Advances in Game-Theoretic, Set-Theoretic and Optimal Control to Enhance Mobility

by

Huayi Li

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in the University of Michigan
2022

Doctoral Committee:

Professor Anouck R. Girard, Co-Chair
Professor Ilya V. Kolmanovsky, Co-Chair
Associate Professor Shan Bao
Dr. Kenneth R. Butts, Toyota Motor North America

Huayi Li

huayil@umich.edu

ORCID iD: 0000-0002-3462-3965

To Mom and Dad.

ACKNOWLEDGMENTS

I would like to begin by acknowledging Professor Anouck Girard and Professor Ilya Kolmanovsky, who I feel extremely lucky to have as advisors. Thank you for your patience, advice, and all the opportunities provided.

I would like to express my appreciation to my dissertation committee member Dr. Ken Butts. My career would be totally different without your mentorship at Toyota and support during my Ph.D. study.

Thank you Professor Shan Bao for your advice during the Mcity project and for serving on my dissertation committee.

I would like to thank Mcity and the National Science Foundation (award number ECCS-1931738) for funding my study, and the Department of Aerospace Engineering at the University of Michigan for the financial support through the Mr. and Mrs. Milo E. Oliphant Fellowship. I would also like to thank TMC and TMNA (Toyota) for offering technical support. Thank you Mr. Tony Kim, Dr. Dominic Liao-McPherson, and Dr. Nan Li for your collaboration and help during my research study.

I wish to acknowledge all my labmates for bringing joy to this journey. Many thanks to Nan, Dominic, Will, and AJ for helping me practice prelim. I would also like to thank Bin, Zhe, and Qian, as well as Mike and Cherry for the friendship.

Deep thanks to everybody in my pandemic pod for being supportive and navigating through this difficult time together.

Finally, to my parents, thank you for your love.

TABLE OF CONTENTS

Dedication . . . . . **ii**

Acknowledgments . . . . . **iii**

List of Figures . . . . . **vii**

List of Tables . . . . . **ix**

List of Algorithms . . . . . **x**

Abstract . . . . . **xi**

Chapter

1 Introduction **1**

1.1 Background and Motivations 1
1.2 Challenges and Strategies 3
1.2.1 Fault-Tolerant Control for Mobility Systems Using Set-Theoretic Methods 3
1.2.2 Energy-Efficient Driving for Autonomous Vehicles in Game-Theoretic Traffic Environment 6
1.2.3 Acceleration Performance Analysis with Jerk Limits Using Trajectory Optimization 7
1.3 Dissertation Outline 9
1.4 Summary of Novel Contributions 10

2 Failure Mode Reconfiguration Based on Constraint Admissible and Recoverable Sets for Zeroed-Out Actuators **11**

2.1 Introduction 11
2.2 Mode-Dependent System Dynamics, Constraints and Problem Formulation . . . 13
2.2.1 System Dynamics 13
2.2.2 Constraints 14
2.2.3 Problem Formulation 15
2.3 Constraint Admissible and Recoverable Sets 15
2.3.1 Constraint Admissible Sets 15
2.3.2 Recoverable Sets 15
2.4 Employing Admissible and Recoverable Sets for Safe Operation and Reconfiguration 16
2.4.1 Safe Operation in Each Mode Using Reference Governor 16
2.4.2 Safe Reconfiguration Upon Failure Detection 16

2.4.3 Command Range and Rate Limiting . . . 17
2.5 Mass-Spring-Damper System Example . . . 18
2.5.1 System Dynamics, Mode Definitions and Constraints . . . 19
2.5.2 FMEM Strategy Design . . . 20
2.5.3 Simulation Results . . . 21
2.6 Aircraft Longitudinal Flight Control Example . . . 22
2.6.1 System Dynamics, Mode Definitions and Constraints . . . 23
2.6.2 FMEM Strategy Design . . . 24
2.6.3 Simulation Results . . . 26
2.7 Concluding Remarks . . . 27

3 Set-Theoretic Failure Mode Reconfiguration for Stuck Actuators . . . 29

3.1 Introduction . . . 29
3.2 Operation Modes, System Dynamics, and Constraints . . . 32
3.2.1 Normal and Failure Modes . . . 32
3.2.2 Open-Loop and Closed-Loop System Models . . . 33
3.2.3 Constraints . . . 36
3.3 Reconfiguration Strategy . . . 36
3.3.1 Constraint Admissible Sets . . . 37
3.3.2 Recoverable Sets . . . 37
3.3.3 Reconfiguration Conditions . . . 38
3.3.4 Safe Reconfiguration Upon Failure Detection . . . 38
3.3.5 Reference Governor . . . 39
3.4 Application to Aircraft Longitudinal Flight Control . . . 40
3.4.1 System Dynamics . . . 40
3.4.2 Constraints . . . 41
3.4.3 Design of Constraint Admissible and Recoverable Sets . . . 41
3.4.4 Simulation Results . . . 43
3.5 Concluding Remarks . . . 43

4 Integrating Failure Detection and Isolation into a Reference Governor-Based Reconfiguration Strategy for Stuck Actuators . . . 45

4.1 Introduction . . . 45
4.2 Preliminaries: Operating Modes, System Dynamics and Constraints . . . 47
4.2.1 Operating Modes . . . 47
4.2.2 System Dynamics . . . 47
4.2.3 Constraints . . . 49
4.3 Online FDI and Failure Mode Reconfiguration Process . . . 51
4.3.1 Structure Overview . . . 51
4.3.2 Phase 0: Reference Tracking and Failure Detection . . . 51
4.3.3 Phase 1: Failure Isolation . . . 53
4.3.4 Phase 2: Failure Reconfiguration . . . 56
4.4 Conditions for Guaranteed Failure Isolation and Reconfiguration . . . 56
4.4.1 Isolation Conditions . . . 56
4.4.2 Reconfiguration Conditions . . . 57

4.5 Numerical Example . . . 58
4.5.1 System Dynamics and Operating Modes . . . 58
4.5.2 Constraints . . . 59
4.5.3 Offline Design for Guaranteed Isolability and Reconfigurability . . . 59
4.5.4 Simulations . . . 60
4.6 Concluding Remarks . . . 61

5 Energy-Efficient Autonomous Driving Control Using Game-Theoretic Interactive Traffic Models and Reinforcement Learning . . . 64

5.1 Introduction . . . 64
5.2 Lane Changes for Energy-Efficient AV Driving . . . 66
5.3 Powertrain Modeling for Battery Electric Vehicles . . . 67
5.3.1 Model Description . . . 67
5.3.2 Model Calibration and Validation . . . 70
5.4 Controller Design . . . 72
5.4.1 Game-Theoretic Traffic Environment . . . 72
5.4.2 Observation and Action Spaces . . . 72
5.4.3 Reward Function . . . 73
5.4.4 Training Algorithm . . . 75
5.4.5 Training Process . . . 76
5.4.6 Autonomous Vehicle Control Policy for Benchmarking . . . 77
5.5 Results . . . 77
5.5.1 Training for RL-Based Policies . . . 77
5.5.2 Control Performance . . . 78
5.6 Concluding Remarks . . . 83

6 Analysis of Multistage Hybrid Powertrains Using Multistage Mixed-Integer Trajectory Optimization . . . 85

6.1 Introduction . . . 85
6.2 Powertrain Modeling . . . 87
6.3 Multistage Mixed-Integer Optimization . . . 92
6.4 Application to a Multistage Hybrid Powertrain . . . 95
6.5 Case Studies . . . 97
6.5.1 Without Jerk Limits . . . 98
6.5.2 With Maximum Jerk Limits . . . 98
6.6 Concluding Remarks . . . 99

7 Conclusions and Future Work . . . 103

Bibliography . . . 107

LIST OF FIGURES

FIGURE

1.1 Transportation energy consumption by mode and type [8]. 2
1.2 Highlights of study areas, strategies and applications of this dissertation. Each background color represents a specific topic. 3
1.3 Problem and assumptions made in the development of Failure Mode and Effect Management strategies. 4
1.4 Demonstration of set-membership relations for failure isolation and reconfiguration. . 5
1.5 Highlights of the reinforcement learning process for training of the energy-efficient autonomous driving control policy. 8

2.1 Possible modes of a system with two actuators labeled by 1 and 2. The numbers in the boxes are the labels of the actuators that work properly. M stands for mode. 12
2.2 Mass-spring-damper system with position and force magnitude constraints and redundant actuation. 19
2.3 Projections of sets and simulation results for the first case. 22
2.4 Projections of sets and simulation results for the second case. 23
2.5 Projections of admissible and recoverable sets as N_M changes. 25
2.6 Projections of admissible sets as η_O changes. 25
2.7 Projections of admissible and recoverable sets as η_{R_M} changes. 26
2.8 Simulation results with failure path from Mode 0 to Mode 1 to Mode 3. 27
2.9 Simulation results with failure path from Mode 0 to Mode 2 to Mode 3. 28

3.1 A flowchart of the proposed FMEM strategy. Signals in round brackets are passed from previous blocks. 31
3.2 The relation between offline designed constraint admissible and recoverable sets exploited in the FMEM strategy. 32
3.3 Example of failure paths, modes M, and labels ℓ_M for $N = 2$. 33
3.4 Aircraft example simulation results (initial states are picked such that recovery is needed). 44

4.1 Structural overview of the online FDI and failure mode reconfiguration strategy. The blocks of Controller and Plant represent the banks of controllers and open-loop system dynamics for different modes. 52
4.2 Isolation condition (4.27) satisfied for $M_0 = 0$. 60
4.3 Reconfiguration conditions (4.34) satisfied for $M_0 = 0$. 61
4.4 State projections of constraint admissible sets and simulated state trajectory. 62

4.5 Simulation results, where M is the actual mode, and M_{est} stands for the mode determined by the FMEM unit. 63

5.1 Battery electric vehicle powertrain system layout. 68
5.2 Time histories of powertrain signals for the UDDS cycle. 71
5.3 Average reward evolution during RL. 78
5.4 Evaluation results for AV control policies in traffic environments of different traffic densities: (a) Constraint violation rate; (b) Average number of lane changes per simulation episode; (c) Average $MPGe$; (d) Average travel speed. 80
5.5 Constraint violation rates for level-k and the proposed AV control policies in traffic environments of different traffic densities. 81
5.6 Average number of lane changes per simulation episode and average travel speed for level-k and the proposed AV control policies in traffic environments of different traffic densities. 82

6.1 The layout of the multistage hybrid powertrain. The powertrain system consists of the motor/generator 1 (MG1), motor/generator 2 (MG2), engine, driveline compliance, 4-speed automatic transmission (4AT/g1), differential (g2), and wheels and tires. . . . 86
6.2 Planetary gear set geometry and kinematics abstraction. The planetary gear set consists of sun gear (orange), ring gear (red), carrier (blue), and pinion gears (green). 88
6.3 Trade-off between acceleration duration and maximum jerk limit. 97
6.4 Simulation with optimal control inputs and without jerk constraint. 100
6.5 Simulation with optimal control inputs and with maximum jerk limit of $40\,\mathrm{m/s^3}$. . . . 101
6.6 Simulation with optimal control inputs and with maximum jerk limit of $20\,\mathrm{m/s^3}$. . . . 102

7.1 Integrating the energy-efficient autonomous driving control policy into a typical autonomous driving architecture. The proposed control policy maps the observations to actions for trajectory planning. 105

LIST OF TABLES

TABLE

5.1 Summary of selected literature on using CAV technologies to improve energy efficiency 65
5.2 Comparison summary on control development for the level-k ($k = 1, 2$) policies and the autonomous vehicle policy considered in this work (AV) 73

LIST OF ALGORITHMS

ALGORITHM

5.1 Training process 77
5.2 Evaluation process 79

6.1 GenerateAllSequences 95

ABSTRACT

Advances in mobility technology are expected to greatly improve our access to transportation and the convenience of life. Many challenges in achieving the safety, efficiency, and comfort of mobility systems still exist. This dissertation leverages advances in modeling and control to highlight several solutions to these challenges.

In particular, for safety-critical applications, it is important to guarantee that the system can operate safely anytime even with the occurrence of failures; hence there is a need for systematic approaches to achieving fault-tolerant control (FTC). Fault tolerance must be ensured in multiple ways: for failure detection and isolation (FDI), system reconfiguration/recovery, and safe operation in all normal and failure modes. Meanwhile, the system availability should be maximized instead of being compromised. In this dissertation, this problem is approached through set-theoretic methods by exploiting nesting between constraint admissible, recoverable, and isolatable sets to satisfy state and control constraints in all phases of failure detection, isolation, reconfiguration, and transition to operation in failure mode. The proposed FTC framework integrates a reference governor for safe reference tracking, and the framework can handle stuck/jammed actuators that are common in mobility applications. An aircraft longitudinal flight control example is used for demonstration.

The second advance in this dissertation is an eco-driving approach for connected and autonomous vehicles (CAVs) that accounts for the interactive vehicle behavior in traffic. By exploiting and fusing game theory, reinforcement learning (RL), and battery electric vehicle (BEV) powertrain modeling and control, significant opportunities for energy efficiency improvements are highlighted. Specifically, a level-k game-theoretic traffic model is considered to capture the interactions between the ego vehicle and the surrounding human-driven vehicles. Also, a model that represents a BEV is established, and a control policy for operating the vehicle in traffic is trained by RL applied to

the reward function that reflects energy efficiency, travel time efficiency, safety, and comfort. The control policy can command both longitudinal and lateral actions, including acceleration, braking, and lane changes, to enhance the possibilities for energy saving.

Finally, the third advance in this dissertation is the development of a trajectory optimizer for multistage hybrid powertrain capable of handling both continuous (engine and motor torques) and discrete (engine on/off or gear selection) degrees of freedom. For driving scenarios prioritizing drivability, the shortening of acceleration duration is often limited by jerk since sudden changes of acceleration can cause discomfort to passengers. Specifically, a 40 to 70 kph acceleration event is considered to simulate a passing maneuver for a multistage hybrid electric vehicle (HEV). Case studies with different jerk limits are set up to investigate the trade-off between acceleration duration and jerk, and the acceleration performance is optimized by solving a constrained trajectory optimization problem formulated as a parallelizable multistage mixed-integer problem. The proposed approach, therefore, addresses a complex optimal control problem in the area of vehicle drivability/rideability that is critical for customer acceptance of energy-efficient and autonomous vehicles. The developed methodology for efficient numerical trajectory optimization with continuous and integer variables can be applied to other engineering domains.

Overall, this dissertation highlights opportunities for applications of game-theoretic, set-theoretic and optimal control towards the development of safer, more efficient and comfortable future mobility systems. The contributions address the gaps in safety-guaranteed FTC, safe and energy-aware CAV control in shared traffic, as well as in unbiased and easy-to-implement drivability analysis processes.

CHAPTER 1

Introduction

1.1 Background and Motivations

Traditionally, we refer to moving people and items by cars, trains, airplanes and ships as transportation [1]. In recent years, however, the word mobility has become more common. This is not simply a shift in the language because mobility means a lot more than moving objects from point A to point B. In fact, mobility also includes considerations such as equity, accessibility, and demand satisfaction [2]. Mobility enables the efficient delivery of supplies such as food and medicine to our doorsteps. In addition, with the growing penetration of connected and autonomous vehicle (CAV) technologies, land travel becomes easily accessible to people who are not capable of driving [3].

To achieve improvements in mobility, many challenges need to be addressed. This dissertation considers control issues for mobility applications with a focus on safety, efficiency, and comfort.

There are many aspects to the safety of mobility systems. The one addressed in this dissertation relates to situations when malfunctions occur. In such cases, these systems should have the ability to continue operation without violating safety constraints while maintaining (to the possible extent) the system function. Failure to do so may not only cause damage to the system and cargo but also cause injury to people and even fatalities [4, 5]. In this dissertation, we consider fault-tolerant control (FTC) approaches for Failure Mode and Effect Management (FMEM) in Chaps. 2 to 4. We illustrate these developments with an application to aircraft longitudinal flight control with redundant actuators where some of the actuators may fail in flight. The general theory that is developed can be useful in other applications, e.g., electric vehicles with redundant motors [6], but their treatment is left to continuing research.

In terms of efficiency and comfort, this dissertation focuses on applications in the automotive domain. The data in [7] by the U.S. Department of Transportation shows that the average American driver traveled 13,476 miles in 2018, which implies on average 37 miles and roughly between 30 minutes and an hour of daily driving per driver. Consequently, improving travel time efficiency and comfort can have a significant societal impact. Additionally, about 26% of the total U.S. energy

consumption in 2020 is for transportation where the operation of light-duty vehicles and heavy-duty trucks accounts for over 77% of the transportation energy consumption (see Fig 1.1 by [8]). Thus, the energy efficiency of automotive applications has a great impact on total energy consumption.

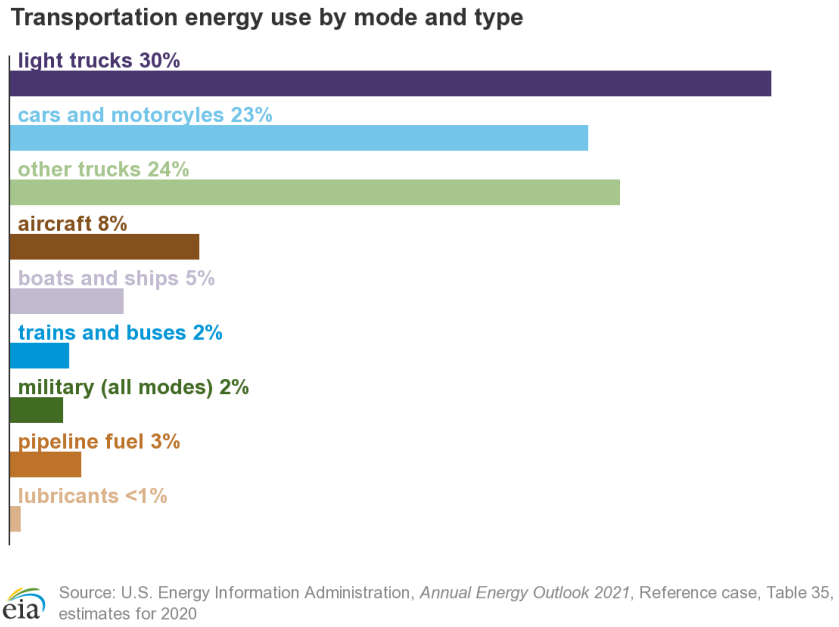

Figure 1.1: Transportation energy consumption by mode and type [8].

The combination of connected and autonomous driving and vehicle electrification is expected to bring significant environmental benefits by reducing energy consumption and emissions [9, 10]. However, approaches to balance safety (e.g., by avoiding collisions) and travel time efficiency with energy consumption reduction are not thoroughly understood [11]. These issues are further considered and addressed in Chap. 5 for the battery electric vehicles (BEVs).

Comfort and drivability are also important areas to customer acceptance of energy-efficient vehicles. Although comfort depends on many aspects such as interior design and convenience features, this dissertation seeks to improve the comfort of driving and riding through control that limits jerk during the acceleration because sudden changes of the longitudinal motion are commonly detectable by passengers and often cause discomfort. In addition, trade-offs exist between the acceleration performance and the jerk constraints [12]. Being able to perform the trade-off analysis is necessary for the powertrain system development, especially during the early design phase. In Chap. 6, we consider a 40 to 70 kph acceleration scenario (simulating a passing maneuver) and study the trade-offs between the minimum acceleration time and the imposed jerk limits. A hybrid electric vehicle (HEV) powertrain is considered since electrified vehicles are promising for the reduction of environmental concerns as mentioned previously and the powertrain configurations of HEVs are usually more complex than those of the other types of vehicles, e.g., BEVs.

1.2 Challenges and Strategies

Figure 1.2 highlights the development areas, methodologies and mobility applications for the three specific topics considered in this dissertation. In the following sections, we discuss the challenges and strategies for these topics.

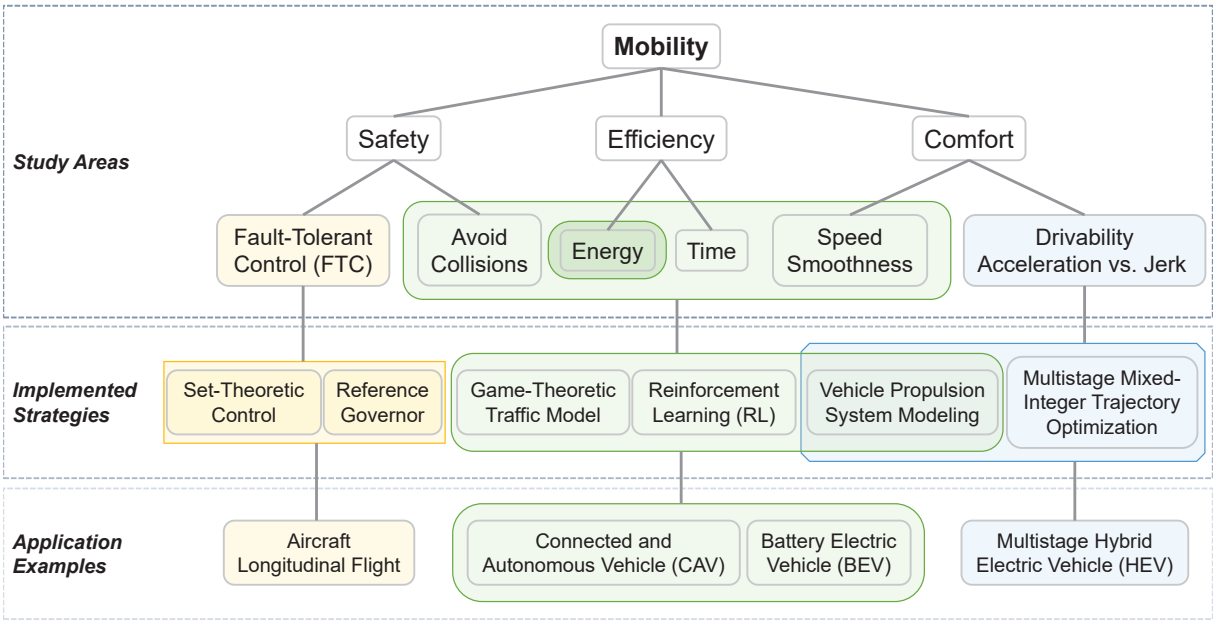

Figure 1.2: Highlights of study areas, strategies and applications of this dissertation. Each background color represents a specific topic.

1.2.1 Fault-Tolerant Control for Mobility Systems Using Set-Theoretic Methods

Figure 1.3 highlights the fault-tolerant control (FTC) problem considered in this dissertation. There are multiple ways to categorize faults and failures of a system. One way is to classify them into communication, software, and hardware failures. In the hardware category, actuator failures are among the most common ones, and failures due to stuck actuators are typical in mobility applications (e.g., [4, 5]). Systems such as those in vehicles and aircraft often have redundancy and operate in constrained environments while tracking reference commands. Even when failures occur, these systems must operate without safety constraint violation at any time. Thus, our FTC strategy should ensure that, in case of actuator failures, the system can still operate safely while maximizing its availability. Specifically, a Failure Mode and Effect Management (FMEM) strategy is needed to handle stuck/jammed actuators that lead to the system operating mode transitions caused by

such failures. The strategy should guarantee that the failure can be isolated and the system can be reconfigured within a certain number of time steps upon failure detection without violating safety constraints while performing reference tracking.

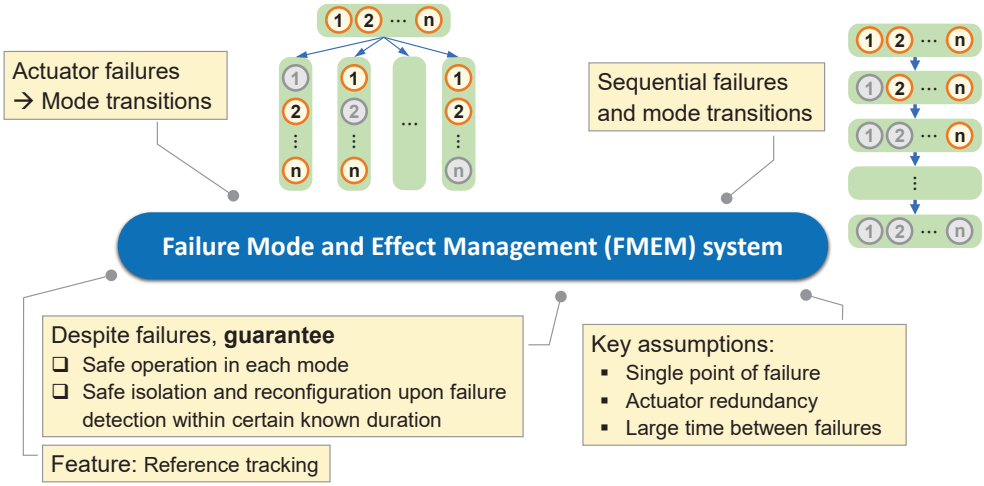

Figure 1.3: Problem and assumptions made in the development of Failure Mode and Effect Management strategies.

There is much literature that considers FTC and reconfigurability analysis, e.g., [13–15], yet safety constraints and reference tracking are usually not addressed. The systems that we study operate with constraints while trying to minimize the tracking errors. Such a problem falls into the category of constrained control and it can be handled by reference governors (see [16] and references therein). Set-theoretic methods, e.g., in [17–20], can be further applied to guarantee constraints satisfaction. Our proposed FTC strategy is inspired by [21] where set-theoretic failure reconfiguration is used for safe trim point to trim point transitions, though [21] does not consider sequential failures. Different methods for FTC, such as the ones based on fault-tolerant model predictive control (FTMPC), have been proposed, e.g., in [22, 23]; however, many of them are designed to address only the stabilization setting, and they may not be able to guarantee that the MPC problem is feasible for all operating scenarios. Lastly, instead of using a systematic FMEM strategy, fault detection and isolation (FDI) and failure mode reconfiguration are often handled separately due to otherwise large problem complexity [14, 24].

In Chaps. 2, 3, and 4, a reference governor-centric set-theoretic FMEM framework is developed to handle stuck/jammed actuators. A mass-spring-damper system and an aircraft longitudinal flight control system are used as application examples. In particular, the proposed FMEM framework exploits nesting between sets to guarantee the feasibility of the online constrained optimization problems for the reference governor, isolation, and reconfiguration. We provide an overview of this strategy here. Firstly, sets of states and reference commands that are constraint admissible,

isolatable, and recoverable are defined. Then, during the online operation (as shown in Fig. 4.1), a reference governor, which is based on a solution to a quadratic programming problem, is applied to track reference commands using the working actuators while imposing constraints informed by the constraint admissible set (in Phase 0). Upon failure detection, an isolation sequence of the open-loop input values is first generated to isolate the failure mode (in Phase 1), and then a recovery sequence of reference commands is generated for failure mode reconfiguration (in Phase 2). Both sequences are generated by solving optimization problems with constraints informed by the isolatable and recoverable sets. Finally, the reference governor is re-activated once the isolation and reconfiguration are finished and the process returns to Phase 0. Note that the above-mentioned optimization problems during the online process may not always have solutions. To guarantee feasibility, by the offline design, we define set-membership conditions and make sure they are satisfied through three mechanisms that manipulate the size of the imposed constraint sets. The idea of nesting between sets is that, at the end of the preceding phase, the system should already be operating in the feasible set (e.g., an isolatable set or a recoverable set) of the next phase, so that once the transition happens, the optimization problem to be solved (e.g., to generate the isolation sequence or recovery sequence) is guaranteed to have a solution (see Figs. 1.4 and 3.2 for reference).

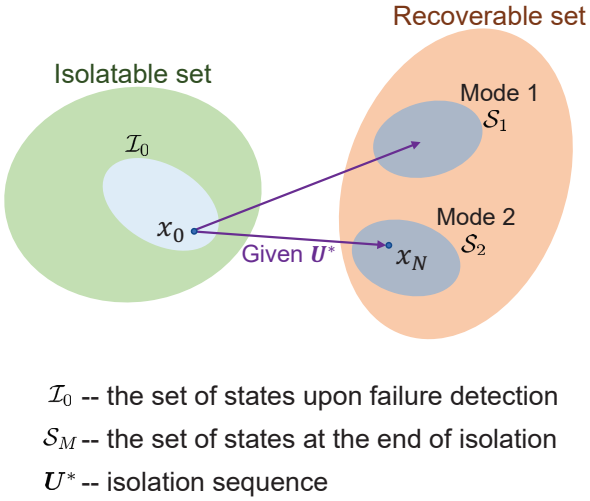

Figure 1.4: Demonstration of set-membership relations for failure isolation and reconfiguration.

Compared with existing methods in the literature, the proposed FTC strategy is distinguished by having the ability to safely perform reference tracking in both normal and failure modes through the use of a reference governor. In addition, through the set-membership conditions enforced in the offline design, our FMEM system explicitly guarantees the isolatability and reconfigurability, and the time durations for the isolating and reconfiguration processes are finite and known prior to the online operation.

For illustration, this set-theoretic FTC strategy is successfully implemented to control the

displacement of a mass-spring-damper system as well as for the longitudinal flight control of a Boeing 747-100 aircraft. Chapters 2, 3, and 4 of this dissertation are written based on the preliminary journal and conference versions in [25–27].

1.2.2 Energy-Efficient Driving for Autonomous Vehicles in Game-Theoretic Traffic Environment

Chapter 5 considers developing an autonomous driving policy focusing on the optimization of energy efficiency. There have been many studies in recent years on using CAV technologies to reduce energy consumption. In the Advanced Research Projects Agency-Energy's (ARPA-E) Next-Generation Energy Technologies for Connected and Automated On-Road Vehicles (NEXTCAR) program by the U.S. Department of Energy, an over 20% of energy saving potential has been demonstrated through connected and automated driving (see Table 5.1 and [28] for reference). The developments in this dissertation are motivated by two considerations. Firstly, in the foreseeable future, CAVs and human-driven vehicles will be operated together in the shared traffic [29]. Hence CAVs will need to account for the surrounding vehicle actions and reactions caused by different types of human driving styles. Secondly, many existing studies considered energy efficiency improvements only in the longitudinal car following situations. Adding lateral control, e.g., lane changing, can provide additional possibilities for energy savings [30]. At the same time, the problem complexity also increases.

When driving, most of the energy waste comes from braking. Consequently, common strategies to improve energy efficiency include route planning and driving as smoothly as possible, particularly, to avoid braking at any time [11]. However, this is not always feasible in real-world driving conditions. In particular, subtle trade-offs exist between collision avoidance and energy saving. One may need to choose lane changing or braking in situations when the ego vehicle (i.e., the vehicle whose behavior is of primary interest in the analysis/evaluation) is approaching a slow vehicle in its front, and whether the decision is good or bad depends on the interactions with the surrounding vehicles. In addition, handling safety and energy efficiency objectives requires considering different time horizons. In regards to safety constraint violations, such as collisions, relevant events typically occur within a time duration of seconds, but meaningful optimization of energy consumption requires much longer time horizons. For example, the Federal Test Procedure (FTP) used by the U.S. Environmental Protection Agency (EPA) for vehicle emissions and fuel economy testing has a 1874-second duration and a travel distance of approximately 11 miles [31].

To address these considerations, the following approaches are used to develop an energy-efficient autonomous driving policy in Chap. 5. Firstly, a level-k game-theoretic interactive highway traffic simulator proposed in [32] (following the methodology originally proposed in [33]) is used to

facilitate the training of the autonomous driving policy. In a level-k game, the level-0 players make decisions without considering other players, while the level-k players for $k > 0$ take the best action assuming everyone else is level-$(k-1)$. This kind of game can be used to model the cognitive behaviors of vehicles driven by human drivers in traffic. Our game-theoretic traffic simulator contains three levels of players with a certain mixed ratio. The level-0 policy uses a heuristic policy while the level-1 and level-2 policies are trained by reinforcement learning (RL).

Secondly, a battery electric vehicle energy model is constructed based on the powertrain and driveline dynamics, and the model is integrated into the interactive traffic simulator to provide explicit information about energy consumption. A backward model, that is, a model that determines battery energy consumption based on wheel traction power, is used due to its low computational complexity; this backward model is verified versus a more detailed forward model (see Remark 5.1 for reference).

Finally, the control policy is trained by RL (see Fig 1.5) using 13 observations that contain vehicle speed and battery state of charge, 7 actions that include both longitudinal actions and lane changes, and the reward function that considers collision rate, vehicle speed, headway, control effort, and energy efficiency. Note that the observations include the relative distance and vehicle speed only between the ego vehicle and its immediate surrounding vehicles. This information is commonly available through the sensors (e.g., Lidars and cameras) on board so the use of vehicle connectivity (e.g., vehicle-to-vehicle communications) is not necessary. Also, note that we choose the Jaakkola RL algorithm [34] for policy training because this algorithm considers both the immediate one-step reward and the infinite-horizon average reward so that optimization objectives with different time horizons can be handled simultaneously.

To summarize the work of Chap. 5, an energy-efficient autonomous driving policy that considers both longitudinal and lateral actions is developed where the safety, efficiency, and comfort are simultaneously optimized. A level-k game-theoretic traffic simulator is used to model the interactions with human-driven vehicles, and a BEV model is developed to provide explicit energy consumption information. The results in Chap. 5 have appeared in [35, 36].

1.2.3 Acceleration Performance Analysis with Jerk Limits Using Trajectory Optimization

As discussed in Sect. 1.1, electrified vehicles hold significant promise for addressing environmental concerns. Hybrid electric vehicles (HEVs) are, in particular, an attractive choice as they do not cause range anxiety concerns. At the same time, HEV powertrains get increasingly complex to satisfy the demand for various features. The power-split configuration of the 2018 Lexus LC-500h (see Fig 6.1), for example, consists of an engine and two motors that are coupled by a planetary

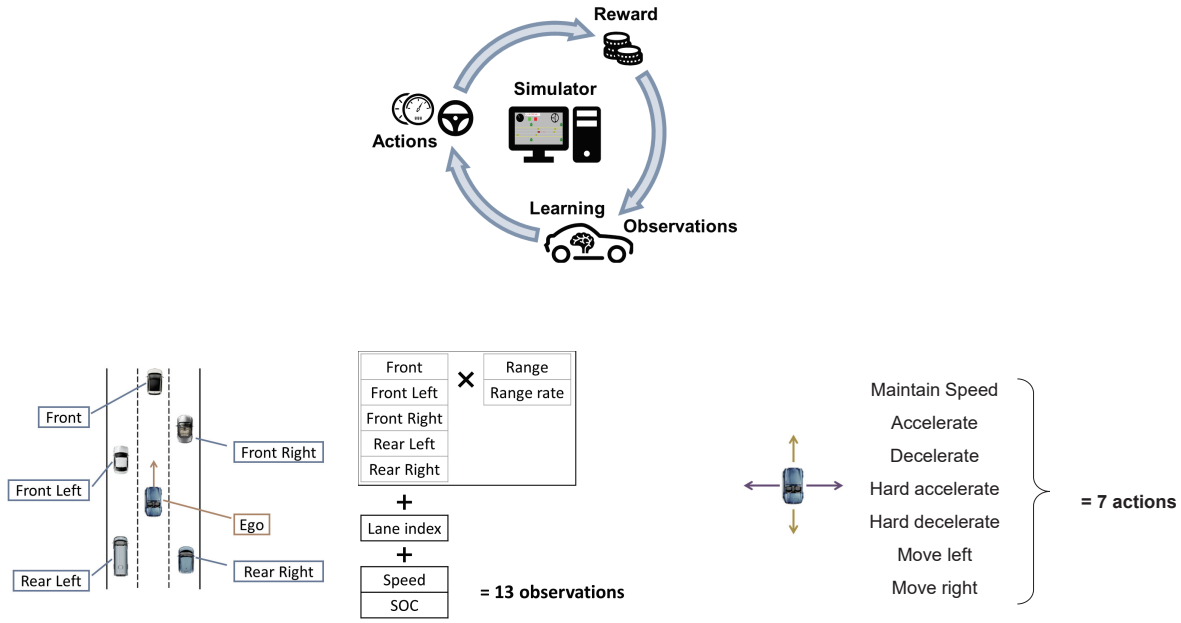

Figure 1.5: Highlights of the reinforcement learning process for training of the energy-efficient autonomous driving control policy.

gear set, in addition to a four-speed automatic transmission located between the differential and the Toyota Hybrid System (THS), i.e., the system that contains the engine, motors, and the planetary gears set [37]. Such powertrains are over-actuated and have high degrees of control freedom. Moreover, unlike the inputs that are continuous such as the engine and motor torques, the engine on/off selection and gear position are discrete signals. As a result, the need to find the optimal control inputs for jerk limited acceleration, which is an important scenario for drivability evaluation, leads to a constrained nonlinear mixed-integer problem that is difficult to solve reliably.

The traditional approach to powertrain control involves using pre-defined strategies and lookup tables [38]. This method is commonly adopted due to its low computational complexity. However, acquiring these lookup tables requires calibration that is labor-intensive and the optimality may not be guaranteed. Instead, trajectory optimization can be used to find the optimal control inputs [39,40]. Although trajectory optimization methods, such as the ones based on dynamic programming (DP) and Pontryagin's maximum principle (PMP), are typically not used for real-time control because they require extensive computational resources and explicitly known information of the driving cycle, they are suitable for offline analysis especially during the early stage of powertrain design, e.g., when comparing different parallel and power-split HEV configurations with several choices of engines and motors. The approach proposed in Chap. 6 of this dissertation instead exploits multistate mixed-integer nonlinear programming.

In Chap. 6, we describe a hybrid powertrain model that can simulate the 40 to 70 kph acceleration

scenario and calculate the jerk. Based on this model, we consider a case study that illustrates a trade-off between the minimum acceleration duration and the jerk limit. To find the optimal control profile for the minimum acceleration duration, a multi-stage mixed-integer nonlinear programming problem is formulated, and a highly parallelizable routine for solving this problem is proposed. The concept of the multi-stage mixed-integer trajectory optimization is as follows. Firstly, the acceleration event is divided into $M \in \mathbb{N}$ stages where the gear position stays constant in each stage. (We assume there is no engine on/off switching and the engine stays on for the whole acceleration event as in reality, the engine is on when the vehicle is at 40 kph.) Then, a set of all possible gear shifting sequences is generated. For each shifting sequence, the optimal control profiles of engine and motor torques are found by solving a constrained nonlinear programming problem. Note that different shifting sequences can be handled in parallel. Finally, we obtain the optimal solution by comparing the values of the acceleration duration for all possible gear shifting sequences and picking the one that has the minimum duration.

This multi-stage mixed-integer trajectory optimization method is distinguished by its broad applicability (e.g., it can be applied to other powertrain configurations that have discrete control inputs) and relative ease of implementation. Moreover, as far as the author is aware, this is the first trajectory optimization study for the powertrain configuration of the type used in the 2018 Lexus LC-500h in the literature. The results in Chap. 6 have been reported in [41].

1.3 Dissertation Outline

The remainder of this dissertation is organized as follows. First, Chaps. 2 to 4 discuss the use of set-theoretic methods for fault-tolerant control of mobility systems. We begin in Chap. 2 considering the case with actuator failures where the failed actuators are zeroed out. Then, the treatment is extended to stuck/jammed actuators in Chap. 3. In Chap. 4, a failure detection and isolation strategy is proposed and integrated with the failure mode reconfiguration strategy proposed in the previous two chapters. This strategy is also extended to consider systems with input disturbances in this chapter. Next, Chaps. 5 and 6 focus on automotive applications for safe, efficient and comfortable driving/riding. A control policy for autonomous driving is developed in Chap. 5 using a game-theoretic traffic model for policy training, focusing on energy efficiency improvement while balancing the trade-offs with collision avoidance, comfort, and time efficiency. Chapter 6 uses trajectory optimization to analyze the acceleration performance of an HEV when jerk limits are applied to improve the comfort level. Finally, concluding remarks are made and directions for future work are discussed in Chap. 7.

1.4 Summary of Novel Contributions

This dissertation develops the following novel results:

- A set membership-based, reference governor-centric fault-tolerant control framework is developed for safe reference tracking in systems with redundant actuators and set-bounded measured disturbances in which actuators can fail by getting stuck/jammed. The framework handles failure mode detection, isolation and reconfiguration phases and ensures that the imposed state and control constraints are satisfied throughout all the involved phases. The abilities are guaranteed for failure isolation, system reconfiguration and safe operation after mode transitions, in addition to that the durations needed for the isolation and reconfiguration phases are finite and known before the online operation. Illustrations based on mass-spring-damper system and aircraft longitudinal flight control are used to show the developments.

- A novel approach to generating a control policy for autonomous eco-driving has been developed that accounts for energy consumption, safety, travel time efficiency and comfort through the application of reinforcement learning and level-k game theory. The main innovations versus previous work include the integration of vehicle interaction and energy consumption into the problem, and the demonstration that control policies can be successfully trained and provide energy consumption reduction for battery electric vehicles in shared traffic. These developments fill the gap in the literature by exploiting lateral vehicle controls and possibilities of lane changes, along with standard longitudinal vehicle controls, for energy consumption reduction and collision avoidance.

- A mixed-integer nonlinear programming-based approach has been proposed and demonstrated for trajectory optimization of advanced hybrid electric vehicle powertrains to enhance their drivability and acceleration performance under jerk limits.

CHAPTER 2

Failure Mode Reconfiguration Based on Constraint Admissible and Recoverable Sets for Zeroed-Out Actuators

2.1 Introduction

The chapter presents an approach to the design of a Failure Mode and Effect Management (FMEM) system based on constraint admissible and recoverable sets for systems with redundant actuators and state and control constraints. Multiple actuator failure modes are considered, and an approach to the design of FMEM system is proposed that ensures constraints are satisfied when operating in normal and failure modes and during mode transitions.

There is great demand for systematic approaches to FMEM system design for industrial systems since the software and algorithmic content of diagnostics and FMEM systems is often larger than what is responsible for the nominal system function. This is notably the case in the area of advanced and autonomous vehicles where redundant actuation (e.g., dual steering motors and multiple brake actuators) is employed to enable safe reconfiguration and the implementation of limited operating strategies in the event of failures. In these applications, a typical requirement for FMEM strategy is to ensure that, in case of a single point of failure, the system operation can be reconfigured so that in the new mode, another single point of failure cannot lead to safety hazards while maximizing system availability.

Figure 2.1 illustrates modes in a system with two redundant actuators, where a normal mode corresponds to both actuators functioning, and three failure modes correspond to either or both actuators to have failed. Notably the modes form vertices of a unidirectional graph.

The proposed approach to FMEM design builds on the idea of using constraint admissible and recoverable sets proposed in [21] but extends it to multiple failure modes and reference tracking. In each mode, a reference governor [16] is applied to enforce the constraints. When an actuator failure occurs, a recovery sequence is computed by solving a quadratic programming problem to bring the

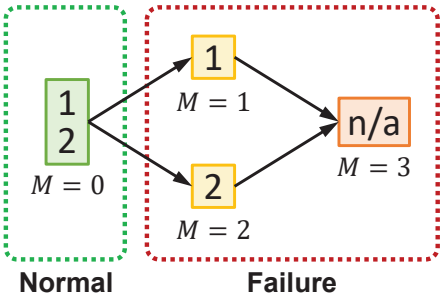

Figure 2.1: Possible modes of a system with two actuators labeled by 1 and 2. The numbers in the boxes are the labels of the actuators that work properly. M stands for mode.

system trajectory into the constraint admissible set for the reference governor in the subsequent mode.

In this setting, constraint admissible sets are sets of initial states and constant reference commands for which the ensuing response satisfies state and control constraints. Recoverable sets are sets of initial states that can be steered into the (state projections of) constraint admissible sets within a specified number of steps without constraint violations.

To be able to perform a safe reconfiguration, the state of the system in the preceding mode must be in the recoverable set for the subsequent mode. In this chapter, the implications of this set-membership condition on the design of FMEM scheme are considered and illustrated with numerical examples. Under the assumptions of a single point of failure (i.e., one actuator failure at a time), instantaneous fault detection and isolation, and large time between subsequent failures, it is sufficient to ensure that a set bounding possible states in the preceding mode is a subset of the interior of the recoverable set in each of the possible subsequent modes. In this chapter, we examine the implications of this condition and illustrate three mechanisms by which it can be ensured: (i) by adding extra state constraints in the preceding mode; (ii) by adding range and rate limits to the command in the preceding mode; and (iii) by temporarily relaxing state constraints when determining the recovery sequence. The latter mechanism is suitable for systems with soft constraints when a temporary constraint violation might be permissible.

Systematic recoverability analysis has been addressed in the fault tolerant control literature [13]; however, the existing methods typically do not handle state and control constraints. In a broader sense, this chapter compliments set-theoretic control methods such as in [17–19,42,43] and provides extensions relevant to handling systems with multiple failure modes and reconfiguration levels. Set theoretic methods for handling failure modes in constrained systems have been considered in [20] and in the reference governor literature (see [16] and references therein). The present chapter is distinguished by addressing multiple failure modes and failure paths/scenarios, by combined use of constrained admissible and recoverable sets and by specific mechanisms used to enforce the

reconfigurability.

The chapter is organized as follows. In Sect. 2.2, the basic problem setting including models, failure modes and constraints is introduced. Section 2.3 introduces constraint admissible and recoverable sets that are used in Sect. 2.4 to define a system capable of safe operation and reconfiguration in the event of failures. Two examples in Sects. 2.5 and 2.6 are used to highlight and illustrate design steps for a mass-spring-damper system and for an aircraft longitudinal flight, respectively. Section 2.7 presents concluding remarks.

2.2 Mode-Dependent System Dynamics, Constraints and Problem Formulation

2.2.1 System Dynamics

The development of our FMEM system relies on a discrete-time model of the form,

$$x_{k+1} = A_M x_k + B_M u_k, \tag{2.1a}$$

$$y_k = C_M x_k, \tag{2.1b}$$

where $k \in \mathbb{Z}_{\geq 0}$, x_k is the state vector, y_k is the output vector, u_k is the input vector, and $M \in \{0, 1, 2, \cdots\}$ designates different operating modes of the system, including normal modes and failure modes caused by actuator failures when specific inputs corresponding to the failed actuators equal zero.

It is assumed that the number of working actuators/inputs are greater than or equal to the number of outputs that need to be tracked. Then in each mode, a stabilizing feedback plus feedforward controller is used of the form,

$$u_k = K_M x_k + G_M v_k, \tag{2.2}$$

where K_M is the feedback gain matrix, G_M is the feedforward gain matrix, and v_k is the vector of the reference commands (set-points). Depending on the mode M, appropriate rows of K_M and G_M are zeroed out to represent the effect of the actuator failures as the inputs corresponding to the failed actuators are forced to zero.

The closed-loop dynamics in mode M can be represented by the following discrete-time model,

$$\begin{aligned} x_{k+1} &= \bar{A}_M x_k + \bar{B}_M v_k, \\ \bar{A}_M &= A_M + B_M K_M, \\ \bar{B}_M &= B_M G_M, \end{aligned} \tag{2.3}$$

where $\bar{A}_M$ is a Schur matrix (where all eigenvalues are inside the unit disk of the complex plane). Since one of the possible modes is when all actuators fail, this assumption implies that the open-loop system must be stable.

2.2.2 Constraints

To ensure safe operation, pointwise-in-time state constraints are defined by a finite set of inequalities and imposed of the form,

$$x_k \in \mathcal{X}(v_k) = \{x : \ \mathcal{A}_{\mathcal{X}} x \leq \mathcal{b}_{\mathcal{X}}(v_k)\}. \tag{2.4}$$

Since these constraints are imposed on the states of the closed-loop system with the given controllers, they can also represent actuator range and rate limits.

Modifications to these constraints are considered to facilitate the subsequent design of our failure mode reconfiguration approach. Firstly, to satisfy subsequent conditions for safe failure mode reconfiguration, it may be necessary to artificially tighten constraints (2.4) to

$$x_k \in \mathcal{X}_M(v_k) = \mathcal{X}(v_k) \cap \bar{\mathcal{X}}_M(v_k), \tag{2.5}$$

where the sets $\bar{\mathcal{X}}_M(v)$ need to be appropriately designed. Secondly, upon detection and isolation of the failure mode, a recovery sequence of control inputs is computed and implemented over a short time horizon before the control is relinquished to the reference governor. In practical applications (see e.g., [44]), some of the state constraints could be imposed conservatively to extend system operating life, and they may be relaxed temporarily during the reconfiguration and recovery. Hence during a short period when the recovery sequence is applied, the constraints can be relaxed to

$$x_k \in \mathcal{X}_{R_M}(v_k), \tag{2.6}$$

where $\mathcal{X}_{R_M}(v) \supseteq \mathcal{X}_M(v)$, and $\mathcal{X}_{R_M}(v)$ should also be appropriately designed.

2.2.3 Problem Formulation

A Failure Mode and Effect Management (FMEM) system is to be designed which is capable of ensuring safety despite failures. Each failure corresponds to a loss of an actuator and system mode transition. The time between failures is assumed to be large.

2.3 Constraint Admissible and Recoverable Sets

2.3.1 Constraint Admissible Sets

Each operating mode M has its corresponding approximation to the maximum constraint admissible set defined by

$$\mathcal{O}_{\infty,M} = \{(v, x_0) : \ x_t \in \mathcal{X}_M(v) \ \forall t \in \mathbb{Z}_{\geq 0}, \ x_{\text{ss}}(v, M) \bigoplus \mathcal{B}_\epsilon \subset \mathcal{X}_M(v)\} \tag{2.7}$$

$$= \{(v, x_0) : \ \mathcal{A}_{\mathcal{O}_{\infty,M}} x_0 \leq \mathcal{b}_{\mathcal{O}_{\infty,M}}(v)\}, \tag{2.8}$$

where x_t is the response of (2.3) to the initial condition x_0 and constant command $v_t = v$, $x_{\text{ss}}(v, M)$ is the steady-state operating point given by $x_{\text{ss}}(v, M) = (I - \bar{A}_M)^{-1}\bar{B}_M v$, and $\mathcal{B}_\epsilon$ is an open ball of radius $\epsilon > 0$. The reasons for adding the constraint $x_{\text{ss}}(v, M) \bigoplus \mathcal{B}_\epsilon \subset \mathcal{X}_M(v)$ in (2.7) are technical: They ensure, under mild additional assumptions [45], that the set $\mathcal{O}_{\infty,M}$ is finitely-determined and can be represented by a finite set of affine inequalities as in (2.8). An important property of $\mathcal{O}_{\infty,M}$ is its invariance under constant commands, that is, if $(v_0, x_0) \in \mathcal{O}_{\infty,M}$ and $v_t = v_0$ for all $t \in \mathbb{N}$ while the system remains operating in mode M, then $(v_t, x_t) \in \mathcal{O}_{\infty,M} \ \ \forall t \in \mathbb{N}$.

2.3.2 Recoverable Sets

The recoverable set for the system in mode M is defined as

$$\begin{aligned}\mathcal{R}_{\infty,M}^{N_M} = \{x_0 : \ &\exists \{v_0, \cdots, v_{N_M-1}\} \text{ such that} \\ &x_t \in \mathcal{X}_{R_M}(v_t) \ \ \forall t \in \{0, 1, \cdots, N_M - 1\}, \ x_{N_M} \in \text{Proj}_x \mathcal{O}_{\infty,M}\}.\end{aligned} \tag{2.9}$$

Any initial condition x_0 in the recoverable set $\mathcal{R}_{\infty,M}^{N_M}$ can be "steered" into $\text{Proj}_x \mathcal{O}_{\infty,M}$ within N_M steps using the command sequence $\{v_0, \cdots, v_{N_M-1}\}$, and the constraints $x_t \in \mathcal{X}_{R_M}(v_t)$ $\forall t \in \{0, 1, \cdots, N_M - 1\}$ and $x_{N_M} \in \text{Proj}_x \mathcal{O}_{\infty,M}$ are satisfied.

2.4 Employing Admissible and Recoverable Sets for Safe Operation and Reconfiguration

2.4.1 Safe Operation in Each Mode Using Reference Governor

The reference/command governor [16] is used for reference tracking and to enforce the constraints in each mode except during the reconfiguration. The reference governor computes the modified command v_k as a function of the state x_k and original reference command r_k based on the solution of the following optimization problem,

$$\begin{aligned} &\min_{v_k} \|r_k - v_k\|^2 \\ &\text{s.t. } (v_k, x_k) \in \mathcal{O}_{\infty,M}. \end{aligned} \tag{2.10}$$

If $(v_t, x_t) \in \mathcal{O}_{\infty,M}$ and the mode remains equal to M, then there exists $(v_{t+k}, x_{t+k}) \in \mathcal{O}_{\infty,M}$, i.e., (2.10) is feasible, for all $k \geq 1$ and the constraints (2.5) remain satisfied (e.g., we can let $v_{t+k} = v_t \; \forall k \geq 1$).

2.4.2 Safe Reconfiguration Upon Failure Detection

The failure mode reconfiguration relies on the condition

$$\text{Proj}_x \mathcal{O}_{\infty,M} \subseteq \mathcal{R}^{N_{M'}}_{\infty,M'} \;\; \forall M' \in \text{succ}(M), \tag{2.11}$$

where $\mathcal{R}^{N_{M'}}_{\infty,M'}$ denotes the set of all states which are recoverable with command sequences of length $N_{M'}$ in mode M', and $\text{succ}(M)$ is the set of all successor modes of mode M.

To ensure that (2.11) is satisfied, the sets $\bar{\mathcal{X}}_M(v)$, $\mathcal{X}_{R_{M'}}(v_t)$ and $N_{M'}$ need to be designed. One can reduce $\bar{\mathcal{X}}_M(v)$ (i.e., tighten constraints for the preceding mode M), enlarge $\mathcal{X}_{R_{M'}}(v_t)$ (i.e., relax constraints for the successor mode M' during the recovery) and increase $N_{M'}$ (i.e., allow more elements in the reconfiguration sequence). The last approach has an impact on online computations as increasing $N_{M'}$ increases the size of the optimization problem which needs to be solved online.

With the condition (2.11) satisfied, if the failure occurs at the time instant t, then $x_t \in \mathcal{R}^{N_{M'}}_{\infty,M'}$ for the new mode $M' \in \text{succ}(M)$, so a recovery sequence $\boldsymbol{v}$ of the reference commands can be

found by solving a quadratic programming problem,

$$\begin{aligned} &\min_{\boldsymbol{v}} \|r_t\mathbf{1} - \boldsymbol{v}\|^2 \\ \text{s.t. } &x_{t+s} \in \mathcal{X}_{R_M}(v_{t+s}) \ \ \forall s \in \{0, 1, \cdots, N_{M'} - 1\}, \\ &x_{t+N_{M'}} \in \text{Proj}_x \mathcal{O}_{\infty,M'}, \end{aligned} \tag{2.12}$$

where $\boldsymbol{v} = \begin{bmatrix} v_t^T & v_{t+1}^T & \cdots & v_{t+N_{M'}}^T \end{bmatrix}^T$ is the recovery sequence, and the reference sequence $r_t\mathbf{1} = \begin{bmatrix} r_t^T & r_t^T & \cdots & r_t^T \end{bmatrix}^T$ means assuming a constant reference equal to r_t during the reconfiguration since we do not know the future references at the begining of the recovery. After the recovery sequence is applied, we have $x_{t+N_{M'}} \in \text{Proj}_x \mathcal{O}_{\infty,M'}$. Hence at the time instant $t + N_{M'}$ the reference governor can be activated to continue operating the system in mode M'.

2.4.3 Command Range and Rate Limiting

An additional mechanism to ensure reconfigurability in the design of our FMEM system is to impose artificial range and rate limits on v_k. This results in state trajectories converging to a subset of $\text{Proj}_x \mathcal{O}_{\infty,M}$ when operating in mode M. Since the time between failures is assumed to be large, $\text{Proj}_x \mathcal{O}_{\infty,M}$ in (2.11) can be replaced by a smaller subset which ultimately bounds state trajectories, thereby weakening the requirement (2.11). These ancillary constraints on the command range and rate have the form,

$$v_k \in \mathcal{V}_M = \{v : \ \mathcal{A}_V v \leq \mathcal{b}_V\}, \tag{2.13}$$

$$\Delta v_k = v_k - v_{k+1} \in \Delta\mathcal{V}_M = \{\Delta v : \ \mathcal{A}_{\Delta V} \Delta v \leq \mathcal{b}_{\Delta V}\}. \tag{2.14}$$

Let

$$\begin{aligned} z_k &= x_k - \Gamma_M v_k, \\ \Gamma_M &= (I - \bar{A}_M)^{-1} \bar{B}_M, \end{aligned} \tag{2.15}$$

then it can be derived that

$$z_{k+1} = \bar{A}_M z_k + \Gamma_M \Delta v_k, \tag{2.16}$$

since

$$\begin{aligned}
z_{k+1} &= x_{k+1} - \Gamma_M v_{k+1} \\
&= \bar{A}_M x_k + \bar{B}_M v_k - \Gamma_M v_{k+1} \\
&= \bar{A}_M (z_k + \Gamma_M v_k) + \bar{B}_M v_k - \Gamma_M v_{k+1} \\
&= \bar{A}_M z_k + \big(\bar{A}_M (I - \bar{A}_M)^{-1} + I\big) \bar{B}_M v_k - \Gamma_M v_{k+1} \\
&= \bar{A}_M z_k + \big(\bar{A}_M (I + \bar{A}_M + \bar{A}_M^2 + \cdots +) + I\big) \bar{B}_M v_k - \Gamma_M v_{k+1} \\
&= \bar{A}_M z_k + (I - \bar{A}_M)^{-1} \bar{B}_M v_k - \Gamma_M v_{k+1} \\
&= \bar{A}_M z_k + \Gamma_M (v_k - v_{k+1}) \\
&= \bar{A}_M z_k + \Gamma_M \Delta v_k.
\end{aligned}$$

With rate limits imposed by (2.14) and assuming the mode remains in M, we have

$$z_k \to \mathcal{F}_M \text{ as } k \to \infty,$$

where $\mathcal{F}_M$ is the minimum invariant set [45] of the system derived from (2.16). This minimum invariant set is defined as an infinite Minkowski sum

$$\mathcal{F}_M = \Gamma_M \Delta \mathcal{V}_M \bigoplus \bar{A}_M \Gamma_M \Delta \mathcal{V}_M \bigoplus \bar{A}_M^2 \Gamma_M \Delta \mathcal{V}_M \bigoplus \cdots,$$

while in the implementation, computable outer approximations of $\mathcal{F}_M$ are used. In addition to the rate limits, if range limits are also imposed on the command as in (2.13), then the state trajectory converges to the set

$$\mathcal{S}_M = \mathcal{F}_M \bigoplus \Gamma_M \mathcal{V}_M. \tag{2.17}$$

With the extra range and rate limits on the commands (2.13), (2.14) added to the optimization problem defining the reference governor (2.10), the state trajectories in mode M will be ultimately bounded in the set $\mathcal{S}_M \cap \text{Proj}_x \mathcal{O}_{\infty,M}$. Hence the condition (2.11) can be weakened to

$$\mathcal{S}_M \cap \text{Proj}_x \mathcal{O}_{\infty,M} \subseteq \text{int} \mathcal{R}_{\infty,M'}^{N_{M'}} \ \ \forall M' \in \text{succ}(M). \tag{2.18}$$

2.5 Mass-Spring-Damper System Example

A low order example of a system with redundant actuation is given by a mass-spring-damper system, where $m_0 = 1\,\text{kg}$ for the mass (Fig. 2.2), $k_0 = 1\,\text{N/m}$ for the spring stiffness and $c_0 = 0.01\,\text{N s/m}$ for the damping coefficient.

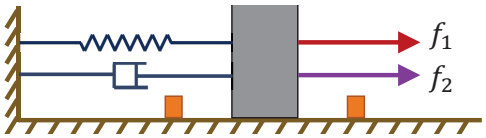

Figure 2.2: Mass-spring-damper system with position and force magnitude constraints and redundant actuation.

2.5.1 System Dynamics, Mode Definitions and Constraints

In the normal operating mode (Mode 0), the continuous-time system model is given by

$$\dot{x} = Ax + Bu, \tag{2.19a}$$

$$y = Cx, \tag{2.19b}$$

where $u = [f_1 \;\; f_2]^T$ is the input with f_1 and f_2 being the forces in N, $x = [d \;\; w]^T$ is the state with d being the displacement of the mass in m, and w being the velocity of the mass in $\mathrm{m/s}$, $y = d$ is the output, and

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k_0}{m_0} & -\frac{c_0}{m_0} \end{bmatrix}, \; B = \begin{bmatrix} 0 & 0 \\ \frac{1}{m_0} & \frac{1}{m_0} \end{bmatrix}, \; C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

Three failure modes are possible in this system and the potential failure paths/scenarios are illustrated in Fig. 2.1: In Mode 1, f_1 operates normally while f_2 fails, that is, $u = [f_1 \;\; 0]^T$; In Mode 2, f_2 operates normally while f_1 fails, that is, $u = [0 \;\; f_2]^T$; In Mode 3, Both f_1 and f_2 fail, that is, $u = [0 \;\; 0]^T$.

The discrete-time model of the form (2.1) with $M \in \{0, 1, 2, 3\}$ is obtained by converting the model (2.19) to discrete-time assuming the sampling period of 0.2 sec.

The controller (2.2) for modes $M = 0, 1, 2$ is designed using Linear Quadratic Regulator (LQR) theory to obtain K_M. The feedforward gain G_M is computed so that the steady-state gain from v to the mass position d is equal to 1. In the selection of the LQR weights Q_M and R_M, it is assumed that the use of f_1 is more expensive than the use of f_2. The closed-loop system of each mode is design using the following parameters:

- For Mode 0, 1, and 2,

$$Q_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \; Q_1 = Q_0, \; Q_2 = Q_0,$$

$$R_0 = \begin{bmatrix} 1000 & 0 \\ 0 & 250 \end{bmatrix},\ R_1 = \begin{bmatrix} 10 & 0 \\ 0 & 0 \end{bmatrix},\ R_2 = \begin{bmatrix} 0 & 0 \\ 0 & 2.5 \end{bmatrix}.$$

- For Mode 3, since both actuators fail, the system operates as it is in open-loop, but to keep the consistency of the notation, let

$$K_3 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},\ G_3 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The constraints are imposed on the position of the mass and on the magnitudes of the actuation forces:

$$|d| \leq y_{\max},\ |f_1| \leq u_{\max,1},\ |f_2| \leq u_{\max,2},$$

where $y_{\max} = 1\,\mathrm{m}$ and $u_{\max,1} = u_{\max,2} = 1\,\mathrm{N}$. Using (2.2), these constraints are converted into the form (2.4).

2.5.2 FMEM Strategy Design

The development of the FMEM strategy based on constraint admissible and recoverable sets proceeds backward beginning from Mode 3 to the predecessor modes.

Since in Mode 3 the system runs in open-loop, the constraint admissible set in Mode 3 can be computed directly for the states (compare with (2.7)) as

$$\begin{aligned} \mathrm{Proj}_x \mathcal{O}_{\infty,3} &= \{x_0 :\ x_t \in \mathcal{X}_3(0)\ \ \forall t \in \mathbb{Z}_{\geq 0}\} \\ &= \{x_0 :\ A_{\mathcal{O}_{\infty,3}} x_0 \leq b_{\mathcal{O}_{\infty,3}}\}, \end{aligned} \tag{2.20}$$

and the recoverable set of Mode 3 is

$$\mathcal{R}_{\infty,3}^{N_3} = \mathrm{Proj}_x \mathcal{O}_{\infty,3}\ \ \forall N_3 \geq 0, \tag{2.21}$$

Equation (2.21) implies that (2.11) can be satisfied only if $x_k \in \mathrm{Proj}_x \mathcal{O}_{\infty,3}$ is imposed as the state constraint during the operation in Modes 1 and 2. To demonstrate using mechanisms in Sects. 2.4.2 and 2.4.3 to satisfy this requirement, two case studies are considered.

In the first case, state constraints of Mode 1 and 2 are tightened while the command range and rate limits are not used. This can be done by imposing state constrains of Mode 3 to Mode 1 and 2.

Remark 2.1. Since the number of the inequalities in the representation of $\mathrm{Proj}_x \mathcal{O}_{\infty,3}$ can be large, this can result in highly complex $\mathcal{O}_{\infty,1}$ and $\mathcal{O}_{\infty,2}$ and large computational effort in using them in the implementation of the reference governor. Consequently, we use a simpler subset $\mathcal{P}_3 \subset \mathrm{Proj}_x \mathcal{O}_{\infty,3}$.

Such a subset is generated by removing close to being redundant inequalities from the representation of $\text{Proj}_x\mathcal{O}_{\infty,3}$ and a scaling transformation to ensure that $\mathcal{P}_3 \subset \text{Proj}_x\mathcal{O}_{\infty,3}$. This leads to

$$\mathcal{P}_3 = \{x_0 : \mathcal{A}_{\mathcal{P}_3} x_0 \leq \mathcal{b}_{\mathcal{P}_3}\}. \tag{2.22}$$

Now for $M \in \{1, 2\}$, we let $\bar{\mathcal{X}}_M = \mathcal{P}_3$. Then, $\mathcal{X}_M(v)$ and $\mathcal{O}_{\infty,M}$ can be constructed by (2.5) and (2.7).

The second case relies on having the command range and rate limits (2.13) and (2.14) of the form $|v| \leq v_{\max}$ and $|\Delta v| \leq \Delta v_{\max}$. No additional state constraints are added to further restrict the operation in Mode 1 and 2. That is, for $M \in \{1, 2\}$, we let $\mathcal{X}_M(v) = \mathcal{X}(v)$ and construct $\mathcal{O}_{\infty,M}$ using (2.7).

Finally, for both cases, $\mathcal{O}_{\infty,0}$ is computed based on (2.7) with $\mathcal{X}_0(v) = \mathcal{X}(v)$.

The value of $\epsilon = 0.01$ was used in computing $\mathcal{O}_{\infty,M}$ for $M \in \{0, 1, 2\}$. The projections $\text{Proj}_x\mathcal{O}_{\infty,M}$ computed using `Bensolve` [46] are shown in Fig. 2.3(a) for the first case and in Fig. 2.4(a) for the second case, with Figs. 2.3(b) and 2.4(b) showing the zoom-in views.

Recoverable sets in the first case for mode switching between Mode 0 and Mode 1 or 2 are based on (2.9) where $\mathcal{X}_{R_M}(v) = \mathcal{X}_M(v)$, and N_M is chosen sufficiently large to satisfy (2.11). Figures 2.3(a) and (b) show the recoverable sets of Mode 2 as examples. After comparing the admissible and recoverable sets for both Mode 1 and Mode 2, N_M should at least be 3 to satisfy (2.11). Hence, we set $N_1 = N_2 = 3$ to minimize the length of the recovery sequence. Note that for transitions to Mode 3, the condition (2.11) is satisfied as by construction so both $\text{Proj}_x\mathcal{O}_{\infty,1}$ and $\text{Proj}_x\mathcal{O}_{\infty,2}$ are subsets of $\text{Proj}_x\mathcal{O}_{\infty,3}$

For the second case, $\text{Proj}_x\mathcal{O}_{\infty,2} \supseteq \text{Proj}_x\mathcal{O}_{\infty,0}$ as shown in Fig. 2.4(a), and so is for $\text{Proj}_x\mathcal{O}_{\infty,1}$. Thus, for mode transitioning from Mode 0 to Mode 1 or 2, since $x_k \in \text{Proj}_x\mathcal{O}_{\infty,M}$ for $M \in \{1, 2\}$ is automatically satisfied, a recovery sequence is not needed. Now, to safely switch from Mode 1 or 2 to Mode 3, $v_{\max}$ and $\Delta v_{\max}$ need to be chosen so that the condition (2.18) holds. In order to maximize the size of $\mathcal{S}_M \cap \text{Proj}_x\mathcal{O}_{\infty,M}$ for $M \in \{1, 2\}$ that restrict the operation of the system in Mode 1 and Mode 2, we set $v_{\max} = 1$ and $\Delta v_{\max} = 0.007$ for Mode 1, and $v_{\max} = 1$ and $\Delta v_{\max} = 0.01$ for Mode 2.

2.5.3 Simulation Results

Results of the two case studies are shown in Figs. 2.3 and 2.4, where the trajectories are plotted in (a) and (b), and time-based signals are in (c) and (d).

In both cases, the system begins with operation in the normal mode (Mode 0), and then sequentially switches to Mode 2 and Mode 3. The reference is set to be switching between $-0.99\,\text{m}$

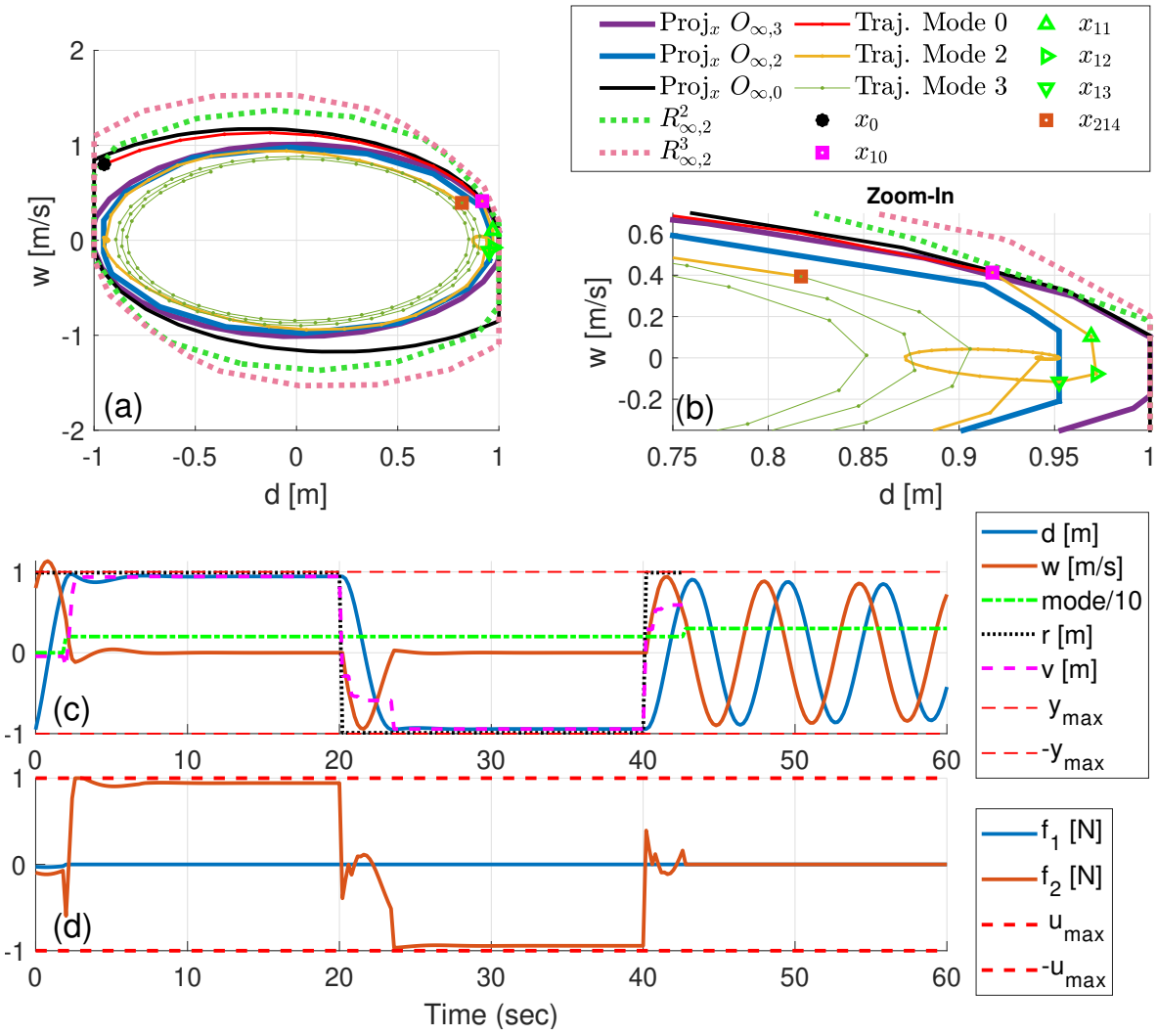

Figure 2.3: Projections of sets and simulation results for the first case.

and $0.99\,\mathrm{m}$. The time-based results show that all state and control input constraints are satisfied throughout the simulation. When the system switches to Mode 2, in comparison with the first case study, the trajectory in the second case immediately starts approaching $\mathcal{S}_2 \cap \mathrm{Proj}_x \mathcal{O}_{\infty,2}$ and stays inside it after the state converges. Effects of command rate limiting are clearly visible from the time-based signals.

2.6 Aircraft Longitudinal Flight Control Example

We now consider a higher order example of aircraft longitudinal flight control with engine thrust and elevator actuators.

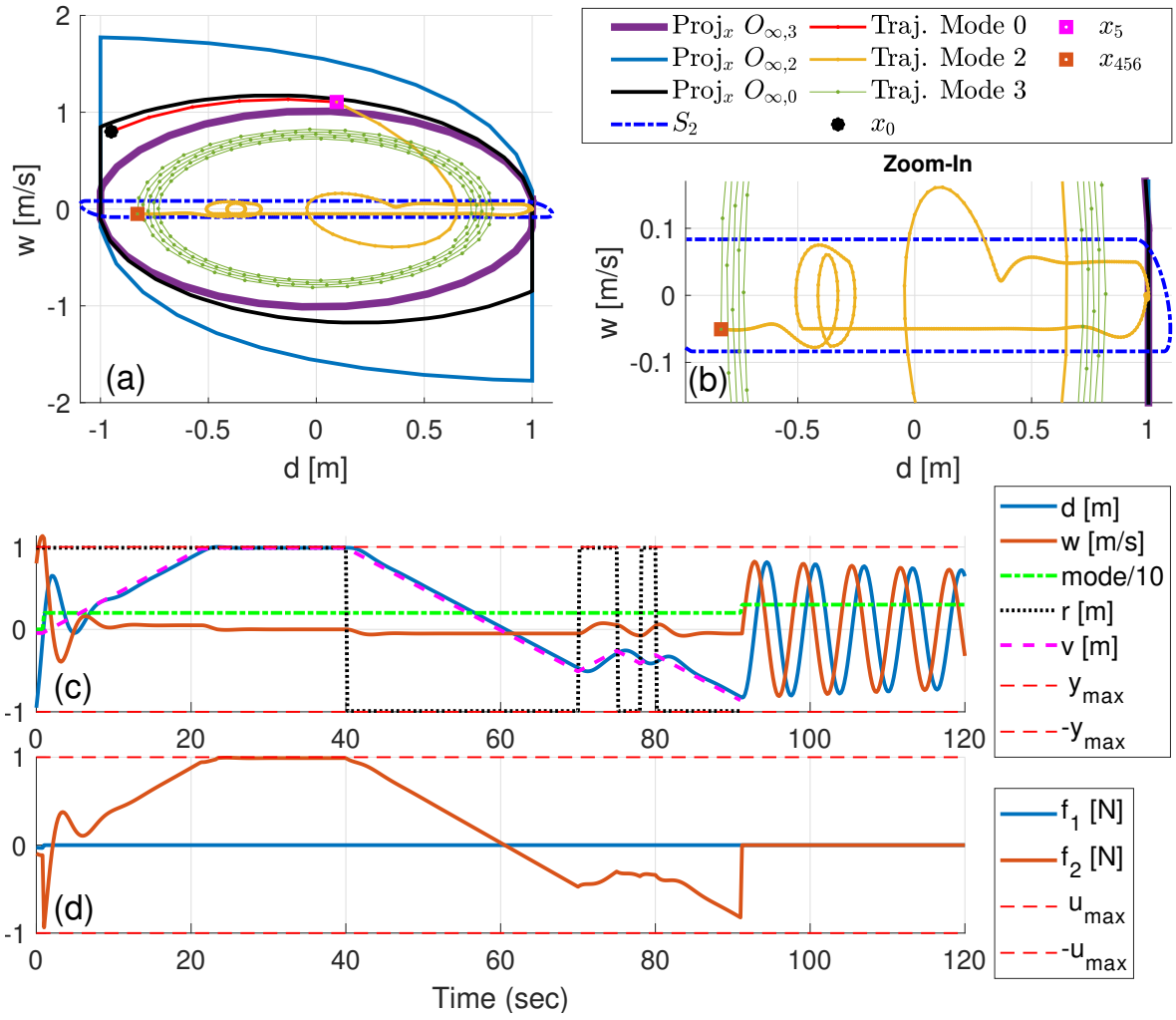

Figure 2.4: Projections of sets and simulation results for the second case.

2.6.1 System Dynamics, Mode Definitions and Constraints

The aircraft model represents a Boeing 747-100 aircraft at steady level flight corresponding to Mach 0.5 cruise at 20,000 feet [47]. The linearized longitudinal flight dynamics under the normal operating conditions can be represented by the form (2.19) where $x = [\Delta\mu \;\; \Delta w \;\; \Delta q \;\; \Delta\theta]^T$, $u = [\Delta a_T \;\; \Delta\delta_e]^T$, and $y = \Delta\dot{h}$, μ and w, respectively, are the projection of the velocity vector on the x-axis and z-axis of the body-fixed frame in $\mathrm{m/s}$, q is the projection of the angular velocity vector on the y-axis in $^\circ/\mathrm{s}$ (degree per second), θ is the pitch angle in $^\circ$ (degree), a_T is the thrust-to-mass ratio in $\mathrm{N/kg}$, δ_e is the elevator deflection in $^\circ$, $\dot{h}$ is the climb rate in $\mathrm{m/s}$, and Δ denotes the

deviation from the trim value. In this model,

$$A = \begin{bmatrix} -0.0073 & 0.0274 & -0.0040 & -0.1713 \\ -0.1205 & -0.4350 & 2.7645 & 0 \\ 0.0188 & -0.3196 & -0.4850 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$
$$B = \begin{bmatrix} 1 & -0.0053 \\ 0 & 0.1170 \\ 0 & 14.5 \\ 0 & 0 \end{bmatrix}, \; C = \begin{bmatrix} 0 & -1 & 0 & 2.7645 \end{bmatrix}. \tag{2.23}$$

The model is then converted to discrete-time assuming 0.2 sec update period.

The normal and failure modes are defined and the controller for each mode is designed similar to Sect. 2.5. In Mode 0, the controller is designed to track a given climb rate deviation and a zero deviation of the velocity magnitude from the nominal, that is, to hold $\sqrt{\mu^2 + w^2}$ at constant. During Modes 2 and 3, $\Delta a_T = 0$ in the linearized model can represent full authority digital engine control (FADEC) system maintaining engine availability when handling an internal engine failure mode but having to restrict the ability to change engine thrust.

The constraints are imposed on the climb rate and on the thrust and the elevator inputs as

$$|\Delta \dot{h}| \leq y_{\max}, \; |\Delta a_T| \leq u_{\max,1}, \; |\Delta \delta_e| \leq u_{\max,2},$$

where $y_{\max} = 5\,\mathrm{m/s}$, $u_{\max,1} = 5\,\mathrm{N/kg}$, and $u_{\max,2} = 5^\circ$. These constraints define $\mathcal{X}(v_k)$ and can be put into the form (2.4).

2.6.2 FMEM Strategy Design

The design process of admissible and recoverable sets for this aircraft example is similar to the first case in Sect. 2.5.2, except for two main differences. First, in Mode 0, the state constraints $\mathcal{X}_0(v)$ in (2.7) had to be tightened. Second, $\mathcal{X}_{R_M}(v_t)$ in (2.9) exploits the relaxation in Mode 1 and 2.

The rationale for these two steps is illustrated in Fig. 2.5 as an example, showing the projection of $\mathcal{O}_{\infty,0}$ and $\mathcal{R}^{N_1}_{\infty,1}$ on the first two coordinates $\Delta\mu$ and Δw. It shows that the change of N_1 has little impact on the shape of $\mathcal{R}^{N_1}_{\infty,1}$ relative to $\mathcal{O}_{\infty,0}$ in the Δw direction. In order to satisfy (2.11), a large value for N_1 is required, meaning the reconfiguration will take a long time. Similar situation also occurs in Mode 2. Thus, in order to have a reasonable value of N_M for $M \in \{1, 2\}$, $\mathcal{X}_0(v)$ and $\mathcal{X}_{R_M}(v_t)$ for $M \in \{1, 2\}$ need to be adjusted.

The first change is to reduce the size of $\mathcal{O}_{\infty,0}$ by adding extra state constraints. This is done by

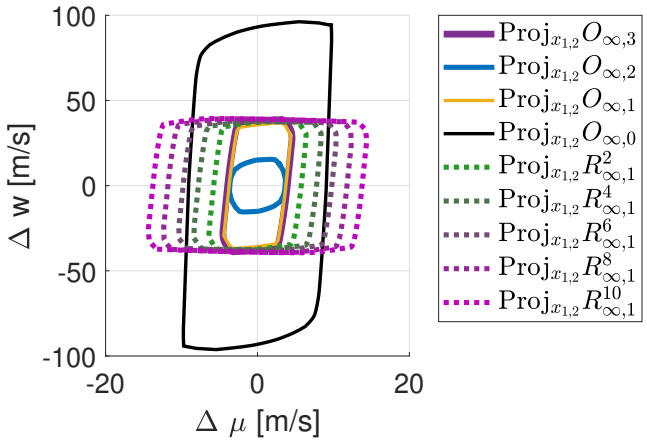

Figure 2.5: Projections of admissible and recoverable sets as N_M changes.

having $\mathcal{X}_0(v) = \mathcal{X}(v) \cap \bar{\mathcal{X}}_0$ where

$$\bar{\mathcal{X}}_0 = \{x : \ \mathcal{A}_{\mathcal{P}_3} x \leq \eta_O b_{\mathcal{P}_3}\}. \tag{2.24}$$

Here, $\eta_O \geq 1$ is a design parameter. Figure 2.6 shows the projection of $\mathcal{O}_{\infty,0}$ with different values of η_O. Smaller values of η_O make it more likely that (2.11) can be satisfied with small N_1 and N_2, but at the same time, $\mathcal{O}_{\infty,0}$ shrinks, meaning that the system operation in the normal mode is more restricted. Thus, the value of η_O needs to be optimized and carefully chosen.

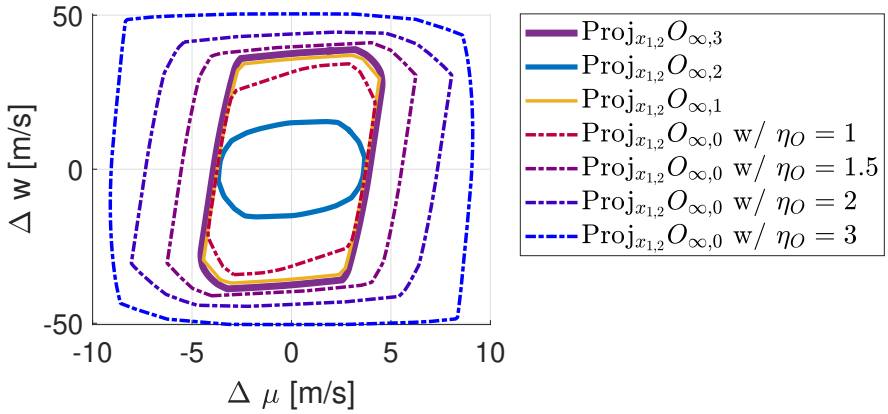

Figure 2.6: Projections of admissible sets as η_O changes.

To avoid the need for over restriction in Mode 0, the second change is introduced, that is, to relax state constraints during the application of the recovery sequence in Mode 1 and 2 based on

(2.9), with

$$\mathcal{X}_{R_M} = \{x : \ \mathcal{A}_{\mathcal{X}} x \leq \eta_{R_M} b_{\mathcal{X}}\}, \tag{2.25}$$

where $\eta_{R_M} \geq 1$ is also a design parameter. Figure 2.7 shows the projection of $\mathcal{R}_{\infty,1}^{N_1}$ for different values of η_{R_M}. The "size" of $\mathcal{R}_{\infty,M}^{N_M}$ increases as the value of η_{R_M} increases, while the trade-off is that the system could operate far outside from the normal constraints if the value of η_{R_M} gets too large. Thus, the value of η_{R_M} also needs to be optimized and chosen with care.

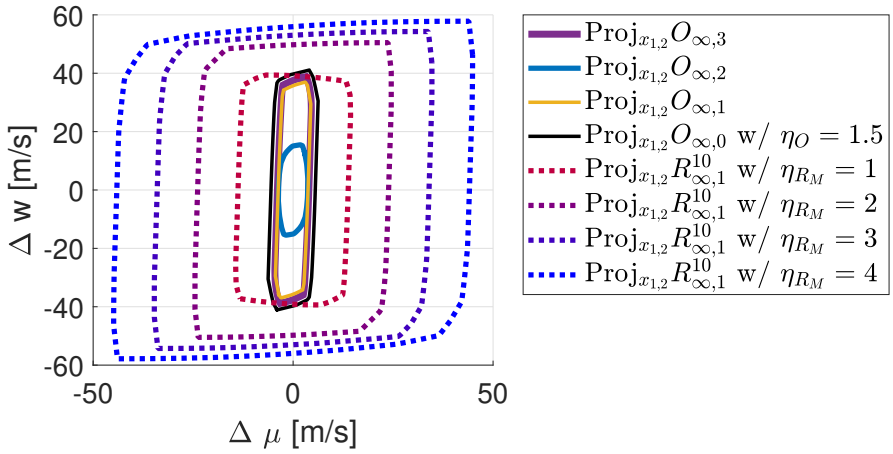

Figure 2.7: Projections of admissible and recoverable sets as η_{R_M} changes.

Based on optimization over a grid of values, we have $\eta_O = 1.5$, and we choose $N_1 = 9$, $\eta_{R_1} = 8$ for Mode 1, and $N_2 = 10$, $\eta_{R_2} = 11$ for Mode 2. Note that in Fig. 2.7, although it appears that for the projected sets we have $\text{Proj}_{x_{1,2}} \mathcal{O}_{\infty,0} \subseteq \text{Proj}_{x_{1,2}} \mathcal{R}_{\infty,1}^{10}$ with $\eta_{R_M} \geq 2$, this does not imply (2.11) is satisfied. Specifically for this aircraft example, large values of η_{R_M} are needed.

2.6.3 Simulation Results

Figure 2.8 shows an example of the simulation with mode switching from Mode 0 to Mode 1 at 72.2 sec and then from Mode 1 to Mode 3 at 120.2 sec, and Fig. 2.9 shows the simulation where the mode switches from Mode 0 to Mode 2 and then to Mode 3 also at 72.2 sec and 120.2 sec. The time-based signals show that all input and output constraints are respected through out the simulation.

2.7 Concluding Remarks

To be able to handle onboard failures safely (i.e., without violation of constraints), Failure Mode and Effect Management (FMEM) systems have to be appropriately designed. In systems with multiple redundant actuators, system operating modes can be defined dependent on functioning actuators. The operation in the preceding mode may have to be restricted either by imposing extra pointwise-in-time state constraints or by deliberately slowing down the response in order to ensure that in the event of a failure there exists a recovery control sequence that can avoid constraint violation. This chapter illustrated some of the ingredients and approaches that can be used in a systematic FMEM system design based on reference governors, and based on recoverable and constraint admissible sets. The development of comprehensive numerical procedures that can be used to guarantee the set inclusion conditions for safe reconfigurability is left to the future work.

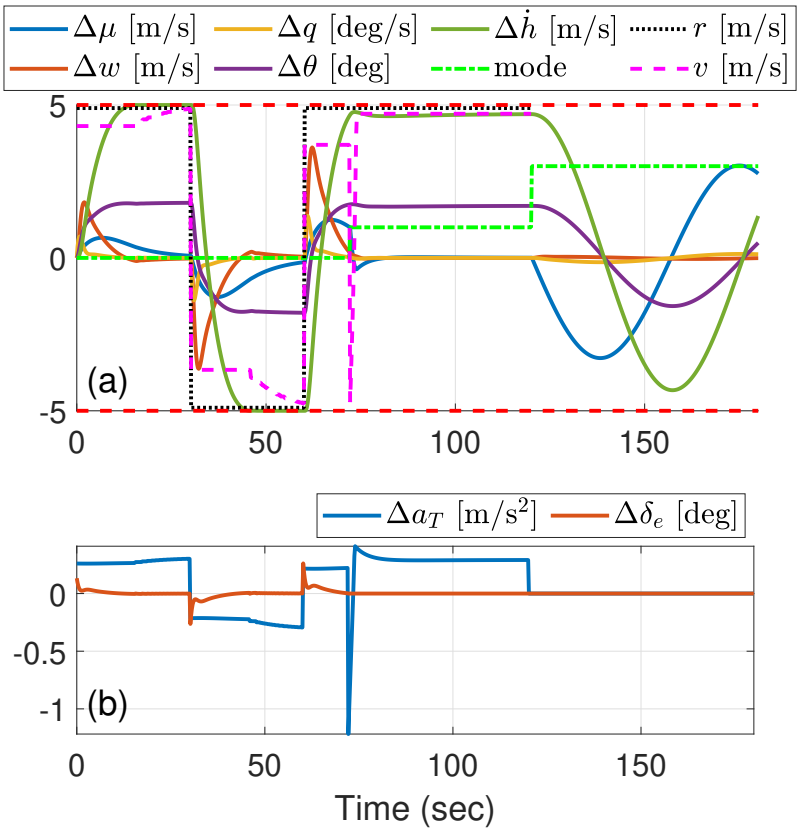

Figure 2.8: Simulation results with failure path from Mode 0 to Mode 1 to Mode 3.

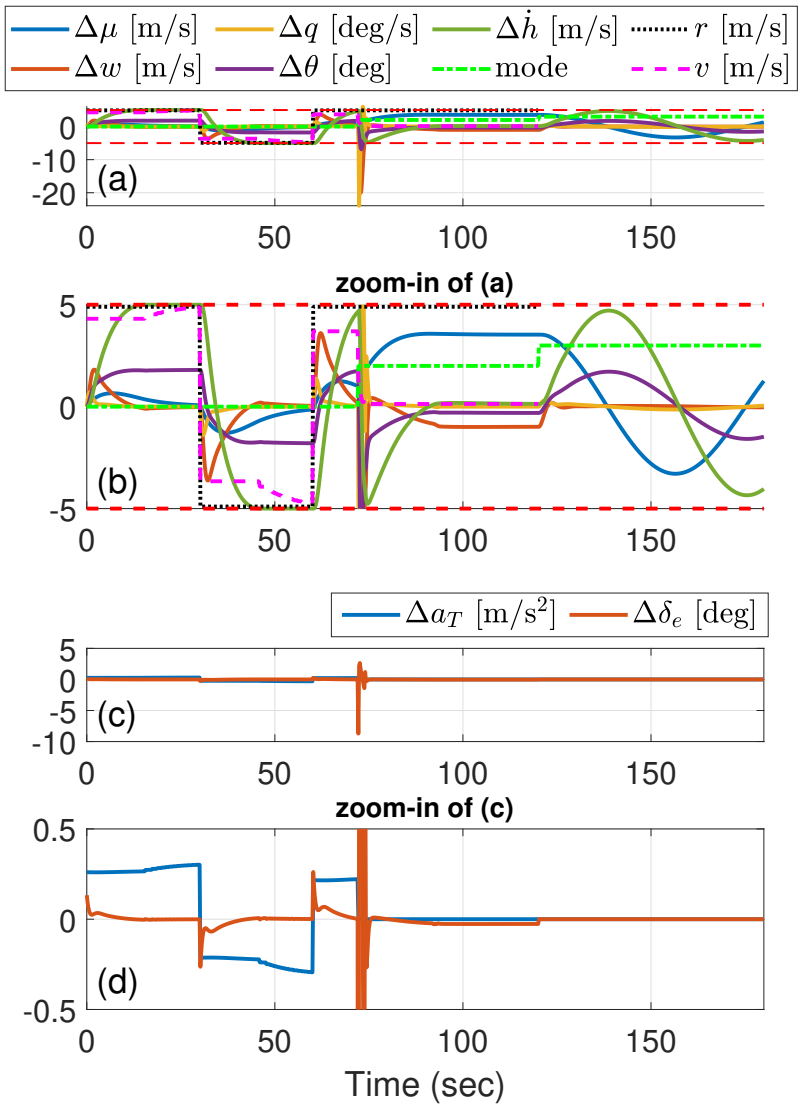

Figure 2.9: Simulation results with failure path from Mode 0 to Mode 2 to Mode 3.

CHAPTER 3

Set-Theoretic Failure Mode Reconfiguration for Stuck Actuators

3.1 Introduction

Maintaining the safety of systems in applications such as aircraft and autonomous vehicles is critical. These systems often operate in constrained environments while tracking reference commands. For example, an autonomous vehicle may be given a planned trajectory to follow but it must simultaneously avoid obstacles such as other vehicles in traffic or pedestrians. Such applications may require the system to operate without constraint violation at all times, even when failures occur.

Continuing to operate without constraint violations is challenging when there are failed hardware components. It is not uncommon for hardware components, such as sensors and actuators, to fail for various reasons. In particular, failures due to stuck actuators often occur in industrial applications. Jammed elevators or rudders, for example, are one of the most common reasons for the failure of aircraft flight control, for which consequences could be fatal [5]. To mitigate failures, redundant actuators are often used (e.g., dual steering systems in automotive applications). In addition, sensors can be added for diagnostic purposes. As the software and algorithmic content for handling failure modes can be large and complex, systematic methods for Failure Mode and Effect Management (FMEM) system design are very much in need. Such FMEM strategies must be able to reconfigure system operation to maintain safety and maximize system availability.

This chapter presents an approach to designing an FMEM strategy for handling actuator failures, where actuators can fail by getting stuck in a constant position. The proposed FMEM system aims to guarantee that pointwise-in-time state and control constraints are satisfied during normal operation, in failure modes, and during mode transitions.

There is much literature on the design of fault-tolerant control systems and on the analysis of the ability to reconfigure, such as [13] and [15]; usually, state and control constraints are not

considered. Constraint handling using control methods similar to ours in this chapter is addressed in [17–19], but failure modes are not considered. In [20], the case of stuck actuators is handled using set-theoretic methods; however, the reference tracking problem is not treated. The use of reference governors for fault-tolerant control is described in [16] and references therein, but these approaches are different from ours and are not combined with the reconfiguration strategy. The use of recoverable sets for safe trim point to trim point transitions is considered in [21], but the reconfiguration is not addressed to handle sequential failures that involve multiple operating mode transitions. Fault-tolerant model predictive control (FTMPC) methods proposed in [22] handle constraints and reference tracking; however, [22] addresses unknown fault intensity instead of stuck/jammed actuators in this chapter and does not consider sequential failures. A comprehensive comparison between the FTMPC approaches and our strategy is left to future work.

In Chap. 2 (based on [25]), an FMEM strategy is proposed for the case when actuator failures result in the corresponding control input being set to zero. This strategy relies on manipulating the reference command to a nominal controller based on the use of constraint admissible and recoverable sets constructed using discrete-time linear models of the closed-loop system in each mode. A maximum constraint admissible set $\mathcal{O}_{\infty,M}$ is the set of all initial states x_0 of the system and reference commands v with which the ensuing response satisfies state and control constraints for all future time instants if operating in mode M. For each failure mode, a recoverable set $\mathcal{R}_{\infty,M}^{N_M}$ is the set of all initial states x_0 of the system for which there exists a reference command sequence $\boldsymbol{v}$ that steers the states into the state projection of $\mathcal{O}_{\infty,M}$ within N_M steps without constraint violations. Then, if the following conditions are satisfied,

$$\text{Proj}_x \, \mathcal{O}_{\infty,M'} \subseteq \mathcal{R}_{\infty,M}^{N_M} \quad \forall M' \in \text{pred}(M), \tag{3.1}$$

where M' is the predecessor mode of M, constraints can be satisfied in each mode and during mode changes. This result is established under the assumptions of a single point of failure (i.e., one actuator failure at a time), instantaneous fault detection and isolation, and large time between subsequent failures. Furthermore, Chap. 2 introduced three mechanisms by which (3.1) can be ensured: (i) by adding extra state constraints in the preceding mode; (ii) by increasing time duration allowed for the reconfiguration; and (iii) by temporarily relaxing state constraints when determining the recovery sequence. The latter mechanism is suitable for systems with soft constraints when a temporary constraint violation is permissible.

A reference governor [16] is used in Chap. 2 for reference tracking in each mode after the reconfiguration is completed and until another failure occurs. The reference governor maintains the state and modified reference in $\mathcal{O}_{\infty,M}$ while minimizing the difference between the modified references and reference commands. When a failure occurs and the reconfiguration begins, the

reference governor operation is suspended. A constrained quadratic programming problem is solved to generate a recovery sequence of modified references, which is then applied until the states enter the constraint admissible set of the current mode, and the operation of the reference governor resumes.

In this chapter, the approach of Chap. 2 is extended to address the practically important case when actuators can get stuck/jammed at a constant position. This requires modifying control laws in each mode to compensate for failed actuator positions, as well as re-formulating the constraint admissible sets, recoverable sets, and reconfiguration conditions to include the dependence on the failed actuator positions. The case treated in this chapter is significantly more general than that in Chap. 2 where inputs corresponding to failed actuators were set to zero.

The proposed FMEM strategy (as shown by Fig. 3.1), upon failure, first modifies the reference command by generating a recovery sequence, and then by using a reference governor once the state enters the maximum constraint admissible set (as shown by Fig. 3.2). The controller responding to the modified reference command is also switched for each specific mode.

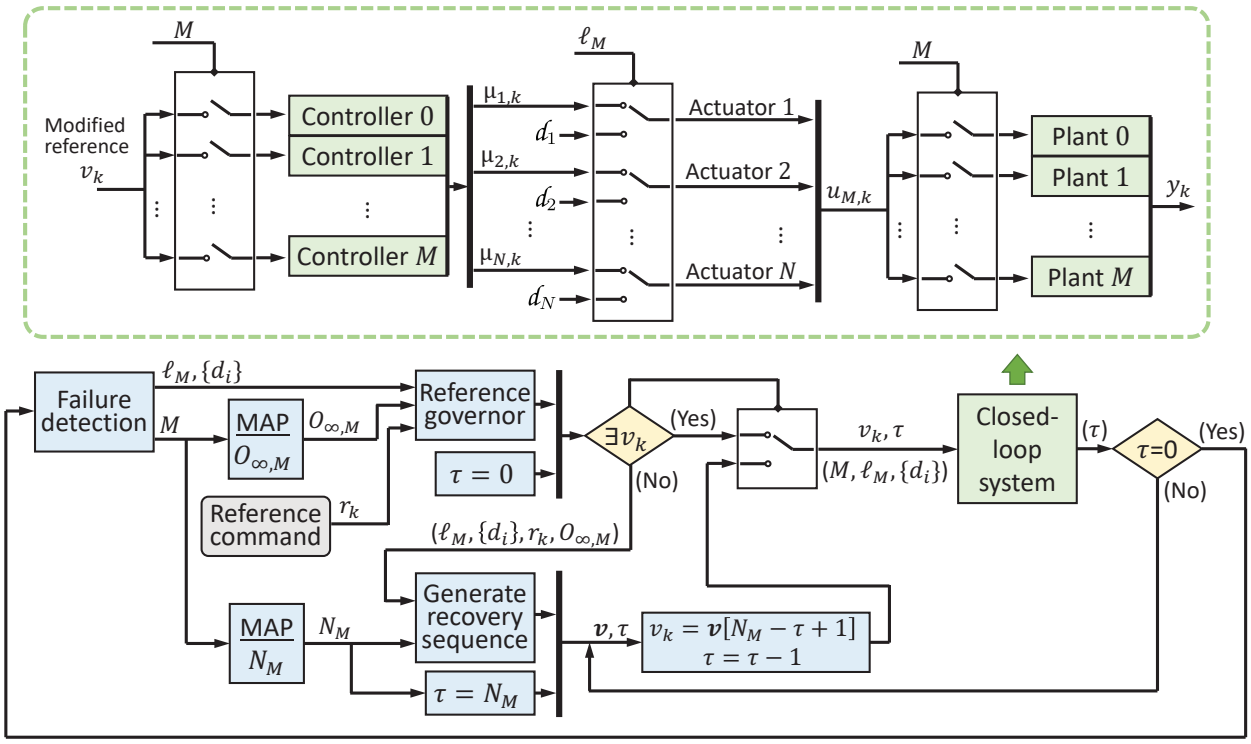

Figure 3.1: A flowchart of the proposed FMEM strategy. Signals in round brackets are passed from previous blocks.

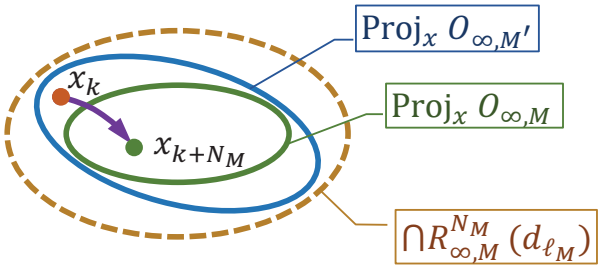

Figure 3.2: The relation between offline designed constraint admissible and recoverable sets exploited in the FMEM strategy.

The rest of the chapter is organized as follows. First, normal and failure modes, open and closed-loop discrete-time system models, as well as constraints are introduced in Sect. 3.2. Section 3.3 defines the problem statement, constraint admissible and recoverable sets, reconfiguration conditions for different mode transitions, and reference governors for reference tracking. Numerical results for an aircraft longitudinal flight application are reported in Sect. 3.4. Finally, conclusions are drawn in Sect. 3.5.

3.2 Operation Modes, System Dynamics, and Constraints

3.2.1 Normal and Failure Modes

Consider an over-actuated system with N actuators. Each actuator may fail because it gets stuck/jammed and generates a constant input equal to the value immediately before the failure happens. Failures of multiple actuators are possible, but only one actuator can fail at a time. The time between subsequent failures is assumed to be large. Furthermore, to simplify the exposition, we assume that the failure is detected instantaneously and the failed actuator position is accurately measured/estimated, while in practice these assumptions can be relaxed.

Figure 3.3 shows an example of potential failure paths for the case of $N = 2$ in the worst case scenario when every combination of actuator failures is possible. Each box represents a potential operating mode of the system. The numbers inside the box are labels of the actuators that are still working. Each mode is denoted by $M \in \{0, 1, \cdots, 2^N\}$, and a vector ℓ_M is defined to label each

mode by

$$\ell_M = \begin{bmatrix} m_1 & m_2 & \cdots & m_N \end{bmatrix}^T,$$

$$m_i = \begin{cases} 0 & \text{if the } i\text{th actuator failed,} \\ 1 & \text{if the } i\text{th actuator works,} \end{cases}$$

for $i \in \{1, \cdots, N\}$. In particular,

$$\ell_0 = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T,$$
$$\ell_\Omega = \begin{bmatrix} 0 & 0 & \cdots & 0 \end{bmatrix}^T,$$

where ℓ_0 is the label of the normal mode when all actuators work, and ℓ_Ω with $\Omega = 2^N$ labels the mode when all actuators fail.

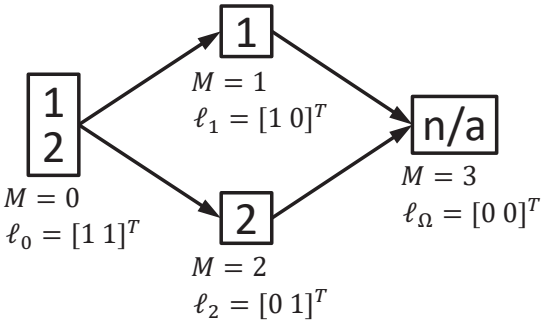

Figure 3.3: Example of failure paths, modes M, and labels ℓ_M for $N = 2$.

3.2.2 Open-Loop and Closed-Loop System Models

We assume that the system in mode M is represented by the following discrete-time linear model:

$$x_{k+1} = Ax_k + Bu_{M,k}, \tag{3.2}$$
$$y_k = Cx_k, \tag{3.3}$$

where x_k is the state, y_k is the output, and $u_{M,k}$ is the input. We have

$$u_{M,k} = \ell_M \odot u_{0,k} + (\ell_0 - \ell_M) \odot u_{\Omega,k} \;\; \forall 0 \leq M \leq \Omega,$$

where

$$u_{0,k} = \begin{bmatrix} \mu_{1,k} & \mu_{2,k} & \cdots & \mu_{N,k} \end{bmatrix}^T,$$
$$u_{\Omega,k} = \begin{bmatrix} \mathcal{d}_1 & \mathcal{d}_2 & \cdots & \mathcal{d}_N \end{bmatrix}^T,$$

and where $\mu_{i,k}$ is the input in the ith channel where the corresponding actuator is working properly, $\mathcal{d}_i$ is the constant input in the ith channel where the corresponding actuator failed and got stuck, and $\odot$ is the element-wise product. For example, in the case with three actuators, if

$$\ell_1 = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^T,$$

we have

$$u_{1,k} = \begin{bmatrix} \mu_{1,k} & \mu_{2,k} & \mathcal{d}_3 \end{bmatrix}^T.$$

In the normal mode ($M = 0$), a stabilizing feedback plus feedforward controller is used, of the form,

$$u_{0,k} = K_0 x_k + G_0 v_k, \tag{3.4}$$

where K_0 is a stabilizing feedback gain, G_0 is a feedforward gain such that the tracking error in steady state is equal to zero, and v_k is the reference command.

In failure modes M, $0 < M < \Omega$, when there are actuators stuck at constant values, we first define $\mu_{\ell_M,k}$ as the vector of inputs of the working actuators and d_{ℓ_M} as the vector of constant inputs of the failed actuators by reducing $u_{M,k}$ to

$$\mu_{\ell_M,k} = \diamondsuit(\ell_M, u_{M,k}),$$
$$d_{\ell_M} = \diamondsuit(\ell_0 - \ell_M, u_{M,k}),$$

where $\diamondsuit(\ell, E)$ is an operator that reduces the dimension of the vector, matrix, or product of sets E by removing the ith, $i \in \{1, \cdots, N\}$, element, row, or set if the corresponding ith element in ℓ is zero.

It is assumed that in all failure modes the number of references is less than or equal to the number of working actuators. Then for each mode M, a stabilizing controller is assumed to have been designed, and given by

$$\mu_{\ell_M,k} = K_M x_k + G_M v_k + H_M d_{\ell_M}, \tag{3.5}$$

where K_M is the stabilizing feedback gain and G_M and H_M are the feedforward gains for the reference commands and stuck inputs. They are designed so that the system has zero steady-state tracking error.

In general, the closed-loop dynamics are represented by

$$x_{k+1} = \bar{A}_M x_k + \bar{B}_M U_{M,k}, \tag{3.6}$$

where $U_{M,k}$ is the closed-loop input, and

- for $M = 0$,

$$\begin{aligned} \bar{A}_0 &= A + BK_0, \\ \bar{B}_0 &= BG_0, \\ U_{0,k} &= v_k, \end{aligned}$$

- for $0 < M < \Omega$,

$$\begin{aligned} \bar{A}_M &= A + B_{M,\mu} K_M, \\ \bar{B}_M &= \begin{bmatrix} B_{M,\mu} G_M & B_{M,\mu} H_M + B_{M,d} \end{bmatrix}, \\ U_{M,k} &= \begin{bmatrix} v_k \\ d_{\ell_M} \end{bmatrix}, \end{aligned}$$

  where

$$\begin{aligned} B_{M,\mu} &= \left(\Diamond(\ell_M, B^T) \right)^T, \\ B_{M,d} &= \left(\Diamond(\ell_0 - \ell_M, B^T) \right)^T. \end{aligned}$$

In the failure mode when all actuators fail and give constant inputs ($M = \Omega$), the system runs in open-loop. In this case we assume that the open-loop system is stable as otherwise state constraints cannot be handled. For consistency of notations, we let

$$\begin{aligned} \bar{A}_\Omega &= A, \\ \bar{B}_\Omega &= B, \\ U_{\Omega,k} &= u_{\Omega,k}. \end{aligned}$$

3.2.3 Constraints

To ensure safe operation, pointwise-in-time state and control constraints are imposed given by inequalities of the form,

$$x_k \in \mathcal{X}_M^*(U_{M,k}) = \{x : \mathcal{A}_M^* \begin{bmatrix} x \\ U_{M,k} \end{bmatrix} \leq \mathcal{b}_M^*\}. \tag{3.7}$$

It is assumed that individual open-loop input range constraints of the form $u_{M,k} \in D_1 \times \cdots \times D_N$ are reflected in (3.7) where D_i, $i = 1, \cdots, N$, is the feasible input range for the ith actuator.

Modifications of constraints may be needed to facilitate the sequential failure mode reconfiguration design. Firstly, to satisfy subsequent conditions for the safe reconfiguration, it may be necessary to restrict the operation in preceding modes by artificially tightening constraints (3.7) to

$$\begin{aligned} x_k \in \mathcal{X}_M(U_{M,k}) = & \ \mathcal{X}_M^*(U_{M,k}) \cap \bar{\mathcal{X}}_M(U_{M,k}) \\ = & \ \{x : \mathcal{A}_M \begin{bmatrix} x \\ U_{M,k} \end{bmatrix} \leq \mathcal{b}_M\}, \end{aligned} \tag{3.8}$$

where the sets $\bar{\mathcal{X}}_M(U_{M,k})$ need to be appropriately designed.

Secondly, in practical applications, some of the state constraints could be imposed conservatively to extend the system operating life and can be relaxed temporarily during the recovery to, for example, reduce the number of steps needed for the recovery. Thus, during the short period when the recovery sequence is applied, the constraints can be relaxed to

$$x_k \in \mathcal{X}_{R,M}(U_{M,k}) = \ \{x : \mathcal{A}_{R,M} \begin{bmatrix} x \\ U_{M,k} \end{bmatrix} \leq \mathcal{b}_{R,M}\}, \tag{3.9}$$

where $\mathcal{X}_{R,M}(U_{M,k}) \supseteq \mathcal{X}_M(U_{M,k})$, and $\mathcal{X}_{R,M}(U_{M,k})$ also need to be appropriately designed.

3.3 Reconfiguration Strategy

The FMEM strategy has two basic goals. Firstly, constraints should not be violated at any time so that safety is preserved. The system needs to operate safely in all modes and during mode transitions. Secondly, the system output should follow the given reference command as closely as possible. The proposed strategy exploits constraint admissible sets, recoverable sets, and reference governors.

3.3.1 Constraint Admissible Sets

For $M \in \{0, \cdots, \Omega\}$, the maximum constraint admissible sets are defined by

$$\begin{aligned}\mathcal{O}_{\infty,M} =&\{(U_M, x_0) : \ x_t \in \mathcal{X}_M(U_M) \ \forall t \in \mathbb{Z}_{\geq 0}, \ x_{M,\mathrm{ss}}(U_M) \oplus \mathcal{B}_\epsilon \subset \mathcal{X}_M(U_M)\} \\ =&\{(U_M, x_0) : \ \mathcal{A}_{\mathcal{O}_{\infty,M}} x_0 \leq \mathcal{b}_{\mathcal{O}_{\infty,M}}(U_M)\},\end{aligned} \tag{3.10}$$

where x_t is the response of (3.6) to the initial condition x_0 and constant closed-loop input U_M, $x_{M,\mathrm{ss}}(U_M)$ is the steady-state point given by

$$x_{M,\mathrm{ss}}(U_M) = (I - \bar{A}_M)^{-1} \bar{B}_M U_M,$$

and $\mathcal{B}_\epsilon$ is an open ball of radius $\epsilon > 0$. By adding the constraint $x_{M,\mathrm{ss}}(U_M) \oplus \mathcal{B}_\epsilon \subset \mathcal{X}_M(U_M)$, one can ensure that, under mild additional assumptions, the set $\mathcal{O}_{\infty,M}$ is positively invariant, finitely determined, and can be represented by a finite set of affine inequalities [45]. Positive invariance means that if $(U_M, x_0) \in \mathcal{O}_{\infty,M}$, then $(U_M, x_t) \in \mathcal{O}_{\infty,M}$ for all future time instants $t \geq 0$ as long as the mode remains equal to M. Being finitely determined means that there exists $T \in \mathbb{Z}_{\geq 0}$ such that $x_t \in \mathcal{X}_M(U_M) \ \forall t \leq T$ is equivalent to $(U_M, x_0) \in \mathcal{O}_{\infty,M}$. As a result, it is possible to represent $\mathcal{O}_{\infty,M}$ by a finite set of inequalities as in (3.10) (see, e.g., [21] for derivation).

3.3.2 Recoverable Sets

The recoverable sets for $M \in \{1, \cdots, \Omega - 1\}$ are defined as

$$\begin{aligned}\mathcal{R}^{N_M}_{\infty,M}(d_{\ell_M}) = \{x_0 : \ &\exists \{v_0, \cdots, v_{N_M}\} \text{ such that} \\ &x_t \in \mathcal{X}_{R,M}(U_{M,t}) \ \forall t = 0, 1, \cdots, N_M - 1, \ (U_{M,N_M}, x_{N_M}) \in \mathcal{O}_{\infty,M}\},\end{aligned} \tag{3.11}$$

where N_M is the number of steps allowed for the reconfiguration, which is a design parameter. We let $\boldsymbol{v} = \begin{bmatrix} v_0^T & \cdots & v_{N_M}^T \end{bmatrix}^T$ designate the reference sequence.

In mode Ω, when all actuators fail, the system runs open-loop, so it needs to be already running inside $\mathcal{O}_{\infty,\Omega}$ in the immediate preceding modes. To keep the consistency of notations, we let

$$\mathcal{R}^{N_\Omega}_{\infty,\Omega} = \mathrm{Proj}_x \ \mathcal{O}_{\infty,\Omega} \quad \forall N_\Omega \geq 0, \tag{3.12}$$

where

$$\mathrm{Proj}_x \ \mathcal{O}_{\infty,M} = \{x_0 : \ \exists U_M \text{ such that } (U_M, x_0) \in \mathcal{O}_{\infty,M}\}.$$

3.3.3 Reconfiguration Conditions

The safe reconfiguration conditions that follow are based on an extension of (3.1). If these conditions are satisfied, upon failure, the states of the system are in the state projection of the constraint admissible set of the predecessor mode, which is a subset of the recoverable set of the successor mode. Therefore, the states are guaranteed to be inside the recoverable set of the successor mode, and there exists a feasible sequence of references for the reconfiguration.

For transitions to mode $M \in \{1, \cdots, \Omega - 1\}$ from mode $M' \in \text{pred}(M)$, where pred(M) is the set of all predecessor modes that can change to the successor mode M due to a single actuator failure, it suffices to require that

$$\text{Proj}_x \, \mathcal{O}_{\infty,M'} \subseteq \bigcap_{d_{\ell_M} \in D_{\ell_M}} \mathcal{R}^{N_M}_{\infty,M}(d_{\ell_M}), \tag{3.13}$$

where

$$D_{\ell_M} = \Diamond(\ell_0 - \ell_M, D_1 \times \cdots \times D_N)$$

is the set of control constraints for d_{ℓ_M}.

For transitions to mode Ω from $M' \in \text{pred}(\Omega)$, since none of the actuators work, we have to ensure that

$$\mathcal{O}_{\infty,M'} \subseteq \mathcal{O}_{\infty,\Omega}, \tag{3.14}$$

for all $M' \in \text{pred}(\Omega)$.

3.3.4 Safe Reconfiguration Upon Failure Detection

At the beginning of the reconfiguration and mode transition to mode $M \in \{1, \cdots, \Omega - 1\}$, the recovery reference sequence $\boldsymbol{v}$ is computed by solving the following quadratic programming (QP) problem

$$\begin{aligned} &\min_{\boldsymbol{v}} \|r_k \mathbf{1} - \boldsymbol{v}\|^2 \\ \text{s.t. } &x_t \in \mathcal{X}_{R,M}(U_{M,t}) \ \ \forall t = k, \cdots, k + N_M - 1, \\ &(U_{M,k+N_M}, x_{k+N_M}) \in \mathcal{O}_{\infty,M}, \end{aligned} \tag{3.15}$$

where r_k equals to the given reference command at the beginning of the transition (at time step k). Then the system runs for N_M steps using the recovery sequence of modified references $\boldsymbol{v}$ until the

end of the reconfiguration.

3.3.5 Reference Governor

A reference governor (RG) is used for reference tracking after the reconfiguration is completed. With the system running in mode $M \in \{0, \cdots, \Omega - 1\}$, the reference governor determines the modified reference based on the solution of the following optimization problem

$$\begin{aligned} &\min_{v_k} \|r_k - v_k\|^2 \\ &\text{s.t. } (U_{M,k}, x_k) \in \mathcal{O}_{\infty,M}. \end{aligned} \tag{3.16}$$

The modified reference v_k is then updated at every time step. By construction, the following result is obtained:

Proposition 3.1 (Reconfiguration Condition)**.** *For system operation in arbitrary mode* $M \in \{1, \cdots, \Omega - 1\}$*, if* (3.13) *is satisfied, the time between failures is larger than* N_M*, and the safe reconfiguration strategy and reference governor based on* (3.15) *and* (3.16) *are used, then constraints given by* (3.9) *and* (3.8) *are satisfied respectively during the reconfiguration and after the recovery is completed.*

Remark 3.1. Note that if (3.13) is satisfied, the system can operate safely not only when the input is stuck at a constant value immediately preceding the failure, but even if it instantaneously jumps to another constant value at the time of failure as long as $d_{\ell_M} \in D_{\ell_M}$, such as for the zero control considered in Chap. 2 where (3.13) coincides with (3.1) by having $d_{\ell_M} \in D_{\ell_M} = \{\mathbf{0}\}$ and $\mathcal{R}_{\infty,M}^{N_M} = \mathcal{R}_{\infty,M}^{N_M}(d_{\ell_M})$.

Remark 3.2. The FMEM strategy is developed offline and involves choosing variables N_M, $\bar{\mathcal{X}}_M(U_{M,k})$, and $\mathcal{X}_{R,M}(U_{M,k})$ to satisfy (3.13) as in Fig. 3.2 for each mode transition. Then during the online operation (as shown in Fig. 3.1), the modified reference command is generated by (3.15) or (3.16). Note that the online chronometric load would not regularly increase with the increase in the number of redundant actuators or possible failure paths as the underlying QP problems are only relevant to the current mode. However, the read-only memory (ROM) size necessary to store sets for different modes can grow.

3.4 Application to Aircraft Longitudinal Flight Control

3.4.1 System Dynamics

3.4.1.1 Open-Loop Model

The aircraft model represents a Boeing 747-100 aircraft in steady level flight corresponding to Mach 0.5 cruise at 20,000 feet [47]. The linearized longitudinal flight dynamics under the normal operating conditions with the classical phugoid approximation are modeled by

$$\begin{aligned}\Delta\dot{u} &= -0.0075\Delta u - 0.1713\Delta\theta + \Delta a_T - 0.0051\Delta\delta_e,\\ \Delta\dot{\theta} &= 0.0436\Delta u - 0.0423\Delta\delta_e,\end{aligned}$$

and the output is

$$\Delta\dot{h} = 2.7645\Delta\theta,$$

where Δ denotes the deviation from the trim value, u is the projection of the velocity vector on the x-axis of the body-fixed frame in $\mathrm{m/s}$, θ is the pitch angle in $^\circ$ (degree), a_T is the thrust-to-mass ratio in $\mathrm{N/kg}$, δ_e is the elevator deflection in $^\circ$, and $\dot{h}$ is the climb rate in $\mathrm{m/s}$.

The system can be written compactly in the form,

$$\dot{x} = Ax + Bu, \tag{3.17a}$$

$$y = Cx, \tag{3.17b}$$

where $x = [\Delta u \;\; \Delta\theta]^T$ is the state vector, $u = [\Delta a_T \;\; \Delta\delta_e]^T$ is the input vector, and $y = \Delta\dot{h}$ is the output. This model is converted to discrete-time assuming a 5-second update period. Note that this system has two redundant actuators, hence $N = 2$ as in Fig. 3.3.

3.4.1.2 Closed-Loop Model

The controllers (3.4) for $M = 0$ and (3.5) for $M \in \{1, 2\}$ are designed using Linear Quadratic Regulator (LQR) theory with the *dlqr* command in `Matlab` to obtain the feedback gain K_M. The feedforward gains G_M and H_M are computed so that the steady-state gain from the reference v to the output y for the climb rate deviation is equal to 1. The weights Q_M and R_M are chosen by trial and error and assuming that the use of thrust is more expensive than the use of the elevator. Their values and those of the resulting G_M and H_M matrices are as follows:

- In Mode 0, the controller is designed to track a given climb rate deviation from nominal and a

zero deviation of the velocity magnitude from nominal, that is, to hold u, using

$$
\begin{aligned}
Q_0 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \\
R_0 &= \begin{bmatrix} 1000 & 0 \\ 0 & 250 \end{bmatrix}, \\
G_0^* &= \left(\begin{bmatrix} 0 & 2.7645 \\ 1 & 0 \end{bmatrix} (I - \bar{A}_M)^{-1} B \right)^{-1} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \\
G_0 &= \left(\Diamond([1\ 0]^T, G_0^{*T}) \right)^T.
\end{aligned}
$$

- In Modes 1 and 2 (for $M \in \{1, 2\}$), the controller only tracks a given climb rate deviation reference, with

$$
\begin{aligned}
Q_1 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ R_1 = \begin{bmatrix} 10 & 0 \\ 0 & 0 \end{bmatrix}, \\
Q_2 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ R_2 = \begin{bmatrix} 0 & 0 \\ 0 & 2.5 \end{bmatrix}, \\
G_M &= [C(I - \bar{A}_M)^{-1} B_{M,\mu}]^{-1}, \\
H_M &= -G_M [C(I - \bar{A}_M)^{-1} B_{M,d}]^{-1}.
\end{aligned}
$$

The closed-loop systems are then obtained based on (3.6).

3.4.2 Constraints

Constraints are imposed on the climb rate, the thrust-to-mass ratio, and the elevator deflection as

$$|\Delta \dot{h}| \leq y_{\max},\ |\Delta a_T| \leq u_{1_{\max}}, \text{ and } |\Delta \delta_e| \leq u_{2_{\max}}, \tag{3.18}$$

where $y_{\max} = 20\,\mathrm{m/s}$, $u_{1_{\max}} = 2\,\mathrm{N/kg}$, and $u_{2_{\max}} = 45°$. These constraints define $\mathcal{X}_M^*(U_{M,k})$ given by (3.7) for all modes of $M \in \{0, 1, 2, 3\}$.

3.4.3 Design of Constraint Admissible and Recoverable Sets

3.4.3.1 Constraint Admissible Sets

The design begins from Mode 3. During simulation, constraints for the failed inputs d_1 and d_2 by (3.18) are automatically satisfied due to the same constraints being applied to the working inputs $\mu_{1,k}$ and $\mu_{2,k}$ in preceding modes. However, the input constraints are still used to define $\mathcal{X}_3^*(U_{3,k})$ so that $\mathcal{O}_{\infty,3}$ is a bounded set. Since Mode 3 is the last mode of the failure sequence, no subsequent conditions need to be considered, so for (3.8), let $\mathcal{X}_3(U_{3,k}) = \mathcal{X}_3^*(U_{3,k})$. Then, the

constraint admissible set of Mode 3, as defined in (3.10), can be represented by

$$\mathcal{O}_{\infty,3} = \left\{(d_1, d_2, x_0) : \mathcal{A}_{O_3,d_1} d_1 + \mathcal{A}_{O_3,d_2} d_2 + \mathcal{A}_{O_3,x_0} x_0 \leq b_{O_3}\right\}. \tag{3.19}$$

Next, in Modes 1 and 2 ($M \in \{1, 2\}$), in order to satisfy the condition (3.14) for transitions to Mode 3, the state constraints can be tightened by imposing constraints from $\mathcal{O}_{\infty,3}$, that is, to have

$$x_0 \in \{x_0 : \mathcal{A}_{O_3,d_M} \mu_{M,k} + \mathcal{A}_{O_3,d_{\ell_M}} d_{\ell_M} + \mathcal{A}_{O_3,x_0} x_0 \leq b_{O_3}\}, \tag{3.20}$$

where $\mu_{M,k}$ is generated by the controller in (3.5), as a function of $U_{M,k}$. By substituting $\mu_{M,k}$ in (3.20) by (3.5), we define

$$\begin{aligned}\bar{\mathcal{X}}_M(U_{M,k}) = \{x_0 : \mathcal{A}_{O_3,d_M} G_M v_k + (\mathcal{A}_{O_3,d_M} H_M + \mathcal{A}_{O_3,d_{\ell_M}}) d_{\ell_M} \\ + (\mathcal{A}_{O_3,d_M} K_M + \mathcal{A}_{O_3,x_0}) x_0 \leq b_{O_3}\}.\end{aligned} \tag{3.21}$$

Then, $\mathcal{X}_M(U_{M,k})$ and $\mathcal{O}_{\infty,M}$ are defined based on (3.8) and (3.10).

Finally, to satisfy the recovery conditions of (3.13) for mode transitions between Modes 0 and 1 and 0 and 2, the constraints are tightened by a scaling coefficient of $\eta_O \in (0, 1]$ that needs to be tuned (beginning from 1 and decreasing until the recovery conditions are satisfied). Then, $\mathcal{X}_0(U_{0,k})$ is defined by these tightened constraints as

$$\mathcal{X}_0(U_{0,k}) = \{x_0 : \mathcal{A}_0^* \begin{bmatrix} x \\ U_{0,k} \end{bmatrix} \leq \eta_O b_0^*\}, \tag{3.22}$$

and $\mathcal{O}_{\infty,0}$ is defined based on (3.10).

3.4.3.2 Recoverable Sets

The recoverable sets for Modes 1 and 2 ($M \in \{1, 2\}$) are designed based on (3.13).

The state constraints during recovery are temporarily relaxed by having

$$\mathcal{X}_{R,M}(U_{M,k}) = \{x : \mathcal{A}_{M,X} \begin{bmatrix} x \\ U_{M,k} \end{bmatrix} \leq \eta_{R,M} b_{M,X}\}, \tag{3.23}$$

where $\eta_{R,M} \geq 1$ is a design parameter. Then, the recoverable sets are constructed based on (3.11).

3.4.3.3 Choosing Design Parameters to Satisfy Recovery Conditions

The final tuning results are $\eta_O = 0.7$, $N_1 = 5$, $\eta_{R,1} = 1.9$, $N_2 = 5$, and $\eta_{R,2} = 1$. Conditions (3.13) are checked using the toolbox `Bensolve` [46]. Figure 3.4 shows the state projections of the $\mathcal{O}_{\infty,M}$ sets for all modes, compared with the intersection of sets $\mathcal{R}_{\infty,M}^{N_M}(d_{\ell_M})$ for all $d_{\ell_M} \in D_{\ell_M}$ for $M = 1$ and 2. It can be seen that the reconfiguration conditions (3.13) are satisfied. By checking the set relations of $\mathcal{O}_{\infty,1}$ and $\mathcal{O}_{\infty,2}$ versus $\mathcal{O}_{\infty,3}$ using the `Bensolve` command *le*, it has been confirmed that the conditions (3.14) are also satisfied.

3.4.4 Simulation Results

Figure 3.4 shows the state trajectories and time-based trajectories of the major signals of a simulation with mode switching from 0 to 2 to 3 at times 0 and 225. In order to demonstrate the reconfiguration process, an initial condition is picked such that $([v_0 \ \mu_{1,0}]^T, x_0) \notin \mathcal{O}_{\infty,2}$ but $x_0 \in \mathcal{R}_{\infty,2}^{N_2}(\mu_{1,0})$, and the simulation starts at the transition from Mode 0 to Mode 2.

The system runs from 0 to 25 seconds with the modified reference sequence $\{v_t\}_0^{N_2}$ generated by the recovery sequence generator. The thrust is stuck while the elevator is working normally. At 25 seconds, the state is steered inside $\mathcal{O}_{\infty,2}$, so after that, the reference governor updates the reference at every time step, until 225 seconds when the mode switches from 2 to 3, in which both actuators fail. Since $x_0 \in \mathcal{O}_{\infty,2}$ prior to the transition, the system continues to operate without constraint violation in open-loop after it switches to Mode 3.

3.5 Concluding Remarks

Failure Mode and Effect Management (FMEM) systems need to be properly designed to avoid safety constraint violations when failures occur. Redundant actuators can be exploited to improve system reliability, but failures could happen due to one or multiple actuators being stuck. A set-theoretic based strategy for system reconfiguration was presented, using constraint admissible and recoverable sets together with the use of a reference governor that enables tracking of references. The strategy considers sequential transitions of normal and failure modes, guaranteeing safe operations in all modes as well as during mode transitions. Mechanisms to help satisfy the reconfiguration conditions were illustrated through a numerical example of an aircraft longitudinal flight control application. By requiring that the recovery sequence be simultaneously feasible for multiple modes, the present approach can be extended to the case when failure detection and isolation are not instantaneous. In such a setting, generating a recovery control sequence directly rather than a modified reference command sequence could be more straightforward.

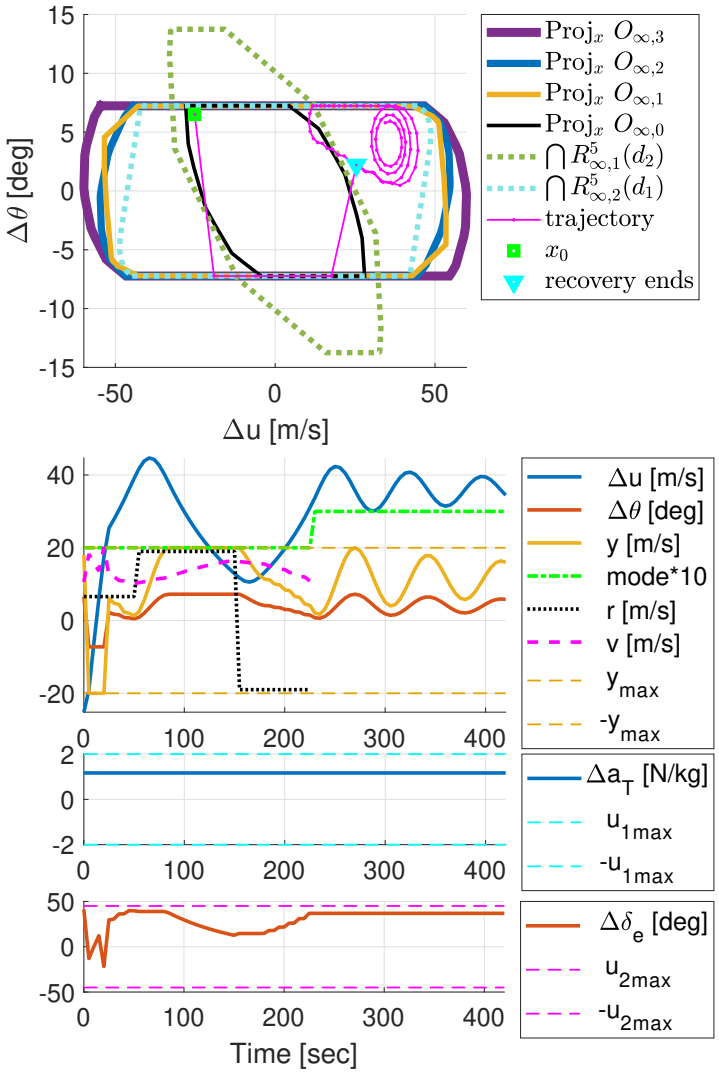

Figure 3.4: Aircraft example simulation results (initial states are picked such that recovery is needed).

CHAPTER 4

Integrating Failure Detection and Isolation into a Reference Governor-Based Reconfiguration Strategy for Stuck Actuators

4.1 Introduction

Stuck/jammed actuators, unless properly handled by a Failure Mode and Effect Management (FMEM) system, can result in a marked degradation of system performance and safety, with notable recent examples in [4, 5]. Systems that are highly automated and designed to fulfill complex missions often exploit actuator redundancy and comprehensive failure detection and isolation (FDI) and fault-tolerant control (FTC) schemes to ensure reliability and safety. To handle the multitude of potential failure modes, a systematic development process of such FMEM strategies that is capable of handling sequential failures and has guaranteed properties by design is highly desirable.

In this chapter, we develop several enhancements to the reference governor-based set-theoretic FMEM that was introduced in Chap. 3 (based on [26]). This strategy handles stuck/jammed actuators while tracking reference commands. It relies on the reference governor for operating the system subject to constraints in different operating modes and generates a recovery reference command sequence once failure is detected to effect the transition into the new operating mode with a different number of functioning actuators. The proposed enhancements include the integration of a scheme that detects and isolates failures in finite time (thereby relaxing the negligibly small time assumption for FDI in Chaps. 2 and 3) before the recovery command sequence is generated. Additionally, set-bounded unmeasured disturbance inputs are handled.

The literature on FDI and failure mode reconfiguration is extensive, see e.g., [13–15]; however, reference command tracking in presence of state and control constraints is often not the focus and the design of the system to handle sequential failures is frequently not considered. In the reference command tracking setting, several reference governor-based approaches have been proposed such as in [16, 48] and references therein, with the approach closest to ours being [48]. A notable difference

is that in Chaps. 2, 3 and this chapter, we pursue a deterministic treatment (based on set-theoretic control) that leads to a possibility of FDI and failure mode reconfiguration in finite time rather than asymptotically in a probabilistic sense, while guaranteeing that the constraints are enforced in all phases of failure detection, isolation, reconfiguration and when operating in the new mode under the reference governor supervision. Other approaches utilizing set-theoretic methods in systems with failures are proposed in, e.g., [20, 21].

Fault detection and diagnosis, isolation, identification, and reconfiguration are essential components of active FTC [49] but they are often studied separately due to the complexity of the individual problems. Notably, existing approaches often do not guarantee explicitly (i.e., by design) the ability to isolate failures and reconfigure the system operation while satisfying constraints. Furthermore, the time required to isolate failures and reconfigure the system operation may not be a priori bounded or even finite. For example, in [23], a fault-tolerant model predictive control (FTMPC) strategy is developed based on similar to ours set-theoretic considerations, yet with important differences. In particular, we focus on the approach that is based on the reference governor for safe command tracking. Furthermore, we guarantee explicitly by design that failures are isolatable and the system can be reconfigured into the new mode within a finite and a priori known duration. We note that in the isolation and reconfiguration phases, the original constraints may need to be temporarily relaxed but the amount of this relaxation is determined by the designer as a part of the offline design process to ensure existence of the solution. For the same reason, in certain modes constraints may need to be tightened to ensure the system state is within the capability of the FDI and reconfiguration scheme and ultimately the next mode controller to handle.

To highlight the contributions, we proposed a set membership-based reference governor-centric FTC strategy that simultaneously (1) guarantees the safe operation in phases of FDI and system reconfiguration and in all normal and failure modes, (2) guarantees abilities and has finite duration that is known before the online operation for isolation and reconfiguration, (3) enables reference tracking, and (4) handles sequential stuck/jammed actuators failures for systems with set-bounded unmeasured disturbances.

The remainder of the chapter is structured as follows. First, the system, its operating modes, and constraints are described in Sect. 4.2. Then, Sect. 4.3 introduces our approach to FDI and failure mode reconfiguration and the online computations involved, followed by the description of the offline design that guarantees the isolability and reconfigurability in Sect. 4.4. A numerical example is reported in Sect. 4.5. Concluding remarks are made in Sect. 4.6.

4.2 Preliminaries: Operating Modes, System Dynamics and Constraints

4.2.1 Operating Modes

Consider a system with N redundant actuators. Each actuator may fail by being stuck/jammed at a constant position. We assume that only one failure can occur at a time, and there is sufficient time between sequential failures for FDI and failure mode reconfiguration. Then, there are $\Omega = 2^N$ possible modes in total, corresponding to different combinations of stuck actuators. Each operating mode is labeled using $M \in \{0, 1, \cdots, \Omega\}$ where, in particular, the normal mode $M = 0$ has all actuators working properly, and the failure mode $M = \Omega$ has all actuators failed.

4.2.2 System Dynamics

We consider a system with a set-bounded input disturbance and discuss the open-loop dynamics, stabilizing nominal controllers, and system response representations in what follows.

4.2.2.1 Open-Loop Systems

Discrete-time linear models are considered to represent system dynamics in each mode. Three representations of the dynamics are used as follows:

$$x_{k+1} = A_M x_k + B_M u_k + F_M w_k, \tag{4.1}$$

$$= A_M x_k + B_{M,\mu}\mu_{M,k} + B_{M,d} d_M + F_M w_k, \tag{4.2}$$

$$= A_M x_k + B_M^{\mu} u_k + B_{M,d} d_M + F_M w_k, \tag{4.3}$$

$$y_k = C_M x_k, \tag{4.4}$$

where M is the mode, x_k is the state, y_k is the output, and $w_k \in \mathcal{W}$ is a set-bounded unmeasured input that can represent the disturbance where $\mathcal{W}$ is assumed to be a closed polytope with $0 \in \texttt{int}\mathcal{W}$. It is assumed that the exact value of the state can be measured without noise. The first representation (4.1) is a general state-space representation where $u_k \in \mathcal{U}$ is the control input and B_M is the input matrix. In (4.2), $B_M u_k$ is split into two parts. The first part $B_{M,\mu}\mu_{M,k}$ is for the working actuators where $\mu_{M,k}$ is the vector of the working actuator inputs. The second part $B_{M,d} d_M$ is for the failed actuators where $d_M \in \mathcal{D}_M$ is the vector of constant inputs of the stuck/jammed actuators. For $B_M^{\mu} u_k$ in (4.3), the columns of B_M corresponding to the failed actuators are set to zero to form B_M^{μ} so that $B_M^{\mu} u_k$ equals $B_{M,\mu}\mu_{M,k}$. Note that $\mu_{M,k}$ and d_M in this chapter are the same variables as $\mu_{\ell_M,k}$ and d_{ℓ_M} defined in Chap. 3.

4.2.2.2 Closed-Loop Systems

To stabilize the system and track the given references (where the number of references is assumed to be less than or equal to the number of working actuators), feedback and feedforward control are used to command the working inputs using

$$\begin{aligned}\mu_{0,k} &= K_0 x_k + G_0 v_k,\\ \mu_{M,k} &= K_M x_k + G_M v_k + H_M d_M \ \ \forall M \in \{1, \cdots, \Omega - 1\},\end{aligned}$$

where K_M is the stabilizing feedback gain, while G_M and H_M are the feedforward gains for the references and the failed actuator positions that are assumed to be measured or accurately estimated. The system is assumed to be stable in mode Ω when the system runs in open-loop so that safety constraints can be handled by restricting the operation of its predecessor modes.

Then, the closed-loop system can be represented by

$$\begin{aligned}x_{k+1} &= \bar{A}_M x_k + \bar{B}_M U_{M,k} + F_M w_k, &(4.5)\\ &= \bar{A}_M x_k + \bar{B}_{M,v} v_k + \bar{B}_{M,d} d_M + F_M w_k, &(4.6)\end{aligned}$$

where

$$\begin{aligned}U_{0,k} &= v_k,\\ U_{M,k} &= \begin{bmatrix} v_k \\ d_M \end{bmatrix} \ \ \forall M \in \{1, \cdots, \Omega - 1\},\\ U_{\Omega,k} &= d_\Omega,\end{aligned}$$

and $\bar{A}_M$ and $\bar{B}_M$ are appropriately defined (see the models in Chap. 3 for details). $\bar{B}_{M,v}$ and $\bar{B}_{M,d}$ are defined similarly to $B_{M,\mu}$ and $B_{M,d}$ by only keeping the corresponding columns to v_k and d_M from $\bar{B}_M$. Note that (4.5) and (4.6) include the model for mode Ω for the consistency of notations.

4.2.2.3 System Response Representations

Let $x_{M,k}$ denote the predicted state of the system at time instant k in mode M. Define, for $k \geq 1$, the set, $\mathcal{Q}_{M,k}$, as a polyhedral overbound on the set of states reachable at time k by applying all possible disturbance sequences when $x_{M,0} = 0$. Since

$$x_{M,k} = B_{M,d} d_M + F_M w_{k-1} + A_M (B_{M,d} d_M + F_M w_{k-2}) + \cdots + A_M^{k-1} (B_{M,d} d_M + F_M w_0),$$

it follows that

$$\mathcal{Q}_{M,k} \supseteq B_{M,d}\mathcal{D}_M \bigoplus F_M \mathcal{W} \bigoplus A_M (B_{M,d}\mathcal{D}_M \bigoplus F_M \mathcal{W}) \bigoplus \cdots \bigoplus A_M^{k-1} (B_{M,d}\mathcal{D}_M \bigoplus F_M \mathcal{W}), \tag{4.7}$$

where $\bigoplus$ stands for the Minkowski sum.

Now, consider the nominal prediction of the state with $w_k = 0 \ \forall k \geq 0$, $d_M = 0$, and the input

command $u_k^* \in \mathcal{U}$. We have

$$x_{M,k}^{\mathtt{u}} = A_M^k x_0 + A_M^{k-1} B_M^\mu u_0^* + A_M^{k-2} B_M^\mu u_1^* + \cdots + B_M^\mu u_{k-1}^*,$$

which can be condensed to the form,

$$x_{M,k}^{\mathtt{u}} = x_{M,k}^{\mathtt{u}}(U^*) = S_{M,k} x_0 + T_{M,k} U^*, \tag{4.8}$$

where

$$\begin{aligned} S_{M,k} &= A_M^k, \\ T_{M,k} &= \begin{bmatrix} A_M^{k-1} B_M^\mu & A_M^{k-2} B_M^\mu & \cdots & B_M^\mu \end{bmatrix}, \\ U^* &= \begin{bmatrix} u_0^{*T} & u_1^{*T} & \cdots & u_{k-1}^{*T} \end{bmatrix}^T. \end{aligned}$$

Note that the elements of the input command U^* sequence corresponding to the failed input channels have no impact to the system response since the corresponding columns of B_M^μ are zero.

Then, by the superposition principle for linear systems, we have

$$\begin{aligned} x_{M,k} \in \mathfrak{R}_{M,k}(U^*) &= \left\{ x_{M,k}^{\mathtt{u}}(U^*) \right\} \bigoplus \mathcal{Q}_{M,k} \\ &= \left\{ S_{M,k} x_0 + T_{M,k} U^* \right\} \bigoplus \mathcal{Q}_{M,k}. \end{aligned} \tag{4.9}$$

Since $\mathcal{W}$ is a closed polytope, it is reasonable to assume that $\mathcal{Q}_{M,k}$ in (4.7) are also polytopes. Note that the reachable sets $\mathfrak{R}_{M,k}(U^*)$ are polyhedral if $\mathcal{Q}_{M,k}$ are polyhedral.

Remark 4.1. The reachable sets above are defined based on the open-loop dynamics. To represent the reachable sets based on the closed-loop dynamics, we define a polyhedral overbound

$$\bar{\mathcal{Q}}_{M,k} \supseteq \bar{B}_{M,d} \mathcal{D}_M \bigoplus F_M \mathcal{W} \bigoplus \bar{A}_M (\bar{B}_{M,d} \mathcal{D}_M \bigoplus F_M \mathcal{W}) \bigoplus \cdots \bigoplus \bar{A}_M^{k-1} (\bar{B}_{M,d} \mathcal{D}_M \bigoplus F_M \mathcal{W}),$$

and let

$$x_{M,k}^{\mathtt{v}} = x_{M,k}^{\mathtt{v}}(V) = \bar{S}_{M,k} x_0 + \bar{T}_{M,k} V,$$

where

$$\begin{aligned} \bar{S}_{M,k} &= \bar{A}_M^k, \\ \bar{T}_{M,k} &= \begin{bmatrix} \bar{A}_M^{k-1} \bar{B}_{M,v} & \bar{A}_M^{k-2} \bar{B}_{M,v} & \cdots & \bar{B}_{M,v} \end{bmatrix}, \\ V &= \begin{bmatrix} v_0^T & v_1^T & \cdots & v_{k-1}^T \end{bmatrix}^T. \end{aligned}$$

Then, the reachable sets can be written as

$$\begin{aligned} x_{M,k} \in \mathfrak{R}_{M,k}(V) &= \left\{ x_{M,k}^{\mathtt{v}}(V) \right\} \bigoplus \bar{\mathcal{Q}}_{M,k} \\ &= \left\{ \bar{S}_{M,k} x_0 + \bar{T}_{M,k} V \right\} \bigoplus \bar{\mathcal{Q}}_{M,k}. \end{aligned} \tag{4.10}$$

4.2.3 Constraints

Pointwise-in-time state and control constraints are imposed to ensure safe operation of the system. Since the working actuator input $\mu_{M,k}$ is a function of the state x_k and the closed-loop input

$U_{M,k}$, these state and control constraints are transformed into the form:

$$\mathcal{A}_M^* \begin{bmatrix} x_k \\ U_{M,k} \end{bmatrix} \leq \mathcal{b}_M^*,$$

and the safety constraints are further defined by

$$x_k \in \mathcal{X}_M^*(U_{M,k}) = \{x : \mathcal{A}_M^* \begin{bmatrix} x \\ U_{M,k} \end{bmatrix} \leq \mathcal{b}_M^*\}. \tag{4.11}$$

To accommodate the sequential failures, the safety constraints of the predecessor modes could be artificially tightened to help the system satisfy the constraints of the subsequent operating modes and during the failure isolation and reconfiguration. On the other hand, the safety constraints could be temporarily relaxed during the failure mode isolation and reconfiguration to, for example, reduce the number of time steps needed during the mode transition, which can help reduce the online computations. Relaxing constraints during failure mode isolation and reconfiguration is possible in many applications as some of the constraints are imposed to reduce system aging and their temporary violation in emergencies does not lead to catastrophic consequences. The tightened safety constraints during the operation in each mode, as well as the relaxed safety constraints for isolation and reconfiguration are defined as follow:

$$\begin{aligned} &x_k \in \mathcal{X}_M(U_{M,k}) = \mathcal{X}_M^*(U_{M,k}) \cap \bar{\mathcal{X}}_M(U_{M,k}), \\ &x_k \in \mathcal{X}_{I,M}(U_{M,k}) \text{ where } \mathcal{X}_{I,M}(U_{M,k}) \supseteq \mathcal{X}_M^*(U_{M,k}), \\ &x_k \in \mathcal{X}_{R,M}(U_{M,k}) \text{ where } \mathcal{X}_{R,M}(U_{M,k}) \supseteq \mathcal{X}_M^*(U_{M,k}), \end{aligned}$$

and the sets $\bar{\mathcal{X}}_M(U_{M,k})$, $\mathcal{X}_{I,M}(U_{M,k})$, and $\mathcal{X}_{R,M}(U_{M,k})$ are to be designed appropriately. One of the easiest way, if all elements of $\mathcal{b}_M^*$ are positive, is to use scaling factors and let

$$x_k \in \mathcal{X}_M(U_{M,k}) = \left\{x : \mathcal{A}_M^* \begin{bmatrix} x_k \\ U_{M,k} \end{bmatrix} \leq \eta_{O,M}\, \mathcal{b}_M^*\right\}, \tag{4.12}$$

$$x_k \in \mathcal{X}_{I,M}(U_{M,k}) = \left\{x : \mathcal{A}_M^* \begin{bmatrix} x_k \\ U_{M,k} \end{bmatrix} \leq \eta_{I,M}\, \mathcal{b}_M^*\right\}, \tag{4.13}$$

$$x_k \in \mathcal{X}_{R,M}(U_{M,k}) = \left\{x : \mathcal{A}_M^* \begin{bmatrix} x_k \\ U_{M,k} \end{bmatrix} \leq \eta_{R,M}\, \mathcal{b}_M^*\right\}, \tag{4.14}$$

where $\eta_{O,M} \in (0, 1]$ and $\eta_{I,M}, \eta_{R,M} \in [1, \infty)$, while they should all be as close to 1 as possible to avoid overly restricted/relaxed operation.

4.3 Online FDI and Failure Mode Reconfiguration Process

4.3.1 Structure Overview

Figure 4.1 illustrates the online process flow of the FDI and failure mode reconfiguration strategy for the proposed FMEM system with three phases. In phase 0, the failure detection is run at every time step to check if there is a new failure. If the failure is detected, the system enters phase 1 for the failure isolation, at the end of which the failure mode M is known. Otherwise, the system continues operating in phase 0, and a reference governor is used to generate the modified reference v_k for reference tracking without constraint violations. Conversely, during the isolation phase, using a reference sequence is less straightforward than using an open-loop input sequence since the closed-loop dynamics are uncertain without knowing the failure mode M. Instead, a sequence of actuator input commands U^* is computed and applied to the plant. Once the isolation process is over, we know the value of M, and it is assumed that the failure has been identified (i.e. the value of d_M is known). The system moves on to phase 2 where the nominal controller for the isolated mode M is enabled and the reconfiguration process is carried out. A recovery sequence of modified references $\boldsymbol{v}$ is generated, and the system operates with this reference sequence until the end of the reconfiguration process. Then, it returns to phase 0 and operates in the new mode. Compared with the strategy in [26] that focuses on reference tracking and failure mode reconfiguration, this proposed strategy integrates FDI by adding phase 1 between phase 0 and phase 2, and replaces the assumption of instantaneous FDI with the process of Detection (in phase 0) and Isolation (in phase 1).

4.3.2 Phase 0: Reference Tracking and Failure Detection

4.3.2.1 Constraint Admissible Sets

We first introduce the definition of a constraint admissible set since both the application of reference governor and the failure detection use this concept. The constraint admissible set of mode $M \in \{0, \cdots, \Omega\}$, denoted by $\mathcal{O}_{\infty,M}$, is defined as a set of the initial state x_0 and the constant closed-loop input U_M such that, with these pairs, the safety constraints given by (4.12) will be satisfied for all future time. We let

$$\begin{aligned}\mathcal{O}_{\infty,M} =&\big\{(U_M, x_0):\ x_t \in \mathcal{X}_M(U_M)\ \forall t \in \mathbb{Z}_{\geq 0},\ x_{M,\mathrm{ss}}(U_M) \bigoplus \mathcal{B}_\epsilon \subset \mathcal{X}_M(U_M)\big\} \\ =&\big\{(U_M, x_0):\ \mathcal{A}_{\mathcal{O}_{\infty,M}} x_0 \leq \mathcal{b}_{\mathcal{O}_{\infty,M}}(U_M)\big\},\end{aligned} \tag{4.15}$$

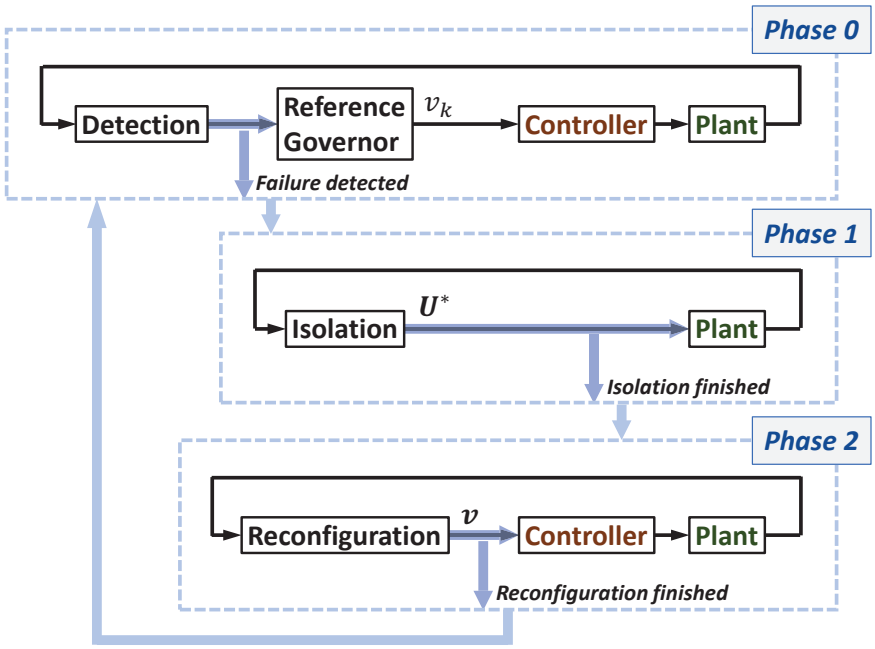

Figure 4.1: Structural overview of the online FDI and failure mode reconfiguration strategy. The blocks of Controller and Plant represent the banks of controllers and open-loop system dynamics for different modes.

where x_t is the system response, $x_{M,\text{ss}}(U_M)$ is the steady-state point (assuming zero-disturbance) given by

$$x_{M,\text{ss}}(U_M) = (I - \bar{A}_M)^{-1}\bar{B}_M U_M,$$

and $\mathcal{B}_\epsilon$ is an open ball of radius $\epsilon > 0$. By this definition, the constraint admissible sets are positively invariant, finitely determined, and can be represented by a finite set of affine inequalities under mild additional assumptions [45].

Remark 4.2. To account for the unknown input disturbance w_k, the computations are modified compared with the process of Chap. 3. Firstly, for $x_t \in \mathcal{X}_M(U_M)$, it is sufficient to have

$$x^{\text{v}}_{M,t}(V_t) \in \mathcal{X}_M(U_M) \sim \bar{\mathcal{Q}}_{M,t},$$

where $\sim$ is the Pontryagin difference, and V_t is given by U_M. Specifically, V_t is a vector of $v_k = v,\ \forall k = 0, \cdots, t-1$, where v is the constant reference in U_M. Moreover, $\mathcal{D}_M = \{d_M\}$ for $\bar{\mathcal{Q}}_{M,t}$ where d_M is the stuck/jammed actuator position in U_M. Secondly, the open ball $\mathcal{B}_\epsilon$ is assumed to be sufficiently large to include the minimum disturbance invariant set [45].

4.3.2.2 Reference Governor

For mode $M \in \{0, \cdots, \Omega - 1\}$, at every time step k in phase 0, in order to track the reference command r_k as close as possible without violating the safety constraints, a reference governor is used to generate the modified reference v_k by solving the following quadratic programming (QP) problem:

$$\begin{aligned} &\min_{v_k} \ \|r_k - v_k\|_2^2 \\ &\text{s.t. } (U_{M,k}, x_k) \in \mathcal{O}_{\infty,M}. \end{aligned} \tag{4.16}$$

The modified reference v_k is then taken by the controller to command the input of the working actuators.

4.3.2.3 Failure Detection

If the system is operating in the current mode M_0 and the control input is u_k, the state x_{k+1} is determined by (4.1) with $M = M_0$. Hence, if the condition

$$x_{k+1} \notin A_{M_0} x_k + B_{M_0} u_k \bigoplus F_{M_0} \mathcal{W} \tag{4.17}$$

holds, a new actuator failure has occurred and the isolation phase is initiated. In the isolation phase, the successor modes of mode M_0, whose set is denoted by $\text{succ}(M_0)$, are considered.

Remark 4.3. It is possible that (4.17) does not hold but the new failure has already happened. In this case, the disturbance $w_k \in \mathcal{W}$ masks the failure. On the other hand, during this time, the system response is close to nominally expected and hence is considered to be safe and satisfactory.

4.3.3 Phase 1: Failure Isolation

4.3.3.1 Isolability

Once a new failure is detected, the system enters the failure isolation phase, at the end of which the failure mode M should be known. Consider mode $M_0 \in \{0, \cdots, \Omega - 1\}$ whose successor mode is not $M = \Omega$ (otherwise it is certain that $\text{succ}(M_0) = \{\Omega\}$), and any two candidate modes $M_1, M_2 \in \text{succ}(M_0)$ where $M_1 \neq M_2$. Based on the measured state, it is sufficient to isolate the failure in N_I steps if there exists an isolation sequence U^* such that the reachable sets have no common states, that is, if

$$\mathfrak{R}_{M_1,N_I}(U^*) \bigcap \mathfrak{R}_{M_2,N_I}(U^*) = \emptyset. \tag{4.18}$$

Then, the failure mode is the one whose reachable set contains the measured state x_{N_I}.

Following (4.9), condition (4.18) can be ensured if and only if

$$\forall z_1 \in \mathcal{Q}_{M_1,N_I}, z_2 \in \mathcal{Q}_{M_2,N_I}, \ S_{M_1,N_I} x_0 + T_{M_1,N_I} U^* + z_1 \neq S_{M_2,N_I} x_0 + T_{M_2,N_I} U^* + z_2, \tag{4.19}$$

where x_0 here represents the state at the beginning of the isolation.

Proposition 4.1. *Equation* (4.19) *is equivalent to*

$$(S_{M_1,N_I} - S_{M_2,N_I})x_0 + (T_{M_1,N_I} - T_{M_2,N_I})U^* \notin \mathcal{Q}_{M_2,N_I} \bigoplus (-\mathcal{Q}_{M_1,N_I}), \tag{4.20}$$

Proof. By definition,

$$\Omega_{N,m_2} \bigoplus (-\Omega_{N,m_1}) = \{z_2 - z_1 : \ z_1 \in \Omega_{N,m_1}, z_2 \in \Omega_{N,m_2}\},$$

that is,

$$z \in \Omega_{N,m_2} \bigoplus (-\Omega_{N,m_1}) \Leftrightarrow \exists z_1 \in \Omega_{N,m_1} \text{ and } z_2 \in \Omega_{N,m_2} \text{ such that } z_2 - z_1 = z.$$

By negation, we have

$$z \notin \Omega_{N,m_2} \bigoplus (-\Omega_{N,m_1}) \Leftrightarrow \forall z_1 \in \Omega_{N,m_1} \text{ and } z_2 \in \Omega_{N,m_2}, \ z_2 - z_1 \neq z.$$

Thus, (4.19) $\Leftrightarrow$ (4.20). □

As a result, we need to find an isolation sequence U^* such that (4.20) holds.

4.3.3.2 Computations to Generate the Isolation Sequence

By the definition of $\mathcal{Q}_{M,k}$, the set on the right-hand-side of (4.20) is polyhedral. Consider its representation in the form of affine inequalities:

$$\mathcal{Q}_{M_2,N_I} \bigoplus (-\mathcal{Q}_{M_1,N_I}) = \{x : \ \mathcal{H}_j x \leq \mathcal{h}_j, \ j = 1, \cdots, q\}.$$

Then, (4.20) can be re-stated as

$$\exists j : \mathcal{H}_j \big((S_{M_1,N_I} - S_{M_2,N_I})x_0 + (T_{M_1,N_I} - T_{M_2,N_I})U^* \big) > \mathcal{h}_j. \tag{4.21}$$

The problem becomes finding an isolation sequence U^* such that (4.21) holds, and at the same time, the safety constraints given by (4.13) are satisfied. This can be handled using mixed-integer optimization applied to (4.21) transformed using the "Big-M" technique (see Sect. 9.3.1 in [50]). We introduce binary integers $\delta_j \in \{0, 1\}, j = 1, \cdots, q$. The condition (4.21) can be written as

$$\begin{gathered} \sum_{j=1}^{q} \delta_j \geq 1, \\ \forall j, \ \mathcal{H}_j \big((S_{M_1,N_I} - S_{M_2,N_I})x_0 + (T_{M_1,N_I} - T_{M_2,N_I})U^* \big) \geq \mathcal{h}_j - (1 - \delta_j)\mathbb{M} + \gamma, \end{gathered} \tag{4.22}$$

where $\mathbb{M} \gg 0$ is a large positive number, and $\gamma > 0$ is a small positive number introduced to replace "$>$" with "$\geq$". Note that to ensure (4.22) holds, one of δ_j has to be equal to 1, which implies (4.21).

The constraints (4.22) can now be integrated into a mixed-integer quadratic programming (MIQP) problem, whose objective is to determine a minimum norm control sequence U^* that ensures the ability to distinguish the operation in mode M_1 versus operation in mode M_2 without

violating the safety constraints, that is, to find the solution of the following problem:

$$\min_{U^*,\delta_1,\cdots,\delta_q} \ \|U^*\|_2^2 \tag{4.23}$$
$$\text{s.t.} \ \ (U^*,\delta_1,\cdots,\delta_q,x_0)\in\mathcal{C}(M_0),$$

where

$$\begin{aligned}\mathcal{C}(M_0)=\mathcal{C}_{M_1,M_2}=\Big\{&(U^*,\delta_1,\cdots,\delta_q,x_0):\\ &\sum_{j=1}^{q}\delta_j\geq 1,\\ &\forall j\in\{1,\cdots,q\},\\ &\quad \delta_j\in\{0,1\},\\ &\quad \mathcal{H}_j(T_{M_1,N_I}-T_{M_2,N_I})U^*-\mathbb{M}\delta_j+\mathcal{H}_j(S_{M_1,N_I}-S_{M_2,N_I})x_0\geq \hbar_j-\mathbb{M}+\gamma,\\ &\forall t\in\{0,\cdots,N_I-1\},\\ &\quad x_t\in\mathcal{X}_{I,M_1}(U_{M,t}),\ x_t\in\mathcal{X}_{I,M_2}(U_{M,t})\Big\}.\end{aligned} \tag{4.24}$$

Remark 4.4. To account for the unknown input disturbance, for $x_t\in\mathcal{X}_{I,M}(U_{M,t})$, it is sufficient to have

$$x_{M,t}^{\mathtt{u}}(U_t^*)\in\mathcal{X}_{I,M}(U_{M,t})\sim\mathcal{Q}_{M,t}.$$

where U_t^* is the vector consists of $u_0^*,\cdots,u_t^*$ from U^*.

Remark 4.5. If there are more than two candidate modes, constraints can be defined similar to (4.24) for all distinct pairs of candidate modes, e.g., if the candidate modes are $M=1,2,3$, then constraints should be defined for mode pairs $(1,2)$, $(2,3)$ and $(3,1)$. The MIQP problem becomes

$$\min \ \|U^*\|_2^2$$
$$\text{s.t.} \ \ (U^*,\delta_{1,2,1},\cdots,\delta_{1,2,q_1},\delta_{2,3,1},\cdots,\delta_{2,3,q_2},\delta_{3,1,1},\cdots,\delta_{3,1,q_3},x_0)\in\mathcal{C}(M_0),$$

where

$$\mathcal{C}(M_0)=\mathcal{C}_{1,2}\cap\mathcal{C}_{2,3}\cap\mathcal{C}_{3,1}.$$

Note that, while handling the combinatorial complexity of failure mode, e.g., due to the exponential growth of modes by the increase of actuators, is not a feature of our approach but a property of the underlying problem that any approach has to deal with in practice, fortunately, in many mobility applications, the number of redundant actuators is limited due to considerations such as cost. Furthermore, it is not uncommon to limit the FMEM strategy to only handling the most probable failure scenarios.

4.3.4 Phase 2: Failure Reconfiguration

When the system finished operating with the isolation sequence, it enters the reconfiguration phase with the known failure mode $M \in \{1, \cdots, \Omega - 1\}$ and the vector of stuck/jammed actuator positions d_M that has been identified. In this phase, a reference sequence $\boldsymbol{v} = \begin{bmatrix} v_0^T & \cdots & v_{N_M}^T \end{bmatrix}^T$ is generated to steer the state into the state projection of the constraint admissible set, $\text{Proj}_x \mathcal{O}_{\infty,M}$, within N_M steps without violating the relaxed safety constraints given by (4.14). The recovery reference sequence is determined by solving the following QP problem:

$$\begin{aligned} \min_{\boldsymbol{v}} \ & \|r_0 \mathbf{1} - \boldsymbol{v}\|_2^2 \\ \text{s.t.} \quad & x_t \in \mathcal{X}_{R,M}(U_{M,t}) \ \ \forall t = 0, \cdots, N_M - 1, \\ & (U_{M,N_M}, x_{N_M}) \in \mathcal{O}_{\infty,M}, \end{aligned} \tag{4.25}$$

where $t = 0$ here corresponds to the beginning of the reconfiguration phase, and by using $r_0\mathbf{1}$ it is assumed that the reference command stays at constant during the reconfiguration process. The system eventually returns to phase 0 once it finished operating with the recovery sequence.

4.4 Conditions for Guaranteed Failure Isolation and Reconfiguration

The set-membership conditions for the offline design procedure is now described. So far, three optimization problems, (4.16), (4.23), and (4.25), have been introduced for different online phases of the FMEM strategy. For the reference governor defined by (4.16), the problem is feasible in the failure modes because of the constraint $(U_{M,N_M}, x_{N_M}) \in \mathcal{O}_{\infty,M}$ imposed in (4.25). To ensure the feasibility of failure mode isolation and reconfiguration, i.e., for (4.23) and (4.25), we can choose the number of steps allowed for isolation N_I, the number of steps allowed for reconfiguration N_M, and the safety constraint scaling coefficients $\eta_{O,M}$, $\eta_{I,M}$, and $\eta_{R,M}$.

4.4.1 Isolation Conditions

When the system is operating in mode $M_0 \in \{0, \cdots, \Omega - 1\}$, its state is contained in $\text{Proj}_x \mathcal{O}_{\infty,M_0}$, but when a failure is detected, the state is in the following set:

$$\mathcal{I}_{M_0} = \bigcup_{M \in \text{succ}(M_0)} A_M \text{Proj}_x \mathcal{O}_{\infty,M_0} \bigoplus B_M \mathcal{U} \bigoplus F_M \mathcal{W}. \tag{4.26}$$

This implies the following result:

Proposition 4.2 (Isolation Condition)**.** *If the following condition holds,*

$$\mathcal{I}_{M_0} \subseteq \textit{Proj}_x \mathcal{C}(M_0), \tag{4.27}$$

then (4.23) *is feasible.*

4.4.2 Reconfiguration Conditions

We begin by introducing the recoverable sets, which, for the successor modes $M \in \text{succ}(M_0)$, are defined as

$$\begin{aligned}\mathcal{R}^{N_M}_{\infty,M}(d_M) = \{x_0 :\ \exists\{v_0, \cdots, v_{N_M}\} \text{ such that} \\ x_t \in \mathcal{X}_{R,M}(U_{M,t})\ \forall t = 0, \cdots, N_M - 1,\ (U_{M,N_M}, x_{N_M}) \in \mathcal{O}_{\infty,M}\}.\end{aligned} \tag{4.28}$$

Remark 4.6. To account for the unknown input disturbance in ensuring $x_t \in \mathcal{X}_{R,M}(U_{M,t})$, it is sufficient to have

$$x^{\mathrm{v}}_{M,t}(V_t) \in \mathcal{X}_{R,M}(U_{M,t}) \sim \bar{\mathcal{Q}}_{M,t},$$

where V_t is the vector consists of $v_0, \cdots, v_t$ from $\boldsymbol{v}$, and $\mathcal{D}_M = \{d_M\}$ for $\bar{\mathcal{Q}}_{M,t}$.

In mode Ω, we let the recoverable set to be the state projection of the constraint admissible set, that is,

$$\mathcal{R}^{N_\Omega}_{\infty,\Omega} = \text{Proj}_x \mathcal{O}_{\infty,\Omega} \quad \forall N_\Omega \geq 0, \tag{4.29}$$

as the system runs open-loop when all actuators have failed.

At the end of the isolation phase, i.e., the beginning of the reconfiguration phase, the system is in the known failure mode M, and the states are in

$$\mathcal{S}'_M(d_M, U^*) = \{x_{N_I} :\ x_0 \in \text{Proj}_x \mathcal{C}(M_0)\}, \tag{4.30}$$

where x_{N_I} results from using the isolation sequence U^* assuming the stuck/jammed actuator input is d_M. Furthermore, since the state at the beginning of the isolation $x_0 \in \mathcal{I}_{M_0}$, if the isolation condition (4.27) holds, the states are contained in

$$\mathcal{S}''_M(d_M, U^*) = \{x_{N_I} :\ x_0 \in \mathcal{I}_{M_0}\}. \tag{4.31}$$

If we have

$$\mathcal{S}''_M(d_M, U^*) \subseteq \bigcap_{d_M \in \mathcal{D}_M} \mathcal{R}^{N_M}_{\infty,M}(d_M), \tag{4.32}$$

which implies that the state at the end of the isolation phase is in the recoverable set for any stuck/jammed actuator position, the optimization problem (4.25) is guaranteed to be feasible.

However, d_M and U^* are a priori unknown. To satisfy (4.32), it is sufficient to define

$$\mathcal{S}(M_0) = \{x_{N_I} :\ x_0 \in \mathcal{I}_{M_0},\ U^* \in \mathcal{U}^{N_I},\ d_M \in \mathcal{D}_M\}, \tag{4.33}$$

i.e., the set of states that are N_I steps later of the states in $\mathcal{I}_{M_0}$ considering all possible inputs (for both working and failed actuators) and disturbances, and impose a stricter condition as

$$\mathcal{S}(M_0) \subseteq \bigcap_{d_M \in \mathcal{D}_M} \mathcal{R}^{N_M}_{\infty,M}(d_M). \tag{4.34}$$

Proposition 4.3 (Reconfiguration Condition)**.** *If the conditions* (4.27) *and* (4.34) *hold, then* (4.25) *is feasible.*

The offline design procedure to satisfy the conditions of Propositions 4.2 and 4.3 is demonstrated through a numerical example in the next section.

4.5 Numerical Example

4.5.1 System Dynamics and Operating Modes

A mass-spring-damper system is used for illustration with the model given by

$$\dot{x} = Ax + Bu + Fw,\ y = Cx, \tag{4.35}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k_0}{m_0} & -\frac{c_0}{m_0} \end{bmatrix},\ B = \begin{bmatrix} 0 & 0 \\ \frac{1}{m_0} & \frac{1}{m_0} \end{bmatrix}, F = \begin{bmatrix} 0 \\ \frac{1}{m_0} \end{bmatrix},$$
$$C = \begin{bmatrix} 1 & 0 \end{bmatrix},\ x = [\alpha\ \ \lambda]^T,\ u = [f_1\ \ f_2]^T,\ y = \alpha,$$

in which α is the displacement, λ is the velocity, f_1 and f_2 are the redundant acting forces, $m_0 = 1$ is the mass, $k_0 = 1$ is the spring constant, and $c_0 = 0.01$ is the damping constant. The system is converted to discrete-time assuming the sampling-period of 0.2 sec.

Since there are $N = 2$ actuators, the system can operate in four modes. We define mode 0 as the normal mode, i.e. the mode in which both actuators work properly. In mode 1, f_1 works properly and f_2 fails, while in mode 2, f_2 works properly and f_1 fails. The last mode is mode 3 when both actuators have failed.

Linear Quadratic Regulator (LQR) theory is used to generate feedback gains K_M for modes $M = 0, 1, 2$ and the closed-loop dynamics (4.5). The design parameters (chosen by trial and error) are as follows:

$$Q_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ Q_1 = Q_0,\ Q_2 = Q_0,$$

$$R_0 = \begin{bmatrix} 1000 & 0 \\ 0 & 250 \end{bmatrix}, R_1 = \begin{bmatrix} 10 & 0 \\ 0 & 0 \end{bmatrix}, R_2 = \begin{bmatrix} 0 & 0 \\ 0 & 2.5 \end{bmatrix},$$

where the subscript corresponds to a specified mode. It is assumed that the use of actuator f_1 is more expensive than f_2 which is reflected in a choice of a larger weight for that input. The closed-loop state matrices $\bar{A}_M$ are derived corresponding to each K_M, and the feedforward matrices are defined

as

$$
\begin{aligned}
G_0 &= \begin{bmatrix} 0.25/g_1 & 0.75/g_2 \end{bmatrix}^T, \\
&\text{where } \begin{bmatrix} g_1 & g_2 \end{bmatrix} = C_0(I - \bar{A}_0)^{-1}B_0, \\
\forall M &\in \{1, 2\}, \\
G_M &= [C_M(I - \bar{A}_M)^{-1}B_{M,\mu}]^{-1}, \\
H_M &= -G_M[C_M(I - \bar{A}_M)^{-1}B_{M,d}]^{-1}.
\end{aligned}
$$

4.5.2 Constraints

The state and control constraints are given by

$$|\alpha| \leq \alpha_{\max},\ |f_1| \leq f_{1_{\max}}, \text{ and } |f_2| \leq f_{2_{\max}}, \tag{4.36}$$

where

$$\alpha_{\max} = 1,\ f_{1_{\max}} = 0.6, \text{ and } f_{2_{\max}} = 0.4.$$

In addition, the input disturbance is bounded by

$$|w| \leq w_{\max}, \tag{4.37}$$

where $w_{\max} = 0.02$.

4.5.3 Offline Design for Guaranteed Isolability and Reconfigurability

In this section, we consider the case of sequential failures where the system begins operating in mode 0, then changes to mode 1 where f_2 fails, and eventually finishes in mode 3 with both actuators failed. The case with 0-2-3 mode transition sequence follows a similar procedure. The `Bensolve` toolbox [46] is used for polyhedral computations. The design parameters, N_I, N_M, $\eta_{O,M}$, $\eta_{I,M}$ and $\eta_{R,M}$, are chosen from a grid of values to minimize the number of time steps needed and the changes from the original constraints due to scaling.

The offline design proceeds backward in terms of the failure sequence, that is, we begin by considering mode 3 and determining its recoverable set given the safety constraints. To ensure the system state is in $\text{Proj}_x\mathcal{O}_{\infty,3}$ during mode 1, we choose $\eta_{O,1} = 0.45$ to satisfy $\text{Proj}_x\mathcal{O}_{\infty,1} \subseteq \text{Proj}_x\mathcal{O}_{\infty,3}$. The result is shown in Fig. 4.4. For mode 2, $\eta_{O,2} = 0.75$.

Next, the isolability condition for the transition from mode 0 given by (4.27) with $M_0 = 0$ is considered. We let $\eta_{O,0} = 1$ so that the normal operation is not artificially restricted, and we use $\eta_{I,1} = 1.2$ to relax the safety constraints for $N_I = 2$ steps during the isolation phase. Note that this is a common step for the 0-1-3 and 0-2-3 mode transitions, as both mode 1 and 2 are the successor modes of mode 0. Hence, the parameters N_I, $\eta_{I,1}$ and $\eta_{I,2}$ are adjusted together to satisfy (4.27).

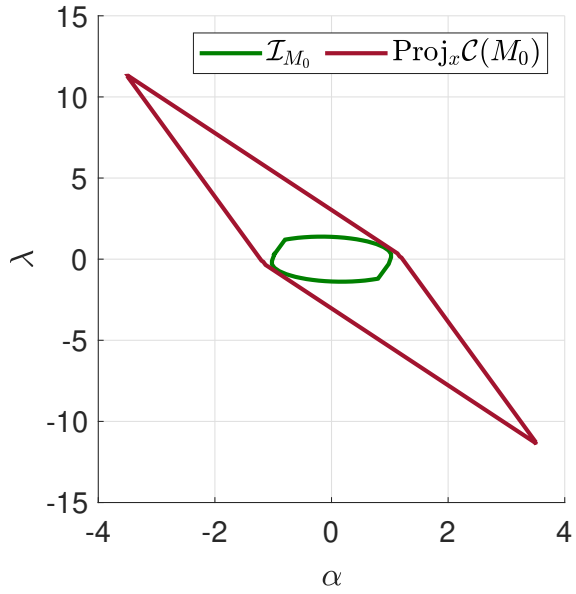

Figure 4.2: Isolation condition (4.27) satisfied for $M_0 = 0$.

For mode 2, we also let $\eta_{I,2} = 1.2$. Figure 4.2 illustrates the set-membership relation.

Finally, we consider the reconfigurablity for the mode transition to $M = 1$. The recoverable set of mode 1 is determined following (4.28), and the condition that needs to be satisfied is (4.34). In this phase, we relax the safety constraints by having $\eta_{R,1} = 1.7$ and allow $N_1 = 15$ steps for the recovery. The set-membership relation is shown in Fig. 4.3. For mode 2, $\eta_{R,2} = 1.9$ and $N_2 = 22$.

4.5.4 Simulations

A simulation is run for the case of 0-1-3 mode transition. The input disturbance follows a normal distribution within $\mathcal{W}$ using zero mean and the standard deviation $\sigma = w_{\max}/6$. The failures are set to happen in 2 and 44 sec, while the displacement reference command switches between -0.99 and 0.99 in 20 and 40 sec. Figure 4.4 shows the state trajectory and Fig. 4.5 shows the time-based signals for a 60-sec simulation. Both failures are detected in one step and the isolation and reconfiguration takes 3.4 sec in total as determined by N_I and N_M for $M = 1$. The difference between the reference command and the modified reference in mode 1 shows that the operation is restricted after the initial failure to prepare for the potential subsequent failure, i.e., transiting to mode 3, which makes the system run in open-loop. No constraint violations are observed during the operation in any of the modes and phases.

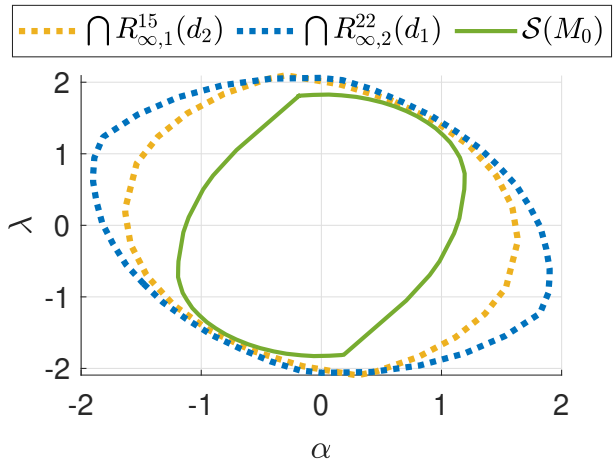

Figure 4.3: Reconfiguration conditions (4.34) satisfied for $M_0 = 0$.

4.6 Concluding Remarks

Enhancements to a reference governor-based Failure Mode and Effect Management (FMEM) strategy for a system with redundant actuators that can become stuck/jammed have been developed. These enhancements ensure the ability to detect and isolate failures within a finite time while tracking reference commands and satisfying state and control constraints. Conditions have been derived that can be used in the offline design phase to guarantee the ability to isolate failures within a pre-determined time duration and to reconfigure the system into the next mode. The proposed system can handle sequential failures and unmeasured set-bounded disturbance inputs. A mass-spring-damper example with two force inputs has been reported to illustrate the design and operation of the proposed FMEM strategy. Possible directions of future work include considering (1) other types of failures, e.g., failures caused by actuator efficiency degradation, failed signals that change slowly with bounded rates, and failures related to sensors and/or communication, and (2) challenges for implementing the strategy in real-time, e.g., in the situation when the isolation and reconfiguration sequences cannot be generated in time for execution due to limitations of computational resources.

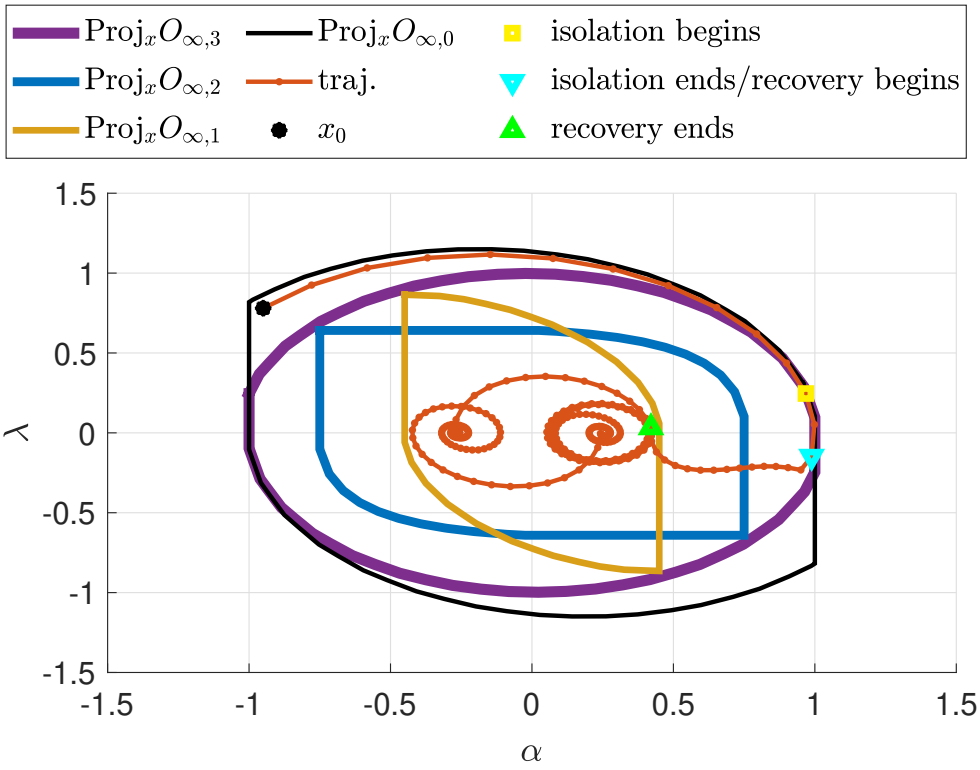

Figure 4.4: State projections of constraint admissible sets and simulated state trajectory.

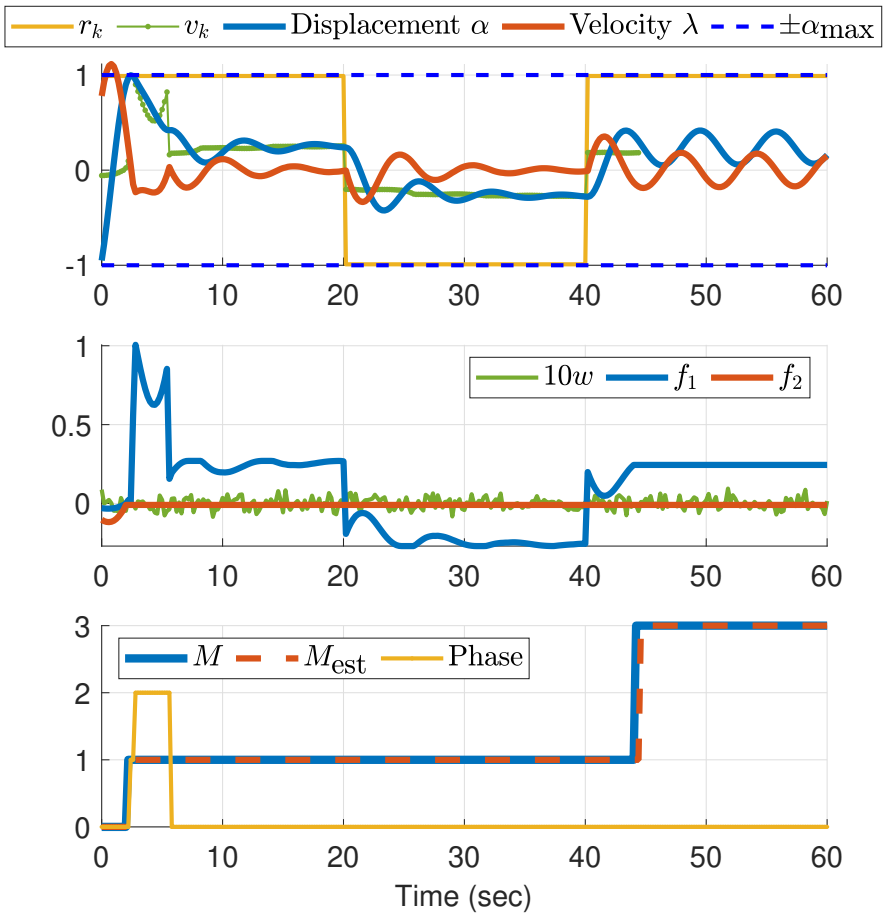

Figure 4.5: Simulation results, where M is the actual mode, and M_{est} stands for the mode determined by the FMEM unit.

CHAPTER 5

Energy-Efficient Autonomous Driving Control Using Game-Theoretic Interactive Traffic Models and Reinforcement Learning

5.1 Introduction

With recent advances in sensing technologies and artificial intelligence, there has been a rapidly growing interest in connected and autonomous vehicles (CAVs) [51,52]. Such vehicles are expected to improve the safety and mobility of transportation and to alleviate traffic congestion.

Another expected benefit of CAVs is a reduction in fuel/energy consumption [11,53]. Since 2016, the U.S. Department of Energy has awarded more than $50 million in funding for studies by the Advanced Research Projects Agency-Energy's (ARPA-E) Next-Generation Energy Technologies for Connected and Automated On-Road Vehicles (NEXTCAR) program for which the goal is to reduce the energy consumption of vehicles in all classes by more than 20% via CAV technologies [54]. Table 5.1 shows selected results by this program. This is not intended to be a comprehensive review, but the collection illustrates the variety of methods and traffic conditions being explored in some of the most recent studies to achieve energy efficiency improvements using CAV technologies.

To realize this encouraging potential in real-world driving circumstances, we observe that at least the following two problems remain to be studied. Firstly, interactions of the ego vehicle with surrounding vehicles in traffic need to be considered. In the foreseeable future, autonomous vehicles will operate together with human-driven vehicles in traffic [29]. Thus, it is necessary to consider the different vehicle actions and reactions caused by different types of human driving styles. In [32, 64–66], level-k game theory is used to model the interactions with the focus on different driving scenarios. Researchers such as of [67–69] have utilized traffic-in-the-loop models and closed-loop control to achieve simultaneous optimization for safety and fuel economy. However, only longitudinal control is considered in these studies. Indeed, the vast majority of recent studies on improving energy efficiency using CAV technologies assume that the ego vehicle is driven in

Table 5.1: Summary of selected literature on using CAV technologies to improve energy efficiency

Ref.	Energy consumption reduction by [%]	Methods	Traffic conditions
[55]	5.4	Thermal management (air conditioning)	Driving cycles
[56]	3.1	Thermal management (battery)	Driving cycles
[57]	41	Thermal management (air conditioning and engine)	Driving cycles
[58]	50	Eco-driving	Intersections
[59]	12	Eco-routing	City
[60]	57.8	Platooning, cooperative merging	Ramps on highway
[61]	8.5	Anticipative lane change	City and highway
[62]	(up to) 59	Cooperative lane change	Highway
[63]	47	Cooperative driving	Intersections and roundabouts

single-lane traffic. Thus, the second problem worth investigating is the simultaneous longitudinal control and lateral control (such as lane changes) of AVs, which increases the dimension of the problem but provides additional possibilities to save energy. A more detailed discussion on lane changes for better energy efficiency is given in Sect. 5.2.

There has been a rich set of research on machine learning (ML) methods for automotive applications to improve energy efficiency and emissions by modeling and control of the powertrain system (see, e.g., [70–75]). In particular, to meet increasingly stringent fuel economy and emissions regulations, the powertrain systems of hybrid electric vehicles (HEVs) have become more and more complex. Consequently, commonly studied model-based control methods for the energy management system (EMS), such as dynamic programming (DP) and model predictive control (MPC) [40], are facing growing difficulties, as they rely on models with good accuracy and many control-oriented models are physics-based. In comparison, machine learning methods can handle this challenge well. For example, for HEVs with small electrical energy storage, there is a significant potential to utilize recurrent neural networks to learn driving patterns and improve energy efficiency [76]. In the CAV domain, example applications of ML include perception and localization, route/path planning/optimization, and motion control, despite challenges such as computation, safety, and adaptability/generalizability that are actively being studied [77–79]. Studies such as [80, 81] use ML to inform energy-efficient acceleration/braking of electric vehicles. A level-k game theory-based traffic simulator is developed in [32] following the methodology originally proposed in [33]. The simulator is based on cognitive driver behavioral models trained by reinforcement learning (RL).

In this chapter, we describe a novel framework for developing energy-efficient AV control policies (including both longitudinal, e.g., vehicle speed, and lateral, e.g., lane change, controls) through RL[1]. We focus our attention on highway driving and autonomous battery electric vehicles (BEVs), as BEVs are getting increasingly popular due to their environmental benefits [82]. A BEV powertrain model is developed to calculate the energy consumption over trips. To enable the AV control policy to properly respond to the interactions with human-driven vehicles on shared roads, the game-theoretic traffic model developed in [32, 64] is used as the RL training environment. Extensions to other powertrain types and traffic environments are possible [66, 83] but are left to future work.

The remainder of this chapter is organized as follows. Firstly, further background on lane changes for energy-efficient AV driving is discussed in Sect. 5.2. Then, we begin the development by building a BEV model and validating it in Sect. 5.3. In Sect. 5.4, the control development is detailed, and another control policy trained by RL and the finite-state-machine (FSM) controller from [64] are introduced to be subsequently used for comparison. Section 5.5 presents results on RL convergence and on performance of the developed control policy in simulations. Finally, concluding remarks are made in Sect. 5.6.

5.2 Lane Changes for Energy-Efficient AV Driving

Including lateral actions such as lane changes may further improve the energy efficiency [30], though the survey results of [11] show that this is still an emerging area that remains to be studied. Anticipating lane selection has been proposed, such as in [84, 85]. Furthermore, instead of focusing on an individual vehicle, cooperative lane change [86] is expected to benefit the neighboring vehicles and harmonize the surrounding traffic. However, these studies assume having connected vehicle technologies, such as vehicle-to-vehicle and vehicle-to-infrastructure communications. A general observation is that the energy efficiency can be improved by reducing the change of speed and acceleration. In contrast, our work uses the position and velocity information of the immediate neighboring vehicles detected by the sensors of the ego vehicle, and a powertrain model is used to accurately predict the energy consumption.

There are two major challenges to combine longitudinal and lateral actions for safe energy-efficient driving. Firstly, considering both longitudinal and lateral actions increases the problem complexity. In particular, there are subtle trade-offs between safety and energy efficiency. For instance, in scenarios such as a sudden cut-in by a slow vehicle, changing lanes rather than hard braking preserves vehicle momentum and avoids energy loss, but if the traffic density is high,

[1]The policy training was performed on the HPC cluster supported through computational resources and services provided by Advanced Research Computing at the University of Michigan, Ann Arbor.

performing a lane change may not be safe or feasible.

Another challenge is that, unlike safety constraint violation scenarios (e.g., collisions) where events typically occur within seconds, energy efficiency evaluation requires longer time horizons of several hundreds of seconds. Hence, optimization-based control algorithms that simultaneously address driving safety and energy efficiency requirements need to account for both short-term and long-term objectives. One approach is to define a terminal cost function for the short horizon optimization reflective of long-term rewards. However, how to practically determine such a terminal cost function is often *a priori* unclear. In special cases, the problem can be reformulated to focus on maintaining component operation in more efficient regions [87] for which short-horizon optimization is sufficient. However, such reformulations are not always feasible and the performance with such an approach could be sub-optimal. In particular, Stackelberg policies and the decision tree policies considered in [88–90] rely on rewards being evaluated short-term. It may not be straightforward to extend these to account for energy efficiency requirements. For our AV control policy, since the RL algorithm updates the value functions with not only the one-step/instantaneous reward but also the average reward over time, it is able to handle multiple optimization objectives that need to be evaluated over different time horizons.

5.3 Powertrain Modeling for Battery Electric Vehicles

5.3.1 Model Description

Accounting for energy efficiency in the controller design requires a longitudinal powertrain model. As an RL process is generally computationally intensive, a powertrain model used in RL should have low computational footprint but sufficiently high accuracy.

Figure 5.1 shows the layout of the BEV model considered in this work. The powertrain system consists of a battery pack, a motor/generator (MG), a drivetrain with a single-speed final drive, the wheels and tires, and the powertrain control unit (PCU).

The powertrain model is of the backward type [40, 91, 92]. (See Remark 5.1 below for details.) It has two states, state of charge (SOC) and total energy consumption. High-fidelity dynamics, such as transient responses of the MG and drivetrain, are not considered. Maps (i.e., look-up tables) are used to represent component operating characteristics as described below. A benefit of using a map-based approach is that it can facilitate the change of component specifications so that potential extensions, such as component sizing or fleet energy efficiency studies, are possible.

Remark 5.1. A backward powertrain model assumes that the actual vehicle speed is always equal to the reference speed command. Rotational speeds of components are coupled/scaled by the gear ratios and wheel radius based on this command. The torque required at the wheels to meet the

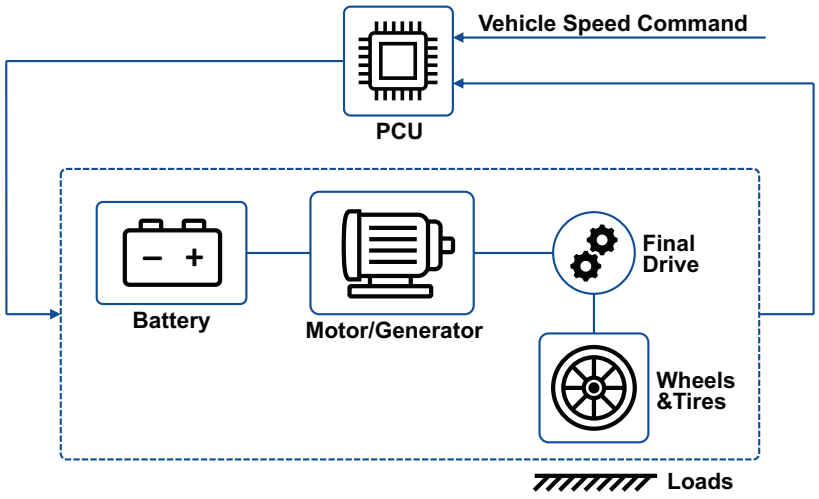

Figure 5.1: Battery electric vehicle powertrain system layout.

acceleration demand is first calculated and then translated component by component to the actuators such as the motor and the brakes. Backward models typically entail low computational costs and are suitable for (rough) energy consumption evaluations. In comparison, forward models involve a driver model, typically modeled as a PID controller, that commands the motor/brake torque in order to track the vehicle speed reference. The torque is then translated to the wheels through drivetrain components. The speed of each component, as well as the actual vehicle speed, is calculated by integrating the acceleration produced by the torques and forces. Consequently, forward models can better capture the component dynamics, at the cost of higher computational complexity than backward models. They are typically used for more detailed (e.g., componentwise) energy efficiency analysis as well as drivability-related simulations [40, 91, 92]. For example, the multistage hybrid electric vehicle model used for acceleration performance and jerk analysis in Chap. 6 is of the forward type. In the current chapter, however, the forward model is not as suitable as the backward model not only because of the computational concerns, but also due to that in the case considering safe driving, the forward model has tracking error between the commanded and actual vehicle speed whose value depends on the tuning of the driver model parameters.

In the BEV model, the MG speed and traction torque at the wheels are first calculated based on the vehicle speed command according to

$$\omega_{mg} = \frac{Vg}{r}, \tag{5.1}$$

$$T_a = \dot{V} M r, \tag{5.2}$$

$$T_l = a_l + b_l V + c_l V^2, \tag{5.3}$$

$$T_{whl} = T_a + T_l, \tag{5.4}$$

where ω_{mg} is the MG speed, V is the vehicle speed, g is the final drive gear ratio, r is the effective wheel radius, T_a is the acceleration torque, M is the effective vehicle mass involving powertrain component inertia, T_l represents lumped external loads including rolling and aerodynamic resistance (while the grade is assumed to be zero) approximated by a quadratic function with coefficients a_l, b_l, and c_l, and T_{whl} is the traction torque at wheels.

The PCU then checks whether the traction torque demand exceeds the battery or MG limits and distributes the torque command to the MG and the friction brake, using the following logic,

$$T_{mgPos} = \begin{cases} 0, & \text{if } T_{whl} < 0, \\ T_{whl}/g, & \text{if } 0 \le T_{whl} \le T_{mgMax}\, g, \\ T_{mgMax}, & \text{if } T_{whl} > T_{mgMax}\, g, \end{cases} \tag{5.5}$$

$$T_{whlBrk} = \begin{cases} 0, & \text{if } -T_{whl} < 0, \\ -T_{whl}, & \text{if } 0 \le -T_{whl} \le T_{brkMax}, \\ T_{brkMax}, & \text{if } -T_{whl} > T_{brkMax}, \end{cases} \tag{5.6}$$

$$T_{mgReg} = \begin{cases} F_{reg}\, T_{whlBrk}/g, & \text{if } T_{whlBrk} \le T_{mgRegLim}\, g, \\ F_{reg}\, T_{mgRegLim}, & \text{if } T_{whlBrk} > T_{mgRegLim}\, g, \end{cases} \tag{5.7}$$

$$T_{mg} = \begin{cases} T_{mgPos}, & \text{if } T_{whl} > 0, \\ -T_{mgReg}, & \text{if } T_{whl} \le 0, \end{cases} \tag{5.8}$$

$$T_{mechBrk} = -(T_{whlBrk} - T_{mgReg}), \tag{5.9}$$

where T_{mgMax} is the maximum MG torque limit, T_{mgPos} is the positive portion of the MG torque limited by T_{mgMax}, T_{whlBrk} is the negative portion of the traction torque at wheels limited by a constant brake torque limit denoted by T_{brkMax}, $T_{mgRegLim}$ is the MG regeneration torque limit, F_{reg} is the regeneration factor, T_{mgReg} is the negative portion of the MG torque limited by $T_{mgRegLim}$, T_{mg} is the MG torque, and $T_{mechBrk}$ is the torque demand assigned to the friction brakes. We obtain the values of T_{mgMax}, $T_{mgRegLim}$, and F_{reg} through maps, where both T_{mgMax} and $T_{mgRegLim}$ depend on ω_{mg}, and F_{reg} is a function of V (to reduce the regenerative braking at low vehicle speeds) and the battery SOC (to reduce the regenerative braking at high SOC).

The power drawn by the MG is then obtained from

$$P_{mg} = \begin{cases} T_{mg}\, \omega_{mg}/\eta, & \text{if } T_{mg}\, \omega_{mg} \ge 0, \\ T_{mg}\, \omega_{mg}\, \eta, & \text{if } T_{mg}\, \omega_{mg} < 0, \end{cases} \tag{5.10}$$

where P_{mg} is the MG electric power, and $\eta \in (0, 1)$ is the MG efficiency as a function of the MG torque and speed, as given by a map.

To calculate the SOC and cumulative energy consumption, the battery is modeled as follows:

$$\begin{aligned} P_{mg} &= (V_{oc} - \frac{1}{N_p} I\, R)\, N_s\, I \\ &= N_s\, V_{oc}\, I - \frac{N_s}{N_p} R\, I^2, \end{aligned} \tag{5.11}$$

which can be re-arranged as

$$\frac{N_s}{N_p} R\, I^2 - N_s\, V_{oc}\, I + P_{mg} = 0, \tag{5.12}$$

where V_{oc} and R are, respectively, the open circuit voltage and the resistance of a single battery cell, I is the battery pack current, N_s is the number of battery cells in series, and N_p is the number of battery cells in parallel. Here, V_{oc} and R are acquired through maps, and both variables depend on SOC, with the assumption that the battery temperature is constant. Then, we can solve for the battery current as

$$I = \frac{N_s\, V_{oc} - \sqrt{(N_s\, V_{oc})^2 - 4\frac{N_s}{N_p} R\, P_{mg}}}{2\frac{N_s}{N_p} R}, \tag{5.13}$$

and the battery dynamics are given by

$$\dot{SOC} = \frac{-I}{3600\, C_{max}\, N_p} \cdot 100, \tag{5.14}$$

where C_{max} is the maximum battery capacity in Ah.

The total discharged electric energy E_{batt} is computed by integrating the battery power P_{batt} based on

$$\dot{E}_{batt} = P_{batt} = N_s\, I\, V_{oc}, \tag{5.15}$$

and the energy consumption can be determined from

$$MPGe = \frac{x}{E_{batt}} \cdot \gamma, \tag{5.16}$$

where $MPGe$ stands for miles per gallon equivalent, x is the total distance traveled, and γ represents the unit conversion coefficient.

5.3.2 Model Calibration and Validation

The BEV powertrain model described above is calibrated using the BEV reference model in the Powertrain Blockset Toolbox (PTBS) version 1.5 developed by MathWorks. The PTBS reference model is a forward model and includes more detailed component controls and dynamics than our model. Our model uses some maps and parameter values from the PTBS reference model, and the other model parameters are hand-tuned to reduce errors between the two models.

After calibration, we validate our model by testing and comparing the $MPGe$ of our model and

that of the PTBS reference model for different driving cycles. The $MPGe$ mismatches between the two models for the Urban Dynamometer Driving Schedule (UDDS), the Highway Fuel Economy Test (HWFET), and the US06 driving cycles are 5.94%, 5.90%, and 7.95%, respectively. Figure 5.2 shows the time histories of powertrain signals for the BEV driving through the UDDS cycle, where the blue curves correspond to our model after calibration and the red curves correspond to the PTBS reference model. It can be observed that the signals of our model closely match those of the PTBS reference model. These results validate that our model (5.1)-(5.16) after calibration can be used to produce sufficiently accurate energy consumption estimates (accurate in terms of matching the estimates produced by the high-fidelity PTBS reference model). Note that our model entails much lower computational footprint than the PTBS reference model, and is thus suitable for RL purposes.

The BEV powertrain model (5.1)-(5.16) is then converted to discrete-time, assuming a 1-second sampling period, and integrated with the traffic simulator of [32, 64], used for the RL-based development of energy-efficient autonomous vehicle control policy.

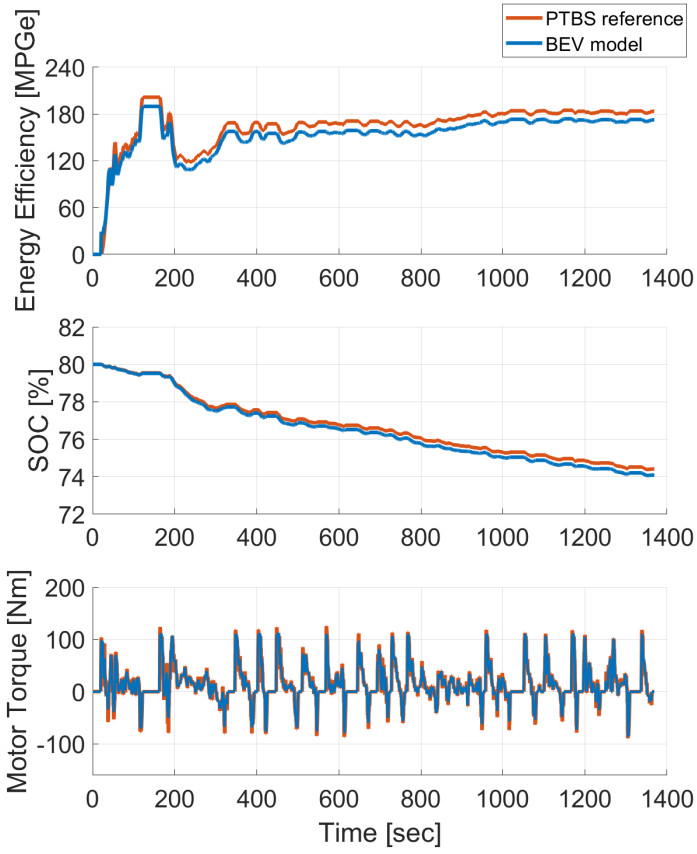

Figure 5.2: Time histories of powertrain signals for the UDDS cycle.

5.4 Controller Design

5.4.1 Game-Theoretic Traffic Environment

In order to train an autonomous vehicle control policy offline, we use an interactive traffic simulator, following the approach described in [32], as the training environment. This level-k game-theory based simulator assumes that the traffic consists of human-driven vehicles that can be modeled by cognitive behavioral models with k levels. Studies such as [66] show that human reasoning process rarely exceeds three steps, so $k = 0, 1, 2$ is used in this work. The level-0 vehicles use a hand-crafted policy that commands one of the three actions, "maintain speed", "decelerate", or "hard decelerate", based on the range and range rate with the vehicle in its front to represent vehicle behavior under minimal rationality; level-1 and level-2 vehicles use policies trained by RL assuming that the surrounding vehicles are all level-0 and level-1, respectively. Moreover, the level-1 and 2 vehicles have a larger action space consisting of seven actions, including acceleration, deceleration, and lane change. Overall, the results of [32] indicate that the level-0 drivers/vehicles have the most conservative behaviors, while level-1 vehicles behave the most aggressively such as driving faster and frequently making lane changes, and the aggressiveness of level-2 vehicles falls between level-0 and level-1.

Remark 5.2. Similarities and differences between the setup considered in this work and that in [32] are highlighted in Table 5.2, in terms of models, reward functions, surrounding traffic, observation space and action space, RL algorithm, and training process. Details are given in the subsequent subsections.

5.4.2 Observation and Action Spaces

The observation space, i.e., input space, for our autonomous vehicle control policy is extended from the observation space for the level-k driver policies. The observation space for the level-k policies has 11 observations, including the range and range rate of the ego vehicle to the vehicles in its front, front right, front left, rear right, and rear left, as well as the lane index of the ego vehicle. The range is categorized by "far", "nominal", or "close", and the range rate is categorized by "moving away", "stable", and "approaching". The simulator is configured for a three-lane highway. As a result, the total number of possible states in the level-k policy is 3^{11}. Here, a state means a unique combination of observations.

To incorporate considerations of energy efficiency, it is necessary to enlarge the observation space by including additional observations. Since vehicle speed and battery SOC are critical factors that affect the PCU decision for the regenerative power distribution, as well as the component efficiencies

Table 5.2: Comparison summary on control development for the level-k ($k = 1, 2$) policies and the autonomous vehicle policy considered in this work (AV)

	Level-k for $k = 1, 2$	AV
BEV model	Not included	Included
Reward function	R_1	$R_1 + R_2$
Surrounding traffic during RL	Level-$(k-1)$	A mixture of level-0, level-1, and level-2 with a certain ratio
Observation space	11 observations	11 observations plus V and SOC
Action space	7 actions	Same as level-k
RL algorithm	Jaakkola RL algorithm	Same as level-k
After training, assign level-0 policy to states visited fewer than n times	$n = 20$	$n = 40$

of the battery and the MG, they are added to the observation space, each being categorized by "high", "medium", and "low". Consequently, the total number of possible states for our autonomous vehicle control policy increases to 3^{13}.

The action space, i.e., output space, for the proposed control design is the same as that for the level-k driver policies. It includes the following seven actions: 1) maintain speed, 2) accelerate, 3) decelerate, 4) hard accelerate, 5) hard decelerate, 6) move left (if the vehicle is not in the left-most lane) and 7) move right (if the vehicle is not in the right-most lane).

The exact definitions of the observation and action spaces for the level-k driver policies are given based on the parameters including the relative longitudinal position and speed thresholds, acceleration rates, deceleration rates, and lane change rates, whose values are the same as in [32] so they are not repeated here. For the two additional observations of the autonomous vehicle control policy, i.e., the vehicle speed and battery SOC, the thresholds that divide the three categories are, respectively, $17.22\ \mathrm{m/s}$ and $22.22\ \mathrm{m/s}$, and 70% and 80%, chosen by trial and error.

5.4.3 Reward Function

The reward function used for RL training is as follows,

$$\mathcal{R} = R_1 + R_2, \tag{5.17}$$

where

$$R_1 = w_1\, c + w_2\, v + w_3\, h + w_4\, u, \tag{5.18}$$

$$R_2 = w_5\, e. \tag{5.19}$$

Here, $w_i > 0$, $i = 1, \ldots, 5$ are weights, and c, v, h, u, and e are reward features. In particular, the reward function consists of two parts: The first part R_1 with the terms c, v, h, and u accounts for the safety, performance, and comfort requirements and shares the same setup as for the level-k policies. The second part R_2 is an additional term that accounts for energy efficiency.

The reward features and their corresponding weights are chosen based on engineering insight and tuning by simulation as follows:

- c accounts for constraint violations,
$$c = \begin{cases} -1, & \text{if a collision occurs to the ego vehicle,} \\ 0, & \text{otherwise,} \end{cases} \tag{5.20}$$
with $w_1 = 10,000$.

- v accounts for travel speed,
$$v = \frac{V - v_n}{a}, \tag{5.21}$$
where V is the speed of the ego vehicle in the longitudinal direction, and the constants v_n, a nominal speed, and a, a nominal acceleration rate, are used to scale this term to the same order of magnitude of the other terms, with $w_2 = 5$.

- h accounts for headway, encouraging the ego vehicle to keep a reasonable distance from preceding vehicles,
$$h = \begin{cases} 1, & \text{if headway} \in \text{“far”}, \\ 0, & \text{if headway} \in \text{“nominal”}, \\ -1, & \text{if headway} \in \text{“close”}, \end{cases} \tag{5.22}$$
with $w_3 = 1$. Here, “headway” means the range of the ego vehicle to the vehicle in its immediate front.

- u accounts for control effort,
$$u = \begin{cases} 0, & \text{if action} = \text{“maintain speed”}, \\ -1, & \text{if action} = \text{“accelerate” or “decelerate”}, \\ -3, & \text{if action} = \text{“move left” or “move right”}, \\ -5, & \text{if action} = \text{“hard accelerate” or “hard decelerate”}, \end{cases} \tag{5.23}$$
with $w_4 = 1$.

- e is for energy efficiency, defined by the time derivative of $MPGe$ as
$$e = \frac{\mathrm{d}MPGe}{\mathrm{d}t} = \frac{VE_{batt} - xP_{batt}}{E_{batt}^2}\,\gamma, \tag{5.24}$$
with $w_5 = 5$.

5.4.4 Training Algorithm

The goal of training is to find a control policy that maximizes the reward averaged over an infinite horizon. Using the settings described above, we formulate this problem as a partially observable Markov decision process (POMDP) problem since only certain observations are available to the ego vehicle. For example, if there are multiple vehicles in front of the ego vehicle on the same lane, the ego vehicle can only observe the relative range and range rate of the vehicle immediately in front of it. Thus, the algorithm used for training the control policy should guarantee convergence of the average reward with respect to POMDP problems. We choose to use the Jaakkola RL algorithm [34] since, under suitable assumptions, this algorithm guarantees convergence of the average reward to a local maximum for POMDP problems. The proof of such a convergence guarantee can be found in Appendix A of [93].

A summary of the Jaakkola RL algorithm is given in [32] and Sect. 1.2.7 of [93]. The key variables and equations are reviewed here. The algorithm iterates with two steps at every simulation time step t. First, the one-step reward R_t is evaluated based on the results of the simulation following the current policy π_t. Then, for each observation state $o \in \mathcal{O}$, and state and action pair $(o, a) \in \mathcal{O} \times \mathcal{A}$, the state-value functions $V(o|\pi_t)$ and the action-value functions $Q(o, a|\pi_t)$, also called Q-values, are updated based on the difference of $R_t - \bar{R}(\pi_t)$ where $\bar{R}(\pi_t)$ is the average reward for an infinite duration with the policy π_t. The state-value $V(o|\pi_t)$ represents the expected cumulative reward starting at state o following policy π_t, while the Q-values $Q(o, a|\pi_t)$ represents the expected cumulative reward if the state starts at o, we take action a first, and then follow policy π_t afterward. Specifically, the state-value functions and Q-values are updated with equations given as

$$\beta_t^o(o) = \left(1 - \frac{\chi_t^o(o)}{K_t^o(o)}\right)\gamma_t \beta_{t-1}^o(o) + \frac{\chi_t^o(o)}{K_t^o(o)}, \tag{5.25}$$

$$V(o|\pi_t) = \left(1 - \frac{\chi_t^o(o)}{K_t^o(o)}\right)V(o|\pi_{t-1}) + \beta_t^o(o)\Big(R_t - \bar{R}(\pi_t)\Big), \tag{5.26}$$

$$\beta_t^a(o, a) = \left(1 - \frac{\chi_t^a(o, a)}{K_t^a(o, a)}\right)\gamma_t \beta_{t-1}^a(o, a) + \frac{\chi_t^a(o, a)}{K_t^a(o, a)}, \tag{5.27}$$

$$Q(o, a|\pi_t) = \left(1 - \frac{\chi_t^a(o, a)}{K_t^a(o, a)}\right)Q(o, a|\pi_{t-1}) + \beta_t^a(o, a)\Big(R_t - \bar{R}(\pi_t)\Big), \tag{5.28}$$

where χ represents a binary (0 or 1) indicator function that equals to one if o or (o, a) is visited at the current time step, K is a function that counts how many times o or (o, a) has been visited, and γ_t is a time-dependent discount factor that takes a value between zero and one and converges to one

as t goes to infinity. In the second step, the policy is updated by the following equation

$$\pi_{t+1}(o,a) = (1-\epsilon)\pi_t(o,a) + \epsilon\hat{\pi}_t(o,a), \;\; \forall (o,a) \in \mathcal{O} \times \mathcal{A}, \tag{5.29}$$

where $\epsilon \in (0,1)$ is the learning rate and $\hat{\pi}_t$ is the greedy policy that maximizes

$$J_t(\pi, o) = \sum_{a \in \mathcal{A}} \pi(o,a)\Big(Q(o,a|\pi_t) - V(o|\pi_t)\Big), \;\; \forall o \in \mathcal{O}. \tag{5.30}$$

The process then moves on to the next time step and the iteration of the above two steps continues.

Note that the Jaakkola RL algorithm updates the value functions and Q-values at each time step using both the immediate one-step reward R_t and the infinite-horizon average reward $\bar{R}$. The one-step reward can make the policy update respond instantly to the large penalty of a safety constraint violation. The average reward, which in actual implementation is estimated by averaging all past instant rewards, eventually contains the weighted energy efficiency $MPGe$ computed over a long horizon. In this way, objectives with different time horizons are handled simultaneously.

5.4.5 Training Process

The level-k policies with $k = 1$ and 2 are obtained following Algorithms 1 and 2 of [32] (similar to Algorithm 1 below) with the reward function R_1 described above, and thus, they do not account for energy efficiency. We then use vehicles operating with level-k policies to provide the traffic environment for the RL training of our autonomous vehicle controller that considers energy efficiency.

The training process for the proposed autonomous vehicle control policy is summarized by the pseudo-code in Algorithm 5.1. Each training episode corresponds to a simulation trajectory with a duration of 200 seconds. This RL training process is similar to the level-k policy training process described by Algorithms 1 and 2 of [32] with the following major differences: 1) When initializing a training episode, we initialize the ego vehicle battery SOC randomly according to a uniform distribution on the range of $[15\%, 90\%]$. For Algorithm 2 of [32], however, since SOC is not a state of the level-k models, this initialization step does not exist; 2) A traffic environment consisting of a mixture of level-0, 1, and 2 vehicles with a ratio of 15%, 55%, and 30% is used to train our autonomous vehicle control policy, while when training a level-k policy, by definition, vehicles in the environment are all level-$(k-1)$; 3) Due to the increase of the size of the observation space, the total number of possible observation combinations is 9 times greater than for the level-k policies. With the same number of training episodes (i.e., 50,000, determined/limited by the affordable computational resources such as training time duration), it is more likely that some states are not sufficiently visited during the training. Thus, an increased value of the parameter n is used in the last step for the autonomous vehicle policy training, where n represents the number of times a state has to be visited during training, lest it be assigned the level-0 policy.

Algorithm 5.1: Training process

Initialize the ego car's policy with equal action probabilities for every state.
$episode \leftarrow 0$.
while $episode \leq 50,000$ **do**
    Randomly select the number of surrounding cars, $n_c \in [21, 30]$.
    Initialize surrounding cars with level-k policies with probabilities of 15%, 55% and 30% for $k = 0$, 1 and 2.
    Initialize the ego car with $SOC \in [15\%, 90\%]$.
    while $t \leq 200$ **do**
        Run simulation and evaluate the ego car's policy with the reward function $\mathcal{R}$.
        Update the ego car's policy.
        if *a collision occurs to the ego vehicle* **then**
            Terminate the current episode.
        end
    end
    $episode \leftarrow episode + 1$.
    Assign the level-0 policy to states visited less than $n = 40$ times.
end

5.4.6 Autonomous Vehicle Control Policy for Benchmarking

For comparison purposes, a second RL-based policy is trained in the mixed traffic environment described above. This benchmark policy uses only R_1, as defined in (5.18), as its reward function, and allows one to study the differences between considering the fuel economy or not, in similar traffic conditions.

In addition to policies trained by RL, the FSM-based policy described in [64] is adopted for comparison. The FSM-based policy is a rule-based controller with three modes including cruise control, adaptive cruise control, and lane change control. Switches between modes are triggered when certain traffic conditions are satisfied. The FSM-based policy is calibrated to optimize safety and performance, while the energy efficiency is not being considered.

5.5 Results

5.5.1 Training for RL-Based Policies

As discussed above, each RL-based policy is trained for 50,000 episodes. Figure 5.3 shows the values of the average reward as the training progresses for the level-1 policy, level-2 policy, the proposed autonomous vehicle control policy with the energy efficiency consideration (AV w/ e), and the benchmark policy that does not account for the energy efficiency (AV w/o e). It can be observed that the average rewards all converge smoothly, suggesting the success of the RL procedures.

The value of the converged average reward of each policy is a combined result of the different reward features in $\mathcal{R}$. For example, the converged average reward of the level-2 policy is higher than those of the other policies. This is because, among the level-0, 1, and 2 policies, the level-1 policy is the most aggressive as concluded in [32], tending to make many lane changes to pursue higher travel speeds. Since the traffic environment for training the level-2 policy is composed of purely level-1 vehicles, the level-2 policy is relatively conservative and collisions are less likely. Moreover, the overall faster traffic flow allows a higher travel speed of the ego vehicle. Thus, the coupled effect of these factors leads to a higher converged average reward for the level-2 policy.

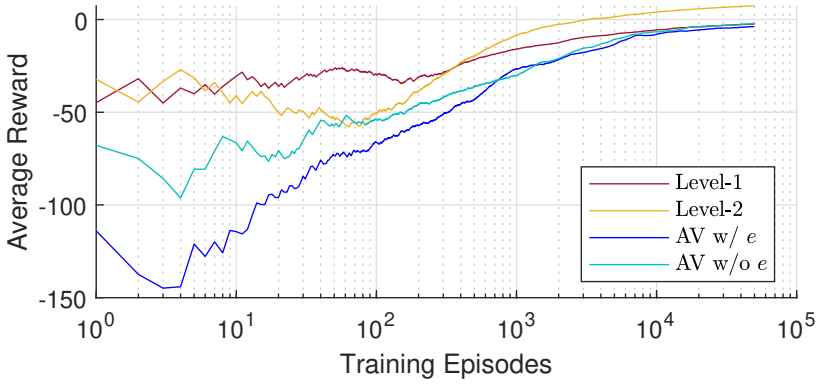

Figure 5.3: Average reward evolution during RL.

5.5.2 Control Performance

5.5.2.1 Evaluation Process

The control policies are evaluated based on simulations using the process described by the pseudo-code in Algorithm 5.2. In particular, for each policy and each traffic density (represented by the number of surrounding vehicles in traffic, ranging from 0 to 30), $10,000$ simulation episodes are run, each with a duration of 200 seconds. Then, the policy is evaluated with respect to four evaluation metrics, including:

- Constraint violation rate, defined as the percentage of simulation episodes where a collision occurs to the ego vehicle;
- Average number of lane changes per simulation episode;
- Average $MPGe$;
- Average travel speed.

Algorithm 5.2: Evaluation process

for $n_c = 0 : 30$ **do**
  $episode \leftarrow 0$.
  while $episode \leq 10,000$ **do**
    Initialize the ego car with $SOC \in [15\%, 90\%]$ and the control policy to be evaluated.
    Initialize surrounding cars with level-k policies randomly with probabilities of 15%, 55% and 30% for $k = 0$, 1 and 2.
    Simulate and record variables relevant to the evaluation metrics.
    $episode \leftarrow episode + 1$.
  end
  Compute and output the evaluation metric values.
end

5.5.2.2 Simulation Result Analysis

Proposed AV Control Policy Figures 5.4 to 5.6 show the results of different policies in regards to the four aforementioned evaluation metrics as functions of the traffic density. Figures 5.4(a) and 5.5 show the constraint violation rate, and Figs. 5.4(b) and 5.6(a) show the average number of lane changes. It can be observed that, when driving in the mixed traffic environment (vs. Mix), the proposed policy (AV w/ e) has the lowest constraint violation rate among all policies that can perform lane changes, indicating good safety features.

Figure 5.4(c) compares the average $MPGe$ among the three AV control policies, i.e., the proposed policy (AV w/ e), the RL-based benchmark policy (AV w/o e), and the FSM-based policy. It shows that the proposed policy with the energy efficiency consideration is more energy-efficient than the other two policies, verifying that the additional observations (vehicle speed and SOC) and the energy efficiency term R_2 in the reward function $\mathcal{R}$ are effective for the improvement of energy efficiency.

The average travel speed of each policy is shown in Figs. 5.4(d) and 5.6(b). It can be observed that at low traffic density, the autonomous vehicle controlled by the proposed AV policy drives at a higher average speed, close to that of level-2 vehicles; but when the traffic gets denser, the autonomous vehicle slows down to an average speed close to that of a level-0 vehicle. This can be explained with the help of Fig. 5.4(b): The average number of lane changes of the proposed policy varies only slightly for different traffic densities, since when the traffic density is low, there is not much need to change lanes, while when the traffic density gets high, it may be neither safe nor energy-efficient to perform lane changes. This feature distinguishes the behavior of the proposed policy from the level-1 policy and the FSM-based policy that prefer higher travel speeds and thus make many lane changes to achieve them.

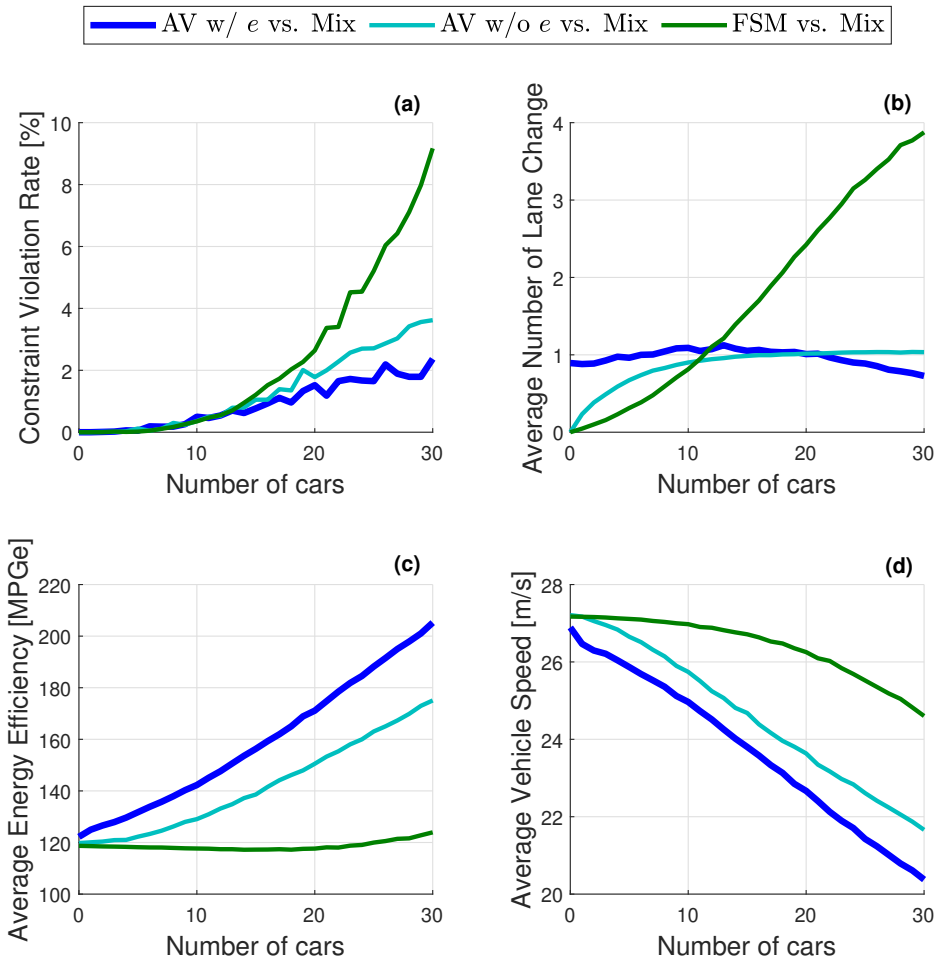

Figure 5.4: Evaluation results for AV control policies in traffic environments of different traffic densities: (a) Constraint violation rate; (b) Average number of lane changes per simulation episode; (c) Average $MPGe$; (d) Average travel speed.

Miscellaneous Observations The results indicate that the effects of the different evaluation aspects, that are closely related to the five features in the reward function, are not decoupled and can largely affect each other. For example, for the level-k policies, although the level-0 policy has the lowest constraint violation rate, it is a very conservative policy that does not allow lane changes (as illustrated in Fig. 5.6(a)) and has the lowest average vehicle speed (as shown in Fig. 5.6(b)). When driving in the level-0 environment (vs. Level-0), consequently, the level-1 policy also has a very low constraint violation rate. However, when driving in the mixed traffic environment, where there are other level-1 cars and level-2 cars, the level-1 policy has the highest constraint violation rate instead.

It is worth highlighting some additional observations in the simulation results. Firstly, for policies trained by RL, the average $MPGe$ increases as the average travel speed decreases in denser traffic. This is attributed to the fact that for highway driving, the energy efficiency at the vehicle

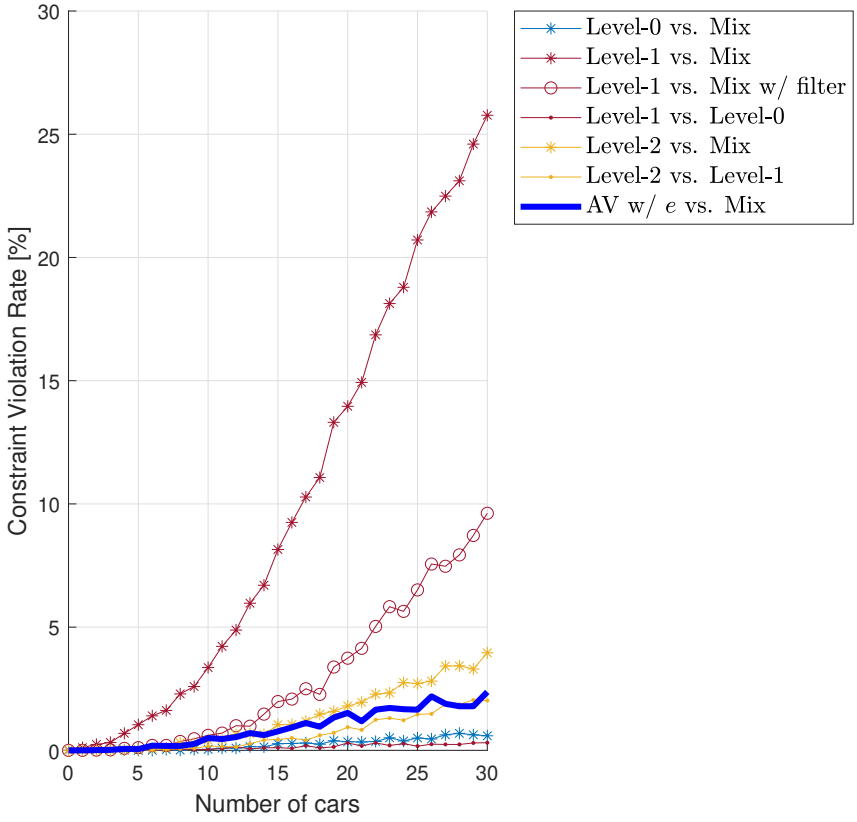

Figure 5.5: Constraint violation rates for level-k and the proposed AV control policies in traffic environments of different traffic densities.

level is affected largely by energy losses from rolling and aerodynamic resistance that increase as the vehicle travel speed increases. To see the significant impact of rolling and aerodynamic resistance on energy consumption, let us consider and compare two cases: In the first case, the vehicle is driving at a constant speed of $20.5\,\mathrm{m/s}$. In the second case, the vehicle is driving at $24.5\,\mathrm{m/s}$. These two cases roughly represent, respectively, the average longitudinal behavior of the proposed AV policy and that of the FSM-based policy with 30 surrounding vehicles, shown in Fig. 5.4(d). Here, we ignore the differences in the MG efficiency by assuming a constant value η_0. Then, we have that the MG power consumed by the rolling and aerodynamic resistance can be calculated as

$$\begin{aligned} P_l &= (T_l/g)\,\omega_{mg}/\eta_0 \\ &= (a_l + b_l\,V + c_l\,V^2)V/(r\eta_0), \end{aligned} \tag{5.31}$$

depending cubically on the vehicle speed V. Without considering the discrepancy in the battery and

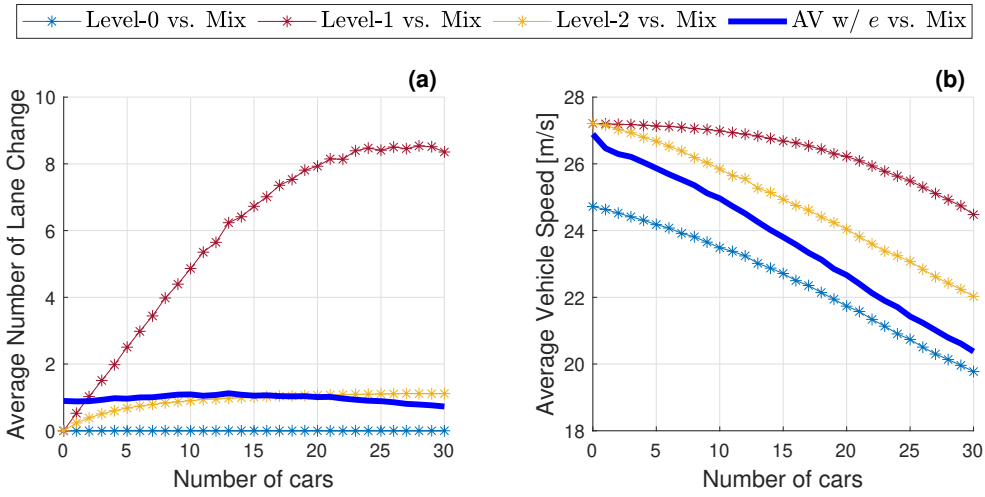

Figure 5.6: Average number of lane changes per simulation episode and average travel speed for level-k and the proposed AV control policies in traffic environments of different traffic densities.

MG efficiency, we then use (5.31) to estimate the difference in the MG power used to counteract the rolling and aerodynamic resistance. We obtain that the discrepancy between the two cases is about 33%. This contributes to the difference in the $MPGe$ shown in Fig. 5.4(b), where the change is about 64%. It is within a reasonable range according to Table 5.1 (e.g., [60, 62]).

Note, however, that the proposed policy does not always operate the autonomous vehicle at a low speed, as the reward function represents several different objectives besides energy efficiency. In general, energy efficiency depends on the traffic scenario, the powertrain type, as well as the component specifications. This fact highlights the benefit of modeling the powertrain system using maps, so that components can be easily sized or swapped.

Secondly, it is observed in Fig. 5.4(b) that when there is no other vehicle in traffic ("zero-traffic"), the autonomous vehicle controlled by the proposed policy makes on average one lane change. This is because the policy by RL converges to a solution where when there is no other vehicle in the immediate vicinity of the ego vehicle, the ego vehicle tends to change to and stay in the right-most lane to reduce the possibility of having interactions/conflicts with other vehicles that may degrade its safety and energy efficiency in the future. Note also that such a solution may only be locally optimal (i.e., not globally optimal), as the Jaakkola RL algorithm used to train the policy guarantees only convergence to a local optimum (and not necessarily the global optimum) [34].

Intuitively speaking, vehicles need not change lanes when there are no slower vehicles in their front blocking their ways. This is the case for most policies shown in Figs. 5.4(b) and 5.6(a) except for the proposed policy for which the average number of lane changes in the zero-traffic environment is close to but slightly less than one. For most cases, when initialized in the middle

lane, the autonomous vehicle controlled by the proposed policy immediately makes a lane change to the right, as explained above. However, for some states with high SOC that were not visited enough during RL training, the policy was overridden by the level-0 policy that does not perform lane changes (see the last line of Algorithm 5.1). This caused the average number of lane changes to be slightly less than one. Note also that the training is conducted only for traffic environments with 21 to 30 surrounding vehicles, i.e., not covering the zero-traffic environment. Such sub-optimal behavior in the zero-traffic environment of the trained policy indicates that it might be beneficial to conduct training for a wider range of traffic environments if computational resources allow.

Thirdly, one of the major contributors to the high constraint violation rate of the level-1 policy when operating in the mixed traffic environment is its high frequency of lane changes. Two problematic scenarios related to lane changes have been identified in [32]. The first case involves a scenario where the ego vehicle originally driving in the right (or left) lane and another vehicle originally driving in the left (or right) lane in an almost parallel longitudinal position with the ego vehicle simultaneously start to perform lane changes to the middle and lead to a side collision between them. The second case involves a scenario where the ego vehicle starts to change lanes trying to overtake a vehicle in its front, but at the same time, the preceding vehicle also starts to change lanes in the same direction (e.g., trying to overtake another vehicle) and blocks the ego vehicle's overtaking. Since the level-1 policy is trained using an environment consisting of only level-0 vehicles that do not change lanes, these two "unrare-in-reality" scenarios have never occurred during the RL training. Consequently, the level-1 policy fails to learn to avoid such scenarios. We have identified all constraint violation cases in the Level-1 vs. Mix data that belong to these two scenarios and computed the constraint violation rate after filtering out these cases. The result is plotted in Fig. 5.5, called Level-1 vs. Mix w/ filter. It can be seen that the constraint violation rate of the level-1 policy after this filtering is significantly reduced.

If such an issue happens when developing autonomous vehicle control algorithms, where problematic scenarios can be clearly identified, they can be handled by specific add-on mechanisms. For example, the autonomous vehicle may be commanded to go back to its original lane when either of the above two cases is detected.

5.6 Concluding Remarks

In this chapter, an autonomous vehicle control policy is developed focusing on energy efficiency optimization while safety, performance, and comfort are balanced. We first discuss the potential of autonomous vehicle (AV) controls for energy-efficient driving and the major challenges to develop such control policies. Then, we show the powertrain model built to capture the energy consumption of a battery electric vehicle (BEV), integrated with the highway traffic simulator consisting of

cognitive driver behavioral models based on level-k game theory. An AV control policy is trained by reinforcement learning (RL) for this BEV and compared with two benchmark policies as well as the level-k policies from different evaluation perspectives.

Analysis of the results indicates that the addition of the energy efficiency term in the RL reward function, in addition to the expanded observation space to include the vehicle speed and SOC, is effective in improving the energy efficiency while maintaining low collision rates. Through analysis of the BEV powertrain model, the increase of the energy efficiency represented by $MPGe$ is likely dominated by the reduction of the average vehicle speed that lowers the rolling and aerodynamic resistance. However, this does not make the vehicle always travel at the lower speed limit, which highlights the capability of the RL-based approach that does a good job in balancing travel speeds, safety, and efficiency. The results also imply the potential to further extend and explore the control design in terms of higher computational efficiency and advanced RL algorithms for control performance improvement. In the future, our AV policy may serve as a baseline control strategy for more advanced autonomous driving control development.

CHAPTER 6

Analysis of Multistage Hybrid Powertrains Using Multistage Mixed-Integer Trajectory Optimization

6.1 Introduction

Modern hybrid powertrains are becoming increasingly complex. They often have multiple controlled degrees of freedom, e.g., gear selection, engine on/off, power split, etc., the operation of which needs to be pre-determined before a meaningful evaluation of the system-level performance can be carried out. Traditionally, this has been accomplished by pre-defining an operating strategy that uses a combination of feedforward and feedback calibration tables that determine power split, engine on/off, and gear selection as functions of, for example, the pedal angle and vehicle speed. These tables must be carefully calibrated to meet design requirements and are placed in closed-loop with a simulation model during performance analyses. However, hybrid powertrains are often highly coupled and over-actuated. Nonlinearities, constraints, and interactions between actuators can make it difficult for engineers to find "optimal" calibrations resulting in biased analyses, reduced overall vehicle performance, and increased development time and costs.

Trajectory optimization is an attractive approach for overcoming the aforementioned difficulties. It allows design engineers to determine optimal input profiles and evaluate specific scenarios without pre-defining a specific controller and without designing calibration tables [38]. For example, studying certain scenarios, such as a 0 to 100 kph acceleration maneuver or a 40 to 70 kph passing maneuver, can help engineers evaluate the capabilities of different powertrain configurations, perform trade-off studies early in the design process, and cascade requirements to powertrain subsystems.

There are several trajectory optimization software packages available, including GPOPS-II [94], OpenOCL [95], Drake [96], and PSOPT [97]. These can efficiently solve nonlinear trajectory optimization problems using sophisticated discretization techniques and leveraging powerful nonlinear programming solvers like IPOPT [98] or SNOPT [99]. However, the presence of integer or binary

valued degrees of freedom in hybrid powertrains, e.g., engine on/off or gear selection, leads to mixed-integer nonlinear programming (MINLP) problems that are difficult to solve reliably.

In this chapter, we present a trajectory optimization-based analysis of a multistage hybrid powertrain (MSHP) of the type used in 2018 Lexus LC500h vehicles, as shown in Fig. 6.1. Compared with the traditional power-split hybrid configuration, which uses a planetary gear set with motor/generators (MGs) to achieve functionality similar to that of a continuously variable transmission (CVT), the MSHP extends and combines a Toyota Hybrid System (THS) power-split hybrid configuration with a four-speed automatic transmission to offer increased performance, drivability, and fuel economy by providing a wider operating range for the powering components [37]. However, the added degree of freedom offered by the gearbox increases the complexity of the powertrain control problem. Moreover, instead of a continuous control input, such as for the engine and MG torques, the gear number is a discrete variable. This requires optimization routines that can simultaneously handle both continuous and discrete control inputs.

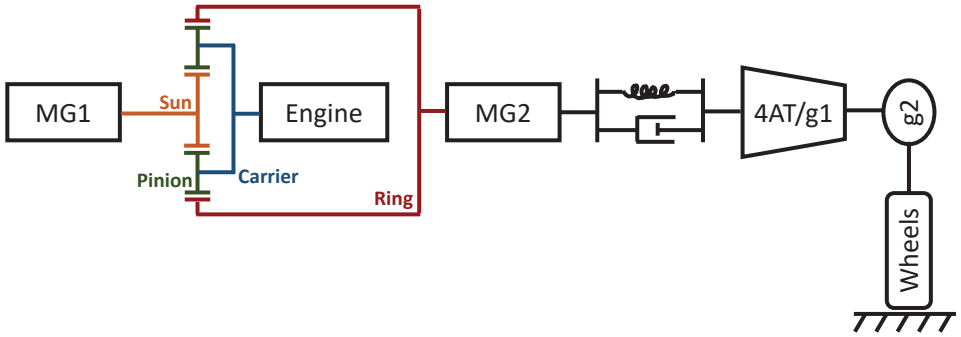

Figure 6.1: The layout of the multistage hybrid powertrain. The powertrain system consists of the motor/generator 1 (MG1), motor/generator 2 (MG2), engine, driveline compliance, 4-speed automatic transmission (4AT/g1), differential (g2), and wheels and tires.

The MSHP is an example of a switched system [100], a class of hybrid systems with discrete modes that are common in the automotive domain [101]. There is a rich literature on optimal control of switched systems [102], often focused around switching time optimization [103, 104], i.e., determining optimal switching instants given fixed modes. Several bi-level descent methods, which alternate between optimizing over the mode sequence (discrete) and the switching times (continuous), with guaranteed convergence properties have also been proposed [105, 106] in addition to a strong variations method based on the maximum principle [107]. Automotive applications of switched systems and switching time optimization include: skid steering [104], roll control [108], mechanical hybrid powertrains [109], and start-up of automated manual transmissions [110, 111].

In this chapter, a highly parallelizable bi-level optimization method is proposed for switched sys-

tems and is used to study an MSHP. To the best of the author's knowledge, this is the first trajectory optimization study for such a powertrain configuration in the literature. Moreover, the proposed optimization methodology is distinguished from others due to its simplicity, broad applicability, and relative ease of implementation. This work focuses on offline trajectory optimization for design phase analysis. Use of these techniques for other purposes, e.g., as a part of a model predictive controller online, and/or to assist the calibration table design, is outside the scope of this chapter and a topic for future research.

The chapter is organized as follows. First, we derive a control-oriented model of an MSHP that is suitable for preliminary design studies and potentially for model-based control design, e.g., model predictive control. Next, we propose a simple and highly parallelizable bi-level trajectory optimization procedure to enable optimization over both discrete and continuous variables. Finally, we present a case study wherein we apply our proposed trajectory optimization methodology to the MSHP model to perform a performance/drivability trade-off study, specifically, 40 to 70 kph acceleration time versus jerk analysis.

6.2 Powertrain Modeling

Figure 6.1 shows the layout of the MSHP. The motor/generator1 (MG1), motor/generator 2 (MG2), and engine are connected to the sun gear, ring gear, and carrier, respectively. The MG2 also connects to the 4-speed automatic transmission (4AT/g1) through a torsional compliance that is represented by a parallel spring and damper to approximate the driveline dynamics. The MSHP has a single-speed differential (g2), and the vehicle is assumed to be driving on level ground.

The model has three continuous inputs,

$$u = \begin{bmatrix} u_{mg1} & u_{mg2} & u_{eng} \end{bmatrix}^T, \tag{6.1}$$

which are the MG1 torque command, MG2 torque command and engine torque command. For the case study of 40 to 70 kph acceleration, we have one discrete input

$$v = v_\eta \in \{1, 2, 3, 4\}, \tag{6.2}$$

which represents the automatic transmission gear selection command. In this chapter, we assume the engine remains on, as indeed would be the case during the acceleration scenarios that we consider, and do not consider engine on/off as a control input.

There are ten states in the model:

$$x = \begin{bmatrix} T_{mg1} & T_{mg2} & T_{eng} & f_{\eta_1} & f_{\eta_2} & f_{\eta_3} & \omega_s & \omega_r & \Delta\alpha_{comp} & s \end{bmatrix}^T, \tag{6.3}$$

including

- $T_{mg1}, T_{mg2}, T_{eng}$: for three first-order low-pass filters that model the MG1, MG2 and engine dynamics,

- $f_{\eta_1}, f_{\eta_2}, f_{\eta_3}$: three states in a third-order Bessel filter used to model the automatic transmission dynamics by filtering the gear number from the control command, with time a constant $\tau_\eta = 5\,\mathrm{s}$,
- ω_s: the angular velocity of the sun gear,
- ω_r: the angular velocity of the ring gear,
- $\Delta\alpha_{comp}$: the angular displacement across the driveline compliance, and
- s: the longitudinal speed of the vehicle.

The actuator dynamics are modeled by first-order filters,

$$\tau_i \dot{T}_i = u_i - T_i, \tag{6.4}$$

where $i \in \{mg1, mg2, eng\}$. The time constants are $\tau_{mg1} = \tau_{mg2} = 0.05\,\mathrm{s}$, and $\tau_{eng} = 0.3\,\mathrm{s}$. The dynamics of the Bessel filter can be expressed in state space form as

$$\begin{bmatrix} \dot{f}_{\eta_1} \\ \dot{f}_{\eta_2} \\ \dot{f}_{\eta_3} \end{bmatrix} = A_\eta \begin{bmatrix} f_{\eta_1} \\ f_{\eta_2} \\ f_{\eta_3} \end{bmatrix} + B_\eta v_\eta,$$

$$\eta = C_\eta \begin{bmatrix} f_{\eta_1} \\ f_{\eta_2} \\ f_{\eta_3} \end{bmatrix}, \tag{6.5}$$

where η is the filtered/actual gear number, and A_η, B_η, and C_η are matrices generated by the MATLAB function "besself" given the order and time constant of the filter.

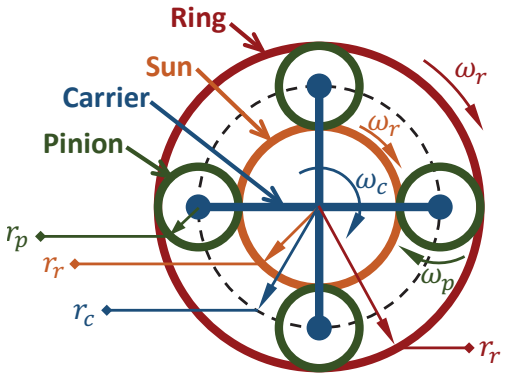

Figure 6.2: Planetary gear set geometry and kinematics abstraction. The planetary gear set consists of sun gear (orange), ring gear (red), carrier (blue), and pinion gears (green).

The geometry and kinematics of the planetary gear set is shown in Fig. 6.2. The model is based on a power-split HEV model in Autonomie (developed by the Argonne National Lab). The dynamic equations for the sun gear angular velocity and ring gear angular velocity are derived from the THS

dynamics and kinematics, described by the following equations:

$$\dot{\omega}_s J_s = T_s - N_p r_s F_s, \tag{6.6}$$

$$\dot{\omega}_c J_{cc} = T_c + N_p r_s F_s - N_p r_r F_r, \tag{6.7}$$

$$\dot{\omega}_r J_r = T_{r_0} + N_p r_r F_r, \tag{6.8}$$

$$\dot{\omega}_p J_p = -r_p F_s - r_p F_r, \tag{6.9}$$

$$r_r \omega_r = r_c \omega_c + r_p \omega_p, \tag{6.10}$$

$$r_s \omega_s = r_c \omega_c - r_p \omega_p, \tag{6.11}$$

with J_{cc} defined by

$$J_{cc} = J_c + J_{pc}, \tag{6.12}$$

$$J_{pc} = N_p(J_p + m_p r_c^2), \tag{6.13}$$

where $\omega_s = \omega_{mg1}$, $\omega_c = \omega_{eng}$, $\omega_r = \omega_{mg2}$, and ω_p are angular velocities for the sun gear (coupled to MG1), carrier (coupled to the engine), ring gear (coupled to MG2), and pinion gears, respectively. We denote by $T_s = T_{mg1}$, $T_c = T_{eng}$, and T_{r_0} the torques applied to the sun gear by MG1, carrier by the engine, and ring gear axles, respectively. Moreover, F_s is the reaction force from a pinion gear to the sun gear, F_r is the reaction force from a pinion gear to the ring gear, r_s, r_c, r_r, and r_p are effective radius of the sun gear, carrier, ring gear and pinion gears, respectively, J_s, J_c, J_r, and J_p are inertias at the sun gear (including MG1), carrier (including engine), ring gear (including MG2), and a single pinion gear, respectively, J_{pc} is the total inertia of the pinion gears, J_{cc} is the total inertia of the pinion gears and the carrier, N_p is the number of pinion gears, and m_p is the mass of a single pinion gear.

By rearranging (6.6) to (6.11) and differentiating the two kinematic equations of (6.10) and (6.11) with respect to time, we obtain

$$T_s = \dot{\omega}_s J_s + N_p r_s F_s, \tag{6.14}$$

$$T_c = \dot{\omega}_c J_{cc} - N_p r_s F_s + N_p r_r F_r, \tag{6.15}$$

$$T_{r_0} = \dot{\omega}_r J_r - N_p r_r F_r, \tag{6.16}$$

$$0 = \dot{\omega}_p J_p + r_p F_s + r_p F_r, \tag{6.17}$$

$$0 = \dot{\omega}_s \rho - \dot{\omega}_c (1 + \rho) + \dot{\omega}_r, \tag{6.18}$$

$$0 = \dot{\omega}_s \rho - \dot{\omega}_r + \dot{\omega}_p (1 - \rho), \tag{6.19}$$

where $\rho = \frac{r_s}{r_r}$ is the ratio of the radii of the sun gear and the ring gear. Equations (6.14) to (6.19) can be written compactly as

$$\begin{bmatrix} T_s & T_c & T_{r_0} & 0 & 0 & 0 \end{bmatrix}^T = Q \begin{bmatrix} \dot{\omega}_s & \dot{\omega}_c & \dot{\omega}_r & \dot{\omega}_p & F_s & F_r \end{bmatrix}^T, \tag{6.20}$$

where

$$Q = \begin{bmatrix} J_s & 0 & 0 & 0 & N_p r_s & 0 \\ 0 & J_{cc} & 0 & 0 & -N_p r_s & N_p r_r \\ 0 & 0 & J_r & 0 & 0 & -N_p r_r \\ 0 & 0 & 0 & J_p & r_p & r_p \\ \rho & -(1+\rho) & 1 & 0 & 0 & 0 \\ \rho & 0 & -1 & 1-\rho & 0 & 0 \end{bmatrix}, \tag{6.21}$$

which is a constant invertible matrix. Thus, we have

$$\begin{bmatrix} \dot{\omega}_s & \dot{\omega}_c & \dot{\omega}_r & \dot{\omega}_p & F_s & F_r \end{bmatrix}^T = Q^{-1} \begin{bmatrix} T_s & T_c & T_{r_0} & 0 & 0 & 0 \end{bmatrix}^T. \tag{6.22}$$

The dynamic equations of the sun gear angular velocity, i.e., MG1 speed, and the ring gear angular velocity, i.e., MG2 speed, are thus

$$\dot{\omega}_s = Q_{11}^{-1} T_s + Q_{12}^{-1} T_c + Q_{13}^{-1} T_{r_0}, \tag{6.23}$$

$$\dot{\omega}_r = Q_{31}^{-1} T_s + Q_{32}^{-1} T_c + Q_{33}^{-1} T_{r_0}, \tag{6.24}$$

where Q_{ij}^{-1}, $\mathrm{i}, \mathrm{j} = 1, 2, \cdots, 6$, denotes the element of Q^{-1} at the ith row and jth column. The angular velocity of the carrier, which is equal to the engine speed, can be computed using the following equation

$$\omega_c = \frac{\omega_r + \rho \omega_s}{1+\rho}. \tag{6.25}$$

The model includes a lumped rotational spring/damper that represents the driveline compliance, where input and output torques (described below) depend on the angular displacement between the MG2 output shaft and the automatic transmission input shaft. The dynamic equation of the angular displacement is given by

$$\Delta \dot{\alpha}_{comp} = \Delta \omega_{comp}, \tag{6.26}$$

where $\Delta \alpha_{comp}$ is the angle difference across the driveline compliance, and $\Delta \omega_{comp}$ is the angular velocity difference across the driveline compliance, calculated as

$$\Delta \omega_{comp} = \omega_r - \omega_{t_i}, \tag{6.27}$$

The angular velocity of the transmission input shaft ω_{t_i} is computed using the following kinematic equations

$$\omega_{t_i} = g_1 \omega_{t_o}, \tag{6.28}$$

$$\omega_{t_o} = g_2 \omega_w, \tag{6.29}$$

$$\omega_w = \frac{s}{r_{tire}}, \tag{6.30}$$

where g_1 is the gear ratio across the automatic transmission, which is computed as

$$g_1 = a_{g_1} \eta^3 + b_{g_1} \eta^2 + c_{g_1} \eta + d_{g_1}, \tag{6.31}$$

based on a third-order polynomial fit with coefficients $a_{g_1}, b_{g_1}, c_{g_1}$ and d_{g_1} that approximate the gear ratio versus gear number curve, and where ω_{t_o} is the angular velocity of the shaft between the automatic transmission and the differential (where t_o means output of the transmission), g_2 is the fixed gear ratio of the differential, ω_w is the angular velocity of the wheels, s is the vehicle longitudinal speed, and r_{tire} is the effective tire radius.

The longitudinal vehicle speed dynamics are derived from Newton's laws and are given by

$$\dot{s} = \frac{g_1 g_2 T_{t_i} - F_l r_{tire}}{M r_{tire}}, \tag{6.32}$$

where s is the vehicle speed, M is the equivalent vehicle mass, F_l is the sum of driving resistance forces, and T_{t_i} is the torque applied to the shaft between the driveline compliance and the automatic transmission. We use a polynomial model for the driving resistance,

$$F_l = a_l s^2 + b_l s + c_l, \tag{6.33}$$

where a_l, b_l, and c_l are the resistance force coefficients. The compliance torque is computed using

$$T_{t_i} = k\Delta\alpha_{comp} + d\Delta\omega_{comp}, \tag{6.34}$$

where k and d are stiffness and damping coefficients of the driveline compliance.

Given T_{t_i}, the torque T_{r_0} applied to the ring gear axle used in the dynamic equations (6.23) and (6.24) is computed as

$$T_{r_0} = T_{mg2} - T_{t_i}, \tag{6.35}$$

where T_{mg2} is the filtered MG2 torque.

Finally, the jerk can be calculated by taking the second time derivative of the vehicle speed and is given by

$$\ddot{s} = \ddot{s}(x, u, v) = \frac{\dot{g}_1 g_2 T_{t_i} + g_1 g_2 \dot{T}_{t_i} - \dot{F}_l r_{tire}}{M r_{tire}}, \tag{6.36}$$

where, by (6.31), (6.33), and (6.34),

$$\dot{g}_1 = 3a_{g_1}\eta^2\dot{\eta} + 2b_{g_1}\eta\dot{\eta} + c_{g_1}\dot{\eta}, \tag{6.37}$$

$$\dot{F}_l = 2a_l s\dot{s} + b_l\dot{s}, \tag{6.38}$$

$$\dot{T}_{t_i} = k\Delta\dot{\alpha}_{comp} + d\Delta\dot{\omega}_{comp}, \tag{6.39}$$

with

$$\dot{\eta} = C_\eta \begin{bmatrix} \dot{f}_{\eta_1} & \dot{f}_{\eta_2} & \dot{f}_{\eta_3} \end{bmatrix}^T, \tag{6.40}$$

from (6.5), and then by (6.28) to (6.30) we have

$$\Delta\dot{\omega}_{comp} = \dot{\omega}_r - \dot{g}_1\omega_{t_o} - g_1 g_2 \frac{\dot{s}}{r_{tire}}. \tag{6.41}$$

6.3 Multistage Mixed-Integer Optimization

The powertrain model in the previous section can be compactly expressed as the following nonlinear switched system

$$\dot{x} = f(x, u, v), \tag{6.42}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the continuous input, $v \in \mathcal{Q} = \{1, 2, 3, 4\}$ is the discrete input and $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathcal{Q} \to \mathbb{R}^n$ is twice continuously differentiable in x and u for any fixed v. Our objective is to solve the following optimal control problem (OCP)

$$\begin{aligned} \min_{T,x,u,v} \ & \Phi\big(x(T), v(T), T\big) + \int_0^T \ell\big(x(\tau), u(\tau), v(\tau)\big) d\tau \\ \text{s.t.} \ & \dot{x}(t) = f\big(x(t), u(t), v(t)\big) \ \ \forall t \in [0, T], \\ & x(0) = x_I, \ x(T) = x_f, \\ & c\big(x(t), u(t)\big) \leq 0, \\ & v(0) = v_I, \ v(T) \in v_f, \end{aligned} \tag{6.43}$$

where $T \in \mathbb{R}$ is the total acceleration duration, $\Phi : \mathbb{R}^n \times \mathcal{Q} \times \mathbb{R} \to \mathbb{R}$ is the terminal cost, $\ell : \mathbb{R}^n \times \mathbb{R}^m \times \mathcal{Q} \to \mathbb{R}$ is the incremental cost, $c : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$ represent inequality constraints, $x_I, x_f \in \mathbb{R}^n$ are initial and terminal state constraints, $v_I \in \mathcal{Q}$ is the initial discrete input constraint, $v_f \subseteq \mathcal{Q}$ is the set of terminal discrete input constraints, $u : [0, T] \to \mathbb{R}^m$ is bounded and measurable, $v : [0, T] \to \mathcal{Q}$ is piecewise constant, and $x : [0, T] \to \mathbb{R}^n$ is continuously differentiable. We also want to enforce that v can change at most once every $\Delta t_s > 0$ seconds (to account for the dynamics of the automatic transmission) and that the gear transitions satisfy a relationship of the following type

$$v_{j+1} \in \Gamma(v_j), \tag{6.44}$$

where the point to set mapping $\Gamma : \mathcal{Q} \rightrightarrows \mathcal{Q}$ represents transition constraints. For example, Γ could be used to enforce that the gear sequence is monotonically increasing in a given scenario. In this situation, Γ would be fully defined by the following,

$$\Gamma(1) = \{1, 2, 3, 4\}, \tag{6.45a}$$

$$\Gamma(2) = \{2, 3, 4\}, \tag{6.45b}$$

$$\Gamma(3) = \{3, 4\}, \tag{6.45c}$$

$$\Gamma(4) = \{4\}. \tag{6.45d}$$

Clutch pairing in automatic transmissions is another common constraint that can be encoded in Γ.

To make this problem tractable, we assume that at most $M \in \mathbb{N}$ stages, i.e., $M - 1$ gear transitions, will occur and that v remains constant in each stage, i.e., $v(t) = v_j \ \forall t \in [t_{j-1}, t_j), \ j = 1, \cdots, M$ where $t_0 < t_1 < \cdots < t_{M-1} < t_M$ with $t_0 = 0$ and $t_M = T$ are the switching times.

This leads to the following multistage OCP,

$$\min_{\tau,x,u,w} \ \phi\big(x(t_M), t_M\big) + \sum_{j=0}^{M-1} l\big(x, u, v_j, t_j\big) \tag{6.46}$$

$$\begin{aligned}
\text{s.t.} \quad & \forall j \in \{1, \cdots, M\}, \\
& \quad \dot{x}(t) = f\big(x(t), u(t), v_j\big) \ \ \forall t \in [t_{j-1}, t_j), \\
& \quad v_j \in \Gamma(v_{j-1}), \\
& \quad t_{j-1} + \Delta t_s \leq t_j, \\
& x(0) = x_I, \ x(T) = x_f, \\
& v_0 = v_I, \ v_M \in v_f, \\
& c\big(x(t), u(t)\big) \leq 0,
\end{aligned}$$

where $\Delta t_s > 0$ is the minimum switching time, $\tau = (t_1, \cdots, t_M)$, $w = (v_0, \cdots, v_M)$, $\phi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is the per-stage terminal cost, and

$$l\big(x, u, v_j, t_j\big) = \phi\big(x(t_j), t_j\big) + \int_{t_j}^{t_{j+1}} \ell\big(x(s), u(s), v_j\big) ds. \tag{6.47}$$

This is an optimal control problem for a switched/hybrid system so, following, e.g., [102, 105, 106], we decompose it into a bi-level optimization problem. At the same time, we also replace the continuous time portions of (6.46) with a discrete approximation. The resulting discretized bi-level problem can be written compactly as

$$\min_{w \in \mathcal{V}} \mathcal{P}^*(w), \tag{6.48}$$

where the multifunction $\mathcal{P} : \mathcal{Q}^{M+1} \rightrightarrows \mathbb{R} \cup [-\infty, \infty]$ represents solutions of a discretized OCP (which is described in detail below) and

$$\mathcal{V} = \big\{(v_0, v_1, \cdots, v_M) : v_{j+1} \in \Gamma(v_j), v_0 = v_I, v_M \in v_f\big\}$$

is the set of all admissible gear sequences. We adopt the convention that $\mathcal{P}^*(w) = \infty$ if the corresponding optimization problem is infeasible. The inner problem in (6.48) is not convex; it may have multiple local minima. In this chapter we use a gradient based optimizer (IPOPT) to solve the inner problems and we accept whichever solution the optimizer provides. One could use a more computationally intensive global optimizer if local solutions are not sufficient.

We form the discretized OCP by applying explicit Euler integration to each stage and introduce the notation x_i^j for the value of the state vector at the ith discretization point in the jth stage. Moreover, we parameterize the length of the stages by their interval lengths, i.e. we use $\Delta t_j = t_j - t_{j-1}$, as optimization variables in place of t_j and introduce a maximum stage length $\Delta t^{\max}$ which serves to ensure that the numerical approximation of the derivative remains reasonable during

the optimizer iterations. The resulting OCP is:

$$\min_{X,U,\Delta t} \sum_{j=1}^{M} \left[\phi(x_N^j, t_j) + \sum_{i=0}^{N-1} \ell(x_i^j, u_i^j, v_j) \right] \tag{6.49}$$

$$\begin{aligned}
\text{s.t.} \quad & \forall j \in \{1, \cdots, M\}, \\
& x_{i+1}^j = f_j(x_i^j, u_i^j, \Delta t_j) \ \ \forall i \in \{0, \cdots, N-1\}, \\
& c(x_i^j, u_i^j) \leq 0 \ \ \forall i \in \{0, \cdots, N\}, \\
& \Delta t_s \leq \Delta t_j \leq \Delta t^{\max}, \\
& x_0^1 = x_I, \ x_N^M = x_f, \\
& x_0^j = x_N^{j-1} \ \ \forall j \in \{2, \cdots, M\},
\end{aligned}$$

where the discrete dynamics are

$$f_j(x, u, \Delta t) = x + \frac{\Delta t}{N} f(x, u, v_j), \tag{6.50}$$

and the optimization variabkles are:

$$X = \begin{bmatrix} x_0^1 & \cdots & x_N^1 & \cdots & x_0^M & \cdots & x_N^M \end{bmatrix}^T, \tag{6.51a}$$

$$U = \begin{bmatrix} u_0^1 & \cdots & u_{N-1}^1 & \cdots & u_0^M & \cdots & u_{N-1}^M \end{bmatrix}^T, \tag{6.51b}$$

$$\Delta t = \begin{bmatrix} \Delta t_1 & \cdots & \Delta t_M \end{bmatrix}^T. \tag{6.51c}$$

By defining $z = (X, U, \Delta t)$ this OCP can be written compactly as a standard smooth nonlinear program of the form

$$\begin{aligned}
\min_z \ \ & f(z, w) \\
\text{s.t.} \quad & g(z, w) = 0, \\
& h(z) \leq 0,
\end{aligned} \tag{6.52}$$

that can be readily solved (for any fixed $w \in \mathcal{V}$) using established nonlinear programming solvers, e.g., IPOPT [98]. We use the CASADI package [112] to automatically generate derivative evaluation functions that are passed to IPOPT to solve (6.52). The set $\mathcal{V}$ of all admissible sequences is generated recursively using Algorithm 6.1.

Many methods in the literature, e.g., [105, 106], solve the upper level problem using gradient or Newton like techniques to minimize the number of times the inner problem (6.52) needs to be solved. Since we only have 4 gears and only need to consider a small number of gear change scenarios, we found it more effective (from an engineering workload perspective) to simply solve (6.52) for each $w \in \mathcal{V}$ in parallel and choose the best solution. If more scenarios need to be considered a genetic algorithm or one of the techniques in [105, 106] could be used instead. To summarize, our algorithm is as follows:

1. Use Algorithm 6.1 to generate the set $\mathcal{V}$ of all admissible discrete transitions.

2. For each sequence $w \in \mathcal{V}$ solve (6.52) to obtain $\mathcal{P}^*(w)$,
3. Pick the solutions corresponding to the lowest value of $\mathcal{P}^*(w)$.

Algorithm 6.1: GenerateAllSequences

Input: v_I, v_f, M
Output: $\mathcal{V}$
Algorithm
  $\mathcal{V} \leftarrow$ `Expand` $(v_I, v_I, M, 1)$
  foreach $w = (v_0, \cdots, v_M) \in \mathcal{V}$ **do**
    if $v_M \notin v_f$ **then**
      $\mathcal{V} \leftarrow \mathcal{V} \backslash w$
    end
  end
  return $\mathcal{V}$
Procedure `Expand` *(s,u,M,d)*
  if $d = M$ **then**
    return s
  end
  $u_s \leftarrow u$
  foreach $v \in \Gamma(s_{[d]})$ **do**
    $s_+ \leftarrow (s, v)$
    $u \leftarrow u \cup$ `Expand` $(s_+, u_s, M, d+1)$
  end
  return u

6.4 Application to a Multistage Hybrid Powertrain

In this section, we use our proposed multistage mixed-integer optimization methodology to study the MSHP system. As a case study, we consider a minimum time 40 to 70 kph acceleration scenario subject to passenger comfort constraints, i.e., jerk constraints.

First, to obtain a suitable steady-state initial condition for the MSHP, we place the model in closed-loop with a pair of proportional-integral (PI) controllers that use the MG1 and MG2 torque commands to track the constant engine speed command and vehicle speed at 40 kph. The engine operating condition is determined based on fuel economy considerations, and the automatic transmission is in the first gear. When the system reaches steady state, values for all model states and inputs are recorded to provide initial conditions for the optimizer. Note that the PI controllers are only used here to acquire the initial conditions. Later, during the optimization and validation simulation, the MSHP model is run open-loop.

Referencing (6.49), the OCP for the minimum time 40 to 70 kph acceleration scenario can be

written explicitly as

$$\min_{\Delta t, X, U, w} \sum_{j=1}^{M} \left[\lambda_1 \Delta t_j + \lambda_2 (\Delta t_j - \Delta t_s)^2\right] + \sum_{j=1}^{M} \sum_{i=0}^{N-1} \left[\lambda_3 (u_{i+1}^j - u_i^j)^T (u_{i+1}^j - u_i^j) + \lambda_4 u_i^{jT} u_i^j\right]$$

$$\begin{aligned}
\text{s.t.} \quad & \forall j \in \{1, \cdots, M\}, \\
& x_{i+1}^j = f_j(x_i^j, u_i^j, \Delta t_j) \ \ \forall i \in \{0, \cdots, N-1\}, \\
& c(x_i^j, u_i^j) \leq 0 \ \ \forall i \in \{0, \cdots, N\}, \\
& \Delta t_s \leq \Delta t_j \leq \Delta t^{\max}, \\
& x_0^1 = x_I, \ x_N^M = x_f, \\
& x_0^j = x_N^{j-1} \ \ \forall j \in \{2, \cdots, M\}, \\
& (v_0, \cdots, v_M) = w \in \mathcal{V},
\end{aligned} \tag{6.53}$$

where $\lambda_\mathfrak{p}, \mathfrak{p} = 1, 2, 3, 4$, are weights to penalize the acceleration duration (λ_1), deviation from the minimum duration (λ_2), change rate of torque commands (λ_3), and excessively high torque commands (λ_4). We set $\lambda_1 = 10^4$ for this time minimization problem, and we found it helpful to stabilize the solver by adding small penalties to the other terms. We found that $\lambda_2 = 10^{-4}$, $\lambda_3 = 10^{-4}$, and $\lambda_4 = 10^{-6}$ works well in practice.

The OCP uses a steady-state initial condition and enforces a terminal equality constraint on the final speed. It includes pointwise in time inequality constraints on the model inputs, states, and outputs (including jerk), given by

$$c(x, u) = \begin{bmatrix}
u_{mg1} - T_{mg1_{\max}} \\
u_{mg2} - T_{mg2_{\max}} \\
u_{eng} - T_{eng_{\max}} \\
T_{mg1_{\min}} - u_{mg1} \\
-u_{mg2} \\
T_{eng_{\min}} - u_{eng} \\
\omega_{eng} - \omega_{eng_{\max}} \\
\omega_{eng_{\min}} - \omega_{eng} \\
\omega_{mg1} T_{mg1} - P_{mg1_{\max}} \\
P_{mg1_{\min}} - \omega_{mg1} T_{mg1} \\
\omega_{mg2} T_{mg2} - P_{mg2_{\max}} \\
P_{mg2_{\min}} - \omega_{mg2} T_{mg2} \\
-s \\
\ddot{s} - \ddot{s}_{\max}
\end{bmatrix}, \tag{6.54}$$

where $\max$ and $\min$ denote the maximum and minimum limits respectively, and P denotes power.

There are only four possible gear ratios for this automatic transmission and skip-shifting is

allowed so we found three stages was sufficient (i.e., if we added a fourth stage the solution did not change). The final gear is left free, i.e., the optimizer is allowed to choose any $v_M \in v_f = \{1, 2, 3, 4\}$.

The OCP is solved using CASADI and IPOPT through the OPTI package. On a desktop with Intel Xeon Processor E3-1246 v3 @ 3.05GHz CPU using MATLAB R2019a and the parallel toolbox with 4 workers, we solve the multistage OCP in an average of 586 seconds when executing the case studies discussed in the next section.

6.5 Case Studies

To better understand the relationship between acceleration and smoothness, we solved (6.53) for several different values of the maximum jerk constraints, that is, for $\ddot{s}_{\max} = 20, 30, 40$ and $50\,\mathrm{m/s^3}$. Figure 6.3 plots the maneuver duration versus maximum jerk limit curve. There is a clear trade-off between these two variables: maneuver duration increases as the maximum jerk limit decreases (the duration for $\ddot{s}_{\max} = 50\,\mathrm{m/s^3}$ is slightly shorter than the duration for the case without jerk limit but are identical within numerical error). Next, we perform a more detailed analysis of the following three scenarios:

1. No jerk limit,
2. A jerk limit of $40\,\mathrm{m/s^3}$,
3. A jerk limit of $20\,\mathrm{m/s^3}$.

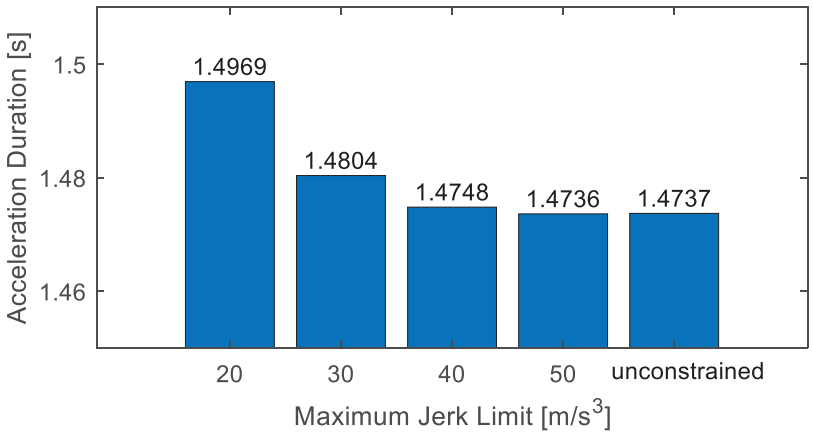

Figure 6.3: Trade-off between acceleration duration and maximum jerk limit.

6.5.1 Without Jerk Limits

Figure 6.4 shows the simulation results of the MSHP, given optimal inputs generated by the trajectory optimization procedure, when there is no constraint on the jerk. In this scenario, the vehicle accelerates from 40 kph to 70 kph in 1.4737 seconds and the maximum jerk is approximately $55\,\mathrm{m/s^3}$. The optimal gear number sequence is $(1, 1, 4)$, and the up shift happens at around 1.01 seconds.

As can be seen in Fig. 6.4, the jerk signal peaks at the beginning of the scenario and then decreases to around zero within 0.34 seconds. For the remainder of the scenario, the absolute value of jerk is below $5\,\mathrm{m/s^3}$. The jerk oscillates around zero because the engine, MG1, and MG2 are operating near their torque/power constraints. For example, at around 0.35 seconds, the MG2 torque reaches its limit and starts decreasing due to the power limit. Around the same time, jerk begins to increase. For another example, at around 0.85 second, the MG1 regenerative torque reaches its limit and starts increasing due to the power constraint. As a result, jerk slightly decreases before it slowly climbs back up to zero.

Figure 6.4 also shows that, in order to minimize the acceleration duration, the engine, MG1, and MG2 operate near their limits throughout the scenario. During the acceleration, both the engine and MG2 operate at their maximum torque and power limits. The actual outputs do not reach the commands instantaneously due to the component dynamics. MG1 outputs a positive torque at the beginning of the acceleration to speed up the engine rotational velocity, and later decreases to output a large negative torque so the engine power can be transmitted through the planetary gear set. Such behavior is consistent with other vehicles that use traditional powersplit configurations.

Instead of staying in the first gear through the whole acceleration duration, the optimal trajectory shifts into the fourth gear near the end of the scenario. We hypothesize that this is advantageous because it shifts the operating points of the engine, MG1, and MG2 away from the constraints, allowing them to increase their power outputs.

6.5.2 With Maximum Jerk Limits

Figures 6.5 and 6.6 show the simulation results when the jerk is set to be no more than $40\,\mathrm{m/s^3}$ and $20\,\mathrm{m/s^3}$, respectively. As shown in Fig. 6.5, the total maneuver duration increases to 1.4748 seconds, and the jerk constraint is respected. The MG2 torque command is lower and the MG1 power is higher at the beginning of the scenario than in the unconstrained scenario (in Fig. 6.4).

Similarly, in Fig. 6.6, the jerk constraint is satisfied by sacrificing the acceleration performance. The total maneuver duration further increases to 1.4969 second. At the beginning of the acceleration, the MG2 torque command is significantly lower compared with two other cases, and the MG1 power increases and then decreases more quickly.

Otherwise, the results in Fig. 6.5 and 6.6 are consistent with the ones of the no jerk limit scenario in Fig. 6.4. The engine operating conditions are similar in all three cases. Moreover, for the cases with jerk constraint, the optimizer still decides to perform a skip-shift directly to fourth gear around 1 second into the maneuver.

6.6 Concluding Remarks

In this chapter, we presented a trajectory optimization methodology for complex hybrid powertrains and used it to analyze a multistage hybrid powertrain through an acceleration duration versus jerk/smoothness tradeoff study. The simulation results show the developed MSHP model is able to capture the system behavior to enable preliminary study of the powertrain design. The results also indicate that the multistage trajectory optimization method can handle control inputs including both continuous and discrete variables, as well as jerk (or other nonlinear output) constraints.

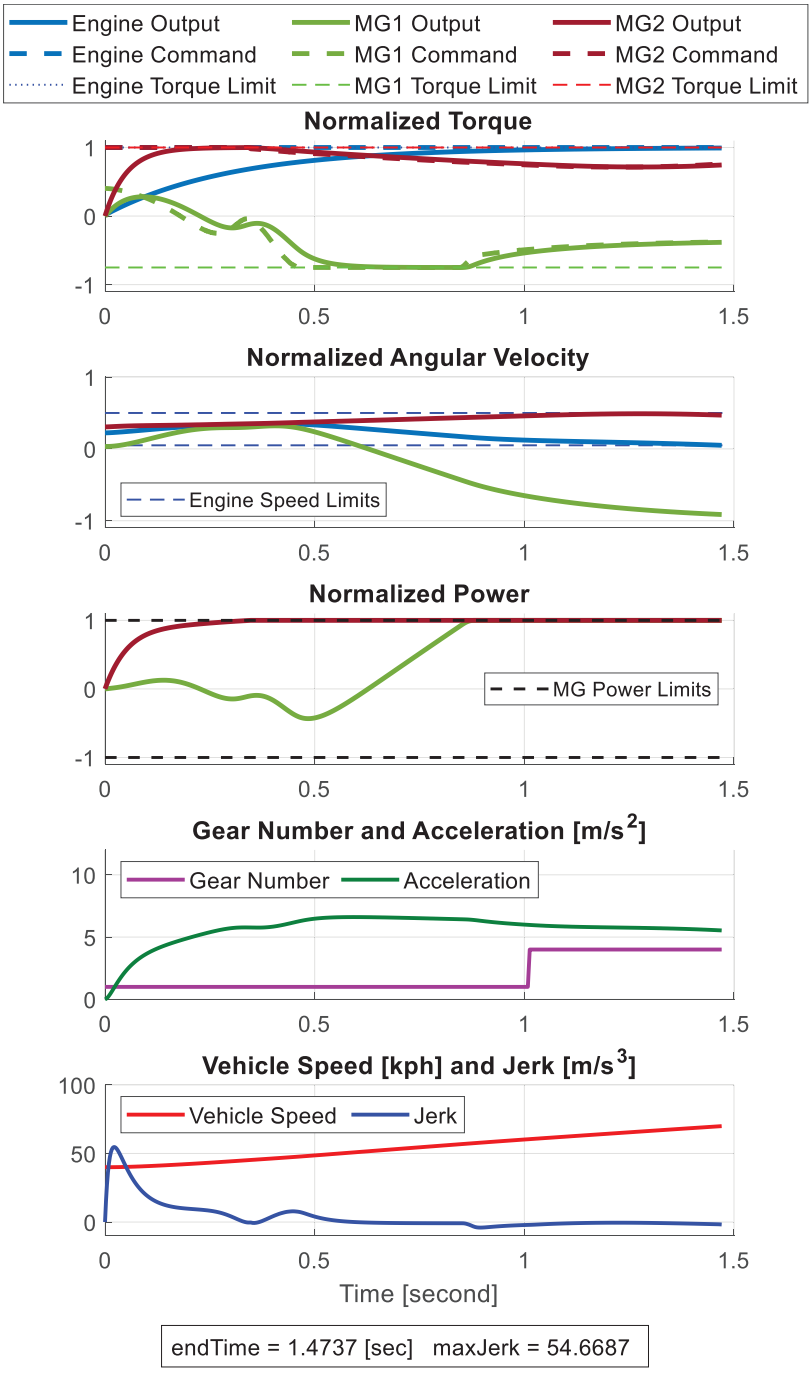

Figure 6.4: Simulation with optimal control inputs and without jerk constraint.

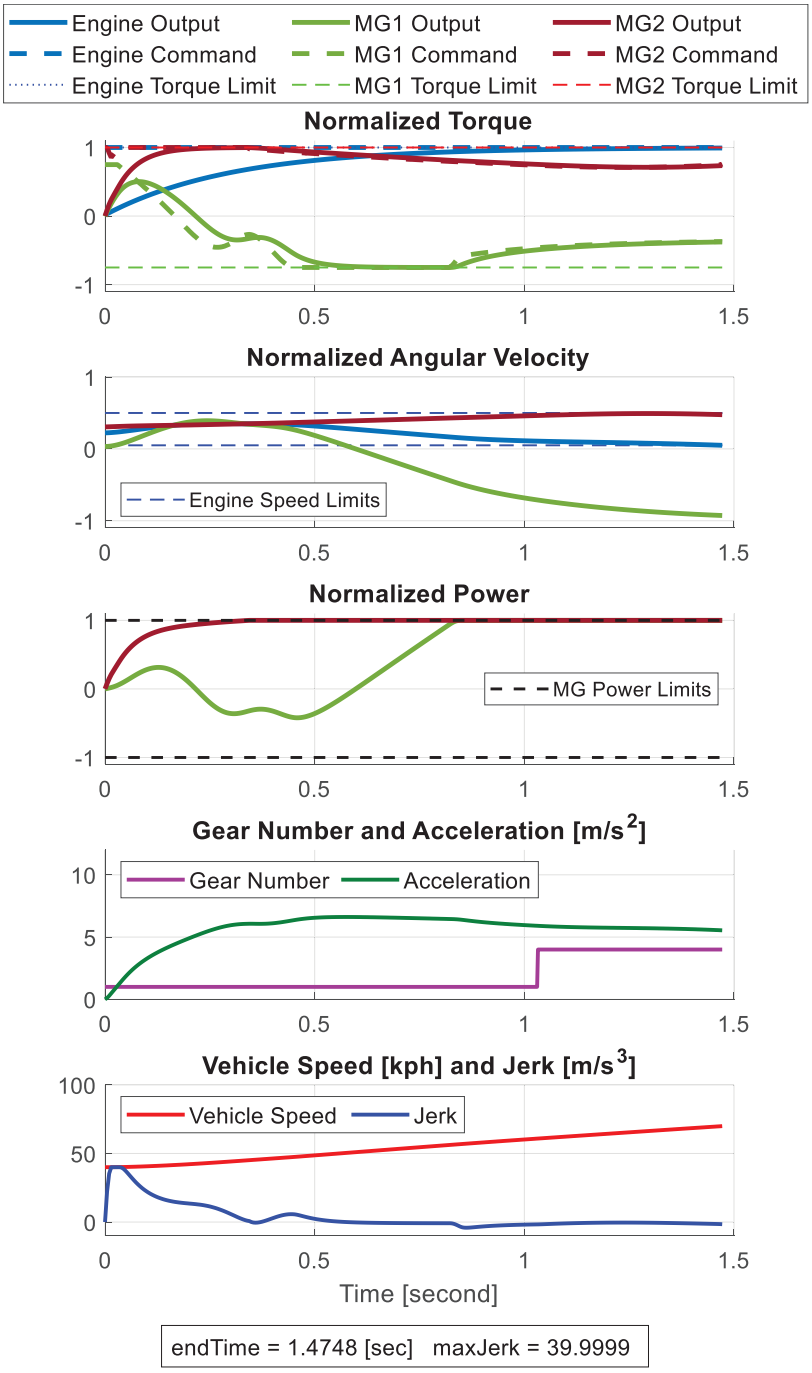

Figure 6.5: Simulation with optimal control inputs and with maximum jerk limit of $40\,\mathrm{m/s^3}$.

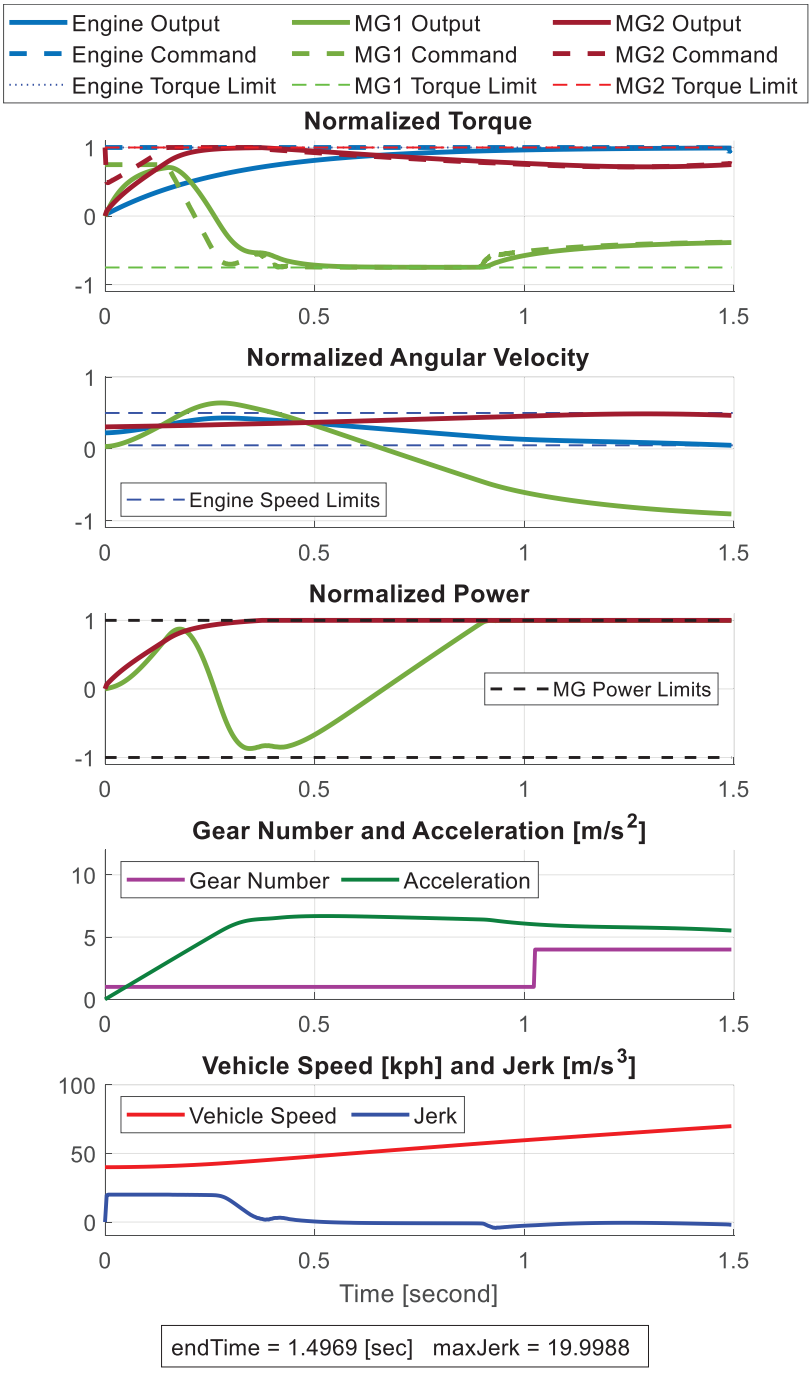

Figure 6.6: Simulation with optimal control inputs and with maximum jerk limit of $20\,\mathrm{m/s^3}$.

CHAPTER 7

Conclusions and Future Work

Autonomy, connected vehicles, electric powertrains and other trends are transforming mobility. Innovative solutions should improve equity and accessibility, while allowing safe, efficient and comfortable movement of people and goods. This dissertation advances decision and control theory for mobility applications. It leverages set theory, game theory, and trajectory optimization to transform the safety, efficiency, and comfort of future mobility systems.

In Chaps. 2, 3, and 4, we focus on developing fault-tolerant control (FTC) and Failure Mode and Effect Management (FMEM) strategies based on reference governor-centric set-theoretic approaches. Failed actuators that are zeroed-out are considered in Chap. 2, and stuck/jammed actuators are considered in Chaps. 3 and 4. Failure mode reconfiguration strategies are proposed in Chaps. 2 and 3, and Chap. 4 integrates the reconfiguration with fault detection and isolation (FDI) to construct a more systematic approach. The methods are demonstrated in simulation on a mass-spring-damper system (displacement control) and an aircraft longitudinal flight control system. The results show that, by imposing set-membership conditions that guarantee isolatability and reconfigurability, the systems are able to handle transitions to failure modes within a certain number of time steps and without violating safety constraints. The function of safe reference tracking by the reference governor is re-activated after the failure mode transition is finished. Possible future work in this direction for a more generic FMEM strategy includes the following:

- The type of failures explored in the current study mainly focused on failed actuators generating constant open-loop inputs. Other types of failures, such as sensor failures and communication failures (e.g., false-positive detection) are common for mobility applications and need to be investigated in the future. Additionally, failed signals in different forms need to be considered, such as for efficiency degradation and slowly changing signals with bounded rate.

- The impact on the system performance, such as the operating range and energy efficiency, should be evaluated if the normal operation must be restricted for guaranteed safety.

- Possible challenges when implementing the FMEM strategy in real-time need to be considered, e.g., to handle the situations when the isolation and reconfiguration sequences cannot be generated in time for execution.

- Situations with multiple simultaneous failures and failures that are not sequential should be addressed.

- Other mobility applications, such as unmanned aerial vehicles (UAVs) and connected and autonomous vehicles (CAVs) that often use system redundancy to achieve fault tolerance, should be considered.

In Chaps. 5, an energy-aware autonomous driving control policy is developed in a game-theoretic traffic environment that models the actions of human-driven vehicles. In addition, a battery electric vehicle (BEV) energy model is built and integrated into the traffic simulator to provide explicit information about energy consumption. The policy is trained by reinforcement learning with a reward function that balances the energy efficiency with other evaluation aspects including safety, time efficiency, and comfort. The ego vehicle is able to perform both longitudinal actions and lane changes. Simulation results indicate that the developed autonomous driving policy outperforms the other non-energy-aware benchmarking policies in energy saving and collision avoidance. The main observation is that, while the reward function for the policy training encourages driving at higher speed for time efficiency considerations, the ego vehicle using the energy-efficient policy tends to drive slower than the benchmarking vehicles to potentially reduce the energy losses due to rolling and aerodynamic resistance. In the future, we can consider extending the proposed methods in the directions below:

- Figure 7.1 shows an example of how the proposed energy-efficient control policy can be integrated into a typical autonomous driving architecture. The challenges in applying the proposed policy in actual vehicles need further investigation. For example, we need to properly design the module of safe tracking control for collision avoidance, whose decisions may affect the overall control performance including energy efficiency.

- Other traffic environments such as city driving with slow traffic should be studied in addition to the current highway traffic. Compared to highway driving where efficiencies of the powertrain components are relatively high, the advances for energy efficiency improvement by optimizing the decision of lane changes and braking may be more notable in traffic with frequent start-and-stops.

- The current autonomous driving policy is trained to optimize its own energy efficiency. The policy may make selfish decisions to save energy such as by changing lanes frequently to

avoid braking and/or driving at a lower average speed than the surrounding vehicles to reduce the resistance force. To achieve a more global energy saving benefit, co-optimization with other vehicles in traffic to develop friendly policies should be considered.

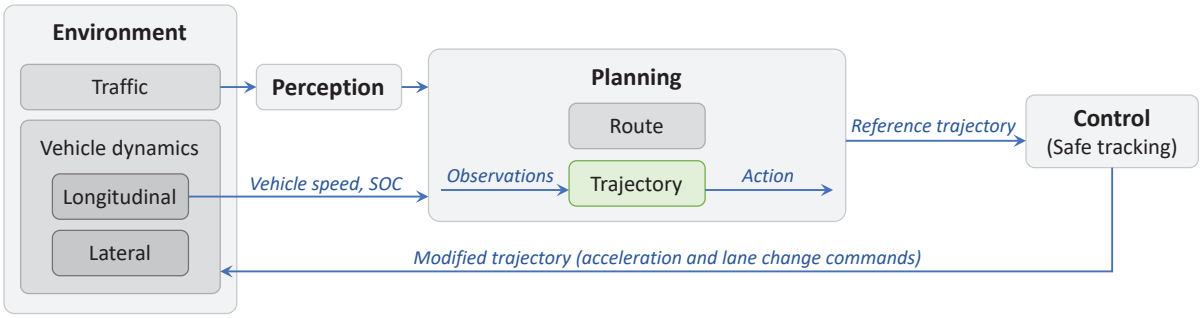

Figure 7.1: Integrating the energy-efficient autonomous driving control policy into a typical autonomous driving architecture. The proposed control policy maps the observations to actions for trajectory planning.

Finally, Chap. 6 takes a hybrid electric vehicle (HEV) powertrain that combines the power-split configuration with a four-speed automatic transmission as an example application. We develop a routine for the analysis of acceleration performance and comfort level measured by jerk. A case study is set up for the 40 to 70 kph acceleration scenario with different jerk limits to study the relations of minimum acceleration duration versus maximum jerk. A vehicle model is built, and we formulate a multistage mixed-integer trajectory optimization problem with the jerk constraint to solve for the minimum acceleration duration. A clear trade-off is seen in the simulation results, i.e., as the allowed jerk value reduces, it takes longer for the vehicle to complete the acceleration event. Moreover, the optimal time-based control profile is insightful. For example, it shows how the engine and motors cooperate to deliver the optimum power, and that the skip-shifting feature is beneficial to improve the acceleration performance. The following are areas of potential future work to be considered:

- As the common vehicle control involves using calibration tables for which the tuning is often labor-intensive, the proposed optimal control method may be integrated to assist and automate a part of the tuning process so that the powertrain control development efficiency can be enhanced.

- To further eliminate the labor effort for tuning and calibration, it is worth investigating potential optimal control methods to be implemented in real-time, e.g., model predictive control (MPC), though the major challenge is to ensure that they are computationally viable.

In this case, the current multistage control strategy can provide benchmarking references for future control development.

Overall, this dissertation highlights opportunities for applications of game-theoretic, set-theoretic and optimal control towards the development of safer, more efficient and comfortable future mobility systems. We also show the ability and benefits of integrating reference governor, reinforcement learning and vehicle propulsion system modeling into the proposed strategies. The contributions address the gaps in safety-guaranteed fault-tolerant control, energy-aware connected and autonomous driving control, and optimal-based vehicle drivability analysis processes.

BIBLIOGRAPHY

[1] J.-P. Rodrigue, *The geography of transport systems*. Routledge, 2020.

[2] J. McKay, "Transport or mobility: What's the difference and why does it matter?." https://www.forumforthefuture.org/blog/transport-or-mobility. Accessed: 2022-02-09.

[3] J. M. Anderson, K. Nidhi, K. D. Stanley, P. Sorensen, C. Samaras, and O. A. Oluwatola, *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, 2014.

[4] M. Bartels, "Russia says 'software failure' caused thruster misfire at space station." https://www.space.com/space-station-nauka-arrival-thruster-fire-update. Accessed: 2022-03-20.

[5] H. Williamson, *Air crash investigations: Jammed rudder kills 132, the crash of USAir Flight 427*. lulu.com, October 2011.

[6] R. Wang and J. Wang, "Fault-tolerant control with active fault diagnosis for four-wheel independently driven electric ground vehicles," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 9, pp. 4276–4287, 2011.

[7] U.S. Department of Transportation, "Average annual miles per driver by age group." https://www.fhwa.dot.gov/ohim/onh00/bar8.htm. Accessed: 2022-01-28.

[8] U.S. Energy Information Administration, "Use of energy explained: Energy use for transportation." https://www.eia.gov/energyexplained/use-of-energy/transportation-in-depth.php. Accessed: 2022-02-09.

[9] E. S. Islam, A. Moawad, N. Kim, and A. Rousseau, "Vehicle electrification impacts on energy consumption for different connected-autonomous vehicle scenario runs," *World Electric Vehicle Journal*, vol. 11, no. 1, p. 9, 2020.

[10] M. Taiebat, A. L. Brown, H. R. Safford, S. Qu, and M. Xu, "A review on energy, environmental, and sustainability implications of connected and automated vehicles," *Environmental science & technology*, vol. 52, no. 20, pp. 11449–11465, 2018.

[11] A. Vahidi and A. Sciarretta, "Energy saving potentials of connected and automated vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 822–843, 2018.

[12] A. Scamarcio, P. Gruber, S. De Pinto, and A. Sorniotti, "Anti-jerk controllers for automotive applications: A review," *Annual Reviews in Control*, vol. 50, pp. 174–189, 2020.

[13] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder, *Diagnosis and Fault-Tolerant Control*, vol. 2. Springer, 2006.

[14] I. Hwang, S. Kim, Y. Kim, and C. E. Seah, "A survey of fault detection, isolation, and reconfiguration methods," *IEEE transactions on control systems technology*, vol. 18, no. 3, pp. 636–653, 2009.

[15] W. Chen and J. Jiang, "Fault-tolerant control against stuck actuator faults," *IEE Proceedings-Control Theory and Applications*, vol. 152, no. 2, pp. 138–146, 2005.

[16] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, 2017.

[17] F. Blanchini and S. Miani, *Set-Theoretic Methods in Control*. Springer, 2008.

[18] C. Danielson, K. Berntorp, A. Weiss, and S. D. Cairano, "Robust motion planning for uncertain systems with disturbances using the invariant-set motion planner," *IEEE Transactions on Automatic Control*, vol. 65, no. 10, pp. 4456–4463, 2020.

[19] F. Blanchini, F. A. Pellegrino, and L. Visentini, "Control of manipulators in a constrained workspace by means of linked invariant sets," *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, vol. 14, no. 13-14, pp. 1185–1205, 2004.

[20] W. Lucia, D. Famularo, and G. Franze, "A set-theoretic reconfiguration feedback control scheme against simultaneous stuck actuators," *IEEE Transactions on Automatic Control*, vol. 63, no. 8, pp. 2558–2565, 2017.

[21] K. McDonough and I. Kolmanovsky, "Fast computable recoverable sets and their use for aircraft loss-of-control handling," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 4, pp. 934–947, 2017.

[22] A. Yetendje, M. M. Seron, and J. A. De Doná, "Robust MPC multicontroller design for actuator fault tolerance of constrained systems," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 4678–4683, 2011.

[23] D. M. Raimondo, G. R. Marseglia, R. D. Braatz, and J. K. Scott, "Fault-tolerant model predictive control with active fault isolation," in *2013 Conference on Control and Fault-Tolerant Systems (SysTol)*, pp. 444–449, IEEE, 2013.

[24] A. Abid, M. T. Khan, and J. Iqbal, "A review on fault detection and diagnosis techniques: basics and beyond," *Artificial Intelligence Review*, vol. 54, no. 5, pp. 3639–3664, 2021.

[25] H. Li, I. Kolmanovsky, and A. Girard, "A failure mode reconfiguration strategy based on constraint admissible and recoverable sets," in *2021 American Control Conference (ACC)*, pp. 4771–4776, IEEE, 2021.

[26] H. Li, I. Kolmanovsky, and A. Girard, "Set-theoretic failure mode reconfiguration for stuck actuators," *IEEE Control Systems Letters*, vol. 6, pp. 1316–1321, 2021.

[27] H. Li, I. Kolmanovsky, and A. Girard, "Integrating failure detection and isolation into a reference governor-based reconfiguration strategy for stuck actuators," in *Proceedings of 2022 American Control Conference (ACC)*, to appear.

[28] U.S. Department of Energy, "NEXTCAR 2020 annual program review." https://arpa-e.energy.gov/sites/default/files/0_NEXTCAR_AR_2020_final.pdf. Accessed: 2022-02-09.

[29] T. Litman, "Autonomous vehicle implementation predictions: Implications for transport planning," tech. rep., Victoria Transport Policy Institute, 2020.

[30] K. McDonough, I. Kolmanovsky, D. Filev, S. Szwabowski, D. Yanakiev, and J. Michelini, "Stochastic fuel efficient optimal control of vehicle speed," in *Optimization and Optimal Control in Automotive Systems*, pp. 147–162, Springer, 2014.

[31] U. E. P. Agency, "Dynamometer drive schedules." https://www.epa.gov/vehicle-and-fuel-emissions-testing/dynamometer-drive-schedules. Accessed: 2022-02-09.

[32] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. R. Girard, "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1782–1797, 2017.

[33] Y. Yildiz, A. Agogino, and G. Brat, "Predicting pilot behavior in medium-scale scenarios using game theory and reinforcement learning," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 4, pp. 1335–1343, 2014.

[34] T. Jaakkola, S. P. Singh, and M. I. Jordan, "Reinforcement learning algorithm for partially observable markov decision problems," in *Advances in Neural Information Processing Systems*, pp. 345–352, 1995.

[35] H. Li, N. Li, I. Kolmanovsky, and A. Girard, "Energy-efficient autonomous vehicle control using reinforcement learning and interactive traffic simulations," in *2020 American Control Conference (ACC)*, pp. 3029–3034, IEEE, 2020.

[36] H. Li, N. Li, I. Kolmanovsky, and A. Girard, "Energy-efficient autonomous driving using cognitive driver behavioral models and reinforcement learning," in *Proceedings of AI-enabled technologies for autonomous and connected vehicles*, Springer, 2022.

[37] Toyota, "Advancement of Toyota Hybrid System II (THS-II)." https://global.toyota/en/powertrain/ths/. Accessed: 2022-1-15.

[38] I. V. Kolmanovsky and A. G. Stefanopoulou, "Optimal control techniques for assessing feasibility and defining subsystem level requirements: An automotive case study," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 3, pp. 524–534, 2001.

[39] H. Waschl, I. Kolmanovsky, M. Steinbuch, and L. Del Re, *Optimization and optimal control in automotive systems*, vol. 455. Springer, 2014.

[40] L. Guzzella, A. Sciarretta, *et al.*, *Vehicle Propulsion Systems*, vol. 1. Springer, 2007.

[41] H. Li, D. Liao-McPherson, I. Kolmanovsky, S. Kim, and K. Butts, "Analysis of multistage hybrid powertrains using multistage mixed-integer trajectory optimization," tech. rep., SAE Technical Paper, 2020.

[42] A. Weiss, C. Petersen, M. Baldwin, R. S. Erwin, and I. Kolmanovsky, "Safe positively invariant sets for spacecraft obstacle avoidance," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 4, pp. 720–732, 2015.

[43] K. Berntorp, A. Weiss, C. Danielson, I. Kolmanovsky, and S. Di Cairano, "Automated driving: safe motion planning using positively invariant sets," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2017.

[44] I. Kolmanovsky, A. Weiss, and W. Merrill, "Incorporating risk into control design for emergency operation of turbo-fan engines," in *Infotech@ Aerospace 2011*, p. 1591, AIAA, 2011.

[45] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Mathematical Problems in Engineering*, vol. 4, pp. 317–367, 1998.

[46] A. Löhne and B. Weißing, "The vector linear program solver bensolve–notes on theoretical background," *European Journal of Operational Research*, vol. 260, no. 3, pp. 807–813, 2017.

[47] A. Girard and I. Kolmanovsky, *Lecture Notes on Control of Aerospace Vehicles*. Department of Aerospace Engineering, The University of Michigan, Ann Arbor, January 2019.

[48] M. Hosseinzadeh, I. Kolmanovsky, S. Baruah, and B. Sinopoli, "Reference governor-based fault-tolerant constrained control," *arXiv preprint arXiv:2107.08457*, 2021.

[49] A. Abbaspour, S. Mokhtari, A. Sargolzaei, and K. K. Yen, "A survey on active fault-tolerant control systems," *Electronics*, vol. 9, no. 9, p. 1513, 2020.

[50] H. P. Williams, *Model building in mathematical programming*. John Wiley & Sons, 2013.

[51] J. Guanetti, Y. Kim, and F. Borrelli, "Control of connected and automated vehicles: State of the art and future challenges," *Annual Reviews in Control*, vol. 45, pp. 18–40, 2018.

[52] H. Waschl, I. Kolmanovsky, and F. Willems, *Control Strategies for Advanced Driver Assistance Systems and Autonomous Driving Functions*. Springer, 2019.

[53] A. Sciarretta, A. Vahidi, *et al.*, *Energy-Efficient Driving of Road Vehicles*. Springer, 2020.

[54] U.S. Department of Energy, "Next-generation energy technologies for connected and automated on-road vehicles." https://arpa-e.energy.gov/technologies/programs/nextcar. Accessed: 2021-07-15.

[55] M. R. Amini, I. Kolmanovsky, and J. Sun, "Hierarchical MPC for robust eco-cooling of connected and automated vehicles and its application to electric vehicle battery thermal management," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 1, pp. 316–328, 2020.

[56] H. Wang, M. R. Amini, Z. Song, J. Sun, and I. Kolmanovsky, "Combined energy and comfort optimization of air conditioning system in connected and automated vehicles," in *Dynamic Systems and Control Conference*, vol. 59148, p. V001T08A001, American Society of Mechanical Engineers, 2019.

[57] N. Doshi, D. Hanover, S. Hemmati, C. Morgan, and M. Shahbakhti, "Modeling of thermal dynamics of a connected hybrid electric vehicle for integrated HVAC and powertrain optimal operation," in *Dynamic Systems and Control Conference*, vol. 59155, p. V002T23A005, American Society of Mechanical Engineers, 2019.

[58] X. Meng and C. G. Cassandras, "Eco-driving of autonomous vehicles for nonstop crossing of signalized intersections," *IEEE Transactions on Automation Science and Engineering*, 2020.

[59] A. Houshmand, C. G. Cassandras, N. Zhou, N. Hashemi, B. Li, and H. Peng, "Combined eco-routing and power-train control of plug-in hybrid electric vehicles in transportation networks," *arXiv preprint arXiv:2004.05161*, 2020.

[60] S. D. Kumaravel, A. A. Malikopoulos, and R. Ayyagari, "Decentralized cooperative merging of platoons of connected and automated vehicles at highway on-ramps," *arXiv preprint arXiv:2002.04826*, 2020.

[61] S. Aoki, L. E. Jan, J. Zhao, A. Bhat, C.-F. Chang, *et al.*, "Multicruise: Eco-lane selection strategy with eco-cruise control for connected and automated vehicles," *arXiv preprint arXiv:2104.11959*, 2021.

[62] R. Chen, C. G. Cassandras, A. Tahmasbi-Sarvestani, S. Saigusa, H. N. Mahjoub, and Y. K. Al-Nadawi, "Cooperative time and energy-optimal lane change maneuvers for connected automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[63] A. I. Mahbub, A. A. Malikopoulos, and L. Zhao, "Decentralized optimal coordination of connected and automated vehicles for multiple traffic scenarios," *Automatica*, vol. 117, p. 108958, 2020.

[64] N. Li, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. Girard, "Game theory-based traffic modeling for calibration of automated driving algorithms," in *Control Strategies for Advanced Driver Assistance Systems and Autonomous Driving Functions*, pp. 89–106, Springer, 2019.

[65] S. Karimi and A. Vahidi, "Receding horizon motion planning for automated lane change and merge using Monte Carlo tree search and level-k game theory," in *2020 American Control Conference (ACC)*, pp. 1223–1228, IEEE, 2020.

[66] R. Tian, N. Li, I. Kolmanovsky, Y. Yildiz, and A. R. Girard, "Game-theoretic modeling of traffic in unsignalized intersection network for autonomous vehicle control verification and validation," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[67] A. V. Rajendran, B. Hegde, Q. Ahmed, and G. Rizzoni, "Design and development of traffic-in-loop powertrain simulation," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 261–266, IEEE, 2017.

[68] S. Qiu, L. Qiu, L. Qian, Z. Abdollahi, and P. Pisu, "Closed-loop hierarchical control strategies for connected and autonomous hybrid electric vehicles with random errors," *IET Intelligent Transport Systems*, vol. 12, no. 10, pp. 1378–1385, 2018.

[69] H. D. Gamage and J. B. Lee, "Reinforcement learning based driving speed control for two vehicle scenario," in *Australasian Transport Research Forum (ATRF), 39th, 2017, Auckland, New Zealand*, 2017.

[70] H. Li, K. Butts, K. Zaseck, D. Liao-McPherson, and I. Kolmanovsky, "Emissions modeling of a light-duty diesel engine for model-based control design using multi-layer perceptron neural networks," tech. rep., SAE Technical Paper, 2017.

[71] B. Xu, X. Hu, X. Tang, X. Lin, H. Li, D. Rathod, and Z. Filipi, "Ensemble reinforcement learning-based supervisory control of hybrid electric vehicle for fuel economy improvement," *IEEE Transactions on Transportation Electrification*, vol. 6, no. 2, pp. 717–727, 2020.

[72] N. Li, K. Han, I. Kolmanovsky, and A. Girard, "Coordinated receding-horizon control of battery electric vehicle speed and gearshift using relaxed mixed integer nonlinear programming," *arXiv preprint arXiv:2102.09014*, 2021.

[73] C. Liu and Y. L. Murphey, "Power management for plug-in hybrid electric vehicles using reinforcement learning with trip information," in *2014 IEEE Transportation Electrification Conference and Expo (ITEC)*, pp. 1–6, IEEE, 2014.

[74] Y. L. Murphey, J. Park, Z. Chen, M. L. Kuang, M. A. Masrur, and A. M. Phillips, "Intelligent hybrid vehicle power control—Part I: Machine learning of optimal vehicle power," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 8, pp. 3519–3530, 2012.

[75] X. Hu, T. Liu, X. Qi, and M. Barth, "Reinforcement learning for hybrid and plug-in hybrid electric vehicle energy management: Recent advances and prospects," *IEEE Industrial Electronics Magazine*, vol. 13, no. 3, pp. 16–25, 2019.

[76] L. Feldkamp, M. Abou-Nasr, and I. V. Kolmanovsky, "Recurrent neural network training for energy management of a mild hybrid electric vehicle with an ultra-capacitor," in *2009 IEEE Workshop on Computational Intelligence in Vehicles and Vehicular Systems*, pp. 29–36, IEEE, 2009.

[77] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 712–733, 2020.

[78] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[79] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[80] G. Delnevo, P. Di Lena, S. Mirri, C. Prandi, and P. Salomoni, "On combining big data and machine learning to support eco-driving behaviours," *Journal of Big Data*, vol. 6, no. 1, p. 64, 2019.

[81] G. Wu, F. Ye, P. Hao, D. Esaid, K. Boriboonsomsin, M. J. Barth, *et al.*, "Deep learning–based eco-driving system for battery electric vehicles," tech. rep., Institute of Transportation Studies, UC Davis, 2019.

[82] G. Zheng and Z. Peng, "Life cycle assessment (LCA) of BEV's environmental benefits for meeting the challenge of ICExit (internal combustion engine exit)," *Energy Reports*, vol. 7, pp. 1203–1216, 2021.

[83] N. Li, Y. Yao, I. Kolmanovsky, E. Atkins, and A. R. Girard, "Game-theoretic modeling of multi-vehicle interactions at uncontrolled intersections," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[84] M. A. S. Kamal, S. Taguchi, and T. Yoshimura, "Efficient vehicle driving on multi-lane roads using model predictive control under a connected vehicle environment," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 736–741, IEEE, 2015.

[85] G. Schildbach and F. Borrelli, "Scenario model predictive control for lane change assistance on highways," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 611–616, IEEE, 2015.

[86] T. Awal, M. Murshed, and M. Ali, "An efficient cooperative lane-changing algorithm for sensor-and communication-enabled automated vehicles," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1328–1333, IEEE, 2015.

[87] S. Di Cairano, W. Liang, I. V. Kolmanovsky, M. L. Kuang, and A. M. Phillips, "Power smoothing energy management and its application to a series hybrid powertrain," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 6, pp. 2091–2103, 2012.

[88] L. Claussmann, A. Carvalho, and G. Schildbach, "A path planner for autonomous driving on highways using a human mimicry approach with binary decision diagrams," in *2015 European Control Conference (ECC)*, pp. 2976–2982, IEEE, 2015.

[89] J. H. Yoo and R. Langari, "Stackelberg game based model of highway driving," in *ASME 2012 5th Annual Dynamic Systems and Control Conference joint with the JSME 2012 11th Motion and Vibration Conference*, pp. 499–508, American Society of Mechanical Engineers Digital Collection, 2013.

[90] J. H. Yoo and R. Langari, "A stackelberg game theoretic driver model for merging," in *ASME 2013 Dynamic Systems and Control Conference*, American Society of Mechanical Engineers Digital Collection, 2014.

[91] K. B. Wipke, M. R. Cuddy, and S. D. Burch, "ADVISOR 2.1: A user-friendly advanced powertrain simulation using a combined backward/forward approach," *IEEE Transactions on Vehicular Technology*, vol. 48, no. 6, pp. 1751–1761, 1999.

[92] G. Mohan, F. Assadian, and S. Longo, "Comparative analysis of forward-facing models vs backwardfacing models in powertrain component sizing," in *IET Hybrid and Electric Vehicles Conference 2013 (HEVC 2013)*, pp. 1–6, IET, 2013.

[93] N. Li, *Game-Theoretic and Set-Based Methods for Safe Autonomous Vehicles on Shared Roads*. PhD thesis, University of Michigan, 2021.

[94] M. A. Patterson and A. V. Rao, "GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Transactions on Mathematical Software (TOMS)*, vol. 41, no. 1, pp. 1–37, 2014.

[95] J. Koenemann, G. Licitra, M. Alp, and M. Diehl, "Openocl–open optimal control library," 2017.

[96] R. Tedrake *et al.*, "Drake: Model-based design and verification for robotics," 2019.

[97] V. M. Becerra, "Solving complex optimal control problems at no cost with PSOPT," in *2010 IEEE International Symposium on Computer-Aided Control System Design*, pp. 1391–1396, IEEE, 2010.

[98] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[99] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM review*, vol. 47, no. 1, pp. 99–131, 2005.

[100] F. Zhu and P. J. Antsaklis, "Optimal control of hybrid switched systems: A brief survey," *Discrete Event Dynamic Systems*, vol. 25, no. 3, pp. 345–364, 2015.

[101] L. Y. Wang, A. Beydoun, J. Cook, J. Sun, and I. Kolmanovsky, "Optimal hybrid control with applications to automotive powertrain systems," in *Control using logic-based switching*, pp. 190–200, Springer, 1997.

[102] X. Xu and P. J. Antsaklis, "Optimal control of switched systems: new results and open problems," in *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334)*, vol. 4, pp. 2683–2687, IEEE, 2000.

[103] X. Xu and P. J. Antsaklis, "Optimal control of switched systems based on parameterization of the switching instants," *IEEE transactions on automatic control*, vol. 49, no. 1, pp. 2–16, 2004.

[104] T. M. Caldwell and T. D. Murphey, "Switching mode generation and optimal estimation with application to skid-steering," *Automatica*, vol. 47, no. 1, pp. 50–64, 2011.

[105] H. Axelsson, Y. Wardi, M. Egerstedt, and E. Verriest, "Gradient descent approach to optimal mode scheduling in hybrid dynamical systems," *Journal of Optimization Theory and Applications*, vol. 136, no. 2, pp. 167–186, 2008.

[106] H. Gonzalez, R. Vasudevan, M. Kamgarpour, S. S. Sastry, R. Bajcsy, and C. J. Tomlin, "A descent algorithm for the optimal control of constrained nonlinear switched dynamical systems," in *Proceedings of the 13th ACM international conference on Hybrid systems: computation and control*, pp. 51–60, 2010.

[107] M. Alamir and S.-A. Attia, "On solving optimal control problems for switched hybrid nonlinear systems by strong variations algorithms," in *Proceedings of 6th IFAC symposium on nonlinear control systems*, pp. 558–563, 2004.

[108] S. Solmaz, R. Shorten, K. Wulff, and F. O. Cairbre, "A design methodology for switched discrete time linear systems with applications to automotive roll dynamics control," *Automatica*, vol. 44, no. 9, pp. 2358–2363, 2008.

[109] K. Van Berkel, T. Hofman, B. Vroemen, and M. Steinbuch, "Optimal control of a mechanical hybrid powertrain," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 2, pp. 485–497, 2011.

[110] R. Amari, M. Alamir, and P. Tona, "Unified MPC strategy for idle-speed control, vehicle start-up and gearing applied to an automated manual transmission," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 7079–7085, 2008.

[111] R. Amari, P. Tona, and M. Alamir, "Experimental evaluation of a hybrid mpc strategy for vehicle start-up with an automated manual transmission," in *2009 European Control Conference (ECC)*, pp. 4271–4277, IEEE, 2009.

[112] J. Andersson, J. Åkesson, and M. Diehl, "Casadi: A symbolic package for automatic differentiation and optimal control," in *Recent advances in algorithmic differentiation*, pp. 297–307, Springer, 2012.