPC for MAMO control systems is introduced as the baseline. Next, the basic algorithm of the MO-AMPC tailored from the consensus ADMM is designed. Then, three different formulations considering system dy-

namics and linear inequality constraints are derived. These formulations achieve the global optimum while managing control regularization and linear inequality constraints. Furthermore, two examples of MAMO control systems using the proposed MO-AMPC are given mathematically. The proposed formulations were evaluated through simulations, demonstrating that the *Local Explicit* formulation offers the highest computational efficiency and has the potential to outperform the integrated MPC.

It has been proven by simulations that the MO-AMPC scheme introduced in this chapter offers a practical and versatile distributed solution for complex vehicle control systems:

1) The MO-AMPC scheme enhances design flexibility by decoupling the system from the objective's perspective.

2) The MO-AMPC scheme can achieve global optimization while meeting the standard requirements of real-world control systems.

3) The MO-AMPC scheme also has the potential to provide superior real-time computational efficiency compared to integrated MPC.

Chapter 6

Learning Multi-Objective Agent-Based Model Predictive Control

This chapter presents the L-MO-AMPC scheme for HVC. It is structured as follows: First, the overview design of the L-MO-AMPC scheme is given in Section 6.1. Then, the detailed algorithm of L-MO-AMPC is developed in Section 6.2. Simulation results are demonstrated following an example of an autonomous path-tracking task in Section 6.3.

6.1 Overview

It can be seen from the simulations results in Section 5.4 and 5.5 that the MO-AMPC control scheme proposed in the previous chapter can converge to the global optimum and potentially has a better computational efficiency compared to the integrated MPC. However, as introduced in Section 1.3, the computational time consumption of the MO-AMPC will increase with the number of iterations, which is a huge challenge for real-time control systems. To this end, the L-MO-AMPC is proposed in this chapter, leveraging learning-based methods to reduce the number of iterations for higher computational efficiency.

The computational time consumption of MO-AMPC is highly correlated with the number of iterations needed for converging. Therefore, if the number of iterations can be reduced through a data-driven approach, it will be very helpful for real-time performance. A simple example of a MAMO control system with one shared agent ($A_{(2)}$) is shown in Figure

6.1. In this example, the trajectories of control actions $U_{(2)}$ generated from the two local optimizations in MO-AMPC converged iteratively to the global optimum, shown as the black point in the right figure. The goal of the proposed L-MO-AMPC in this chapter is to generate closer initial control actions, shown as the red circles. In this way, the iterations needed for converging will be significantly reduced, shown as dashed lines.

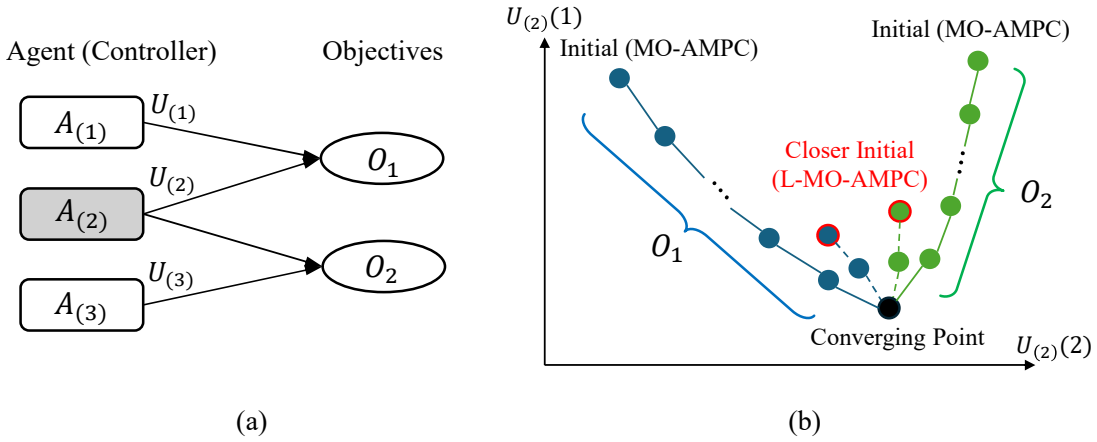


Figure 6.1: Example of a MAMO control system with three agents and two objectives. Figure (a) is the structure of the system where agent $A_{(2)}$ is the shared agent. Figure (b) shows the converging process of control action $U_{(2)}$ with $n_{u(2)} = 2$. The blue and green solid lines represent the iterative trajectory of $U_{(2)}$ from MO-AMPC, respectively. The dashed lines represent the iterative trajectory from L-MO-AMPC.

In complex MAMO control systems, objectives are usually coupled because shared agents affect more than one objective simultaneously, where negotiation between objectives and coordination between agents exist. The negotiation or interaction between objectives should be learned to generate better initial guesses of the shared control action. Therefore, the GP-based learning method introduced in Section 3.4 is developed to predict the contributions for each objective and initialize the iterations to accelerate the convergence process.

6.2 Algorithm Design

The overall structure of the proposed L-MO-AMPC scheme is shown in Figure 6.2.

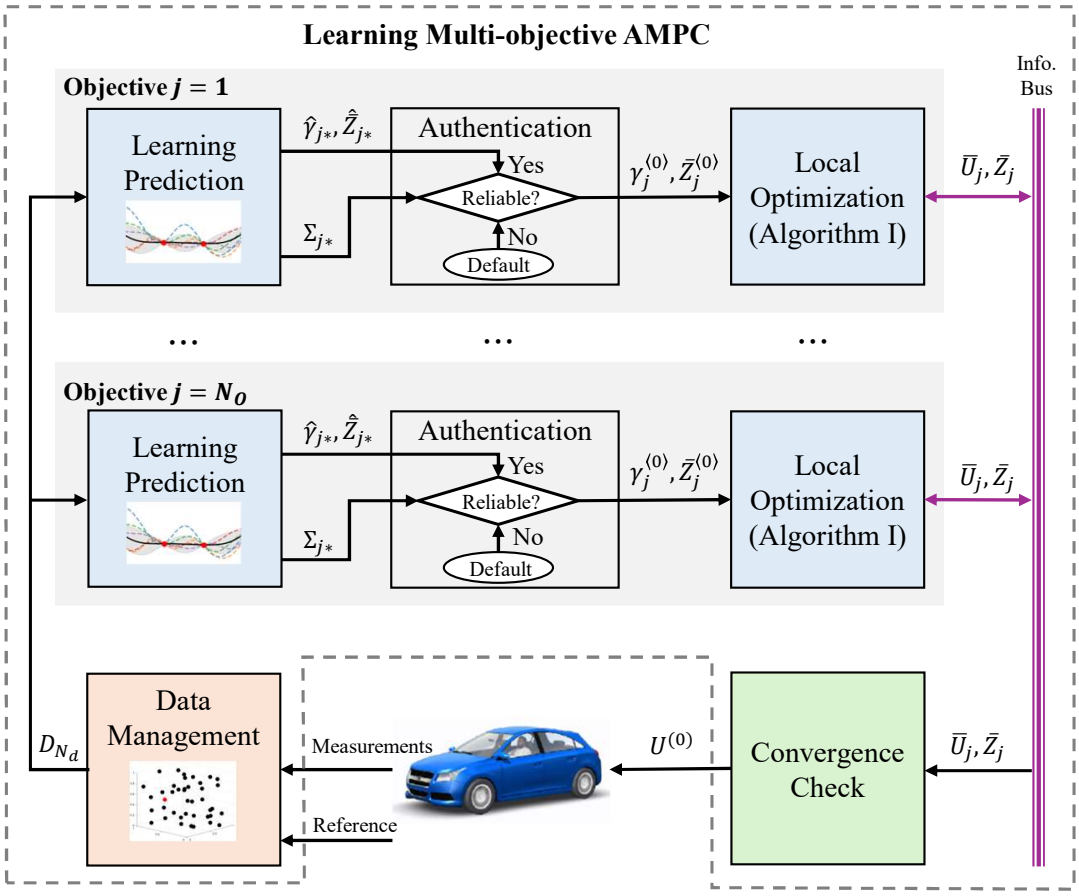


Figure 6.2: The overall structure of the L-MO-AMPC. Each objective has its local optimization module, initialized by a learning prediction module after a reliability authentication.

- The data management introduced in Section 4.2.1 and 4.2.2 is adopted as the pre-processing for learning modules. Data density is controlled, and the most relevant data subset is selected for fast online inference. The dataset is updated online and could be saved as or loaded from an off-line dataset. The data management module can be designed to be either centralized or fully distributed.

- Each objective has a learning prediction model to predict the negotiations from others for closer initial control actions. At the same time, the predicted variance is used as a criterion for the reliability check. The predicted mean value will be used to generate initial control actions only if the reliability check is passed. Otherwise, a default value of zeros is used as in MO-AMPC.

- Local optimizations for each objective are based on the MO-AMPC proposed in the previous chapter. The control actions from shared agents will converge iteratively by exchanging information.
- The convergence check is based on the convergence conditions of all objectives. It can also be designed to be either centralized or fully distributed. The iteration will stop, and the final control actions will be applied as long as the convergence check is passed.

6.2.1 Negotiation between Objectives

The core idea of the L-MO-AMPC is to learn the negotiation between objectives. In the Algorithm 2 of MO-AMPC, γ_j and $\bar{Z}_j$ will eventually converge to fixed values, indicating the compromise of the j-th objective during the convergence process. Furthermore, in the local update of the *Local Explicit* formulation (Equation 5.17 $\sim$ 5.20), the control action $\bar{U}_j$ is determined if the value of γ_j and $\bar{Z}_j$ is known because other elements, like H_j, ρ_j, $\bar{A}_{in,j}$, and $\bar{b}_{in,j}$, are determined by the system configuration and parameters. Therefore, if the values of γ_j and $\bar{Z}_j$ can be predicted from collected data and used as the initial values $\gamma_j^{\langle 0 \rangle}$ and $\bar{Z}_j^{\langle 0 \rangle}$ during local update in the Algorithm 2, the optimized control input $\bar{U}_j^{\langle 0 \rangle}$ from the first iteration will be closer to the final global optimum. This way, the number of iterations can be reduced, and the convergence process can be accelerated.

Therefore, for each objective, the first iteration of the local update in Algorithm 2 can be rewritten in L-MO-AMPC as,

$$\begin{aligned} \bar{U}_j^{\langle 0 \rangle} &= \arg\min_{\bar{U}_j} L_\rho(\gamma_{j*}^{\langle 0 \rangle}, \bar{Z}_j^{\langle 0 \rangle}) \\ [\gamma_{j*}^{\langle 0 \rangle T}, \bar{Z}_j^{\langle 0 \rangle T}]^T &= \begin{cases} \hat{y}_{j*} & \text{if} \Sigma_* \leq \epsilon_l, \\ \mathbf{0}_{2 \cdot n_{zj} \cdot N_p} & \text{otherwise.} \end{cases} \\ \hat{y}_{j*} &= [\hat{\gamma}_{j*}^T, \hat{\bar{Z}}_{j*}^T]^T \end{aligned} \tag{6.1}$$

where L_ρ is the augmented Lagrangian in Equation 5.4. $\hat{y}_{j*} \in \mathbb{R}^{2 \cdot n_{zj} \cdot N_p}$ and $\Sigma_* \in \mathbb{R}^1$ are the predicted mean and variance from GPR in Equation 3.17, respectively. ϵ_l is the threshold of the learning reliability check. $\hat{y}_{j*}$ includes the predicted mean $\hat{\gamma}_{j*} \in \mathbb{R}^{n_{zj} \cdot N_p}$ and $\hat{\bar{Z}}_{j*} \in \mathbb{R}^{n_{zj} \cdot N_p}$. Then, each j-th objective in the scheme has a local optimization based on MO-AMPC. The optimized control actions $\bar{U}_j$ and slack variables $\bar{Z}_j$ will be updated at each iteration and shared through the information bus. Iterations will end, and the

final control action $U^{(0)}$ will be output if the convergence check is passed. Each j-th local optimization has a corresponding learning module predicting the mean values of the Lagrange multiplier γ_j and slack variables $\bar{Z}_j$, indirectly reflecting the compromise during the convergence process. An authentication module will arbitrate based on the prediction variance Σ_{j*}: Predictions are considered reliable as initial values $\gamma_j^{\langle 0\rangle}$ and $\bar{Z}_j^{\langle 0\rangle}$ only when the variance Σ_{j*} is less than a given threshold; otherwise, default values (usually are 0 vectors) will be used.

6.2.2 Learning and Post-Processing

GPR prediction in Equation 3.17 is used as the learning-based method in L-MO-AMPC. The input variables are vehicle states and are selected using the model selection method introduced in Section 4.2.1. The hyper-parameters are trained using an offline dataset as in Section 3.4. The online data management process ensures continuous learning ability and high inference efficiency by managing the data density to prevent the data from being too aggregated or clustered. The minimum Euclidian distance is used to determine the action for a new data point:

1. *Add the new input point*: The new data point is far from all the existing points in the dataset.
2. *Replace the nearest point*: The new data point is close to an existing point, but their outputs are significantly different. New information needs to be updated.

Furthermore, the KNN based on Euclidian distance are used to select a subset D_{N_d} for efficient GPR inference. Then the time complexity will reduce from $\mathcal{O}(N_D^3)$ to $\mathcal{O}(N_d^3)$ if only $N_d \leq N_D$ data points are selected.

6.3 Example: Autonomous Path-Tracking Control

An extended single-track vehicle model considering the independent left and right torque control and an error-based path-tracking model proposed in [129] are used in this work, as shown in Fig. 6.3. The single-track model is obtained as,

$$\begin{aligned}
\dot{v}_y &= -\frac{C_{\alpha f}+C_{\alpha r}}{m v_x} v_y - (v_x + \frac{l_f C_{\alpha f} - l_r C_{\alpha r}}{m v_x}) r + \frac{C_{\alpha f}\delta_f}{m} \\
\dot{r} &= \frac{l_r C_{\alpha r} - l_f C_{\alpha f}}{I_z v_x} v_y - \frac{l_f^2 C_{\alpha f} + l_r^2 C_{\alpha r}}{I_z v_x} r + \frac{l_f C_{\alpha f}\delta_f + M_{zt}}{I_z}
\end{aligned} \tag{6.2}$$

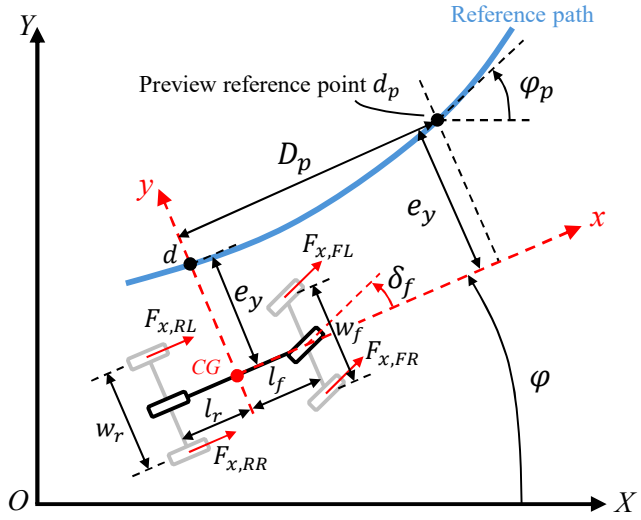


Figure 6.3: Vehicle and path tracking model.

where m is the vehicle mass; I_z is the yaw inertia; v_x, v_y and r are the longitudinal velocity, lateral velocity and yaw rate, respectively; $C_{\alpha f}$ and $C_{\alpha r}$ are the cornering stiffness of the front and rear axles, respectively; l_f and l_r are the distance from the vehicle CG to front and rear axles, respectively; δ_f is the front wheel steering angle, and M_{zt} is the total active yaw moment other control agents generate. The error-based path-tracking model can be derived as,

$$\begin{aligned} \dot{e}_y &= v_x e_\varphi + v_y + r D_p \\ \dot{e}_\varphi &= \dot{\varphi} - \dot{\varphi}_{des} = r - r_{des} \\ \dot{e}_u &= F_{xt}/m - a_{x,des} = \dot{v}_x - \dot{v}_{x,des} \end{aligned} \tag{6.3}$$

where e_y is the lateral error from the vehicle CG to the preview point d_p on the reference path; the preview point d_p is calculated based on a preview distance D_p from the closest point d on the reference path; e_φ is the heading error between the actual heading angle of vehicle φ and the desired heading angle φ_{des}; e_u is the longitudinal speed error between the actual speed v_x and the desired speed $v_{x,des}$; $a_{x,des}$ is the desired longitudinal acceleration; F_{xt} is the total longitudinal forces. The preview distance D_p is usually selected based on experience and can vary depending on vehicle speed.

The F_{xt} and the active yaw moment M_{zt} can be expressed with the small steering angle assumption as,

$$\begin{aligned} F_{xt} &= (Q_{fl} + Q_{fr} + Q_{rl} + Q_{rr})/R_w \\ M_{zt} &= \Big[(Q_{fl} + Q_{fr})T_{wf} + (Q_{rl} + Q_{rr})T_{wr}\Big]/2R_w \end{aligned} \tag{6.4}$$

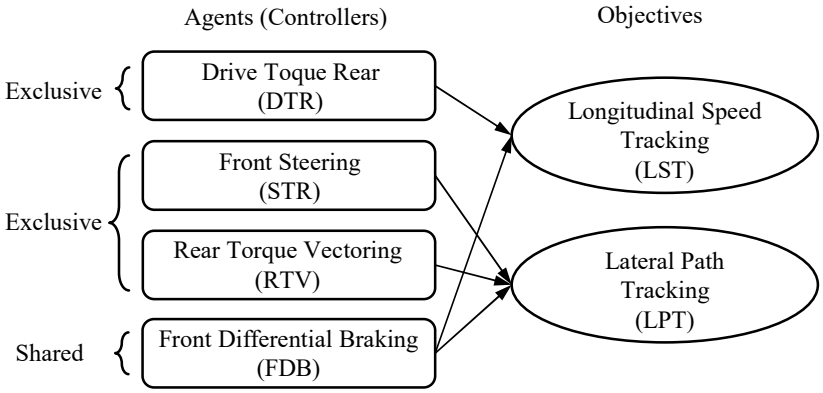


Figure 6.4: Example of the MAMO control system for path-tracking. The two objectives share the FDB agent. For objective: $S_{\text{LST}} = \{\text{DTR}, \text{FDB}\}$, $S_{\text{LPT}} = \{\text{STR}, \text{RTV}, \text{FDB}\}$. For agents: $S_{(\text{DTR})} = \{\text{LST}\}$, $S_{(\text{STR})} = \{\text{LPT}\}$, $S_{(\text{FDB})} = \{\text{LST}, \text{LPT}\}$, $S_{(\text{RTV})} = \{\text{LPT}\}$.

where $Q_i, i = \{fl, fr, rl, rr\}$ are wheel torques at front-left, front-right, rear-left, and rear-right corners, respectively; T_{wf} and T_{wr} are the front and rear track widths; R_w is the wheel radius. Wheels torques are generated jointly by various agents.

The agent configuration is shown in Figure 6.4. Agents collaborate to achieve LST and lateral path tracking (LPT) simultaneously. δ_f, Q_{DTR}, Q_{RTV}, $Q_{\text{FDB},fl} \leq 0$, and $Q_{\text{FDB},fr} \leq 0$ are steering from STR agent, and torque inputs from the DTR, RTV, and FDB at left and right wheels, respectively. Then, the wheel torques can be expressed as,

$$\begin{aligned} Q_{fl} &= Q_{\text{FDB},fl}, \quad Q_{fr} = Q_{\text{FDB},fr}, \\ Q_{rl} &= \frac{Q_{\text{DTR}}}{2} - Q_{\text{RTV}}, \quad Q_{rr} = \frac{Q_{\text{DTR}}}{2} + Q_{\text{RTV}} \end{aligned} \tag{6.5}$$

As a baseline, Equations (6.2) $\sim$ (6.5) can be combined into an integrated state-space form as in (5.1), where

$$\begin{aligned} X &= [e_u, e_y, e_\varphi, v_y, r]^T, CX = [e_u, e_y, e_\varphi]^T \\ U &= [\delta_f, Q_{\text{DTR}}, Q_{\text{RTV}}, Q_{\text{FDB},fl}, Q_{\text{FDB},fr}]^T \end{aligned} \tag{6.6}$$

The controller agents can be split into two groups based on Figure 6.4: longitudinal tracking (j = LST) and lateral tracking (j = LPT), sharing the FDB agent. For each subsystem, the states, control inputs, and outputs in (5.17) can be defined as,

$$\begin{aligned} U_{\text{LST}} &= [Q_{\text{DTR}}, Q_{\text{FDB},fl}, Q_{\text{FDB},fr}]^T, Y_{\text{LST}} = [e_u]^T \\ U_{\text{LPT}} &= [\delta_f, Q_{\text{RTV}}, Q_{\text{FDB},fl}, Q_{\text{FDB},fr}]^T, Y_{\text{LPT}} = [e_y, e_\varphi]^T \end{aligned} \tag{6.7}$$

As discussed in Section 6.2, γ_j and $\bar{Z}_j$ from the previous final iteration at the previous time step are collected as data outputs y. The data inputs x are often empirically selected and filtered through statistical criteria. According to the system equation, we select the states from measurements and the reference path as inputs. Thus, the inputs and outputs of a data point $(x^{(t)}, y_j^{(t)})$ are collected from the previous time step as follows:

$$\begin{aligned} x^{(t)} &= [e_u^{(t-1)}, e_y^{(t-1)}, e_\varphi^{(t-1)}, v_y^{(t-1)}, r^{(t-1)}, v_{x,des}^{(t-1)}, \dot{\varphi}_{des}^{(t-1)}]^T \\ y_j^{(t)} &= [\gamma_j^{(t-1)}, \bar{Z}_j^{(t-1)}]^T \end{aligned} \tag{6.8}$$

The new data point will be collected if it brings new information (experience). Subsequently, the most relevant data subset will be selected for online GPR inference to predict $\hat{y}_{j*} = [\hat{\gamma}_{j*}^T, \hat{\bar{Z}}_{j*}^T]^T$ as in Equation 6.1.

6.4 Simulation Results

This section presents several comparison simulation results on a racing track scenario in CarSim, with the controller running in Matlab/Simulink, as shown in Fig. 6.5.

The integrated MPC was running as a baseline using the `active-set` algorithm in the Matlab built-in `quadprog` function. The reference path will be processed at the beginning of each time step. All the controllers run in real-time at a fixed 20Hz frequency on a laptop with an i7-12700H CPU and 16 GB RAM. A high-fidelity calibrated CarSim model of this vehicle from a previous experimental study [127] is adopted for simulations, whose main parameters are specified in Appendix C.

6.4.1 Convergence

First, numerical verification of convergence is verified through simulations. The control actions from the proposed L-MO-AMPC scheme were compared with the integrated MPC, as shown in Figure 6.6. The median errors between the L-MO-AMPC and integrated MPC results are very small and acceptable in practice, as shown in Table 6.1, showing the convergence of the proposed scheme to the integrated scheme, which generates the global optimum.

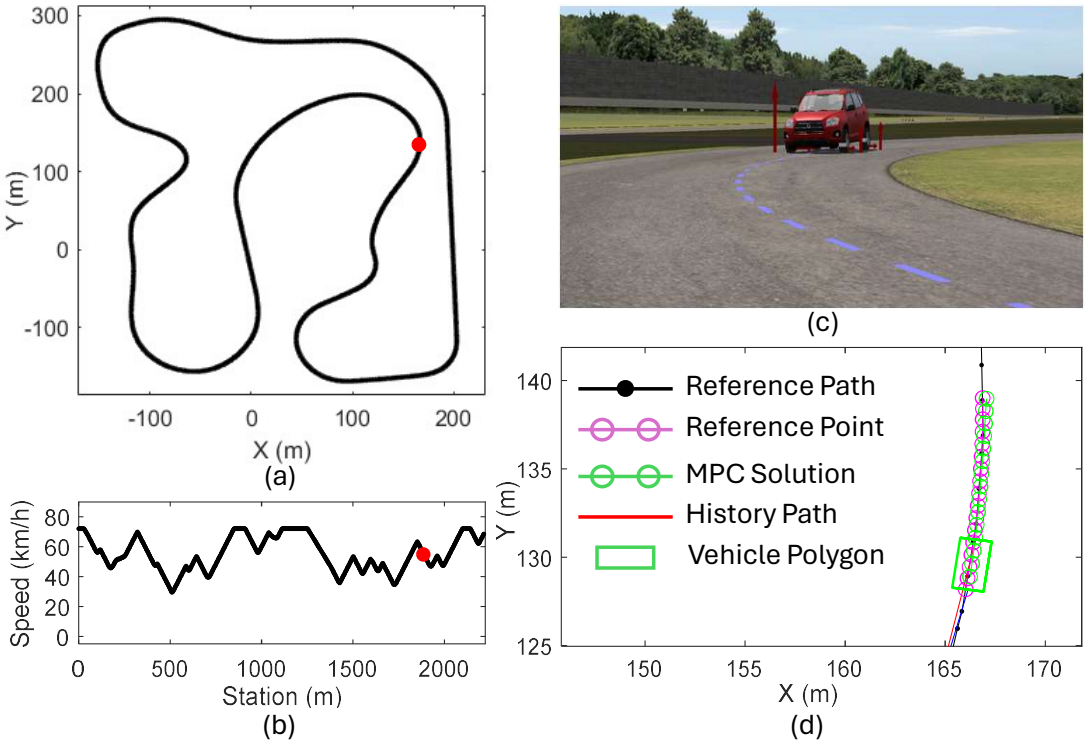


Figure 6.5: Real-time simulation of the combined path-tracking on a racing track. (a) Global reference path. (b) Reference speed profile. (c) Visualization in CarSim. (d) Control results. (c) and (d) are at station $\approx 1900m$ and marked as red dots in (a) and (b). A comparison of the converging processes at this position is shown in Section 6.4.2.

Table 6.1: Quantitative Convergence Accuracy of L-MO-AMPC

Control action	STR (deg)	DTR (Nm)	RTV (Nm)	FDB-L (Nm)	FDB-R (Nm)
Absolute error*	0.0001	0.9544	0.0998	3.3464	3.4309

*The median value of one lap data.

6.4.2 Improvements by Learning

Several consecutive laps were simulated, starting with an empty dataset, as shown in Figure 6.7. The number of data points grew fast in the first lap but slower after because of the data density management. As more data was collected, the GPR predictions became more

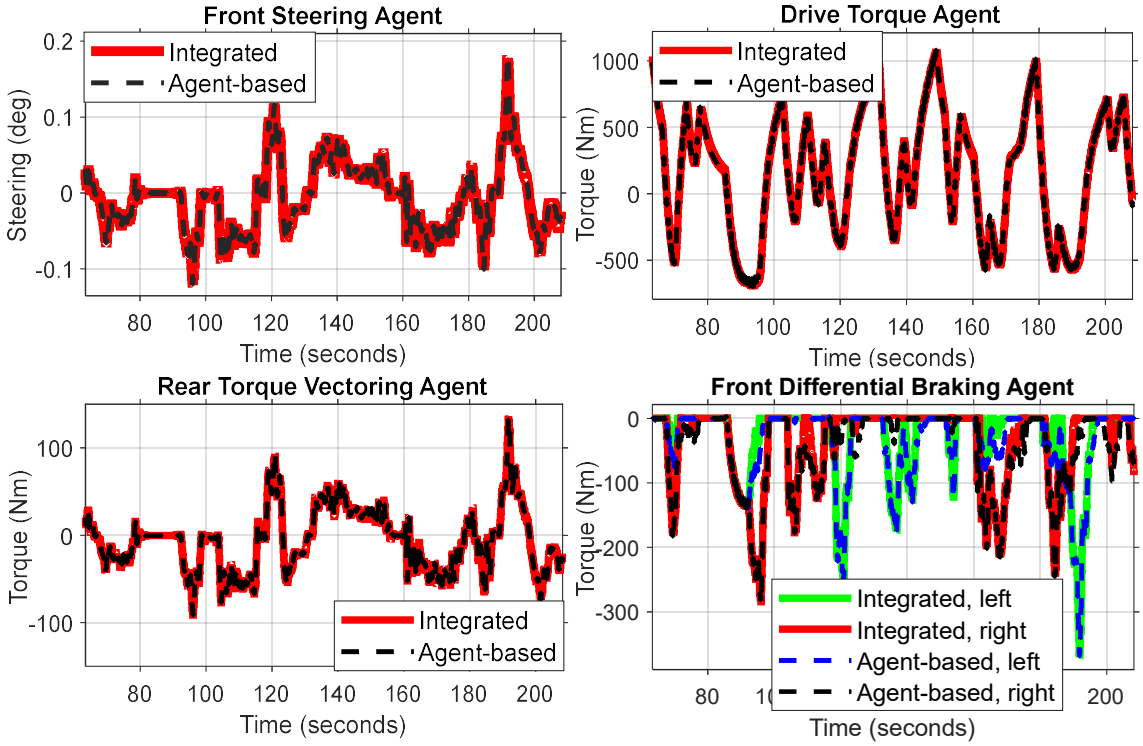


Figure 6.6: Comparison of control actions between the proposed L-MO-AMPC scheme and the integrated scheme.

accurate as the variance decreased.

The Comparison results on computational efficiency during the second lap are also shown in Fig. 6.7. The proposed MO-AMPC needed about $10 \sim 20$ iterations at each time step to converge, reducing the computational time by 42.5% compared to the integrated MPC on average. Furthermore, learning the negotiations among objectives required only less than 5 iterations. As a result, the L-MO-AMPC significantly reduced the computational time by 88.6% and 92.8% compared to without learning and the integrated MPC, respectively.

Detailed converging processes of the shared FDB agent at the station $\approx 1900m$ are compared in Figure 6.8. Results from the first lap without learning predictions are shown in (a) on the left, where FDB torques from the two objectives needed 13 iterations to converge. However, results from the second lap with reliable learned initial values $\gamma_j^{\langle 0 \rangle}$ and $\bar{Z}_j^{\langle 0 \rangle}$ are shown in (b) on the right, where only three iterations were needed. In (b),

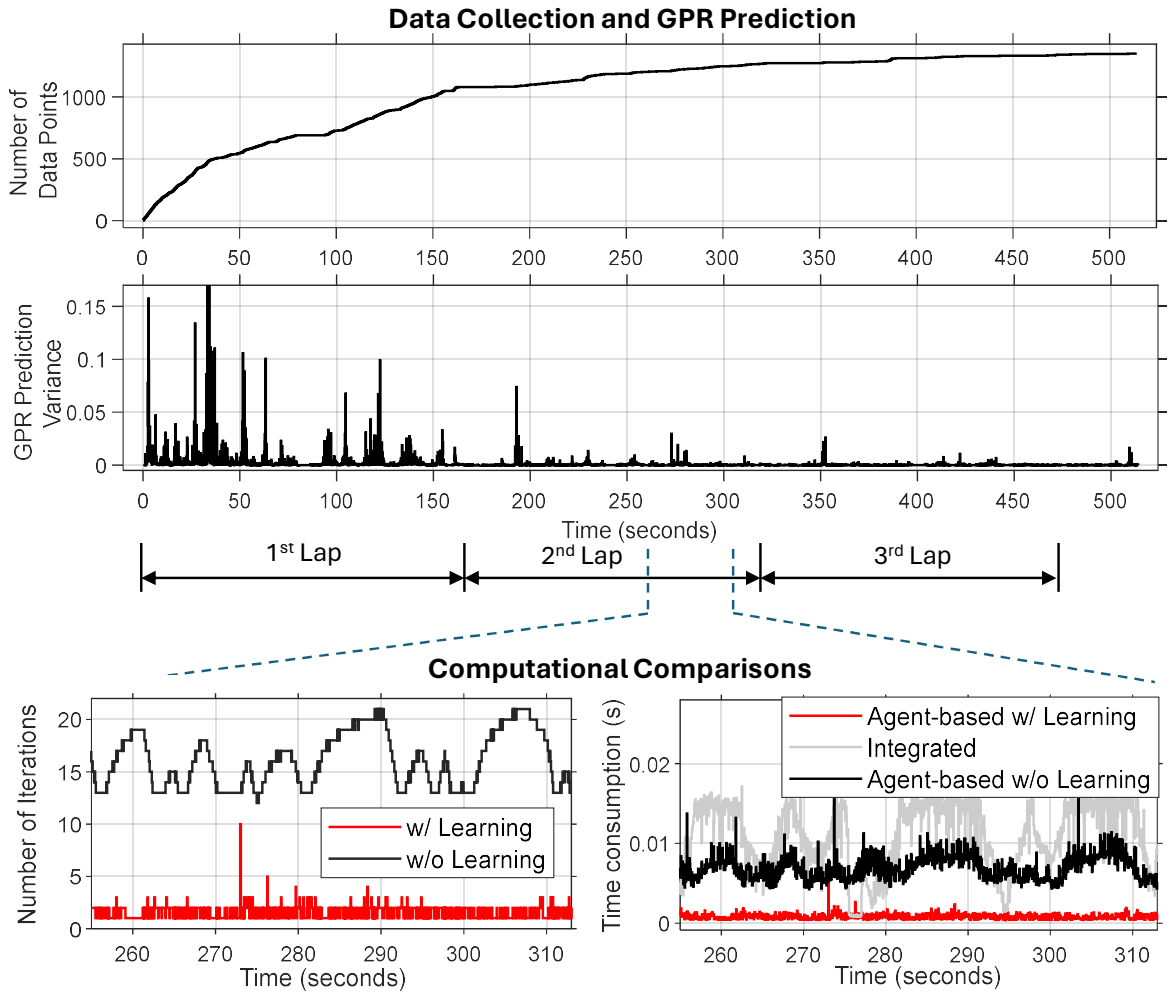


Figure 6.7: Data collection and learning process. A comparison shows the improvement in computational efficiency.

the control actions $\bar{U}_j$ calculated in the first iteration are already very close to the global optimum, significantly reducing the convergence difficulty.

6.4.3 Limited Computational Resource

Improvements in computing efficiency can help save computing resource costs. Thus, an interesting assumption is that the maximum number of iterations in a controller is limited

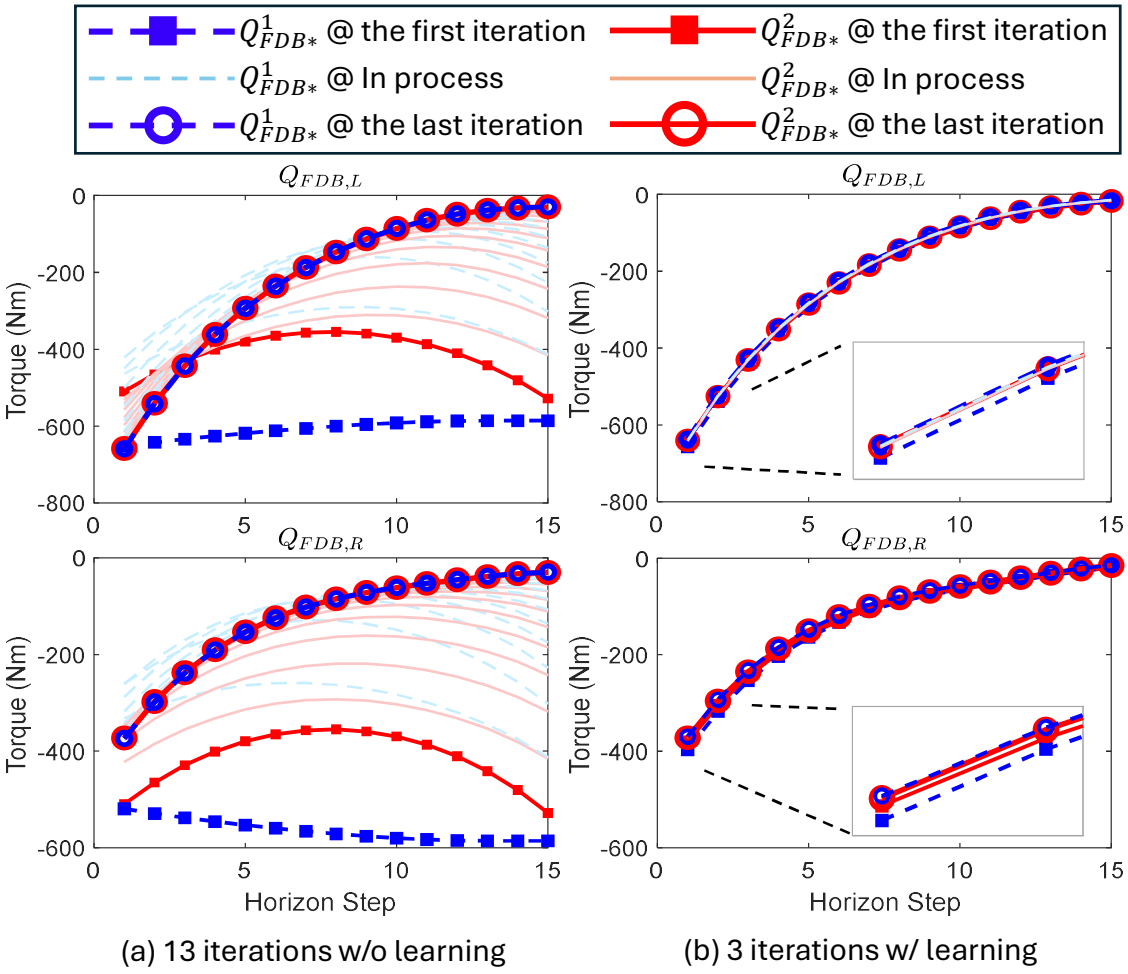


Figure 6.8: Detailed converging processes of the shared FDB agent at the station $\approx 1900m$ on the racing track, which marked as a red dot in Figure 6.5.

to only 5. Moreover, priorities in the objective are set: lateral tracking is preferred when the distributed control scheme cannot converge in the limited number of iterations. The comparison simulations of three laps with the same setting as the simulation in Section 6.4.2 are conducted, and the results are shown in Figure 6.9.

Since insufficient data was in the first lap, the proposed distributed scheme could not converge within five iterations, so lateral tracking was prioritized. The longitudinal velocity tracking error in the first lap is significantly higher than the last two. The vehicle can neither decelerate in time when entering the corner nor keep up with the acceleration out

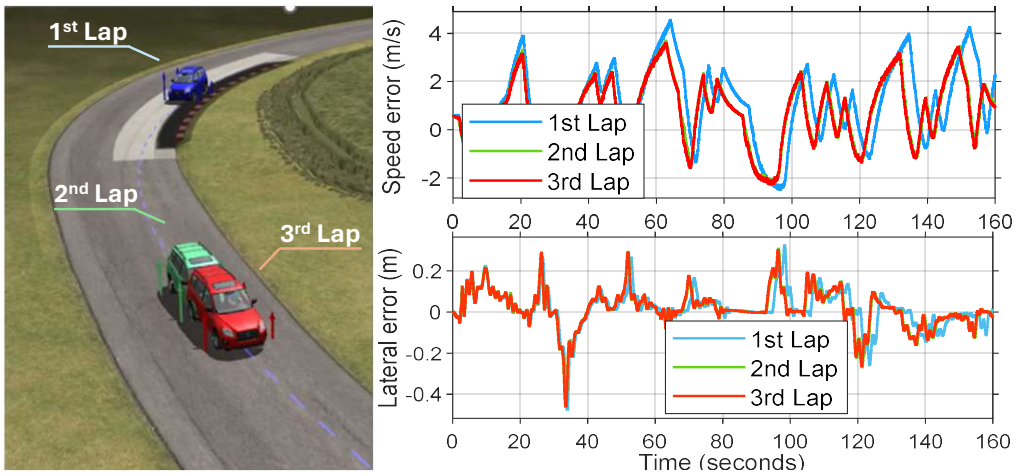


Figure 6.9: Comparison results of three laps under limited maximum iterations. The speed tracking was improved in the last two laps.

of the corner. By overlapping the simulation animations together, it's shown that the vehicle in the first lap is much slower than the last two laps.

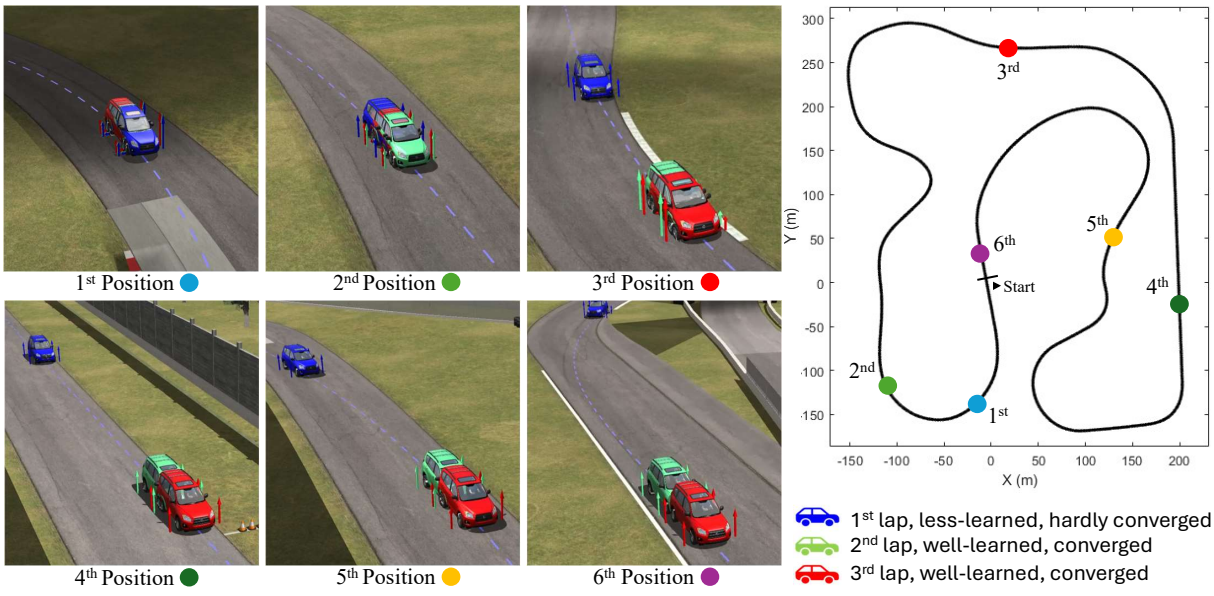


Figure 6.10: Six slices of the comparison results under limited maximum iterations.

Furthermore, six slices of the comparison during a lap are shown in Figure 6.10. In the first deceleration into the turn (first position), the car in the first lap (blue) will not decelerate enough, so its position will be ahead. But when exiting the turn (second position), the car in the first lap (blue) will fall behind due to insufficient acceleration. In the subsequent process, the vehicle in the first lap (blue) gradually falls behind, and the speed tracking error becomes larger. The vehicle in the third lap (red) will have the best speed tracking performance, gradually take the lead, and gradually pull away from the vehicle in the first lap (blue) together with the vehicle in the second lap (green).

6.5 Summary

This paper proposes a data-driven distributed control scheme - the L-MO-AMPC- upgraded based on MO-AMPC and can decouple the integrated system into an equivalent but more flexible distributed form. GPR was leveraged to learn the negotiations between objectives for faster convergence. Simulations on a path-tracking task demonstrate its convergence and high computing efficiency.

The main contributions of this section are as follows:

1) Based on MO-AMPC, the negotiations between objectives are analyzed. The inputs and outputs of the prediction data point have been mathematically selected.

2) GPR learning method is developed to predict the contributions for each objective and initialize the iterations to accelerate the convergence process.

3) The proposed scheme is applied to a path-tracking task. It achieves the same control performance as the integrated one while reducing the computational time by 43.5%. Furthermore, the learning-based method saves 88.6% more computing time than without learning.

The proposed scheme is not confined to path-tracking but is promising in many applications as a general data-driven distributed control scheme. It is especially ideal for physically distributed systems with available information exchange, such as interaction-aware planning, multi-vehicle avoidance control, and fleet management. In the future, the proposed method will be expanded to include those interesting topics. Real-time efficiency and convergence performance for systems with a larger number of agents and with stronger non-linearity will be further explored.

Chapter 7

Experimental Study

This chapter presents the experimental Studies of the proposed schemes. First, the experiment results of yaw stability control using the proposed L-AMPC is demonstrated in Section 7.2. Next, the experiment results of cruise control using the proposed MO-AMPC is presented in Section 7.3. Finally, the results of holistic stability control using the proposed MO-AMPC is presented in Section 7.4.

7.1 Experiment Setup

Real-time experiments were conducted on an electric vehicle with four independent-drive electric motors, as shown in Figure 7.1, whose main parameters are specified in Appendix C. In addition to driver-controlled drive, braking and steering, this test vehicle also enables independent differential torque control for the front and rear axles, as torque can be individually controlled for all four wheels, as well as an additional active front steering input. At the same time, this vehicle also provides a wealth of sensor interfaces, including global navigation satellite system (GNSS), inertial measurement unit (IMU), wheel speed, etc., to facilitate the provision of more accurate reference speed estimation and other key signals through sensor fusion.

This vehicle was equipped with dSPACE to support rapid control prototype testing with Matlab/Simulink. The controllers were running in real-time at 20Hz $\sim$ 50Hz on a laptop with an i7-12700H CPU and 16 GB RAM. The controller ran in the Matlab Real-time Desktop model and sent the control command to motor drivers through the CAN bus at every time step.

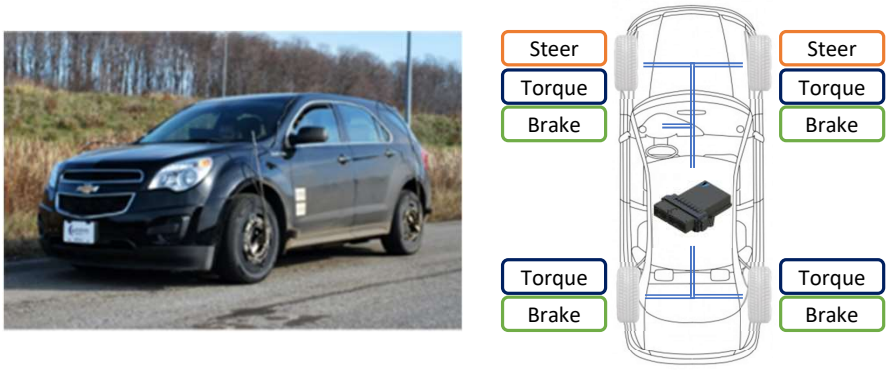


Figure 7.1: Test vehicle with independent wheel control actuators. Distributed controllers, such as torque-vectoring and differential braking, are coordinated as connected agents to achieve multiple objectives.

7.2 Yaw Stability Control using L-AMPC

The agent configuration and the L-AMPC controller design introduced in Section 4.4 were used in the experimental tests. The experimental tests focused on the well-learned scenario as the simulation demonstrated in Section 4.5.2. The detailed experimental configuration is shown in Table 7.1.

Table 7.1: Experimental Configurations using L-AMPC in Well-Learned Scenarios

Scenario		Agent configuration	Driving Maneuver	Vehicle Speed	Road Friction
Black-Box Agents	Test 1	Controllable: RDT White-box: STR, DT Black-box: FDT	Sinewaves	40 kph	Dry Asphalt
	Test 2	Controllable: RDT White-box: STR, DT Black-box: FDT	DLCs	60 kph	Dry Asphalt

In addition to the above two tests, the comparison between the proposed multi-step GPR prediction and the traditional single-step GPR prediction is also presented.

7.2.1 Test 1: SineWaves

Several consecutive sinewave maneuvers were implemented. The learning process is shown in Figure 7.2.

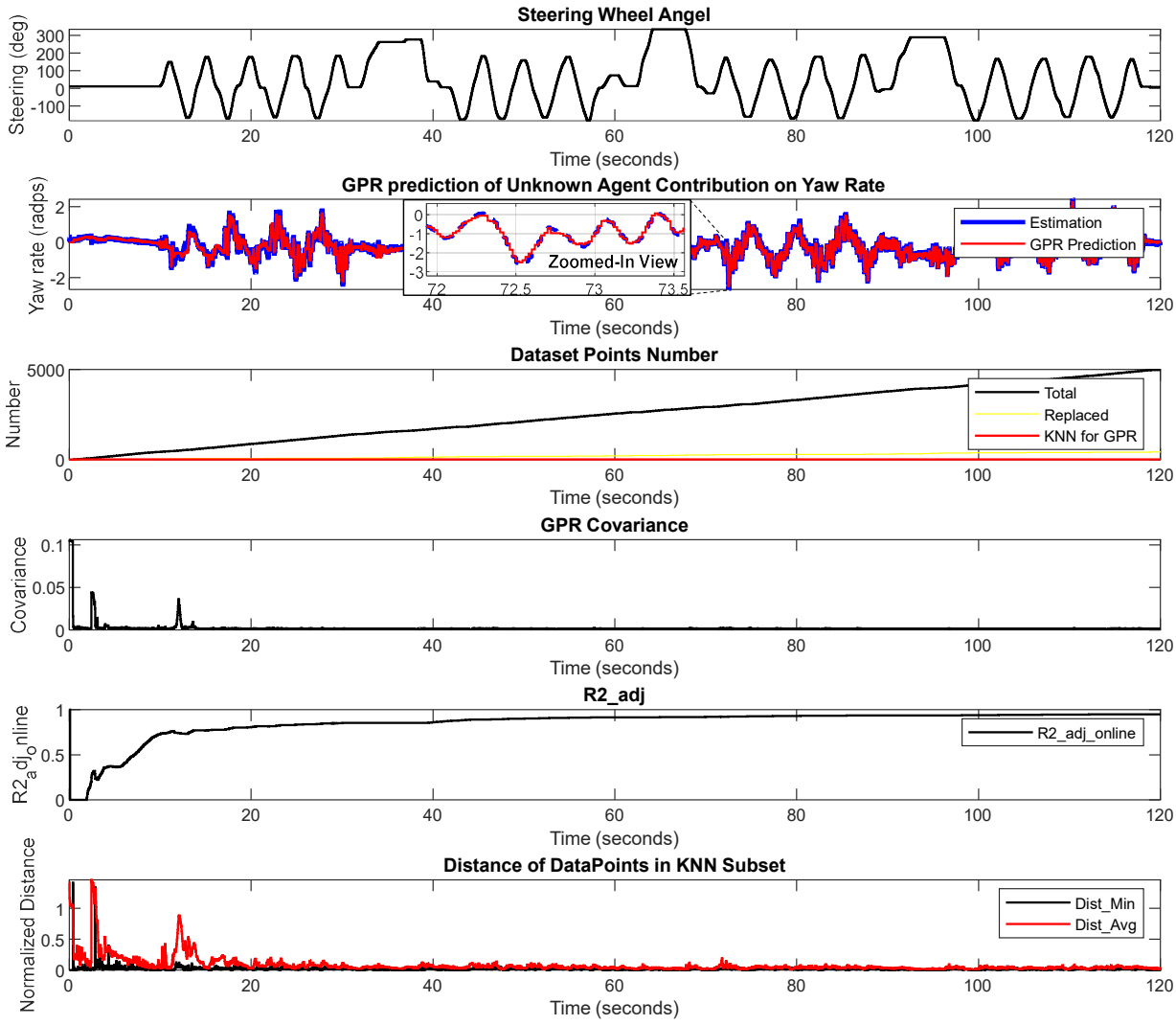
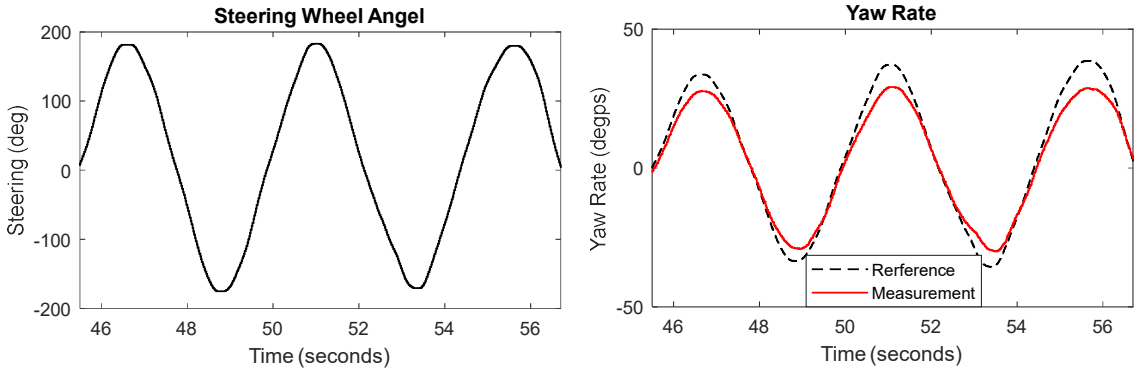


Figure 7.2: Experiment of L-AMPC: learning process of yaw stability control in Sinewaves.

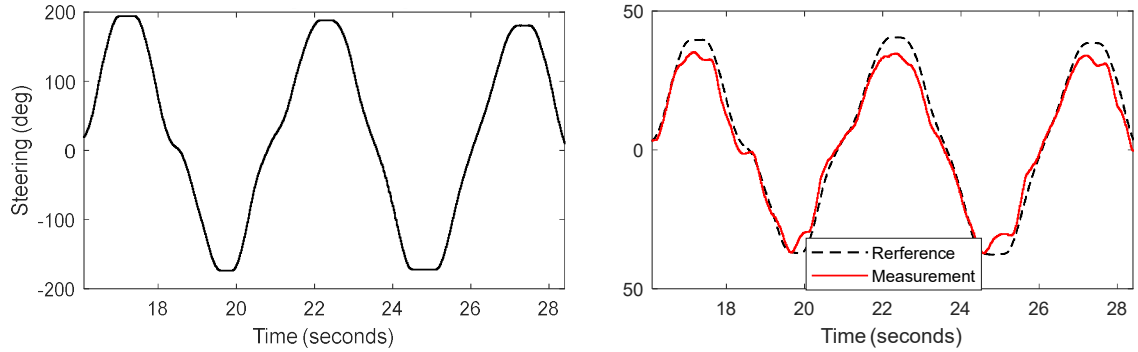
The proposed learning scheme can accurately predict the contribution of the unknown agents. As the number of collected data kept growing, the shortest and average distance to the test point decreased in the 20-point subset selected by KNN. Therefore, the predictions

continually became more accurate, which could be proved by decreasing the prediction variance and R^2_{adj} criterion.

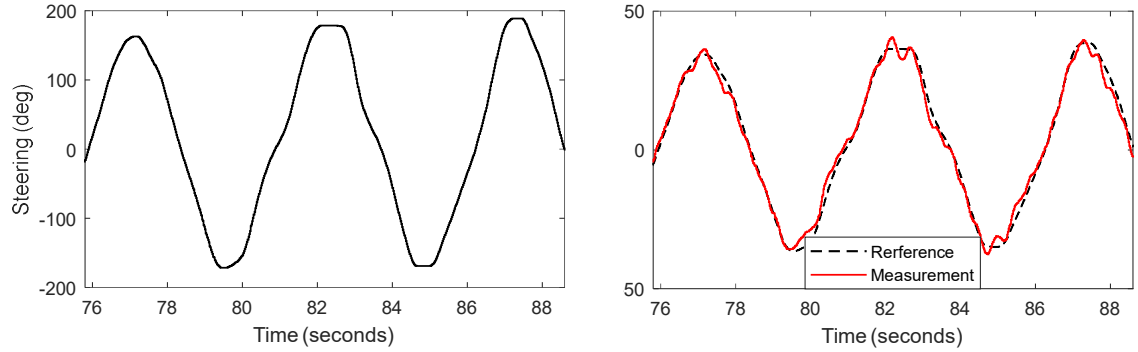
Three similar sinewave maneuvers with different settings were also implemented for comparison, as shown in Figure 7.3:



(a) Only the driver (STR & DT) is controlling the vehicle



(b) All agents are working, RDT is controlled by **AMPC**



(c) All agents are working, RDT is controlled by **L-AMPC**

Figure 7.3: Experiment of L-AMPC: Comparison of yaw stability control in Sinewaves.

(a) RDT and FDT agents were turned off, and the vehicle was only controlled by the DT and STR agents.

(b) All agents are activated where FDT was the black-box agent, and only a traditional AMPC controller was working on the RDT agent.

(c) All agents are activated where FDT was the black-box agent, and the proposed L-AMPC with learning function is working on the RDT agent.

It is shown that when all agents are activated, and the traditional AMPC controller controls the controllable RDT agent, the tracking performance is improved compared to the first experiment, where only the driver controls the vehicle. However, without the information of the black-box agent, there is still a significant tracking error. In the last experiment, since the proposed L-AMPC controller can learn the contribution of the black-box FDT agent from data, the measured yaw rate is always closer to the reference.

Intuitively, it is shown in Figure 7.4 that when the proposed L-AMPC controls the RDT agent, the yaw rate error is significantly reduced, meaning the yaw rate tracking performance is improved.

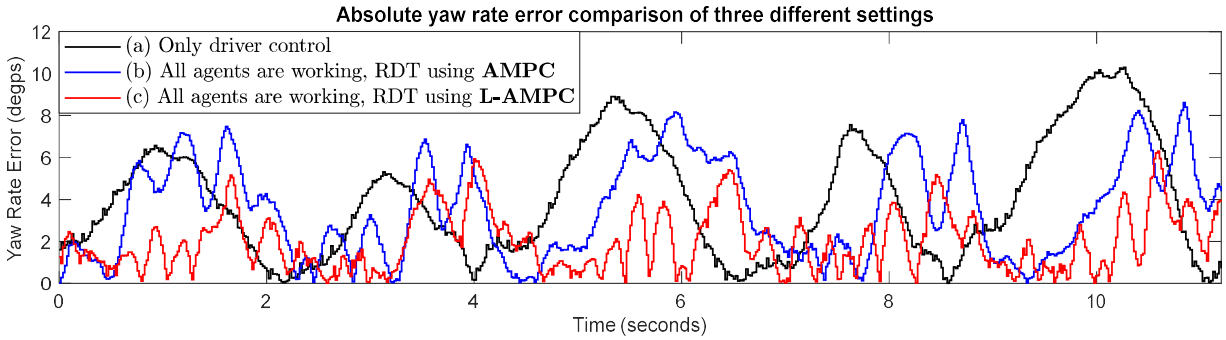


Figure 7.4: Experiment of L-AMPC: MSE of yaw stability control in Sinewaves.

Quantitatively, the comparison of the mean squared error (MSE) of desired and actual yaw rates is also shown in Table 7.2. The MSE of yaw rate reduced from 18.2 deg/s^2 to 5.9 deg/s^2 when the LAMPC controls the RDT agent.

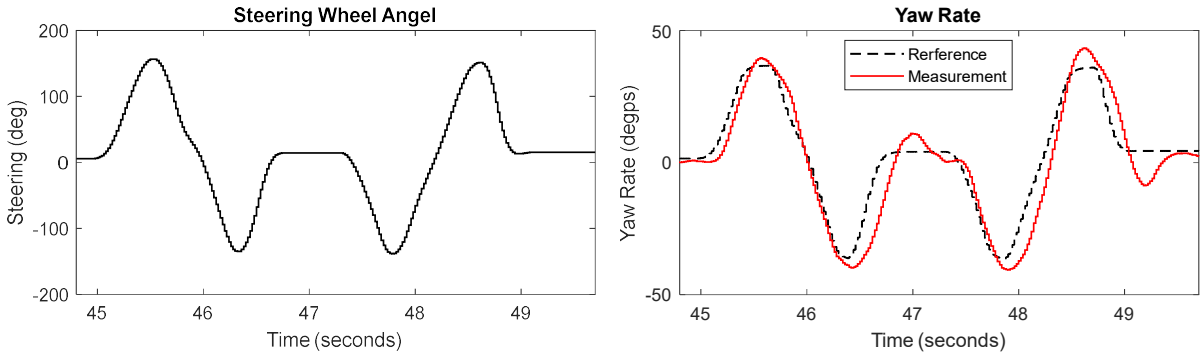
7.2.2 Test 2: DLCs

Several DLC maneuvers are implemented in succession to demonstrate the improvement of L-AMPC better. experimental settings are the same as Test 1. Two settings (b) and

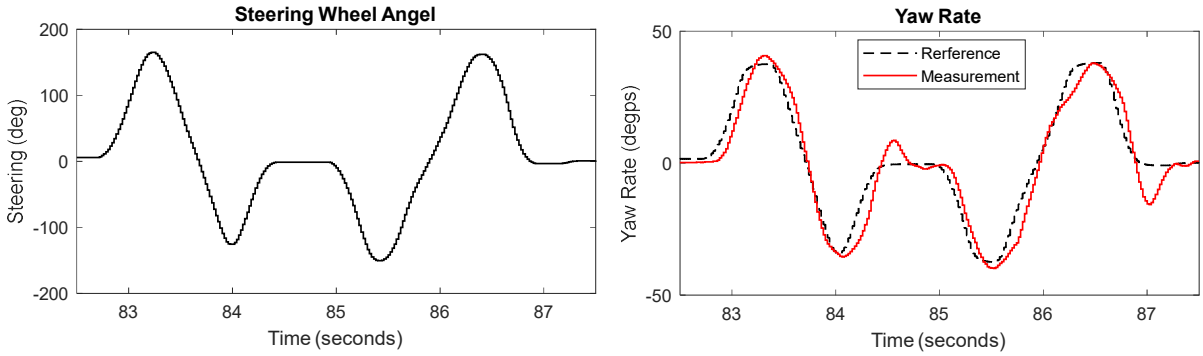
Table 7.2: MSE improvement of yaw stability control in Sinewaves.

Test Settings	MSE (deg/s^2)	Perception of Improvement
(a) Only the driver (STR & DT) is controlling the vehicle	23.6	-
(b) All agents are working, RDT is controlled by AMPC	18.2	22.8 %
(c) All agents are working, RDT is controlled by L-AMPC	5.9	74.8 %

(c) were implemented for comparison, as shown in Figure 7.5.



(b) All agents are working, RDT controlled by **AMPC**



(c) All agents are working, RDT controlled by **L-AMPC**

Figure 7.5: Experiment of L-AMPC: Comparison of yaw stability control in DLCs.

Intuitively, it is shown that when the RDT agent is controlled by the proposed L-AMPC, the yaw rate error is reduced, meaning the yaw rate tracking performance is improved. The comparison of the absolute value of the synchronized yaw rate error is shown in Figure 7.6.

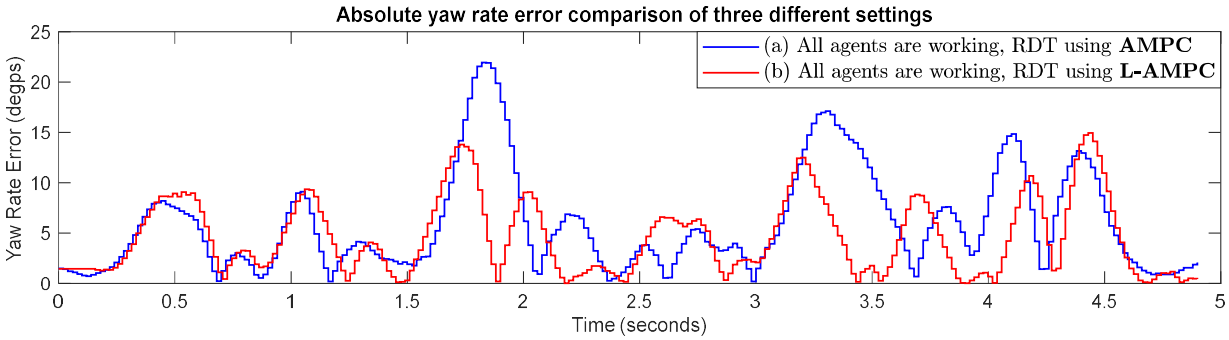


Figure 7.6: Experiment of L-AMPC: MSE of yaw stability control in Sinewaves.

Quantitatively, the comparison of yaw rate MSE is shown in Table 7.3.

Table 7.3: MSE improvement of yaw stability control in DLCs.

Test Settings	MSE (deg/s^2)	Perception of Improvement
(b) All agents are working, RDT is controlled by AMPC	60.4	-
(c) All agents are working, RDT is controlled by L-AMPC	34.9	42.2 %

7.2.3 Multi-step GPR Prediction

All the results above are based on the multi-step GPR method proposed in Section 4.2.3. The L-AMPC controller runs in Matlab/Simulink on an i7 CPU/16GB RAM laptop in real-time at 50Hz, proving that the proposed multi-step GPR has high computational efficiency and can support real-time implementation.

In order to verify the accuracy of multi-step GPR, the experimental results of different methods for horizon prediction in Sinewave, DLC and another scenario with a fast-changing yaw rate are shown in Figure 7.7 (a), (b), and (c), respectively. In each scenario, three prediction methods are compared:

(a) Single-step GPR with a constant assumption.

(b) Single-step GPR with a linear assumption.

(c) Multi-step GPR.

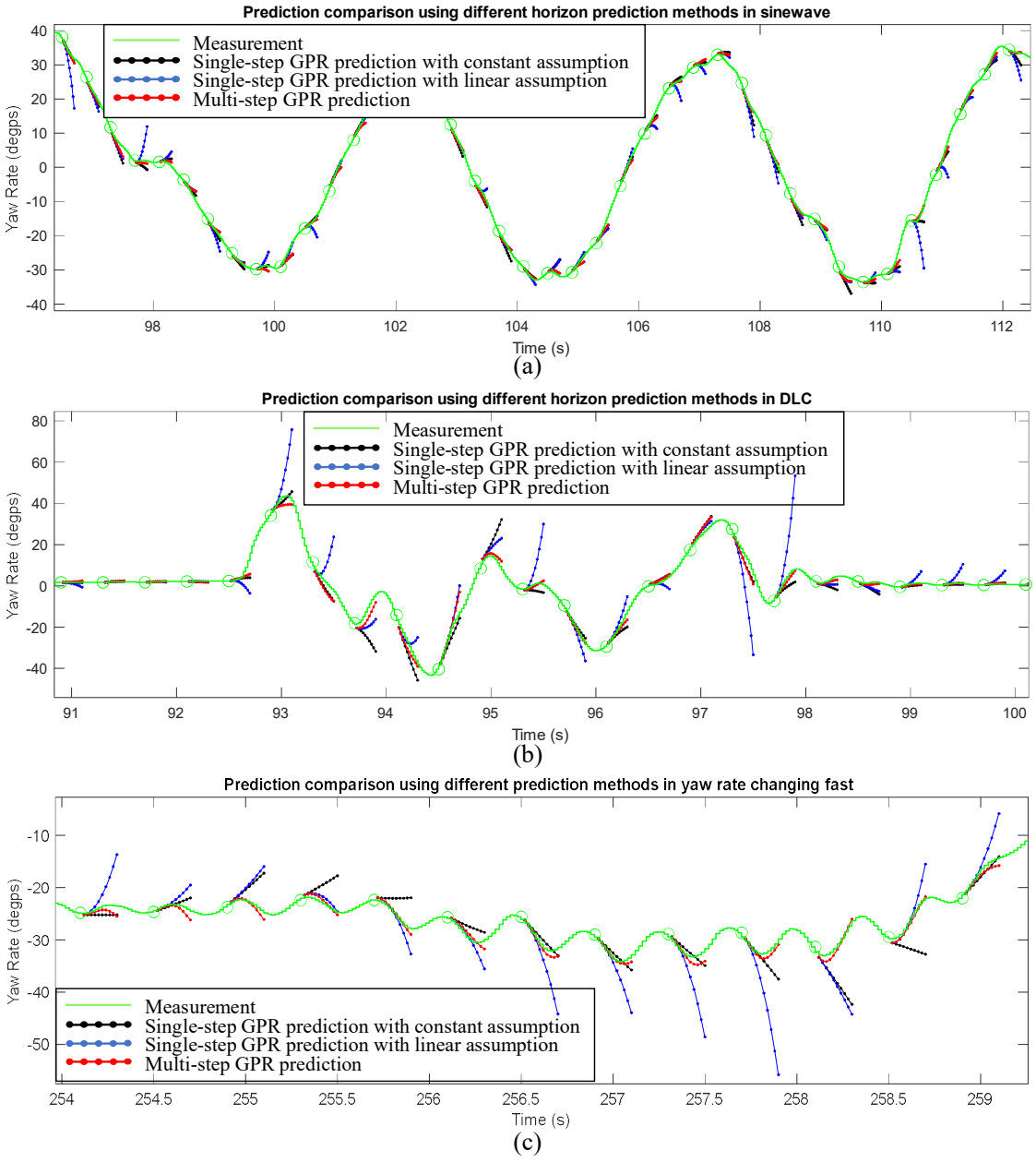


Figure 7.7: Experiment of L-AMPC: Multi-step GPR predictions.

It is evident from the results that the predicted yaw rate using multi-step GPR will be significantly closer to the ground truth, especially when the yaw rate is changing fast.

7.3 Cruise Control using MO-AMPC

The agent configuration and the MO-AMPC controller design introduced in Section 5.4 were used in the experimental tests. The experimental tests focused on the efficacy of multiple objectives, the convergence of the MO-AMPC scheme, and the computational efficiency. The detailed experimental configuration is shown in Table 7.4.

Table 7.4: Experimental Configurations using MO-AMPC in Cruise Control

Scenario	Driving Maneuver	Vehicle Speed	Road Friction
Test 1 Aggressive Lane Change	DLC in Acceleration -> SLC in Aeceleration	0 ->60 ->0 kph	Dry Asphalt
Test 2 Normal Cruise Driving	Circle Cruise	20 ∼40 kph	Dry Asphalt

The DLC in aggressive lane change simulates lane change and overtaking during cruising in real driving scenarios, while the single lane change (SLC) simulates emergency obstacle avoidance during cruising in real driving scenarios.

7.3.1 Test 1: Aggressive Lane Change

Two MAMO control system configurations were tested under the same experimental conditions. As shown in Figure 7.8.

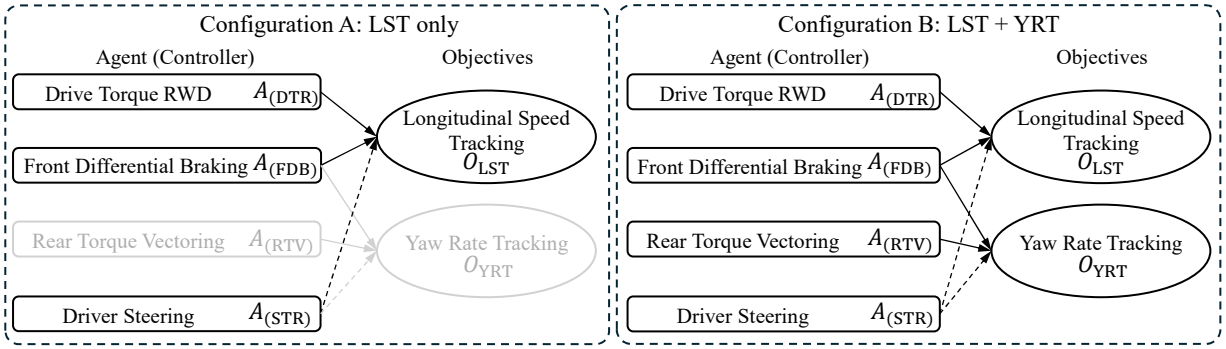


Figure 7.8: Experiment of MO-AMPC: MAMO configuration for test of cruise control.

Only the objective of LST and its agent group were enabled in the first configuration; both objectives and all the agents were enabled in the second configuration. hose two tests

had the same desired speed profile and similar steering maneuvers. Two comparisons were conducted for this example:

(1) Comparison between partially disabling one objective and enabling all the objectives to verify the effectiveness of the proposed MO-AMPC.

(2) Comparison between the proposed MO-AMPC and integrated MPC to demonstrate convergence and improved computational efficiency.

The results of the vehicle response are shown in Figure 7.9. The acceleration and deceleration are smooth when tracking the target speed in both tests. The error between the target and measured yaw rates is significantly reduced when the objective of YRT is enabled, indicating the improved vehicle's yaw stability.

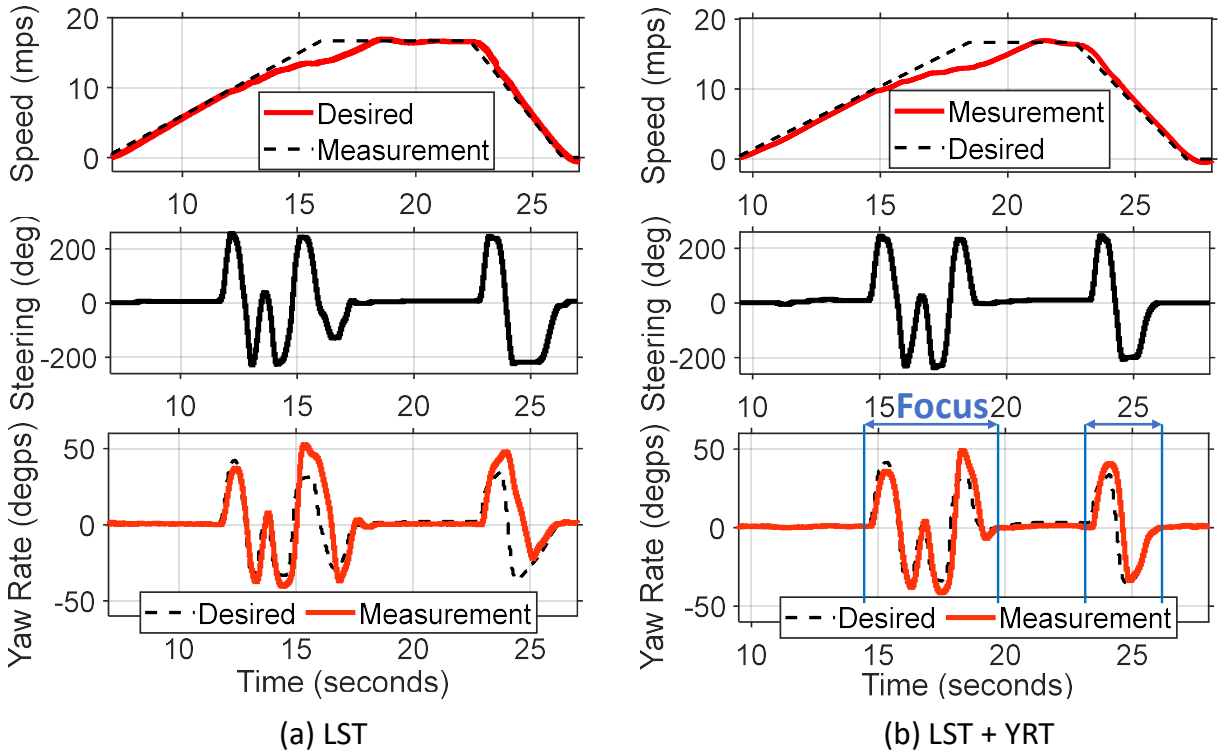


Figure 7.9: Experiment of MO-AMPC: Comparison of the vehicle response for the aggressive lane change in the example of cruise control. The over-steering was significantly suppressed when the objective of YRT was enabled. The yaw stability was improved.

The results of torque commands on each wheel are shown in Figure 7.10, proving that the proposed MO-AMPC can successfully collaborate with multiple controllers to achieve multiple objectives simultaneously.

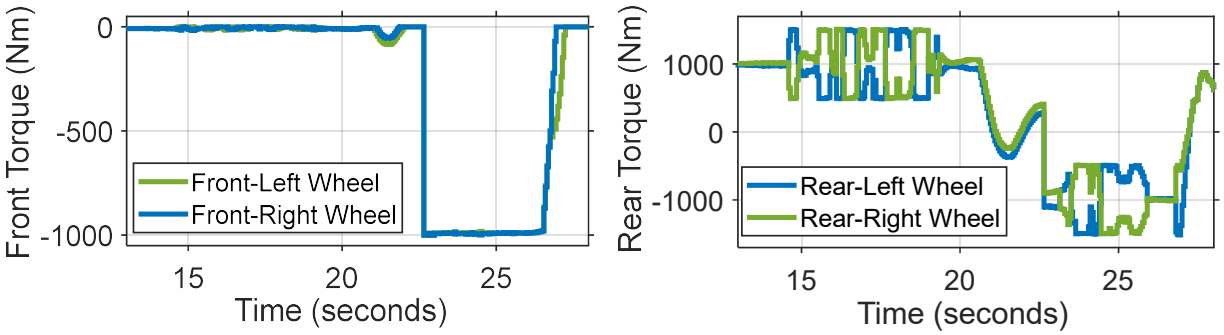


Figure 7.10: Experiment of MO-AMPC: Wheel torque commands from the MO-AMPC during the aggressive lane change in cruise control. The *Local Explicit* formulation with optimal parameters was adopted. All control agents have cooperated: DTR generated the driving and braking torque on the rear wheels; RTV generated differential torque on the rear wheels; FDB generated independent braking torques on the front wheels.

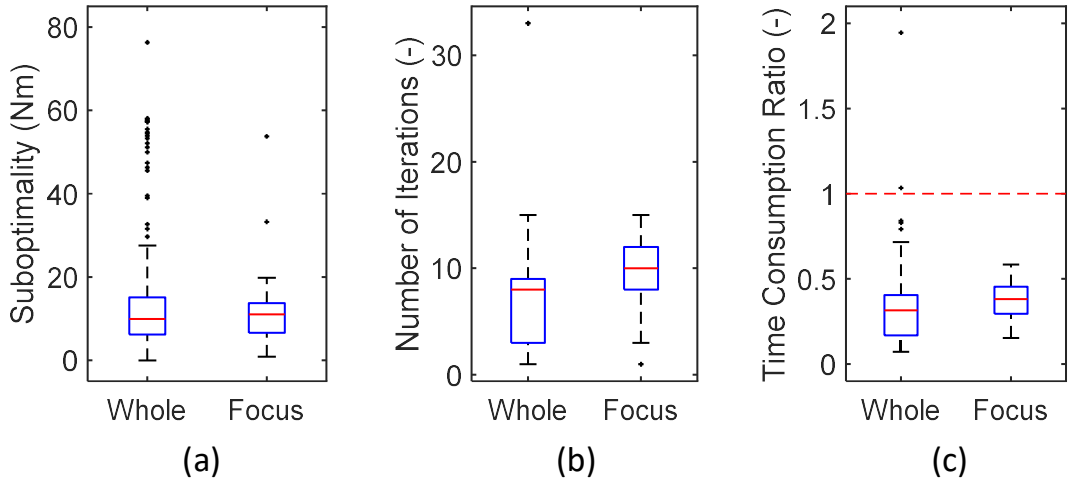


Figure 7.11: Experiment of MO-AMPC: Results of the convergence and computational efficiency during the aggressive lane change in cruise control: (a) torque suboptimality (b) number of iterations (c) time consumption ratio. "Whole": the whole process; "Focus": the period marked in Figure 7.9 where both objectives require significant control.

The suboptimality, iterations, and time consumption ratio results are shown in Figure 7.11. The majority of the torque suboptimality was always less than 20 Nm. The number of iterations required less than 15. The time consumption ratio was significantly less than 1. Besides, it is easy to notice that a few more iterations are needed during the "focus" period to achieve convergence. Accordingly, the time consumption ratio is slightly higher but less than 1. The results demonstrate that even for such a small system, the proposed MO-AMPC in *Local Explicit* formulation with the optimal parameters is computationally efficient and outperforms the integrated MPC while always converging to the global optimum.

7.3.2 Test 2: Normal Cruise Driving

The normal cruise driving was conducted on a circle road in a testing proving ground, as shown in Figure 7.12. The configuration of the MAMO control system was the same as in Section 5.4.1. Both objectives and all the agents were enabled in this test.



Figure 7.12: Experiment of MO-AMPC: Testing ground of the cruise control in normal cruise driving.

The results of the vehicle response and the performance of the MO-AMPC controller are shown in Figure 7.13. The longitudinal speed and yaw rate are tracked very well. In most cases, the proposed MO-AMPC needs less than five iterations to converge, and

the required computational time consumption is significantly lower than that of integrated MPC. When the speed changes suddenly, the MO-AMPC will require more iterations, but the time consumption will not exceed that of integrated MPC in most cases.

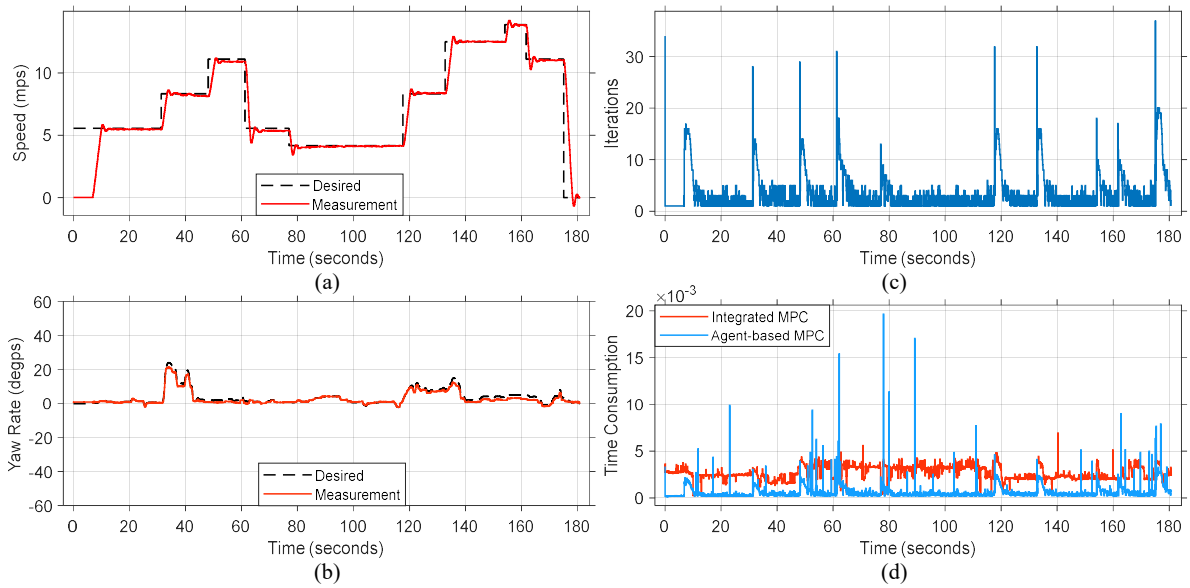


Figure 7.13: Experiment of MO-AMPC: Results of the cruise control in normal cruise driving. (a) speed tracking results. (b) yaw rate tracking results. (c) number of iterations. (d) comparison of time consumption between the MO-AMPC and the integrated MPC.

7.4 Holistic Stability Control using MO-AMPC

The agent configuration and the MO-AMPC controller design introduced in Section 5.5 were used in the experimental tests. The experimental tests focused on the efficacy of multiple objectives, the convergence of the MO-AMPC scheme, and the computational efficiency. The detailed experimental configuration is shown in Table 7.5.

Two comparisons were conducted for this example:

(1) Comparison between partially disabling one objective and enabling all the objectives to verify the effectiveness of the proposed MO-AMPC.

(2) Comparison between the proposed MO-AMPC and integrated MPC to demonstrate convergence and improved computational efficiency.

Table 7.5: Experimental Configurations using MO-AMPC in Holistic Stability Control

Scenario	Driving Maneuver	Vehicle Speed	Road Friction
Test 1 Straight Launch and Brake	Full Pedal Launch ->Full Pedal Brake	0 ->45 ->0 kph	Sandy Road
Test 2 DLC during Acceleration	DLC during Full Pedal Acceleration	0 ->70 kph	Dry Asphalt

Therefore, in each test, two MAMO control system configurations were tested under the same experimental conditions. As shown in Figure 7.14.

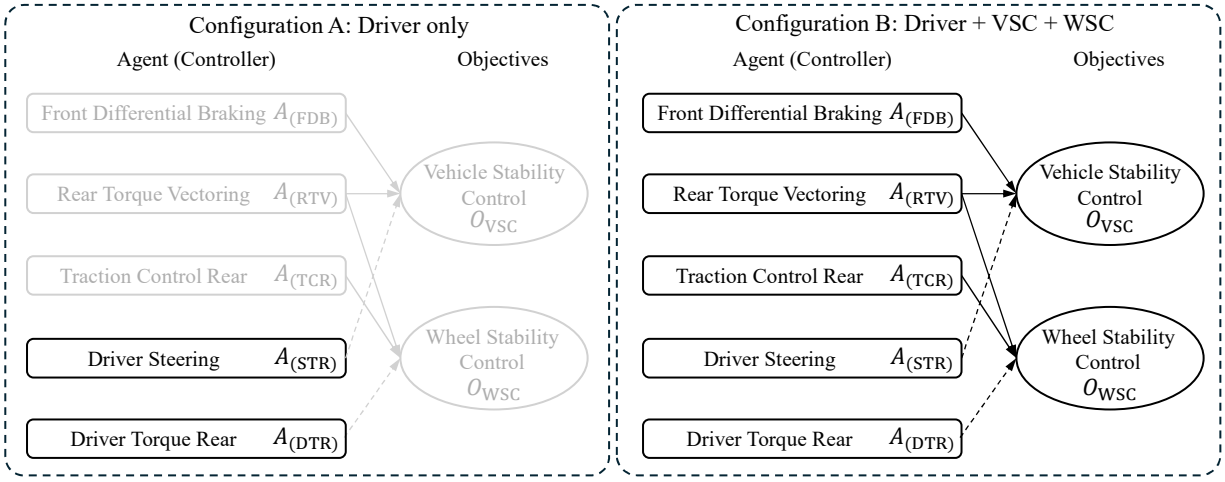


Figure 7.14: Experiment of MO-AMPC: MAMO configuration for test of holistic vehicle stability control.

7.4.1 Test 1: Straight Launch and Brake

Aggressive accelerating or decelerating on a low-adhesion road surface can easily cause severe wheel slippage, while slight steering wheel operation can easily lead to lateral instability. Therefore, this is challenging driving behaviour that requires considering both wheel stability and body yaw stability. In this test, the vehicle was full-pedal straight launched and then full-pedal braked on a low-friction sandy road surface. The compared results of vehicle response are shown in Figure 7.15.

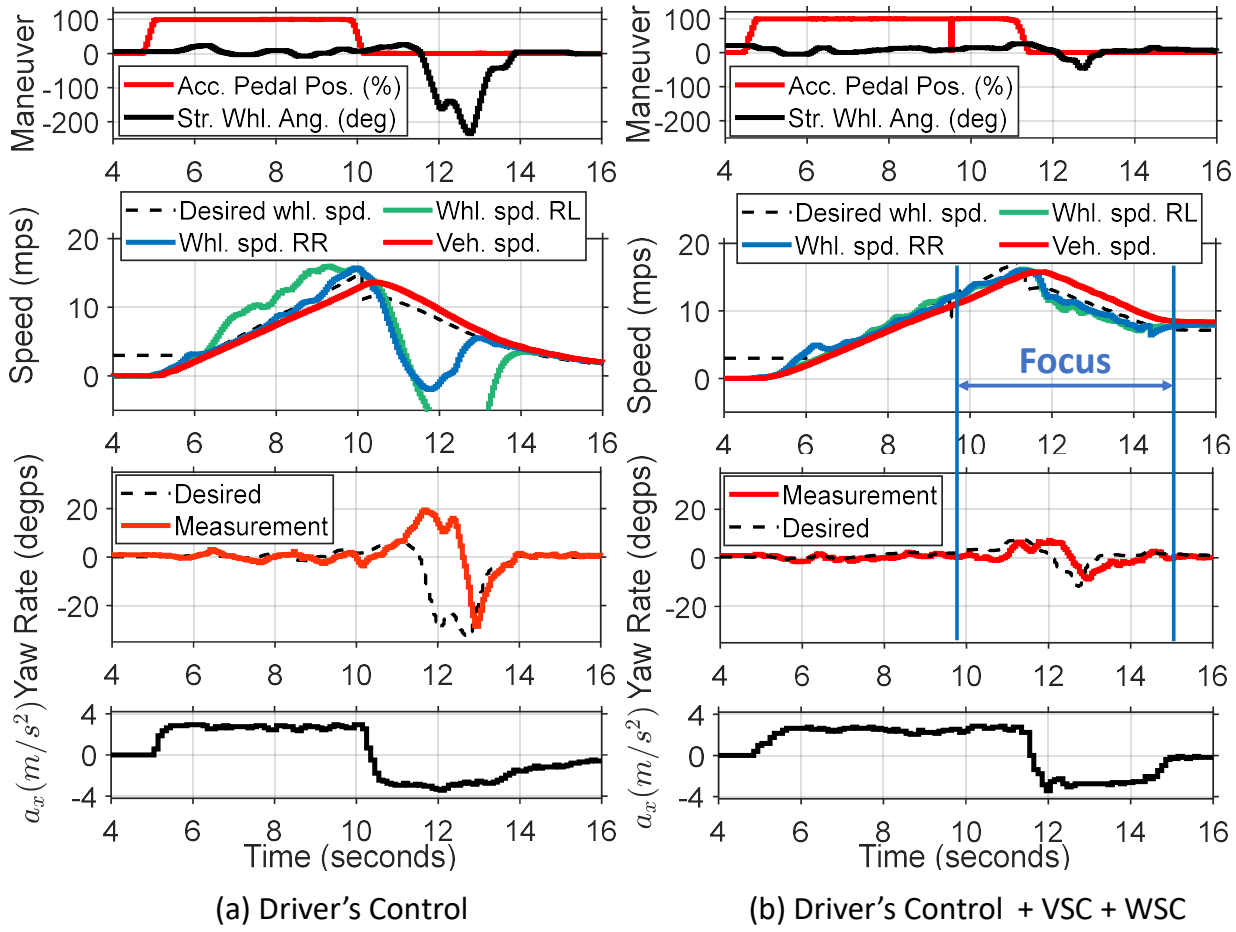


Figure 7.15: Experiment of MO-AMPC: Comparison of the vehicle response for straight launch and braking in the example of holistic stability control. The over-steering was significantly suppressed, and no obvious wheel slip occurred during the second test. The holistic stability was improved when both VSC and WSC were enabled.

Due to the low friction, severe wheel slip and yaw rate errors were observed in test 1-A (left plots) when only the driver was controlling the vehicle. Especially during deceleration, due to the rear-wheel drive configuration of the vehicle, the violent locking of the rear wheels caused the lateral force margin of the rear tires to be extremely reduced, which eventually led to severe over-steering of the vehicle. The driver tried his best to steer in the opposite direction during this process, which was manifested as a significantly opposite desired yaw rate and then stabilized the vehicle to prevent danger. These vehicle performances and

driver's behaviour show the danger and challenge of this situation. The driver's aggressive steering also shows that a large effort was required to keep the vehicle straight. However, when both stability control objectives were enabled in test 1-B (right plots), wheel speeds were nicely controlled at desired values during the launching and braking processes. Also, both the yaw rate absolute value and the error were significantly reduced, meaning the driver can easily control the vehicle to go straight without much effort. Besides, the maximum acceleration and deceleration of the vehicle are basically the same as those in test 1-A, which means that the control process does not improve stability at the expense of vehicle power but significantly improves stability while keeping the power of the vehicle. Therefore, the overall stability of the vehicle has been significantly improved, and vehicle stability control does not lose longitudinal acceleration or deceleration.

The torques on four wheels in test 1-B are shown in Figure 7.16.

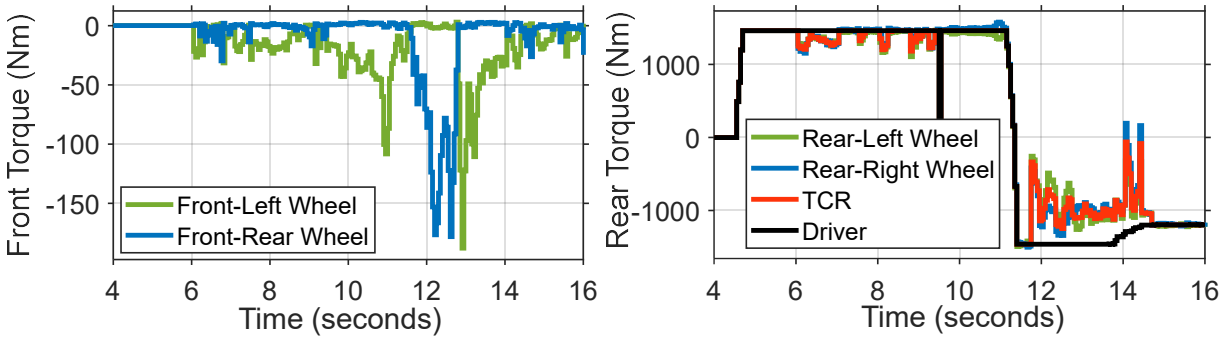


Figure 7.16: Experiment of MO-AMPC: Torque commands from the MO-AMPC using the *Local Explicit* formulation with optimal parameters for straight launch and braking in the example of holistic stability control. All control agents cooperated: FDB generated independent braking torques on the front wheels; the drive's torque was first reduced by TCR and then adjusted by RTV on the rear wheels.

As shown in Figure 7.16, the FDB agent generated independent braking torques on the front-right and front-left wheels, generating additional yaw moment to help improve the yaw stability. The TCR agent first reduced the overall drive's torque to depress the wheel slip and then adjusted by the RTV agent for both objectives. It is worth noting that in the controller, the penalty weights for control actions of RTV and FDB are the same, but the torque generated by FDB is significantly greater than that of RTV. This is because RTV is a shared agent for two goals. If RTV generates a larger torque to help improve yaw stability, it will inevitably sacrifice wheel stability. Therefore, the optimal agent collaboration is to use FDB to help improve yaw stability as much as possible, which is consistent with

the global optimal control, as the comparison of the control action between the proposed MO-AMPC and the integrated MPC shown in Figure 7.17.

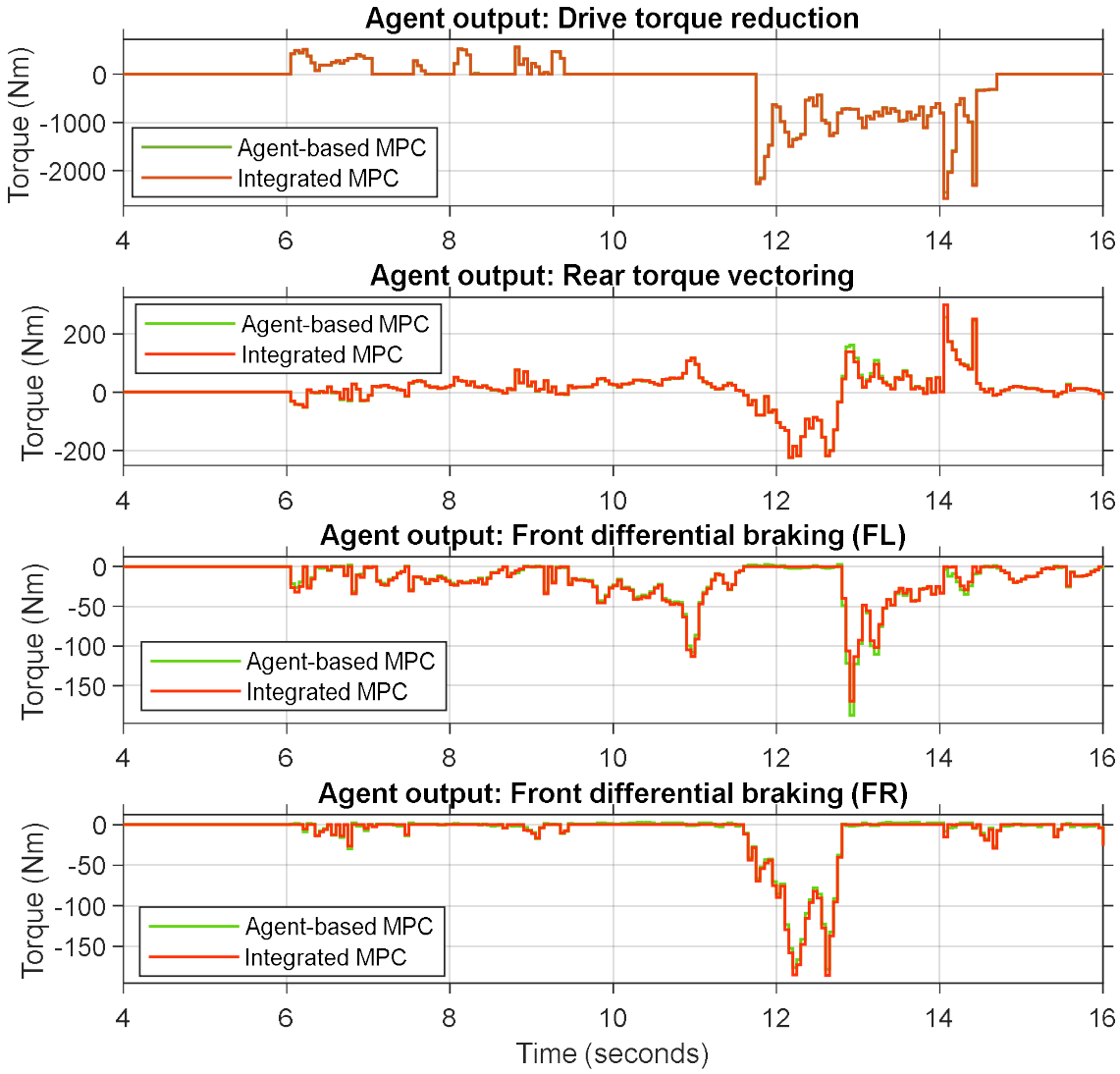


Figure 7.17: Experiment of MO-AMPC: comparison of the control action between the proposed MO-AMPC and the integrated MPC for straight launch and braking in the example of holistic stability control. Red lines: control action from the integrated MPC; green lines: control action from the proposed MO-AMPC.

As shown in Figure 7.17, the control action from the MO-AMPC controller is very close

to that of the integrated MPC, which verifies the convergence of the MO-AMPC scheme.

The suboptimality, iterations, and time consumption ratio results of the proposed MO-AMPC scheme are shown in Figure 7.18.

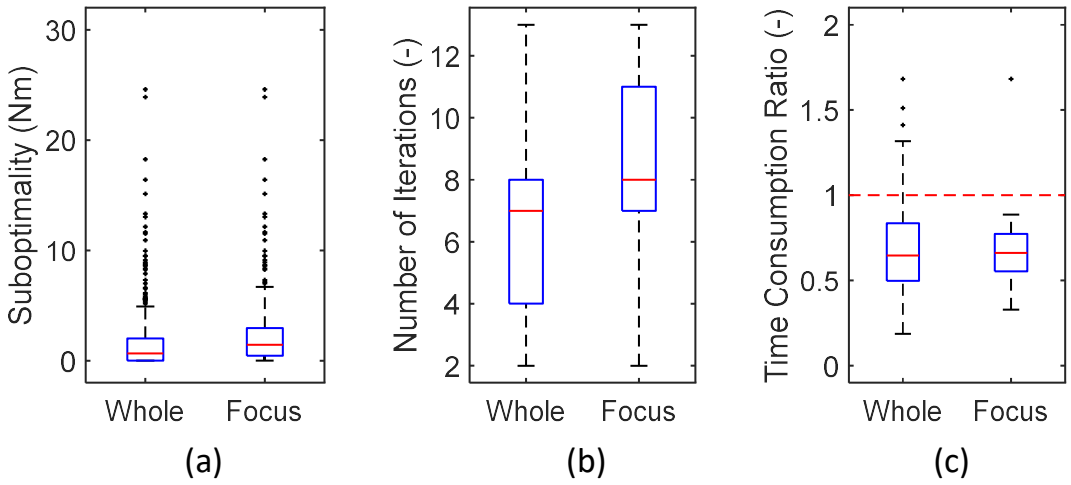


Figure 7.18: Experiment of MO-AMPC: Results of the convergence and computational efficiency for straight launch and braking in the example of holistic stability control. "Focus" represents the specific period marked in Figure 7.15 where both objectives require significant control.

As shown in Figure 7.18, the majority of the torque suboptimality was less than 5 Nm, which means the control actions from the MO-AMPC were very close to the integrated MPC and their error can be neglected in practice. The number of iterations required was less than 15. The time consumption ratio was less than 1. A few more iterations are needed to achieve convergence during the "focus" period, so the time consumption ratio is slightly higher but still less than 1.

7.4.2 Test 2: Double Lane Change in Acceleration

Changing lanes during a launch or accelerating to overtake are also common driving behaviours. In these cases, there is also a risk of wheels slipping and vehicle instability, even on high-friction road surfaces. Therefore, this is also challenging driving behaviour that requires considering both wheel stability and body yaw stability. In this test, a DLC was conducted during the acceleration on high-friction road surfaces. The compared vehicle performance is shown in Figure 7.19.

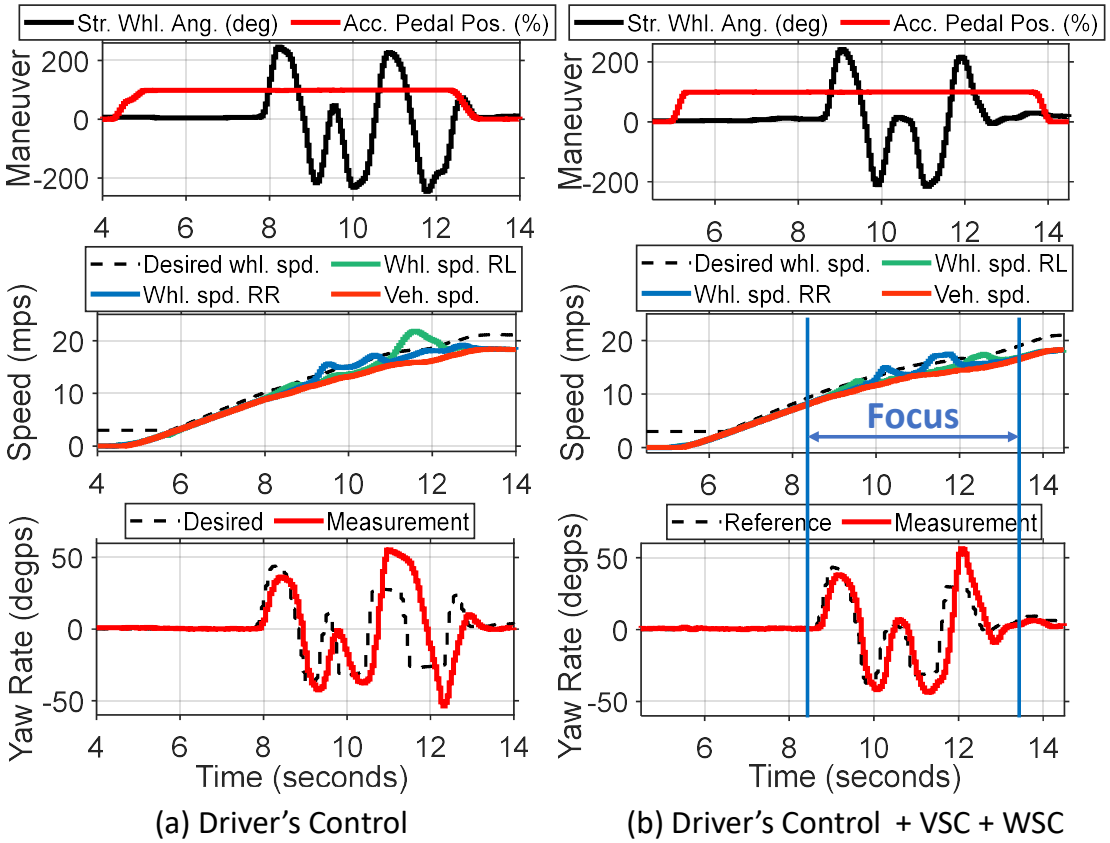


Figure 7.19: Experiment of MO-AMPC: Comparison of the vehicle response for DLC acceleration in the example of holistic stability control. The over-steering was significantly suppressed, and no obvious wheel slip occurred during the second test. The holistic stability was improved when both VSC and WSC were enabled.

As shown in the left plots of Figure 7.19, significant longitudinal wheel slip and over-steering occurred in test 2-A when the driver needed aggressive counter-steering to stabilize the vehicle. However, As shown in the right plots, when both objectives were enabled in test 2-B, yaw rate error was significantly reduced with no excessive wheel slip. Therefore, all agents collaborate to achieve both objectives simultaneously.

The suboptimality, iterations, and time consumption ratio results are shown in Figure 7.20. The majority of the torque suboptimality was always less than 5 Nm, which means the control actions from the MO-AMPC were very close to the integrated MPC, and their error can be neglected in practice. The number of iterations required was less than 25.

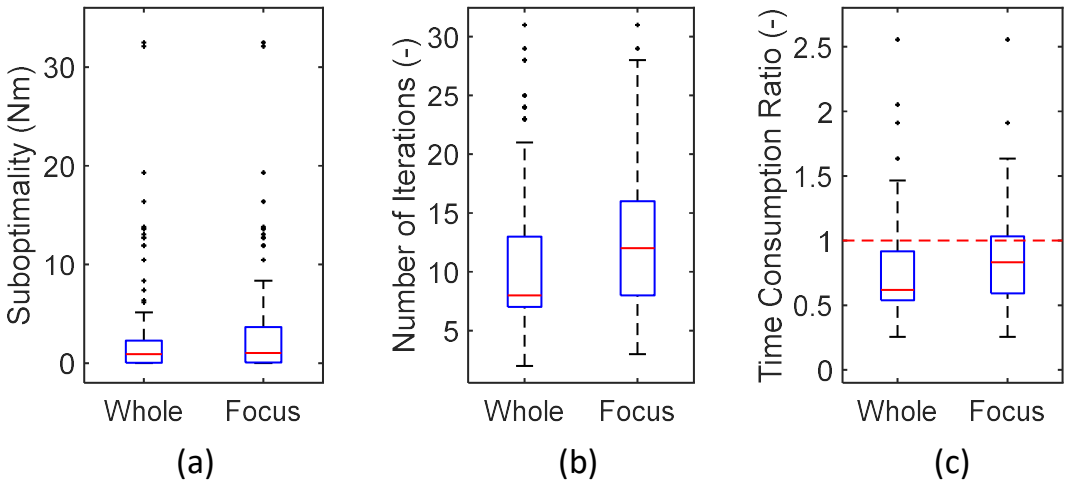


Figure 7.20: Experiment of MO-AMPC: Results of the convergence and computational efficiency for DLC acceleration in the example of holistic stability control. "Focus" represents the specific period marked in Figure 7.19 where both objectives require significant control.

The time consumption ratio was less than 1. A few more iterations are needed to achieve convergence during the "focus" period, so the time consumption ratio is slightly higher than 1.

7.5 Summary and Discussion

In this chapter, examples using the proposed control schemes are verified by real-vehicle experiments. Test vehicles and platforms are introduced first. Then, three experiments on three examples are introduced respectively:

1) The L-AMPC scheme proposed in Section 4 was first verified on a yaw stability control example as demonstrated in Section 4.4. Two maneuvers of Sinewaves and DLCs are tested using different controllers. By using the L-AMPC, the control performances were significantly improved because the unknown contributions were well-learned.

2) The MO-AMPC scheme proposed in Section 5 was verified on a cruise control example as demonstrated in Section 5.4. Both aggressive lane changes and normal cruise driving were tested. The efficacy of all the objectives and the collaboration of all agents

were verified. The convergence of the MO-AMPC to the integrated MPC scheme was proved. The feasibility of real-time implementations was also demonstrated.

3) The MO-AMPC scheme was further verified on a holistic vehicle stability control example as demonstrated in Section 5.5. Both straight launch and brake and DLC during acceleration were tested. The efficacy of collaboration and convergence, as well as the real-time feasibility of MO-AMPC, were further proved.

Chapter 8

Conclusion and Future Work

8.1 Conclusions

This thesis presented three practical and efficient distributed control schemes for MAMO systems in HVC. The two pain points of the AMPC scheme summarized in Section 1 - uncertain contributions from unknown controllers and limitations on objective-oriented decomposition - have been solved. The core concept of this study is combining data-driven learning methods with model-based distributed control schemes. Machine learning techniques are used not only to predict the model of unknown contributions but also to predict the interaction and negotiation between objectives. Meanwhile, model-based distributed control schemes guarantee the safety of constraint satisfaction under uncertainty and the global optimality of the distributed structure, improving the flexibility of controller design. The performance of proposed schemes is verified through simulations and experiments on several different vehicle control examples.

First, the learning agent-based model predictive control (L-AMPC) is proposed to solve the first pain point that the traditional AMPC highly depend on a perfect prediction model, but in practice, there may be some unknown "black-box" controllers contributing to the objective that is not presented in the known system dynamics in AMPC. This model mismatch will cause a significant error in predicting vehicle behaviour, leading to unexpected or even harmful control results. To this end, this scheme combines the model-based AMPC approach with the data-driven machine learning methods to improve the holistic vehicle performance for multi-agent systems with unknown contributions.

One of the most important requirements for incorporating data-driven learning methods in control systems is safety. Currently, any learned prediction cannot be guaranteed

to be completely reliable in all situations. Therefore, probabilistic prediction becomes a key requirement for safety-critical control systems, where distributions can provide information about the possibility of prediction deviations. This is why GPR, a non-parametric Bayesian method that produces probabilistic predictions, was chosen to be a very suitable learning method in control systems. For most control (and planning) optimization problems, safety is reflected in the satisfaction of constraints, such as torque limit, angle limit, obstacle avoidance safety, etc. Fortunately, the variance predicted by GPR can be smoothly converted to chance constraint under the set confidence level. Then, the chance constraint is used in the framework of stochastic MPC, which shrinks the constraints of the nominal model according to the size of the predicted variance to achieve the satisfaction of the actual constraints under the set confidence interval. Compared with the Robust MPC framework, the advantage of the stochastic MPC framework is to use the feedback assumption to suppress the expansion of uncertainty along the prediction horizon. In other words, the proposed L-AMPC based on the stochastic MPC framework can ensure the safety of the constraints while being as less conservative as possible. Moreover, soft constraints are added to the framework of stochastic MPC to ensure the feasibility of L-AMPC.

The core idea of GPR is to model the correlation between data points as a covariance function (kernel function), and data is the top priority. In order to adapt to the uncertainty of the vehicle driving environment, a continuously updated data management method is developed in L-AMPC. This data management system can absorb new experiences when the vehicle is driving in a new environment so that the L-AMPC can self-learn. However, an important shortcoming of GPR is that its computational complexity is proportional to the cube of the number of data points used for inference or training, which is a challenge for real-time control systems. To overcome it, multiple solutions are used for L-AMPC. First, hyper-parameter training based on a large amount of data is performed only offline because GPR is not so sensitive to the accuracy of hyper-parameters, and the correlation between the test point (prediction point) and the data used for inference is more important. Second, the inference sub-dataset is selected based on the L2-norm distance. Thus, a few but the most important data points are used for online real-time inference. In addition, the novel multi-step prediction framework makes GPR's predictions more accurate throughout the prediction horizon and, at the same time, meets the real-time requirements. Besides, model selection methods from statistical learning are used to select the minimum subset of the potential variables as the output of data points. For the input of data points, a contribution estimation method is developed to learn the unknown contribution on vehicle CG. This estimation method can apply to any number and type of unknown agents because a two-stage system dynamics is used in AMPC to unify the contributions of each controller, improving the generalization of L-AMPC.

The proposed L-AMPC is tested through simulations using a yaw stability control example. In this example, the multi-agent system consists of several controllable, white-box, and black-box agents contributing to the common objective: yaw rate tracking. The real-time simulation results show that the proposed L-AMPC can still satisfy the constraint in less-learned scenarios to keep safe while improving the tracking accuracy in well-learned scenarios since unknown contributions are leaned and predicted. By being proved on different agent configurations and driving conditions, the L-AMPC is general enough to apply to any type or number of unknown agents.

Second, the multi-objective agent-based model predictive control (MO-AMPC) is proposed to solve the second pain point of traditional AMPC that AMPC only decomposes the system from the controller's perspective. As vehicle automation increases, this pain point greatly limits the system's flexibility and generalizability for today's increasingly complex MAMO control systems, where multiple features designed as objectives with coupled controller agents should be achieved simultaneously. To this end, the MO-AMPC decompose the MAMO control systems from the objective perspective while guaranteeing convergence to the global optimum as the integrated MPC and satisfying the constraints as well, greatly improve the controller design flexibility with a "plug-and-play" fashion. Each objective has a local optimization, and information exchange between objectives is available. The local optimizations will be solved iteratively until the convergence checks on residuals are passed.

Three formulations of MO-AMPC tailored from consensus ADMM are proposed. In each formulation, every iteration in local optimization consists of three steps: the local update step optimizes the local control actions of the corresponding agent groups, the global update step optimizes the global variable based on all the results from the local updates, and the dual update step calculates the Lagrange multiplier. Although the convergence of ADMM to the global optimum has been proven, its application in practical distributed control systems requires more exploration. In practice, the distributed scheme should converge to the same global optimum as the integrated scheme while always satisfying constraints, and the computational efficiency cannot be too low. To achieve this, three different formulations of MO-AMPC with different allocations of tracking penalty, control regularization, and constraints are presented. The *Fully Distributed* formulation puts every component related to the local control action into the local optimization. The global and dual updates are simple and computationally efficient, but the optimization in the local update is relatively complex because the regularization of the control action is considered in each local node. The *Virtual Center Node* formulation handles the regularization of the control action centrally while distributing the tracking penalty in local optimizations, where the local update is a bit simpler than the *Fully Distributed* formulation. The *Local Explicit*

formulation is the most computationally efficient one because it converts the inequality constraints into indicator functions in the cost functions, allowing local updates to be solved explicitly through simple matrix operations. The satisfaction of constraints is achieved through the "max" operation of the virtual center node. Furthermore, the convergence and parameter selection of the *Local Explicit* formulation are also discussed in this thesis.

Simulations on two examples, cruise control and holistic vehicle stability control, are implemented using MO-AMPC. In both examples, two objectives need to be satisfied simultaneously, as well as a coupled controller agent. The tested scenarios all involve negotiation between the two objectives. The convergence to the global optimum of three formulations is verified in simulations. Besides, the *Local Explicit* formulation shows the best computational efficiency compared to the other two formulations. In addition, it shows the potential of superior computational efficiency to the integrated MPC.

Third, the learning multi-objective agent-based model predictive control (L-MO-AMPC) is proposed to improve further the computational efficiency of the MO-AMPC. As an iterative optimization method, the computational complexity of the MO-AMPC is positively correlated to the number of iterations required for convergence, which is a huge challenge for real-time control systems. To improve this, a learning-based method for initializing iterations is used in L-MO-AMPC to reduce the number of iterations needed.

The overall structure of the L-MO-AMPC is presented, where each objective has a learning prediction model to predict the negotiations from others for closer initial control actions. At the same time, the predicted variance is used as a criterion for the reliability check. The predicted mean value will be used to generate initial control actions only if the reliability check is passed. A data management system used in L-AMPC is adopted as the pre-processing for learning modules. The global variable and the Lagrangian multiplier are selected as the output of the learning prediction, which can represent the negotiation between objectives. Simulations on an autonomous path-tracking example are presented to test the performance of L-MO-AMPC. In this example, the longitudinal speed tracking and the lateral path tracking are decomposed with a shared FDB agent. The convergence to the global optimum has been verified. It achieves the same control performance as the integrated one while significantly reducing the computational time. Furthermore, if running on a chip with limited computational resources, L-MO-AMPC can improve the overall performance with easily defined objective priority.

Finally, experimental studies of the L-AMPC and the MO-AMPC are presented with the same examples in the simulations, respectively. All the experiments are implemented in real-time on an electric vehicle with four independent-drive electric motors, which can be designed as multiple collaborated controllers. The proposed control schemes are exten-

sively validated in different driving scenarios, different road conditions, and different agent configurations. The results show that the control performances were significantly improved on the yaw stability control example when using L-AMPC compared to using traditional AMPC because the unknown contributions were well-learned. For the MO-AMPC, the control performance of enabling both objectives is significantly improved compared to one or both objectives are disabled, which proves the efficacy of the distributed control scheme. Moreover, the convergence to the global optimum and the real-time feasibility of MO-AMPC are also validated on both cruise control and holistic stability control examples.

8.2 Limitations

This section will discuss the limitations of the method proposed in this thesis, although the method proposed in this thesis has good results. Some of these limitations can be ignored in practical applications, and some may require further research and exploration in the future.

Limitations of L-AMPC: Only the Linear Model Has Been Investigated

For L-AMPC, this thesis only discusses uncertainty propagation and shrinking methods for inequality constraints for linear models. For nonlinear models, uncertainty propagation may be more complicated. In addition, the closed-loop propagation with feedback assumption used in this paper to suppress uncertainty growth is no longer used.

However, it should be clarified that learning from data nonlinear or unknown models that are difficult to model, thereby simplifying model-based design (such as using simple linear models), is one of the beauties of the hybrid solution. In addition, in the linear model used in this article, the tire cornering stiffness is not a fixed constant but is updated at each time step based on the current vehicle state and the nonlinear tire model, which effectively improves the accuracy of the linear model.

Limitations of MO-AMPC: Convergence is Only Valid for QP Problems

The proposed MO-AMPC currently discusses the decomposition of the QP problem from the objective's perspective, i.e., quadratic loss function, linear model and inequality constraints. The convergence guarantee is also only applicable to QP problems. Further

research is needed for non-QP problems in the future. However, QP problems are sufficient for most engineering problems, especially vehicle control problems with high real-time requirements.

8.3 Future Work

This thesis could be the beginning of several promising research directions on intelligent autonomy. A few outlooks are made in this section for future research directions based on this thesis.

Safety-oriented learning-based autonomy

Modern approaches to hybrid data-driven and model-based planning and control are a hot topic. Data-driven has obvious advantages in dealing with highly complex scenarios and modelling highly nonlinear features, which can simplify the model-based design. However, the lack of interpretability and the inability to guarantee reliability in all scenarios remain challenges for purely data-driven learning methods, which pose safety risks. However, safety is the highest priority for planning and control, especially for automation related to people or life (such as vehicle control, autonomous driving, etc.). In addition, the real-time and adaptability of hybrid architectures are also one of the difficulties in engineering applications. The L-AMPC proposed in this paper provides a feasible glimpse into this safety-oriented field. It converts probabilistic learning prediction results into chance constraints and then uses a stochastic model-based optimization framework to ensure safety, maximizing the advantages of data-driven intelligence within the constraints. This core idea can unleash great potential in safety-oriented learning-based autonomy. As with GPR used in this article, probabilistic prediction distributions can be viewed as indirect interpretability for learning predictions. In addition, methods such as conformal prediction (CP), probabilistic deep time series prediction, variational autoencoder (VAE), etc., can also generate probabilistic predictions and be used in learning-based planning and control. Nowadays, more advanced and human-like interpretability can be provided by the large language model (LLM) and combined with model-based methods. For model-based methods that ensure safety, in addition to the robust and stochastic MPC mentioned in this article, there are also some tools worth exploring, for example, the control barrier function (CBF). In addition, ensuring safety while not making the results too conservative due to the loss of too many feasible domains is also the key to improving actual efficiency, as well as real-time performance. The above methods are all worth exploring in the future

and may produce more satisfactory results in the area of safety-oriented learning-based autonomy.

Scalable "plug-and-play" planning and control

Multi-agent planning and control has always been a hot research topic with high practical value. The flexible and configurable modular design with the "plug-and-play" fashion can facilitate scalability and reduce costs. The proposed MO-AMPC scheme has the potential to be extended beyond single-vehicle systems, serving as a general distributed control approach for systems that are physically distributed. Although the examples implemented in this thesis are only simple prototype systems containing a small number of agents and objectives, due to the mathematically guaranteed convergence, it is capable of large-scale complex systems. However, it is still necessary to explore interaction mechanisms and topology management to apply the MO-AMPC scheme in more complex practical systems. Many physically distributed systems, such as interaction-aware planning, multi-vehicle collision avoidance, and fleet management, are also potentially well suited to applying the proposed control scheme. Future works could focus on expanding the proposed method to these areas, particularly for MAMO systems with a larger number of agents and higher degrees of non-linearity.

Learning-based interactive collaboration

The proposed L-MO-AMPC scheme provides a new direction for the hybrid data-driven and model-based distributed control and planning framework. There were few similar studies at the time the author completed this thesis. The proposed scheme is not confined to path-tracking but is promising in many applications as a general data-driven distributed control scheme. It is especially ideal for physically distributed systems with available information exchange, such as interaction-aware planning, multi-vehicle avoidance control, and fleet management. The core idea of this scheme is to use a data-driven approach to learn interactions or negotiations between conflicting or cooperative objectives to reduce iterations or better help the overall system reach the global optimum or even make it possible to eliminate physical connections. Taking the interaction between autonomous vehicles and pedestrians as an example, traditional autonomous driving control based on interaction may only be one-way; that is, it only predicts the behaviour of people within a certain period of time so that the autonomous vehicle can avoid pedestrians accordingly. In a human-friendly traffic environment, autonomous vehicles should not only consider the impact of pedestrians on their behaviour but also take into account the feelings of

pedestrians in the environment by adjusting the ego vehicle's behaviour. For example, overly aggressive autonomous vehicles will make pedestrians feel uncomfortable. Therefore, considering two-way interaction to achieve global optimality to satisfy both parties or even multiple parties is a topic worthy of in-depth research. Some existing methods, such as game theory, may be able to achieve this function, but their complex calculations and non-scalability limit their application. On the contrary, the L-MO-AMPC architecture proposed in this paper not only has the characteristics of "plug-and-play" but also can achieve excellent real-time and self-learning capabilities with the help of data-driven learning, which is very suitable for application in such interactive collaboration scenarios. In the future, the proposed method will be expanded to include those interesting topics. Real-time efficiency and convergence performance for systems with a larger number of agents and stronger non-linearity will be further explored in real-world implementations.

Letter of Copyright Permission

IEEE, as the publisher of the three manuscripts mentioned above, fully or partly adopted in Chapters 4, 5, 6, and 7 allow the reuse of published papers in theses without formal permissions. The waivers of copyright from IEEE are achieved by displaying the following statement:

Policy Regarding Thesis/Dissertation Reuse, from IEEE Copyright Clearance Centre

"The IEEE does not require individuals working on a thesis to obtain a formal reuse license; however, you may print out this statement to be used as a permission grant".

References

[1] International Energy Agency. Global ev outlook 2023. https://www.iea.org/reports/global-ev-outlook-2023, 2023. (Accessed: September 17, 2024).

[2] Amin Rezaei, Mengqiu Cao, Qihao Liu, and Jonas De Vos. Synthesising the existing literature on the market acceptance of autonomous vehicles and the external underlying factors. *Journal of Advanced Transportation*, 2023(1):6065060, 2023.

[3] Paolo Falcone, H Eric Tseng, Francesco Borrelli, Jahan Asgari, and Davor Hrovat. Mpc-based yaw and lateral stabilisation via active front steering and braking. *Vehicle System Dynamics*, 46(S1):611–628, 2008.

[4] Chris Vermillion, Jing Sun, and Ken Butts. Predictive control allocation for a thermal management system based on an inner loop reference model—design, analysis, and experimental results. *IEEE transactions on control systems technology*, 19(4):772–781, 2010.

[5] Victor Mazzilli, Stefano De Pinto, Leonardo Pascali, Michele Contrino, Francesco Bottiglione, Giacomo Mantriota, Patrick Gruber, and Aldo Sorniotti. Integrated chassis control: Classification, analysis and future trends. *Annual Reviews in Control*, 51:172–205, 2021.

[6] Chen Tang and Amir Khajepour. Agent-based model predictive controller (ampc) for flexible and efficient vehicular control. *IEEE Transactions on Vehicular Technology*, 70(10):9877–9885, 2021.

[7] Kanghua Zhang, Jixin Wang, Xueting Xin, Xiang Li, Chuanwen Sun, Jianfei Huang, and Weikang Kong. A survey on learning-based model predictive control: Toward path tracking control of mobile platforms. *Applied Sciences*, 12(4):1995, 2022.

[8] Kang Yuan, Hong Shu, Yanjun Huang, Yubiao Zhang, Amir Khajepour, and Lin Zhang. Mixed local motion planning and tracking control framework for autonomous vehicles based on model predictive control. *IET Intelligent Transport Systems*, 13(6):950–959, 2019.

[9] William Dunham, Brandon Hencey, Anouck R Girard, and Ilya Kolmanovsky. Distributed model predictive control for more electric aircraft subsystems operating at multiple time scales. *IEEE transactions on control systems technology*, 28(6):2177–2190, 2019.

[10] Milad Jalaliyazdi. *Integrated vehicle stability control and power distribution using model predictive control*. PhD thesis, University of Waterloo, Waterloo, Canada, 2016.

[11] Zhang Hao, Li Xian-sheng, Shi Shu-ming, Liu Hong-fei, Guan Rachel, and Liu Li. Phase plane analysis for vehicle handling and stability. *International journal of computational intelligence Systems*, 4(6):1179–1186, 2011.

[12] HK Sachs and M Singh. Automobile stability–a study of the domain of attraction. *Vehicle System Dynamics*, 6(2-3):169–177, 1977.

[13] Hans B Pacejka. Tyre factors and vehicle handling. *International journal of vehicle design*, 1(1):1–23, 1979.

[14] Kong-Hui Guo. A study of a phase plane representation for identifying vehicle behavior. *Vehicle System Dynamics*, 15(sup1):152–167, 1986.

[15] Shoji Inagaki, Ikuo Kushiro, and Masaki Yamamoto. Analysis on vehicle stability in critical cornering using phase-plane method. *Jsae Review*, 2(16):216, 1995.

[16] Wei Liu, Lu Xiong, Bo Leng, Haolan Meng, and Renxie Zhang. Vehicle stability criterion research based on phase plane method. Technical report, SAE technical paper, 2017.

[17] Taeyoung Chung and Kyongsu Yi. Design and evaluation of side slip angle-based vehicle stability control scheme on a virtual test track. *IEEE Transactions on control systems technology*, 14(2):224–234, 2006.

[18] Carrie G Bobier. *A phase portrait approach to vehicle stabilization and envelope control*. Stanford University, 2012.

[19] A Stotsky and X Hu. Stability analysis of robustly decoupled car steering system with nonlinear tire model. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 5, pages 4750–4755. IEEE, 1997.

[20] Matthijs Klomp. *Longitudinal force distribution and road vehicle handling*. Chalmers Tekniska Hogskola (Sweden), 2010.

[21] Masato Abe, Yoshio Kano, Kazuasa Suzuki, Yasuji Shibahata, and Yoshimi Furukawa. Side-slip control to stabilize vehicle lateral motion by direct yaw moment. *JSAE review*, 22(4):413–419, 2001.

[22] Wanki Cho, Jangyeol Yoon, Jeongtae Kim, Jaewoong Hur, and Kyongsu Yi. An investigation into unified chassis control scheme for optimised vehicle stability and manoeuvrability. *Vehicle System Dynamics*, 46(S1):87–105, 2008.

[23] Siqi Zhang, Shuwen Zhou, and Jun Sun. Vehicle dynamics control based on sliding mode control technology. In *2009 Chinese Control and Decision Conference*, pages 2435–2439. IEEE, 2009.

[24] Moustapha Doumiati, Olivier Sename, J Martinez, Péter Gáspár, Z Szabo, L Dugard, and Jozsef Bokor. Vehicle yaw control via coordinated use of steering/braking systems. *IFAC Proceedings Volumes*, 44(1):644–649, 2011.

[25] Mansour Ataei, Amir Khajepour, and Soo Jeon. A novel reconfigurable integrated vehicle stability control with omni actuation systems. *IEEE Transactions on Vehicular Technology*, 67(4):2945–2957, 2017.

[26] Chen Tang and Amir Khajepour. Integrated stability control for narrow tilting vehicles: An envelope approach. *IEEE Transactions on Intelligent Transportation Systems*, 22(5):3158–3166, 2020.

[27] Junmin Wang and Raul G Longoria. Coordinated and reconfigurable vehicle dynamics control. *IEEE Transactions on Control Systems Technology*, 17(3):723–732, 2009.

[28] Haiyan Zhao, Weixuan Chen, Jinyang Zhao, Yilin Zhang, and Hong Chen. Modular integrated longitudinal, lateral, and vertical vehicle stability control for distributed electric vehicles. *IEEE transactions on vehicular technology*, 68(2):1327–1338, 2019.

[29] Shih-Ken Chen, Youssef Ghoneim, Nikolai Moshchuk, Bakhtiar Litkouhi, and Valery Pylypchuk. Tire-force based holistic corner control. In *ASME International Mechanical Engineering Congress and Exposition*, volume 45271, pages 133–140. American Society of Mechanical Engineers, 2012.

[30] Yamin Wang, Shouxiang Wang, and Lei Wu. Distributed optimization approaches for emerging power systems operation: A review. *Electric Power Systems Research*, 144:127–135, 2017.

[31] Shengbo Eben Li, Yang Zheng, Keqiang Li, Yujia Wu, J Karl Hedrick, Feng Gao, and Hongwei Zhang. Dynamical modeling and distributed control of connected and automated vehicles: Challenges and opportunities. *IEEE Intelligent Transportation Systems Magazine*, 9(3):46–58, 2017.

[32] Mohammad Pirani, Ehsan Hashemi, Amir Khajepour, Baris Fidan, Bakhtiar Litkouhi, Shih-Ken Chen, and Shreyas Sundaram. Cooperative vehicle speed fault diagnosis and correction. *IEEE Transactions on Intelligent Transportation Systems*, 20(2):783–789, 2018.

[33] Ruben Van Parys and Goele Pipeleers. Distributed mpc for multi-vehicle systems moving in formation. *Robotics and Autonomous Systems*, 97:144–152, 2017.

[34] Qi Yan, Yue Ma, and Shumin Ruan. Research on cooperative control based on dmpc for intelligent vehicle platoon. In *Proceedings of 2021 Chinese Intelligent Systems Conference: Volume III*, pages 622–633. Springer, 2022.

[35] Lubomir Bakule. Decentralized control: An overview. *Annual reviews in control*, 32(1):87–98, 2008.

[36] N Sandell, Pravin Varaiya, Michael Athans, and M Safonov. A survey of decentralized control methods for large scale systems. *Systems Eng. for Power, US Dept. of Commerce*, pages 334–335, 1975.

[37] Dragoslav D Siljak. *Decentralized control of complex systems*. Courier Corporation, 2013.

[38] Lu Xiong, Ming Liu, Xing Yang, and Bo Leng. Integrated path tracking for autonomous vehicle collision avoidance based on model predictive control with multiconstraints. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 554–561. IEEE, 2022.

[39] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on control systems technology*, 15(3):566–580, 2007.

[40] Francesco Borrelli, Paolo Falcone, Tamas Keviczky, Jahan Asgari, and Davor Hrovat. Mpc-based approach to active steering for autonomous vehicle systems. *International journal of vehicle autonomous systems*, 3(2-4):265–291, 2005.

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

[42] Ligang Wu, Xiaojie Su, and Peng Shi. Sliding mode control with bounded l2 gain performance of markovian jump singular time-delay systems. *Automatica*, 48(8):1929–1933, 2012.

[43] Ehsan Hashemi and Amir Khajepour. Integrated path-tracking and combined-slip force controls of autonomous ground vehicles with safe constraints adaptation. *IEEE Transactions on Intelligent Vehicles*, 2024.

[44] Changfang Chen, Yingmin Jia, Junping Du, and Fashan Yu. Lane keeping control for autonomous 4ws4wd vehicles subject to wheel slip constraint. In *2012 American Control Conference (ACC)*, pages 6515–6520. IEEE, 2012.

[45] Craig Earl Beal and J Christian Gerdes. Model predictive control for vehicle stabilization at the limits of handling. *IEEE Transactions on Control Systems Technology*, 21(4):1258–1269, 2012.

[46] Milad Jalali, Saeid Khosravani, Amir Khajepour, Shih-ken Chen, and Bakhtiar Litkouhi. Model predictive control of vehicle stability using coordinated active steering and differential brakes. *Mechatronics*, 48:30–41, 2017.

[47] Giovanni Palmieri, Osvaldo Barbarisi, Stefano Scala, and Luigi Glielmo. An integrated ltv-mpc lateral vehicle dynamics control: Simulation results. *Automotive Model Predictive Control: Models, Methods and Applications*, pages 231–255, 2010.

[48] P Giovanni, B Osvaldo Barbarisi, and G Luigi. A preliminary study to integrate ltv-mpc lateral vehicle dynamics control with a slip control. In *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference Shanghai, PR China*, 2009.

[49] Shaosong Li, Guodong Wang, Bangcheng Zhang, Zhixin Yu, and Gaojian Cui. Vehicle yaw stability control at the handling limits based on model predictive control. *International Journal of Automotive Technology*, 21:361–370, 2020.

[50] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N Pistikopoulos. The explicit solution of model predictive control via multiparametric quadratic programming. In *Proceedings of the 2000 American control conference. ACC (IEEE Cat. No. 00CH36334)*, volume 2, pages 872–876. IEEE, 2000.

[51] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. Geometric algorithm for multiparametric linear programming. *Journal of optimization theory and applications*, 118:515–540, 2003.

[52] Tomas Gal. *Postoptimal Analyses, Parametric Programming, and Related Topics: degeneracy, multicriteria decision making, redundancy*. Walter de Gruyter, 2010.

[53] Alberto Bemporad, Francesco Borrelli, and Manfred Morari. Piecewise linear optimal controllers for hybrid systems. In *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334)*, volume 2, pages 1190–1194. IEEE, 2000.

[54] Seyedeh Asal Nahidi. *Reconfigurable integrated vehicle stability control using optimal control techniques*. PhD thesis, University of Waterloo, Waterloo, Canada, 2017.

[55] Giovanni Palmieri, Paolo Falcone, H Eric Tseng, and Luigi Glielmo. A preliminary study on the effects of roll dynamics in predictive vehicle stability control. In *2008 47th IEEE Conference on Decision and Control*, pages 5354–5359. IEEE, 2008.

[56] Hongyan Guo, Feng Liu, Fang Xu, Hong Chen, Dongpu Cao, and Yan Ji. Nonlinear model predictive lateral stability control of active chassis for intelligent vehicles and its fpga implementation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(1):2–13, 2017.

[57] H.J. Ferreau, C. Kirches, A. Potschka, H.G. Bock, and M. Diehl. qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 2014.

[58] Andreas Wächter and Carl D. Laird. *Algorithm 857: IPOPT–A Flexible Interior Point Algorithm for Large-Scale Nonlinear Programming*. ACM, 2006. Software available at https://github.com/coin-or/Ipopt.

[59] Giancarlo Ferrari-Trecate, Francesco Alessandro Cuzzola, Domenico Mignone, and Manfred Morari. Analysis of discrete-time piecewise affine and hybrid systems. *Automatica*, 38(12):2139–2146, 2002.

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

[61] Stefano Di Cairano, Hongtei Eric Tseng, Daniele Bernardini, and Alberto Bemporad. Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain. *IEEE Transactions on Control Systems Technology*, 21(4):1236–1248, 2013.

[62] Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In *Robustness in identification and control*, pages 207–226. Springer, 2007.

[63] Zhi Qiang Alex Zheng. *Robust control of systems subject to constraints*. PhD thesis, California Institute of Technology, 1995.

[64] Alberto Bemporad, Francesco Borrelli, and Manfred Morari. Robust model predictive control: Piecewise linear explicit solution. In *2001 European Control Conference (ECC)*, pages 939–944. IEEE, 2001.

[65] Stefano Riverso and Giancarlo Ferrari-Trecate. Tube-based distributed control of linear constrained systems. *Automatica*, 48(11):2860–2865, 2012.

[66] Massimo Canale, Lorenzo Fagiano, and MARIA CARMELA Signorile. A model predictive control approach to vehicle yaw control using identified models. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of automobile engineering*, 226(5):577–590, 2012.

[67] Halit Zengin. *Multiple-model robust adaptive vehicle motion control*. PhD thesis, University of Waterloo, 2019.

[68] Ali Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6):30–44, 2016.

[69] Marco Bichi, Giulio Ripaccioli, Stefano Di Cairano, Daniele Bernardini, Alberto Bemporad, and Ilya V Kolmanovsky. Stochastic model predictive control with driver behavior learning for improved powertrain control. In *49th IEEE conference on decision and control (CDC)*, pages 6077–6082. IEEE, 2010.

[70] Stefano Di Cairano, Daniele Bernardini, Alberto Bemporad, and Ilya V Kolmanovsky. Stochastic mpc with learning for driver-predictive vehicle control and its application to hev energy management. *IEEE Transactions on Control Systems Technology*, 22(3):1018–1031, 2013.

[71] Lukas Hewing, Juraj Kabzan, and Melanie N Zeilinger. Cautious model predictive control using gaussian process regression. *IEEE Transactions on Control Systems Technology*, 28(6):2736–2743, 2019.

[72] Eugenio Cinquemani, Mayank Agarwal, Debasish Chatterjee, and John Lygeros. Convexity and convex approximations of discrete-time stochastic control problems with constraints. *Automatica*, 47(9):2082–2087, 2011.

[73] Pierre OM Scokaert and David Q Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic control*, 43(8):1136–1142, 1998.

[74] R Tyrrell Rockafellar, Stanislav Uryasev, et al. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

[75] I. Batina. *Model predictive control for stochastic systems by randomized algorithms*. Phd thesis 1 (research tu/e / graduation tu/e), Mathematics and Computer Science, 2004.

[76] Mark Cannon, Basil Kouvaritakis, and Xingjian Wu. Model predictive control for systems with stochastic multiplicative uncertainty and probabilistic constraints. *Automatica*, 45(1):167–172, 2009.

[77] Jun Yan. *Constrained model predictive control, state estimation and coordination*. University of California, San Diego, 2006.

[78] Giuseppe Calafiore and Marco C Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102:25–46, 2005.

[79] D Munoz de la Penad, Alberto Bemporad, and Teodoro Alamo. Stochastic programming applied to model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 1361–1366. IEEE, 2005.

[80] Marco C Campi, Simone Garatti, and Maria Prandini. Scenario optimization for mpc. *Handbook of Model Predictive Control*, pages 445–463, 2019.

[81] Giulio Ripaccioli, Daniele Bernardini, Stefano Di Cairano, Alberto Bemporad, and Ilya V Kolmanovsky. A stochastic model predictive control approach for series hybrid electric vehicle power management. In *Proceedings of the 2010 American control conference*, pages 5844–5849. IEEE, 2010.

[82] Lars Blackmore, Masahiro Ono, Askar Bektassov, and Brian C Williams. A probabilistic particle-control approximation of chance-constrained stochastic predictive control. *IEEE transactions on Robotics*, 26(3):502–517, 2010.

[83] Georg Schildbach and Francesco Borrelli. Scenario model predictive control for lane change assistance on highways. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 611–616. IEEE, 2015.

[84] Changchun Liu, Ashwin Carvalho, Georg Schildbach, and J Karl Hedrick. Stochastic predictive control for lane keeping assistance systems using a linear time-varying model. In *2015 American Control Conference (ACC)*, pages 3355–3360. IEEE, 2015.

[85] Chen Tang and Amir Khajepour. Wheel modules with distributed controllers: A multi-agent approach to vehicular control. *IEEE Transactions on Vehicular Technology*, 69(10):10879–10888, 2020.

[86] Matthew S Elliott and Bryan P Rasmussen. Model-based predictive control of a multi-evaporator vapor compression cooling cycle. In *2008 American Control Conference*, pages 1463–1468. IEEE, 2008.

[87] Yi Zheng, Shaoyuan Li, and Hai Qiu. Networked coordination-based distributed model predictive control for large-scale system. *IEEE Transactions on Control Systems Technology*, 21(3):991–998, 2012.

[88] Jinfeng Liu, Xianzhong Chen, David Muñoz de la Peña, and Panagiotis D Christofides. Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. *AIChE Journal*, 56(8):2137–2149, 2010.

[89] JM Maestre, D Munoz De La Pena, EF Camacho, and T Alamo. Distributed model predictive control based on agent negotiation. *Journal of Process Control*, 21(5):685–697, 2011.

[90] Yang Zheng, Shengbo Eben Li, Keqiang Li, Francesco Borrelli, and J Karl Hedrick. Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies. *IEEE Transactions on Control Systems Technology*, 25(3):899–910, 2016.

[91] Stefano Riverso, Marcello Farina, and Giancarlo Ferrari-Trecate. Plug-and-play decentralized model predictive control for linear systems. *IEEE Transactions on Automatic Control*, 58(10):2608–2614, 2013.

[92] Chen Tang, Mansour Ataei, and Amir Khajepour. A reconfigurable integrated control for narrow tilting vehicles. *IEEE Transactions on Vehicular Technology*, 68(1):234–244, 2018.

[93] Mansour Ataei, Amir Khajepour, and Soo Jeon. Reconfigurable integrated stability control for four-and three-wheeled urban vehicles with flexible combinations of actuation systems. *IEEE/ASME Transactions on Mechatronics*, 23(5):2031–2041, 2018.

[94] Chen Tang, Mehdi Abroshan, and Amir Khajepour. Agent-based model predictive controller (ampc) for vehicular stability with experimental results. *IEEE Transactions on Vehicular Technology*, 71(7):7104–7112, 2022.

[95] Jeremy H. Gillulay and Claire J. Tomlin. Guaranteed safe online learning of a bounded system. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2979–2984, 2011.

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

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

[98] Anil Aswani, Patrick Bouffard, and Claire Tomlin. Extensions of learning-based model predictive control for real-time application to a quadrotor helicopter. In *2012 American Control Conference (ACC)*, pages 4661–4666. IEEE, 2012.

[99] Anil Aswani, Humberto Gonzalez, S Shankar Sastry, and Claire Tomlin. Statistical results on filtering and epi-convergence for learning-based model predictive control. *arXiv preprint arXiv:1208.0864*, 2012.

[100] Patrick Bouffard, Anil Aswani, and Claire Tomlin. Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In *2012 IEEE International Conference on Robotics and Automation*, pages 279–284. IEEE, 2012.

[101] Anil Aswani, Neal Master, Jay Taneja, Andrew Krioukov, David Culler, and Claire Tomlin. Energy-efficient building hvac control using hybrid system lbmpc. *IFAC Proceedings Volumes*, 45(17):496–501, 2012.

[102] Saket Adhau, Sayli Patil, Deepak Ingole, and Dayaram Sonawane. Embedded implementation of deep learning-based linear model predictive control. In *2019 sixth Indian control conference (ICC)*, pages 200–205. IEEE, 2019.

[103] Chao Ning and Fengqi You. Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming. *Computers & Chemical Engineering*, 125:434–448, 2019.

[104] Jonggu Lee, Seungwan Ryu, Taewan Kim, Wonchul Kim, and H Jin Kim. Learning-based path tracking control of a flapping-wing micro air vehicle. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7096–7102. IEEE, 2018.

[105] Benjamin Recht. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):253–279, 2019.

[106] Alonso Marco, Philipp Hennig, Jeannette Bohg, Stefan Schaal, and Sebastian Trimpe. Automatic lqr tuning based on gaussian process global optimization. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 270–277. IEEE, 2016.

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

[108] Haimin Hu, Konstantinos Gatsis, Manfred Morari, and George J Pappas. Non-cooperative distributed mpc with iterative learning. *IFAC-PapersOnLine*, 53(2):5225–5232, 2020.

[109] Yvonne R Stürz, Edward L Zhu, Ugo Rosolia, Karl H Johansson, and Francesco Borrelli. Distributed learning model predictive control for linear systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 4366–4373. IEEE, 2020.

[110] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[111] Ramin Rostami, Giuliano Costantini, and Daniel Görges. Admm-based distributed model predictive control: Primal and dual approaches. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 6598–6603. IEEE, 2017.

[112] Euhanna Ghadimi, André Teixeira, Iman Shames, and Mikael Johansson. Optimal parameter selection for the alternating direction method of multipliers (admm): Quadratic problems. *IEEE Transactions on Automatic Control*, 60(3):644–658, 2014.

[113] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[114] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[115] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[116] Ian Goodfellow. Deep learning, 2016.

[117] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[118] Mikhail Belyaev, Evgeny Burnaev, and Yermek Kapushev. Exact inference for gaussian process regression in case of big data with the cartesian product structure. *arXiv preprint arXiv:1403.6573*, 2014.

[119] Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch. *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*, volume 3176. Springer, 2011.

[120] Juš Kocijan. *Modelling and control of dynamic systems using Gaussian process models*. Springer, 2016.

[121] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[122] Saurabh Suradhaniwar, Soumyashree Kar, Surya S Durbha, and Adinarayana Jagarlapudi. Time series forecasting of univariate agrometeorological data: a comparative performance evaluation via one-step and multi-step ahead forecasting strategies. *Sensors*, 21(7):2430, 2021.

[123] Michael Maiworm, Daniel Limon, Jose Maria Manzano, and Rolf Findeisen. Stability of gaussian process learning based output feedback model predictive control. *IFAC-PapersOnLine*, 51(20):455–461, 2018.

[124] David Q Mayne. Control of constrained dynamic systems. *European Journal of Control*, 7(2-3):87–99, 2001.

[125] Michael Maiworm, Daniel Limon, and Rolf Findeisen. Online learning-based model predictive control with gaussian process models and stability guarantees. *International Journal of Robust and Nonlinear Control*, 31(18):8785–8812, 2021.

[126] Alexander Rose, Maik Pfefferkorn, Hoang Hai Nguyen, and Rolf Findeisen. Learning a gaussian process approximation of a model predictive controller with guarantees. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 4094–4099. IEEE, 2023.

[127] Ehsan Hashemi, Milad Jalali, Amir Khajepour, Alireza Kasaiezadeh, and Shih-ken Chen. Vehicle stability control: Model predictive approach and combined-slip effect. *IEEE/ASME Transactions on Mechatronics*, 25(6):2789–2800, 2020.

[128] ZhouDan Pan and Mark Cannon. Asynchronous admm via a data exchange server. *IEEE Trans. Control Netw. Syst.*, pages 1–12, 2024.

[129] Yixiao Liang, Yinong Li, Amir Khajepour, and Ling Zheng. Holistic adaptive multi-model predictive control for the path following of 4wid autonomous vehicles. *IEEE Transactions on Vehicular Technology*, 70(1):69–81, 2020.

[130] Larry Wasserman. *All of nonparametric statistics*. Springer Science & Business Media, 2006.

[131] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[132] George Casella and Roger Berger. *Statistical inference*. CRC Press, 2024.

[133] Richard A Berk et al. *Statistical learning from a regression perspective*, volume 14. Springer, 2008.

[134] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

APPENDICES

Appendix A

Non-parametric Kernel-based Regression

A.1 Formulation of Kernel-based Regression

As introduced in Section 3.4, the output of the data point y_p is related to x_p through [117],

$$y_p = f(x_p) + \varepsilon_p \tag{A.1}$$

where ε_p are additive noises and follow independent, identically distributed Gaussian distributions with zero mean and variances $\sigma^2_{N_d}$. Now, the problem is to estimate the latent function $\hat{f}(x)$ and predict the output $\hat{y}_*$ given inputs [117]. According to the basis method [114][130], the latent function $\hat{f}(x)$ can be expanded as,

$$\hat{f}(x) = \sum_{i=1}^{N} w_i \psi_i(x) = \boldsymbol{\psi}(x)^T \boldsymbol{w} \tag{A.2}$$

where $\boldsymbol{\psi}(x) = [\psi_1(x), \psi_2(x), \ldots, \psi_N(x))]^T$ are some known basis functions, such as splines and wavelets, mapping a p-dimensional input vector x into an N-dimensional feature space. $\boldsymbol{w} = (w_1, w_2, \ldots, w_N)^T$ are parameters. Therefore, an estimate of $f(x)$ can be obtained by estimating $\boldsymbol{w}$. In fact, this is a typical implementation of the core idea in machine learning: first, project the inputs into some high dimensional space using a set of basic feature space functions, and then apply the linear model in this space instead of directly on the inputs themselves.

Similar to the rigid regression [131] in linear regression, the regularization could be realized by adding a roughness penalty λ when minimizing the penalized sum of squares (in Appendix A.3 the Bayesian linear formalism will be considered, according to the likelihood function based on the noise assumption. For now, just non-Bayesian formalism is discussed.):

$$\min_{w} \left\{ (\boldsymbol{y} - \boldsymbol{\Psi w})^T (\boldsymbol{y} - \boldsymbol{\Psi w}) + \lambda \|\boldsymbol{w}\|^2 \right\} \tag{A.3}$$

where $\boldsymbol{\Psi}$ denotes the aggregation of columns $\boldsymbol{\psi}(x)$ for all cases in the training set. The optimal solution is given by,

$$\boldsymbol{w} = (\boldsymbol{\Psi}^T \boldsymbol{\Psi} + \lambda I_N)^{-1} \boldsymbol{\Psi}^T \boldsymbol{y} = \boldsymbol{\Psi}^T (\boldsymbol{\Psi} \boldsymbol{\Psi}^T + \lambda I_N)^{-1} \boldsymbol{y} \tag{A.4}$$

By replacing $\boldsymbol{\Psi}^T \boldsymbol{\Psi}$ with the Gramian matrix K and defining $\alpha = (K + \lambda I_n)^{-1} \boldsymbol{y}$, Equation A.4 becomes:

$$\boldsymbol{w} = \boldsymbol{\Psi} \boldsymbol{\alpha} = \sum_{p=1}^{N_d} \alpha_i \psi(x_i) \tag{A.5}$$

$$\hat{f}(x) = \boldsymbol{w}^T \boldsymbol{\psi}(x) = \sum_{p=1}^{N_d} \alpha_i \psi(x_i)^T \psi(x) = \sum_{p=1}^{N_d} \alpha_i K(x_i, x) \tag{A.6}$$

where K denotes the kernels, a kernel is a positive-definite function of two inputs and can be seen as a measure of the similarity and compute an inner product in a high-dimensional (implicit) feature space. That's why a kernel is also often called a covariance function, kernel function, or covariance kernel.

A kernel is also called a kernel smoother, smooth function, or weight function [114]. These concepts characterize another role of the kernel, which is used as a device for localization. That is, estimating the unknown function $f(x)$ by fitting a different but simple model separately at each query point x. This is done by fitting the simple model using only observations close to the target point x and ensuring that the resulting estimated function $f(x)$ is smooth. From Equation A.4, the estimation of the latent function can be written as,

$$\hat{f}(x) = \boldsymbol{\psi}(x)^T \boldsymbol{w} = \boldsymbol{\psi}(x)^T \left(\boldsymbol{\Psi}^T (\boldsymbol{\Psi} \boldsymbol{\Psi}^T + \lambda I_N)^{-1} \boldsymbol{y} \right) = S \boldsymbol{y} \tag{A.7}$$

where $S = \boldsymbol{\psi}(x)^T \left(\boldsymbol{\Psi}^T (\boldsymbol{\Psi} \boldsymbol{\Psi}^T + \lambda I_N) \right)^{-1}$ is called smoothing matrix or the hat matrix, which can be seen as a series of kernel operations. Thus, this localization is achieved via a weighting function or kernel $K(\cdot, \cdot)$, which assigns a weight to x_p based on its distance from x.

A.2 Model Selection for Regression

Regression is, in fact, a trade-off between bias and variance. To reduce the excessive modelling bias, a large number of predictors are usually introduced at the initial stage of the modelling. However, this increases the variance and reduces the predictive power. To enhance the predictive power, one needs to select the significant variables. The term "model selection" has many different definitions according to the occasion; here, the definition in statistical learning [117] is used, namely the best subset selection of input variables (or called predictors in regression).

According to the order of selection, there are usually two methods: forward selection and backward selection. Forward selection starts from the minimum model and then adds variables, while backward selection starts from the most complete model and gradually reduces variables. During this process, the best set of variables is selected according to some criteria. Since kernel-based regression could be regarded as a linear smoother, thus the model selection criteria used in multiple linear regression also applies to this study. The commonly used standards can be roughly divided into three categories, as introduced in Section 4.2.1: (a) predictive risk function, e.g., PMSE and Stein's Unbiased Risk Estimator (SURE); (b) Coefficient of determination: R^2_{adj}; (c) likelihood-based information criteria: Mallows' C_p, AIC, and BIC.

PMSE represents the expected value of the squared difference between the fitted values and the observation, which is often calculated by the squared error loss function,

$$\text{PMSE} = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2\right] = \frac{1}{n}\sum_{i=1}^{n} E\left[(f(x_i) - \hat{f}(x_i))^2\right] + \sigma_n^2 \tag{A.8}$$

However, the predictive risk $R(f, \hat{f}) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[(f(x_i) - \hat{f}(x_i))^2\right]$ is unknown. Thus, leave-one-out cross-validation (LOOCV) is usually used to estimate PMSE. Partitioned training data is used to fit the model (estimate f), and the left test data is used to evaluate the model:

$$\text{PMSE} \approx \text{LOOCV} = -\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[(f(x_i) - \hat{f}_{-i}(x_i))^2\right] + \sigma_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}(x_i)}{1 - S_{ii}}\right)^2 \tag{A.9}$$

where s_{ij} represents the (i, j)-th entry of the smooth matrix S introduced in Equation A.7, and $\hat{f}$ is the estimation based on the entire dataset D_n and therefore saves the computation time. In addition, there is a modified form called generalized cross-validation (GCV),

which basically replaces factors $1 - s_{ii}$ in LOOCV by their average, for the convenience of calculation,

$$\text{PMSE} \approx \text{GCV} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{f}(x_i)}{1 - \frac{n^{-1}\text{trace}(S)}{n}} \right)^2 \tag{A.10}$$

The coefficient of determination, R^2_{adj}, basically represent the proportion of the variation in the variables that is predictable,

$$R^2_{adj} = 1 - \frac{SS_{\text{res}}/df_{\text{res}}}{SS_{\text{tol}}/df_{\text{tol}}} \tag{A.11}$$

where $SS_{\text{res}} = \sum_{i=1}^{n}(y_i - f(x_i))^2$ represents the residual sum of squares (RSS); $SS_{\text{tol}} = \sum_{i=1}^{n}(y_i - \bar{y})^2$ represents the total sum of squares (TSS); $df_{\text{res}} = n - p$ is the degrees of freedom of the estimate of the population variance around the model; $df_{\text{tol}} = n - 1$ is the degrees of freedom of the estimate of the population variance around the mean.

AIC and BIC are those criteria that estimate the relative amount of information lost by a regression model. They are defined in the literature as,

$$\text{AIC} = -2(\text{maximum loglikelihood}) + 2(\text{number of free parameters}), \tag{A.12}$$

$$\text{BIC} = -2(\text{maximum loglikelihood}) + (\text{number of free parameters})\log(n), \tag{A.13}$$

which could be calculated through,

$$\text{AIC} = n \log\left(\frac{SS_{\text{res}}(S)}{n}\right) + 2(|S| + 1) + \text{constant}, \tag{A.14}$$

$$\text{BIC} = n \log\left(\frac{SS_{\text{res}}(S)}{n}\right) + (|S| + 1)\log(n) + \text{constant}. \tag{A.15}$$

Usually, BIC favours smaller models than AIC, as BIC puts a harsher penalty for the complexity of a model.

A.3 GPR Training Process: Hyper-parameter Optimization

Machine learning models contain many parameters that need to be determined before the inference. In practical applications, a point-like prediction, which is optimal in some sense,

is usually needed. To this end, a loss function, $L(y_{\text{true}}, y_{\text{guess}})$ is needed to specify the loss (or penalty) incurred by guessing the value y_{guess} when the true value is y_{true}. For example, the loss function could equal the absolute deviation between the guess and the truth.

In non-Bayesian paradigms, the model is typically trained Bayesian paradigm by minimizing the empirical risk (or loss), which is usually the error-based loss function. In contrast, in the Bayesian setting, there is a clear separation between the likelihood function (used for training, in addition to the prior) and the loss function. The likelihood function describes how the noisy measurements are assumed to deviate from the underlying noise-free function. The loss function, on the other hand, captures the consequences of making a specific choice, given an actual true state. Besides, the likelihood function should be given the prior and assuming the noise distribution. The likelihood and loss function need not have anything in common.

In non-Bayesian paradigms, the goal is to make the point prediction y_{guess}, which incurs the smallest loss. The expected loss or risk should be minimized by averaging y_{guess} w.r.t. the model's opinion since the y_{true} is unknown,

$$\hat{R}(y_{\text{guess}}|x) = \int (y_{\text{guess}}, y) p(y|x, \theta)\, dy \tag{A.16}$$

Thus, the best guess, in the sense that it minimizes the expected loss, is,

$$y_{\text{optimal},k} = \arg\min_{y_{\text{guess}}} \hat{R}(y_{\text{guess}}, x_k) \tag{A.17}$$

In Bayesian paradigms, the likelihood function (often simply called the likelihood) describes the joint probability of the observed data as a function of the parameters of the chosen statistical model [132]. It is common to use a hierarchical specification of models. At the lower level are the parameters, $\boldsymbol{w}$. For example, the parameters could be the parameters in a linear model or the weights in a neural network model. At the higher level are hyper-parameters $\boldsymbol{\theta}$, which control the distribution of the parameters at the bottom level. For example, the "weight decay" term in a neural network. For the kernel-based regression model, usually the length-scale (or called bandwidth) l is one of the most important hyper-parameters which controls the smoothness of the underlying function $f(x)$. From the Bayes' rule, the posterior over the parameters in the lower level,

$$p(w|y, X, \theta) = \frac{p(y|X, w)p(w|\theta)}{p(y|X, \theta)} \tag{A.18}$$

where $p(w|\theta)$ is the parameter prior, the $p(y|X,w)$ is the likelihood, and the $p(y|X,\theta)$ is the marginal likelihood, which is given by,

$$p(y|X,\theta) = \int p(y|X,w)p(w|\theta)\,dw \tag{A.19}$$

In the higher level, the posterior over the hyper-parameters could be analogously expressed in the way that the marginal likelihood from the first level plays the role of the likelihood,

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} \tag{A.20}$$

where,

$$p(y|\theta) = \int p(y|X,w,\theta)p(w|\theta)\,dw \tag{A.21}$$

In practice, it is common to maximize the marginal likelihood $p(y|X,\theta)$ in Equation A.19 w.r.t. the hyper-parameters, θ, instead of calculating additional integral in Equation A.21. Also, the marginal likelihood $p(y|X,\theta)$ is independent of the parameters w. Besides, it is primarily the marginal likelihood from Equation A.19 involving the integral over the parameter space, which distinguishes the Bayesian scheme of inference from other schemes based on optimization. It is a property of the marginal likelihood that it automatically incorporates a trade-off between model fit and model complexity. Notice that the trade-off between data fit and model complexity is automatic; there is no need to set a parameter externally to fix the trade-off. This is the reason why the marginal likelihood is valuable in solving the hyper-parameter selection problem.

Both the loss function and marginal likelihood function could be used in cross-validation (CV). CV is almost always used in the k-fold cross-validation setting: the data is split into k disjoint, equally sized subsets; validation is done on a single subset and training is done using the union of the remaining $k-1$ subsets, the entire procedure is repeated k times, each time with a different subset for validation. Thus, a large fraction of the data can be used for training, and all cases appear as validation cases. The price is that k models must be trained instead of one. Often, the computational cost of CV (training k models) is prohibitive, but in certain cases, such as GPR, there are computational shortcuts [133].

As one of the memory-based regression methods, glsgpr requires little or no training; all the work gets done at evaluation time. In general, the only parameter that needs to be determined from the training data is the length-scale (or called bandwidth) l. The marginal likelihood is commonly used in solving the hyper-parameter selection for Bayesian formalism, for example, glsgpr training.

Under the Gaussian process model, the prior is Gaussian. Then, the log marginal likelihood from (A.30) could be written as

$$\log p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{y}^\top [K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1} \boldsymbol{y} - \frac{1}{2}\log \left| K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I \right| - \frac{n}{2}\log 2\pi \quad \text{(A.22)}$$

The three terms of the marginal likelihood in Equation A.22 have readily interpretable roles: the only term involving the observed targets ($-\frac{1}{2}\boldsymbol{y}^\top [K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I]^{-1} \boldsymbol{y}$) is the data-fit; the term $\frac{1}{2}\log \left| K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I \right|$ is the complexity penalty depending only on the covariance function and the inputs, and $-\frac{n}{2}\log 2\pi$ is a normalization constant. The (log-) marginal likelihood function considers both fitting accuracy and model complexity. So, over-fitting models would be penalized.

To set the hyper-parameters by maximizing the marginal likelihood, the partial derivatives of the marginal likelihood w.r.t. the hyper-parameters is a standard tool:

$$\frac{\partial}{\partial \theta_h} \log p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{2}\mathrm{tr}\left(\left(\alpha_y \alpha_y^T - K_y^{-1} \right) \frac{\partial K_y}{\partial \theta_h} \right) \quad \text{(A.23)}$$

where θ_h is the h-th hyper-parameter in $\boldsymbol{\theta}$. $K_y = K(\boldsymbol{x}, \boldsymbol{x}) + \sigma_{N_d}^2 I$ is the Gramian matrix and $\alpha_y = K_y^{-1}\boldsymbol{y}$.

Appendix B

Convergence Discussion of MO-AMPC

B.1 Proof of Convergence

The first two formulations, *Fully Distributed* and *Virtual Center Node*, strictly follow the ADMM framework, whose convergence has been proven in literature, such as [110]. However, the last formulation of *Local Explicit* is slightly more special since there is an indicator penalty function L_+ showing up in the dual optimization solved by a max operation in Equation 5.22. We found through simulation, and it is also mentioned in the literature that dual convergence is usually significantly slower than primary convergence. Therefore, This section will focus on the dual convergence for the last formulation. A similar proof has been discussed in ([112]) based on the assumption that ρ is constant for all the dimensions in Z. But, this study will discuss the more general situation that does not depend on this assumption.

The main steps of the *Local Explicit* formulation in Equation 5.18, 5.22, and 5.23 could be respectively transformed as,

$$\begin{aligned} U^{j(q+1)} =& -(H^j + \bar{A}_{in}^{j^T}\rho^j\bar{A}_{in}^j)^{-1} \\ & \cdot \left[E^j + \bar{A}_{in}^{j^T}\rho^j(v^{j(q)} - b_{in}^{j(q)})\right] \\ v^{j(q+1)} =& F^{j(q+1)}(\bar{A}_{in}^j U^{j(q+1)} + D^{j(q)}v^{j(q)} - b_{in}^{j(q)}) \\ D^{j(q+1)} =& \frac{1}{2}(I + F^{j(q+1)}) \end{aligned} \tag{B.1}$$

where, $u^{j(q)} := \rho^{j^{-1}}\gamma^{j(q)}$, $v^{j(q)} := u^{j(q)} + \tilde{Z}^{j(q)}$. $d_p^q = \{0,1\}$ are indicator variables that $d_p^q = 0$ if $z_p^{(q)} = 0$ and $d_p^q = 1$ if $z_p^{(q)} = 1$. $F^{(q)} := \text{diag}(\text{sign}((1/m_p)\sum_{j=1}^{m_p}(-\bar{A}_{in}^j U^{j(q+1)} - u^{j(q)} + b_{in}^{j(q)})|_p))$ and $D^{(q)} := \text{diag}(\{d_p^q\})$ are all diagonal matrices. $F^{j(q)}$ and $D^{j(q+1)}$ are the local copy of $F^{(q)}$ and $D^{(q+1)}$, respectively. By substituting the first and third equations into the second and considering $\|v^{j(q)} - v^{j(q-1)}\|_2^2 \leq \|F^{j(q)}v^{j(q)} - F^{j(q-1)}v^{j(q-1)}\|_2^2$, we can get an inequality as:

$$\begin{aligned} &\|F^{j(q+1)}v^{j(q+1)} - F^{j(q)}v^{j(q)}\|_2^2 \\ &\leq \left[\frac{1}{2}(I - 2M^j) + \frac{1}{2}\right]\|(F^{j(q)}v^{j(q)} - F^{j(q-1)}v^{j(q-1)})\|_2^2 \end{aligned} \tag{B.2}$$

where, $M^j = \bar{A}_{in}^j(H^j + \bar{A}_{in}^{j^T}\rho^j\bar{A}_{in}^j)^{-1}\bar{A}_{in}^{j^T}\rho^j$. It is not difficult to expand the proof in [112] that the prime and dual residuals in Equation 5.5 will satisfy the following inequalities if the matrix A_{in} is at least full column-rank:

$$\begin{aligned} \|r^{j(q)}\|_2^2 &\leq \|F^{j(q+1)}v^{j(q+1)} - F^{j(q)}v^{j(q)}\|_2^2 \\ \|s^{j(q)}\|_2^2 &\leq \|\bar{A}_{in}^{j^T}\rho^j\|\|F^{j(q+1)}v^{j(q+1)} - F^{j(q)}v^{j(q)}\|_2^2 \end{aligned} \tag{B.3}$$

The inequalities in Equation B.3 clearly tells us that the sequence $\{\|r^{j(q)}\|_2^2\}$ and $\{\|s^{j(q)}\|_2^2\}$ must converge to 0 if the sequence $\{\|F^{j(q+1)}v^{j(q+1)} - F^{j(q)}v^{j(q)}\|_2^2\}$ could converge to 0. Therefore, let's define the convergence factor ζ^j for Equation B.2 as,

$$\begin{aligned} \zeta^j &:= \frac{\|F^{j(q+1)}v^{j(q+1)} - F^{j(q)}v^{j(q)}\|_2^2}{\|F^{j(q)}v^{j(q)} - F^{j(q-1)}v^{j(q-1)}\|_2^2} \\ &\leq \frac{1}{2}\|I - 2M^j\|_2^2 + \frac{1}{2} \end{aligned} \tag{B.4}$$

Then the residuals will finally converge to 0 if $\zeta^j < 1$. Since ρ^j is a positive-definite diagonal real matrix, it could be decomposed as $\rho^j = L^{j^T}L^j$ by the Cholesky factorization, where L^j is a positive diagonal real upper triangular matrix. Then according to the Woodbury formula of matrix inversion lemma, a polynomial function could be derived as $L^jM^jL^{j^T} = t - t(I + t)^{-1}t$, where $t = L^j\bar{A}_{in}^jH^{j^{-1}}\bar{A}_{in}^{j^T}L^{j^T}$. It's proven in ([134]) that if λ_i^t is one of the eigenvalue of t, then accordingly $\lambda_i^{L^jM^jL^{j^T}} = \lambda_i^t - \lambda_i^t(1 + \lambda_i^t)^{-1}\lambda_i^t = \lambda_i^t/(1 + \lambda_i^t)$ should be an eigenvalue of $L^jM^jL^{j^T}$. Furthermore, it's easy to prove that matrix M^j and $L^jM^jL^{j^T}$ have the same eigenvalues by multiplying both sides of the equation $L^jM^jL^{j^T}x_i = \lambda_i^{L^jM^jL^{j^T}}x_i$ by $L^{j^{-1}}$, where x_i is the eigenvector. Since H^j is positive-define, the eigenvalue of M^j satisfies:

$$\lambda_i^{M^j} = \frac{\lambda_i^t}{1 + \lambda_i^t} \in [0, 1),\ t = L^j\bar{A}_{in}^jH^{j^{-1}}\bar{A}_{in}^{j^T}L^{j^T} \tag{B.5}$$

This result is the same as in [112], but the condition is expanded to no constant assumption on ρ^j, which can be a positive definite diagonal matrix with different values on its diagonal elements. Then [112] has proven that the 0-eigenspace of M^j corresponds to the stationary points of the primal optimization and could be disregarded if $\bar{A}_{in}^j$ is full column-rank, which makes the convergence result holds by,

$$\|I - 2M^j\|_2^2 < 1,\ \zeta^j \in (\frac{1}{2}, 1) \tag{B.6}$$

Finally, the convergence of the residuals sequences $\{\|r^{j(q)}\|_2^2\}$ and $\{\|s^{j(q)}\|_2^2\}$ follows from the convergence of $\{\|F^{j(q+1)}v^{j(q+1)} - F^{j(q)}v^{j(q)}\|_2^2\}$.

B.2 Optimal Parameter Selection

ρ^j is an important parameter in the proposed multi-objective AMPC because it affects the speed of convergence. For the first two formulations, it is difficult to have a deterministic method to determine the value of ρ^j, which is usually selected based on experience. However, for the last formulation - local explicit, there can be an optimal rule for choosing ρ^j since the upper limit of the convergence speed can be expressed explicitly by the convergence factor ζ^j in Equation B.4. Therefore, the selection of ρ^j can be regarded as an optimization problem that minimizes the convergence factor ζ^j as,

$$\rho_*^j = \arg\min_{\rho^j} \zeta^j(\rho^j) = \arg\min_{\rho^j} \left\{ \max_i \left| \frac{\lambda_i^t}{1+\lambda_i^t} - \frac{1}{2} \right| + \frac{1}{2} \right\} \tag{B.7}$$

Since the function $l(\lambda_i) = \lambda_i/(1+\lambda_i)$ is a monotonically increasing function in $\lambda_i > 0$, the maximum values exist for the two extreme non-zero eigenvalues λ_1^t and λ_n^t. Besides, the function of the convergence factor $\zeta^j(\lambda_i^t)$ is monotonically increasing when $\lambda_i^t > 1$ and decreasing when $0 < \lambda_i^t < 1$. Thus, $\lambda_1^t \leq 1 \leq \lambda_n^t$. So the convergence factor in Equation B.7 can be written as,

$$\zeta^j(\rho^j) = \max\left\{ \frac{1}{1+\lambda_1^t}, \frac{\lambda_n^t}{1+\lambda_n^t} \right\} \tag{B.8}$$

where $\lambda_i^t = \lambda_i(L^j \bar{A}_{in}^j H^{j^{-1}} \bar{A}_{in}^{j^T} L^{j^T}) = \lambda_i(\bar{A}_{in}^j H^{j^{-1}} \bar{A}_{in}^{j^T} \rho^j)$. In fact, there is no direct equation to derive the ρ_*^j. Therefore, let's add a sufficient condition that all the diagonal elements of ρ^j are the same constant, i.e. $\rho^j = \rho^j \cdot I$, to achieve a near-optimal convergence speed. Then, the left brace of the max in Equation B.8 monotone decreases in ρ^j, and the right is

monotone increasing. Thus, the minimum with respect to ρ^j happens at the intersection point: $1/(1+\rho_*^j\lambda_1^{t'}) = \rho_*^j\lambda_n^{t'}/(1+\rho_*^j\lambda_n^{t'})$, where $t' = \bar{A}_{in}^j H^{j^{-1}} \bar{A}_{in}^{j^T}$. That is:

$$\rho_*^j = \frac{1}{\sqrt{\lambda_1(\bar{A}_{in}^j H^{j^{-1}} \bar{A}_{in}^{j^T})\lambda_n(\bar{A}_{in}^j H^{j^{-1}} \bar{A}_{in}^{j^T})}} \tag{B.9}$$

Appendix C

Parameters of the Chevrolet Equinox Electric Vehicle

The test vehicle in this study is the Chevrolet Equinox electric vehicle, as shown in Figure C.1, whose main parameters are specified in Table C.1. In addition to driver-controlled drive, braking and steering, this test vehicle also enables independent differential torque control for the front and rear axles, as torque can be individually controlled for all four wheels, as well as an additional active front steering input. At the same time, this vehicle also provides a wealth of sensor interfaces, including GNSS, IMU, wheel speed, etc., to facilitate the provision of more accurate reference speed estimation and other key signals through sensor fusion.

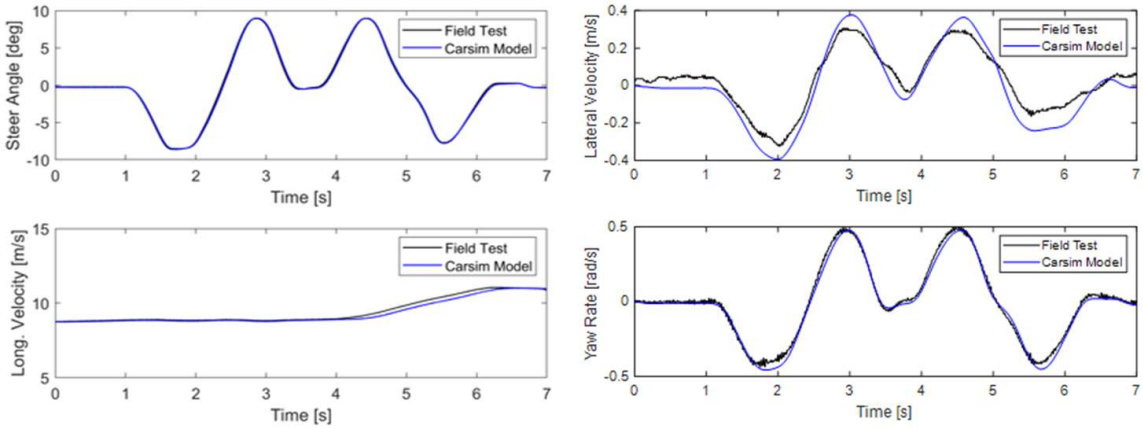


Figure C.1: Test vehicle: the Chevrolet Equinox electric vehicle.

Table C.1: Specification of the Test Vehicle

Parameter	Description	Value
m	Vehicle mass	2272 kg
I_z	Vehicle inertia	4600 kg $\cdot$ m^2
I_w	Wheel inertia	34 kg $\cdot$ m^2 m
l_f	Front distance from axles to CG	1.42 m
l_r	Rear distance from axles to CG	1.43 m
L_b	Wheel base	2.85 m
R_w	Effective radius of wheels	0.351 m
K_{us}	Desired under-steering gradient	0.005 s^2/m
T_w	Wheel track	1.66 m
$C_{\alpha f}, C_{\alpha r}$	Cornering stiffness of tires	130000 N/rad

A high-fidelity CarSim model was built for this vehicle and used for simulation. The accuracy of the CarSim model was verified using experimental data obtained on the Equinox vehicle under a DLC maneuver, shown in Figure C.2.



(a) Road wheel angle and longitudinal velocity

(b) Lateral velocity and yaw rate

Figure C.2: Equinox CarSim model verification under a DLC maneuver.

Plot (a) shows the road wheel steering angle and the longitudinal velocity applied to the field test and the CarSim model simulation, and Plot (b) shows the lateral velocity and yaw rate responses. The experimental and CarSim simulation results fit well for both lateral velocity and yaw rate, indicating the sufficient accuracy of the CarSim model.