Development of a distributed model predictive controller for over-actuated autonomous vehicle path tracking

by

Ted Ecclestone

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2023

Author's declaration

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

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

Abstract

Widespread interest in the advancement of autonomous vehicle technology is motivated by multiple outstanding issues associated with vehicular travel despite the decades-long ubiquity of this mode of transportation. It is well known that the leading cause of accidents on the road is human error. Furthermore, vehicle hardware faults and harsh environmental circumstances are also common collision factors due to the challenges that they introduce to the driving task. Autonomous vehicles have the potential to greatly exceed the perception, decision making, and control capabilities of human drivers in some applications, and the large-scale adoption of this technology will thereby mitigate the primary driving-related safety concerns. Numerous additional benefits will be realized as a result; for instance, complex planning algorithms will help to reduce traffic congestion, and transportation- and insurance-related costs will be minimized due to the lower collision rates. Though it may be many years before the technology sees extensive use for passenger transportation applications due to the complexity of standard driving environments, autonomous vehicles will likely find use over the short-term in other specialized domains. For example, these vehicles can be used to transport payloads over short distances in a wide variety of applications, including agriculture, mining, and shipping, where the operating environment is less complex. In these scenarios, autonomous vehicle technology will help to lessen the effects of labour shortages while enabling longer operating hours at a lower cost.

A key component of the autonomous stack is the motion controller, which serves to regulate the longitudinal and lateral motion of the vehicle according to a defined set of objectives by precisely manipulating the available actuators. Model predictive control (MPC) is a powerful control strategy commonly used for this purpose; the algorithm can coordinate a large set of control inputs such that the system meets all defined objectives while satisfying any constraints on the states and inputs. Many prior works investigate the use of MPC, and its variants, for vehicle path tracking and stability control applications. One such variant is distributed MPC; with this approach, the controlled plant is modelled as a set of interacting subsystems, each subsystem using its own MPC controller to select a set of optimal control actions in combination with all others. An extension of distributed MPC, agent-based MPC (AMPC), enhances the control capabilities by allowing the controller to additionally consider both the effect of subsystems that are not controllable by the optimal controller and the effects of hardware faults on the system dynamics. While previous works have investigated the application of AMPC to vehicle stability control tasks, in this thesis, AMPC is utilized to perform path tracking.

The vehicle hardware platform considered in this work, WATonoTruck, is modular

and over-actuated in design, making it a suitable test platform for AMPC. Built using the corner module platform, the wheels at each corner can be independently driven and steered. A vehicle dynamics reference model to represent the behaviour of WATonoTruck is constructed; this model utilizes a nonlinear tire force model to accurately characterize the tire-road interaction, and incorporates Ackermann geometry to prevent unecessary wheel slip and reduce the control task complexity that results from the over-actuated nature of the system. This model serves as the prediction model for the designed AMPC controller. The controller also considers numerous constraints on the vehicle states, inputs, and input rates to ensure stability, and can incorporate an external longitudinal controller and account for actuator faults. The controller is validated over several simulated and experimental tests that demonstrate its ability to provide effective path tracking and velocity control performance in a varied set of scenarios, including those where actuator failures occur or the driving environment is harsh.

Acknowledgements

The successful completion of this project and of all others that I have been involved with throughout my master's program would not have been possible without the guidance and support that I received from many people. First of all, I would like to express my deepest thanks for my supervisor, Professor Amir Khajepour, for all of the insights, suggestions, mentorship, and encouragement that he has provided to me throughout my time at the University of Waterloo. It has been a privilege to work as a member of the MVS Lab and I am grateful for the opportunity to have been involved with many of the projects conducted by the group.

Furthermore, I am also thankful for the members of our lab that I have had the pleasure of working with over the last few years. Thank you to Ben MacCallum, in particular; I have learned a lot from our close collaboration while working on many projects together. Thank you as well to Neel Bhatt, Ruihe Zhang, Pouya Panahandeh, Ehsan Mohammadbagher, Ahmad Alghooneh, Minghao Ning, and Yaodong Cui for your support throughout our time working together; also, to Neel, along with Steven Tuer and Jin Park, for their help in preparing the WATonoTruck for experimental testing. Additionally, I would like to thank the MVS Lab technicians and associates Aaron Sherratt, Jeff Graansma, Mike Duthie, Adrian Neill, and Bek Dakibay for supporting all experimental testing that I conducted in the lab, as well as Chao Yu, Dr. Amin Habibnejad Korayem, Dr. Chen Tang, Dr. Mehdi Abroshan, Dr. Amir Soltani, Yukun Lu, Dr. Reza Valiollahi Mehrizi, Dr. Mohammad Pirani, and Dr. Ahmad Mozaffari for the technical guidance that they have provided.

I am also grateful to Dr. Bakhtiar Litkouhi, Dr. Gary Zhang, Dr. Alireza Kasaiezadeh, Dr. Annie Zhao, and the rest of the ADVCS team at General Motors Global R&D for the opportunity to intern with their group this past summer; it was an exciting opportunity to apply what I had learned at Waterloo to a project in industry, and I learned a lot as a member of their team.

Finally, I am forever appreciative of my family, friends, and my partner Gab for their endless love, kindness, and generosity. Thank you for everything that you do for me.

Dedication

Thank you to my family and Gab, for always being there to support and inspire me.

Table of contents

List of tables

List of figures

Chapter 1

Introduction

1.1 Motivation

Despite the significant worldwide reliance on vehicles to meet daily transportation needs, there are many outstanding issues with the current state of car-based travel. Between 2005 and 2007, the U.S. Department of Transportation conducted a study to determine the leading causes of passenger vehicle accidents that involved collecting evidence at numerous collision scenes around the country [1]. The study's findings indicated that approximately 94% of crashes were the result of driver error, 2% were related to a vehicle hardware failure, 2% were the result of hazardous environmental conditions, and the remaining 2% were of unknown cause. The U.S. Department of Transportation further reported that motor vehicle crashes were a leading cause of death, with over 37,000 lives lost in collisions throughout the United States in 2017 [2]. In addition to the safety concerns associated with driving, issues related to traffic congestion, transportation and insurance-related costs, and the vast amount of space occupied by infrastructure built to support vehicular travel hamper the experience of vehicle users and society at large.

The widespread adoption of autonomous vehicles (AVs) is expected to both resolve these outstanding challenges and provide additional benefits for society. AV technology has the potential to exceed the perception, prediction, decision making, planning, and control capabilities of human drivers, which will significantly reduce the possibility of human error. The use of model-based and data-driven control algorithms will enable the vehicle to respond effectively to harsh conditions, such as wet or snowy road surfaces, or a vehicle malfunction, such as a steering or electric motor failure, by utilizing a predictive model that characterizes the system's behaviour in such scenarios. The resulting

safety improvements will allow for faster vehicle speeds, shortening travel times; provide insurance and medical savings due to minimized collision rates; and reduce the area that vehicles require to safely travel and park, freeing space for new bicycle lanes and pedestrian walkways [3]. Furthermore, AV technology can provide value in domains beyond human passenger transportation. AVs may be used to transport payloads for a wide variety of applications, including agriculture, mining, and shipping. AVs operating in these spaces will alleviate the impact of later shortages, provide cost savings, and enable longer operational hours. In comparison to urban environments, the relative simplicity of the driving task in these scenarios will likely result in faster adoption of AV technology and the corresponding realization of its benefits.

A popular and successful method to develop a safe, reliable, and understandable AV architecture involves the integration of a set of modules that each undertake a unique aspect of the driving task and operate concurrently to govern the behaviour of the vehicle. These modules include perception, which utilizes measurements collected using sensors such as LIDAR, RADAR, and cameras to determine the position and orientation of obstacles in the vehicle's environment; prediction, which considers the historical evolution of obstacle states to predict their trajectory over a finite period of time into the future; decision making, which considers the state of the vehicle and the environment to determine high-level desired behaviour, such as stopping at a stop sign; motion planning, which plans a local path that is safe and feasible for the vehicle to follow; and the controller, which manipulates the vehicles actuators such that the vehicle perform safe maneuvers by following the planned trajectory while maintaining stability. A successful AV motion controller must achieve a set of path following and velocity regulation control objectives while adhering to a set of constraints imposed to ensure stability and safety, considering vehicle dynamics to ensure maneuver feasibility, and coordinating the manipulation of the available actuators. The control task is further complicated by nonlinear tire force behaviour and the nonlinear relationship between system states.

In recent years, researchers have studied the control of over-actuated AVs; features of these vehicles may include, for example, four-wheel independent steering (4WIS) and four-wheel independent drive (4WID) [4–13]. While an increasing number of available actuators enables a growing ability to precisely manipulate the vehicle to meet its control objectives while satisfying the defined constraints, the problem of coordinating these actuators to enable this capability becomes increasingly complex. Model predictive control (MPC) is a powerful control algorithm suitable for application to the vehicle motion control task; the technique can coordinate a large number of actuators to achieve a set of objectives while adhering to a set of constraints. MPC utilizes a reference model to predict system behaviour over a finite prediction horizon given the current state and selects a feasible

input sequence that will result in the optimal system behaviour; therefore, knowledge of vehicle dynamics may be incorporated directly into the control input selection process.

The University of Waterloo's Mechatronic Vehicle Systems (MVS) Lab is currently developing WATonoTruck, an over-actuated autonomous flatbed truck equipped with independent drive and steering at each wheel. The novelty of WATonoTruck stems from its corner module (CM)-based design. CMs, also developed by the MVS Lab, are effectively small electric vehicles with a single wheel; each contain independent suspension, steering, drive, and brake systems, and a microcontroller [14]. Four modules are installed at each corner of the truck; they communicate with each other and a central computer using a controller area network (CAN) bus. While most existing vehicle motion control research treats the vehicle as a single system to be controlled by a controller, the modular nature of the WATonoTruck hardware platform motivates its representation as a distributed system. With this framework, each subsystem, or "agent", utilizes its own controller and continuously selects its control actions in collaboration with all others. Agent-based MPC (AMPC), a distributed, cooperative optimal control algorithm, is suitable for this purpose; in comparison to standard, centralized MPC, this control strategy facilitates increased flexibility to changes in the system's actuator topology and tolerance to actuator faults, while providing equivalent optimal control performance [15–17].

1.2 Objectives and contributions

The primary objective of this work is to design an AMPC controller for WATonoTruck. This task includes the formulation of a vehicle dynamics reference model that incorporates Ackermann steering geometry and nonlinear tire force behaviour; the reconfiguration of the reference model to account for the distributed nature of the system; defining state, input, and input rate constraints to ensure system safety and maneuver feasibility; the design of an external longitudinal controller that can be incorporated into the agent-based scheme, if desired; and the AMPC formuation. While this work focuses on the control of WATonoTruck, the vehicle model and controller formulation are presented in a generalized way to allow for the application of the controller, with or without modification, to other vehicle hardware platforms.

The second objective is to implement and validate the AMPC scheme through both simulation and experimental testing. The capability of the designed framework to effectively control WATonoTruck, an over-actuated vehicle platform, will be evaluated by utilizing different actuator configurations to demonstrate flexibility to hardware topology changes,

simulating actuator faults to exhibit robustness to hardware failure, and driving the vehicle in harsh simulation scenarios.

The central novel contribution of this work is the application of AMPC to the task of autonomous path tracking. Additional novel contributions include the ability of the controller to easily handle different actuator configurations, including cases where certain actuators are disabled, and to consider the effects of "non-controllable" agents that are controlled externally to the AMPC scheme on the overall system dynamics; a method to prevent reference model singularities at low speeds without causing model errors at higher speeds; and the derivation of constraints applied to the imaginary front- and rear-center steering angles that prevents violations of the Ackermann geometry at all corners.

1.3 Outline

Chapter 2 includes a review of the literature relevant to the objectives of this thesis. Topics covered include vehicle path tracking and stability control, with a focus on studies that utilize MPC; distributed model predictive control and its prior application to vehicle motion control tasks; and control strategies used when considering over-actuated vehicles.

In Chapter 3, the formulation of the AMPC prediction model is outlined. This includes an overview of the WATonoTruck hardware platform, the presentation of the equations of motion needed to represent the dynamics of the system, the state space representation of the system in both centralized and agent-based forms, the method to use the model to predict the system state over the MPC prediction horizon, an approach to account for hardware faults within the model, and a model modification that prevents singularities at low speeds.

In Chapter 4, the formulation of the AMPC controller is presented. First, the constraints enforced on the system states, inputs, and input rates are given. The MPC formulation is then outlined before the details of the AMPC algorithm are discussed. A non-controllable longitudinal velocity controller that can be incorporated into the AMPC framework is also presented.

Simulation and experimental results conducted using the AMPC controller are presented in Chapter 5. The simulations demonstrate that the AMPC framework is capable of handling varied actuator configurations, accounting for hardware faults, and effectively controlling the vehicle in harsh scenarios. The path tracking performance demonstrated through the experimental results further reinforces these findings.

Chapter 6 concludes this thesis with a summary of its contents and an outline of possible future directions for the project to continue the development of the AMPC framework.

Chapter 2

Literature review and background

2.1 Vehicle motion control

Vehicle motion control is a complex task that involves regulating the longitudinal, lateral, and vertical components of a vehicle's motion. For AV applications, the purpose of the motion controller is to enable the vehicle to track a desired reference path and velocity profile while maintaining stability. These high-level requirements further comprise a set of lower-level objectives. For instance, to ensure vehicle stability, the controller may constrain the vehicle yaw rate and limit the longitudinal and lateral slip of the wheels to confine the system state to a stable region while preventing tire force saturation. Path tracking objectives include minimizing both the lateral deviation and heading error of the vehicle with respect to the reference path. These objectives must be met while coordinating all available actuators with consideration for the constraints on actuator states and rates of change.

MPC is a popular control approach that is suitable for vehicle motion control; the technique can coordinate multiple system inputs to achieve optimal control performance while considering constraints on both the inputs and system states. Linear MPC considers the current system state and a desired reference trajectory over a finite time horizon that extends into the future to select the optimal control input vector at every samping instant. The MPC objective is formulated as a convex quadratic programming (QP) optimization problem that considers the difference between the reference and actual states over the horizon to select globally optimal inputs. MPC requires a linear reference model to predict system behaviour over the horizon; standard vehicle dynamics models, while typically nonlinear, can be linearized about the operating point at each sampling instant and used

as the MPC reference. The following subsections describe the application of MPC to vehicle motion control problems.

2.1.1 Stability control

Many researchers apply MPC to achieve vehicle stability control objectives. One recurring strategy involves the use of tire forces as the MPC reference model inputs. Palmieri et al. [18] design an MPC controller used to stabilize a vehicle when performing sudden lane change maneuvers or taking corners at high speeds. The controller selects longitudinal force modifications to the driver's intended drive force at each wheel that result in optimal lateral behaviour. The forces are realized using an integrated slip controller that selects brake torque while preventing excessive wheel spin. Since the forces are selected by the controller, it is not necessary to consider nonlinear tire force behaviour when constructing the MPC reference model, which simplifies the optimization. Instead, the nonlinear pure slip magic formula tire model is used to determine the tire force bounds, which serve as a controller constraint. Beal and Gerdes [19] devise a safe handling envelope that defines bounds on the vehicle yaw rate and sideslip angle that the vehicle must not exceed to remain laterally stable. The authors use MPC to perform lateral vehicle control; a linear bicycle model serves as the reference model, the lateral tire force at the front wheel serves as the input, and the rear lateral force is estimated using a linearized brush tire model; the driver's steering angle input is corrected by considering the optimal lateral tire force. In addition to the stability constraints, a yaw rate reference is incorporated so that the driver's intended maneuver is executed. Experimental results indicate that the vehicle stability is maintained regardless of the driver input.

Another effective approach to MPC-based vehicle stability control involves utilizing an overall yaw moment that acts on the vehicle as an control action in addition to, or in place of, the forces that act on the vehicle. Bernardini et al. [20] employ a hybrid MPC technique to combine active front steering (AFS) and differential braking for lateral stability control. The effect of differential braking on the vehicle is represented by a yaw moment input. The prediction model is hybridized through the inclusion of an integral term to the yaw rate tracking objective that mitigates the effect of both disturbances and model inaccuracies and facilitates zero steady state tracking error. Additionally, wheel sideslip angles are minimized in the MPC optimization. Di Cairano et al. [7] similarly utilize AFS and differential braking via MPC to ensure yaw stability while tracking the yaw behaviour intended by the driver as indicated by the steering input. Differential braking is again emulated using an overall yaw moment input. The steering angle rate is included as a model input while the steering angle is incorporated into the system state

vector. Wheel slip angle constraints are imposed, and a piecewise affine approximation of a nonlinear tire force model is used; the wheel status is assumed to be either linear or saturated over the prediciton horizon according to its operating condition. Nahidi et al. [8] design a multi-layer integrated longitudinal and lateral stability controller. The upper layer uses MPC to determine the overall longitudinal force and yaw moment that will lead to stable behaviour. The center of gravity (CG)-based nature of this controller makes it applicable to any passenger vehicle, regardless of the powertrain configuration. A low-level optimal torque distributor is used to compute the torque at each wheel that will result in the desired CG longitudinal force and yaw rate. The controller is evaluated using an electric vehicle in many harsh maneuvers on slippery roads, including double lane change, full throttle launch, and acceleration in turn; the controller is able to maintain the stability of the vehicle in each scenario.

Considering the CG forces and moments as inputs to the MPC scheme reduces model simplicity and computational intensity by removing the need to include nonlinear tire force behaviour in the prediction model; tire models can instead serve to constrain the MPC inputs. Further, this method provides a high-level controller that is adaptable to different powertrain configurations. However, by utilizing wheel torque values as inputs to the model rather than CG forces and moments, the MPC controller can consider how wheel dynamics will influence the behaviour of the system over the prediction horizon. Jalali et al. [9] design an integrated vehicle stability and slip controller using MPC that considers torque applied via differential braking and torque vectoring as inputs to the system. A double track vehicle model is used; the yaw rate, lateral velocity, and wheel angular velocities are regulated to maintain stability. The controller can be modified as needed to accommodate different driveline configurations, such as front-wheel drive or rear-wheel drive, by removing the relevant torque inputs from the reference model. Ataei et al. [21] present a reference model that can be reconfigured to perform stability control using MPC for vehicles with different wheel configurations. The model considers torque applied by differential braking and torque vectoring, active front and rear wheel steering, and active camber. Ataei et al. [10] use MPC to perform slip control when accelerating or braking, lateral stability and handling improvement, and rollover prevention. Torque at each corner and active steering at the front wheels are the inputs considered. The capability of the controller to sucessfully handle all objectives demonstrates the proficiency of MPC at handling multiple integrated control tasks.

2.1.2 Path tracking

While it is necessary to consider vehicle stability requirements when developing AV motion controllers, additional objectives must be incorporated when implementing path tracking capabilities. Many researchers use MPC to control autonomous vehicles while satisfying path tracking and stability requirements. Falcone et al. [22] use MPC to control an AV equipped with AFS to track a predefined path in a double lane change scenario on a slippery road. Two different approaches are considered. The first uses a nonlinear vehicle model, resulting in a nonlinear optimization problem that cannot be solved at a frequency suitable for high-speed experimental testing. The second involves linearizing the nonlinear model at the operating point at each sampling instant to generate a linear time-varying (LTV) model to be used with linear MPC. Falcone et al. [23] extend the previous work by considering an additional constraint to guarantee stability; AFS is used to perform path tracking. Cui et al. [24] present a path tracking controller using MPC that incorporates an unscented Kalman filter (UKF) to estimate unmeasurable states including the vehicle sideslip angle, the wheel sideslip angles, and the tire-road friction coefficient (TRFC). A 3DOF vehicle model combined with the magic formula tire force model are used to represent the system dynamics. The controller and estimator combination was found to provide accurate path tracking performance over a wide range of road conditions. Zhang et al. [25] investigate the application of MPC to low-speed autonomous valet parking path tracking scenarios. A kinematic vehicle model was used as the MPC reference model, and experimental results demonstrated the efficacy of the technique in various scenarios.

MPC may be combined with other techniques to mitigate issues including hardware faults or the unexpected appearance of an obstacle. Geng et al. [26] devise a fault-tolerant MPC scheme that provides robust path tracking. A fault signal detection algorithm, developed using Kalman filtering and chi-squared detectors, is used to detect sensor faults and fuse multiple redundant sensor measurements to maintain accuracy in the presence of a fault. This strategy enables the system to maintain optimal path tracking performance when faults occur. Brown et al. [27] present an MPC-based path tracking controller that prevents unsafe behaviour in the presence of an obstacle. Using the stability envelope developed by Beal and Gerdes [19] in addition to an environmental envelope that defines the safe driving area, the controller enables the vehicle to deviate from the defined reference path to avoid collisions while maintaining stability and minimizing lateral and heading error. Alsterda et al. [28] propose the contingency MPC algorithm. This novel technique enables the vehicle to track the desired path while maintaining a contingency plan, an alternative path that may be taken should an emergency occur. The technique enables the vehicle to deviate from the desired path to prevent stability when the road surface condi-

tion changes and a loss of friction occurs; the efficacy of the controller was demonstrated in experimental scenarios conducted on snow- and ice-covered roads. Hajiloo et al. [29] use MPC for combined path planning and tracking. Their controller employs active steering and, when needed, differential braking to track the reference while avoiding collisions with obstacles. Ji et al. [30] use MPC perform path tracking along a reference path generated by an artificial potential field. The potential field spans the drivable area of the vehicle; areas of high intensity in the field indicate locations where obstacles exist and the vehicle should avoid. The reference path is constructed by considering the negative gradient of the potential field to determine the direction of steepest descent. Vehicle dynamics are not considered when generating the path, which may render it infeasible to follow. Rasekhipour et al. [31] address this issue; in this work, the potential field is integrated into the MPC objective function in addition to a nominal reference path corresponding to the lane centerline. This enables the controller to consider both the dynamics of the vehicle and position of nearby obstacles when planning and tracking the path.

Effective MPC-based path tracking is only possible if the controller can run in real time. Reducing the computational complexity of the algorithm enables the use of increasingly accurate vehicle models that will aid prediction accuracy, thereby improving path tracking performance. Wang et al. [32] develop an approach to, in real-time, adjust the MPC parameters with the greatest effect on the optimization time, namely the prediction horizon length, control horizon length, and sampling frequency. The method improved the path tracking performance given the capacity of the CPU used to perform the optimization. Zhang et al. [33] highlight various challenges regarding the application of MPC to vehicle path tracking, including the computational complexity that results from long horizon lengths and a high number of optimization variables, and propose an improved MPC using Laguerre functions and exponential weights to mitigate these issues. The Laguerre functions reduce the number of variables to be optimized; they formulate the input sequence over the horizon as a linear combination of orthogonal functions. The coefficients that comprise the linear function are optimized; because the set of coefficients is smaller than the set of control inputs over the horizon, optimization efficiency is improved. Exponential weights are used to reduce tracking error priority further along the horizon to minimize the effect of compounding model prediction error. Choi et al. [34] introduce a variable sampling time MPC that is applied to vehicle path tracking. The optimal sampling time is selected by considering the magnitude of the steering angle and lateral acceleration while the vehicle is running. When compared to an MPC controller with a fixed sampling time, tracking performance was improved while computational efficiency remained the same.

The accuracy of the MPC reference vehicle model greatly impacts the controller's efficacy. Model accuracy may be improved by utilizing online parameter estimations or

incorporating data-driven methods. Lin et al. [35] present an adaptive MPC controller that can estimate the tire cornering stiffness and TRFC in varying road conditions. The controller demonstrates effective path tracking capabilities in changing conditions. Liang et al. [36] use adaptive MPC for a combined decision making, motion planning, and path tracking scheme. The proposed controller can accurately track the path in lane change scenarios on straight and curved roads. Aswani et al. [37] develop a learning-based MPC (LBMPC) controller that uses machine learning to augment the reference model by estimating the error between the model predictions and the actual evolution of the system state. In this work, the method is applied to various non-ground vehicle applications; to extend this work, Ostafew et al. [38, 39] apply the strategy to mobile robot path tracking tasks. In these papers, Gaussian process regression is the machine learning technique used. Rokonuzzaman et al. [40] apply LBMPC to AV path tracking using neural networks as the learning method. In this case, the neural network replaces the dynamic reference model, rather than augmenting it, once enough data has been collected.

2.2 Distributed model predictive control

The MPC technique employed by all works cited in Section 2.1 uses a single, centralized optimization problem to select all inputs at the each sampling input by considering all objectives of the system. Because of this characteristic, the technique is commonly referred as "centralized MPC". Unlike centralized MPC, alternative approaches such as distributed and decentralized MPC must perform more that one optimization to select all system inputs at each sampling instant. One such approach, cooperative distributed MPC (DMPC), was first presented by Venkat et al. [41] and extended by Rawlings et al. [42]. The authors reason that centralized MPC is unsuitable for the control of large, networked systems due to organizational rather than computational issues. For example, a centralized controller would be inflexible to changes in the plant and would not be able to effectively control all subsystems simultaneously if some were operated externally to the MPC controller. Although alternative DMPC frameworks had been proposed prior to this work, the preceding techniques did not guarantee optimal control performance equivalent to that of a centralized MPC scheme; conversely, the distributed controller of Venkat et al. is guaranteed to provide optimal control performance equivalent to centralized MPC. To achieve this, each subsystem is modelled as an independent, decentralized system with its own states and inputs. Interaction models are used to characterize the relationships between each subsystem. The decentralized and interaction models are combined to determine the effect of each subsystems' actions on the evolution of all others. Christofides

et al. [43] compare the cooperative DMPC technique developed in the previous works to other non-centralized MPC techniques, namely decentralized MPC and non-cooperative distributed MPC. Similar to cooperative DMPC, both decentralized and non-cooperative distributed MPC involve segmenting the system into multiple subsystems, each with their own controllers and inputs to optimize. With decentralized MPC, there is no communication between the subsystems. Therefore, in scenarios where some subsystems have a strong influence on the others, stability and control optimality issues may arise. For decentralized MPC to be successful, the interactions between subsystems must be weak enough to be considered minor disturbances. Conversely, cooperative DMPC allows for communication between subsystems to enable better control performance. In the non-cooperative case, while subsystems can communicate, each subsystem considers their own local objectives that may be in conflict with the objectives of others. This may result in non-optimal behaviour for certain subsystems. This serves as a contrast to cooperative DMPC, where the objectives are shared globally among all agents, and an optimal solution can be found.

Given the number of microcontrollers utilized on modern passenger vehicles and the increasing amount of motion control features that they implement, it is reasonable to view vehicles as a large, networked system that would benefit from a distributed control approach. Recognizing this opportunity, Tang and Khajepour [15] apply cooperative DMPC to the vehicle stability control task for modular urban vehicles. The plant considered in this work was developed using CMs [14], which are effectively small, one-wheel electric vehicles that contain their own microcontrollers and suspension, steering, drive, and brake systems. The cooperative DMPC methodology is applied by considering each CM as its own subsystem, or "agent". Each agent selects the drive torque, brake torque, steering angle, and active suspension inputs by cooperating with all other agents to achieve optimal control performance. Tang and Khajepour [16] extend this work by applying cooperative DMPC to passenger vehicle equipped with active steering, torque vectoring, and differential braking. Different methodologies to cluster the actuators into agents are discussed, including clustering actuators by corner as in to the previous work, or by actuator type. Here, the performance of the controller is validated in simulation; it is further validated experimentally by Tang et al. [17] on an equivalent passenger vehicle hardware platform. In these three studies, the authors augment the standard cooperative DMPC formulation by incorporating non-cooperative agents into the system. These agents represent groups of actuators that cannot be controlled by the MPC controller as they are handled, for example, by some proprietary, third party logic that cannot be accessed. In this case, the actions of the non-controllable agents are provided to the controllable agents, i.e., those whose actions are selected by the cooperative DMPC scheme. This allows the controllable agents to consider the effect of the non-controllable agents' actions on the dynamics of the

system while selecting their own. In these works, the authors choose to name their control methodology agent-based MPC (AMPC) to indicate that it coordinates the control efforts of multiple agents, defined according to existing actuator topologies, while accounting for the non-controllable agent inputs, which are not considered in the standard cooperative DMPC formulation.

2.3 Control of over-actuated vehicles

Vehicles that have independently steered and driven wheels, such as those developed using the CM platform, are highly over-actuated. In comparison to conventional passenger vehicles, these systems have the potential for increased maneuverability and safety due to the large number of redundant inputs. For example, a vehicle that can steer both its front and rear wheels can make much sharper turns in low speed scenarios in comparison to a vehicle that only has front wheel steering capabilities. Furthermore, in the event of an actuator hardware failure, the redundant actuators that remain functional can compensate for the failure and maintain the desired system behaviour. However, the drawback of this feature is the challenge of coordinating the large set of actuators required to control the vehicle while meeting all objectives.

Many studies involve the design of vehicle stability controllers for over-actuated vehicles. Li et al. [4] propose the application of an optimal torque distribution controller to a four-wheel independent drive (4WID) vehicle to improve stability and handling. The algorithm considers the drivers' drive and brake commands to determine the desired longitudinal force at each wheel, and uses this information to determine torque corrections to be applied. Moreover, direct yaw moment control (DYC) is commonly applied to this class of problems. Chen et al. [5] utilize DYC, a wheel force distribution controller, and a UKF estimator to improve vehicle stability. The DYC controller uses sliding mode control (SMC) to select the CG yaw moment that will result in the desired yaw rate and sideslip angle. The wheel force distributor determines the longitudinal wheel forces at each wheel that will result in this yaw moment, along with an overall longitudinal force acting on the vehicle that will result in the driver's desired acceleration as indicated by the accelerator pedal input. The optimization attempts to minimize the longitudinal force at each wheel while satisfying the other objectives. Yim [6] implements a technique similar to Chen et al. [5], where SMC-based DYC is used to determine the the desired yaw moment. In this case, the vehicle considered is both 4WID and four-wheel independent steer (4WIS); the desired forces at each wheel are thus determined by considering both the wheel torque and steering angles. In this case, the desired longitudinal behaviour of the vehicle is not considered,

so the vehicle may not accelerate as the driver intends when the stability control features are engaged. The works by Bernardini et al. [20], Di Cairano et al. [7], Nahidi et al. [8], Jalali et al. [9], and Ataei et al. [10] discussed in Section 2.1.1 similarly discuss how control actions are coordinated in 4WID systems or those equipped with differential braking.

Over-actuated vehicle path tracking has also been previously investigated. Tan et al. [11] utilize MPC for path tracking control of a two-wheel independent steer, 4WID vehicle. The method provides good path tracking performance in simulated scenarios. Hang et al. [12] use MPC for path tracking control of a 4WIS vehicle. A single track vehicle model is used, and the front and rear steering angles selected by the controller are applied to both the front and rear two wheels, respectively. While the control performance is acceptable, the Ackermann steering geometry is not considered, which may result in unecessary large sideslip angles in experimental scenarios. Tan et al. [13] develop a 4WIS vehicle path tracking controller that, rather than imposing a restrictive relationship between the front and rear steering angles, derives a model that avoids this restriction by considering Ackermann steering geometry.

Multiple works use MPC to perform combined path tracking and velocity control. Ren et al. [44] propose an integrated controller that considers both path tracking and stability objectives for a vehicle with torque vectoring capabilities. In this approach, an upper-level MPC controller first selects a steering angle and total longitudinal force necessary to follow the path and desired velocity. Then, a yaw rate controller determines a desired yaw rate by considering the desired steering angle and current yaw rate and lateral velocity. The desired velocity is passed to a stability controller, which determines the overall yaw moment to be applied to the vehicle; this, along with the total longitudinal force, is used by an optimal torque distributor to determine the torque at each corner that will satisfy all objectives. Wu et al. [45] develop an MPC-based path tracking controller for a 4WID vehicle equipped with AFS. The magic tire formula is used to incorporate accurate tire behaviour into the reference model; the tire parameters are determined using nonlinear least squares fitting on tire force data. Wheel sideslip angles are constrained to be small, and slack variables are used with these constraints to prevent optimization infeasibility. Tire force constraints are imposed by considering the friction circle and actuator torque limits. Jeong and Yim [46] use MPC for path tracking and velocity control of a 4WID, 4WIS vehicle. The steering angle and longitudinal force at each wheel were taken to be the control inputs. Wheel sideslip angle and slip ratio where minimized in the MPC cost function to prevent tire saturation, and a linear tire model was used to estimate the lateral forces.

2.4 Summary

In this chapter, vehicle path tracking- and stability control-related literature was first reviewed; in particular, the application of MPC to vehicle motion control problems was explored due to the large body of existing work demonstrating the efficacy of the algorithm. The existing approaches can be categorized according to the choice of control inputs; namely, the tire forces, overall yaw moment, and wheel torque. While the use of tire forces and yaw moments as inputs improves the reference model simplicity by removing the need to incorporate nonlinear tire force behaviour, the model accuracy decreases. Further, a separate torque distribution controller is required to determine the torque at each wheel that will result in the desired forces and moments. Additionally, researchers consider methods to improve controller robustness to faults and the appearance of obstacles, reduce computational complexity, and enhance prediction model accuracy.

Subsequently, an overview of DMPC and its prior application to vehicle motion control problems was provided. Cooperative DMPC provides an effective alternative to centralized MPC for controlling large, distributed systems. By representing a vehicle as a distributed system, researchers apply AMPC to the task of vehicle stability control.

Finally, the techniques used to control over-actuated vehicles were investigated. In particular, due to the documented success of the algorithm, MPC-based techniques were again the primary focus of this section. The reviewed studies demonstrate that excellent control performance can be provided when all actuators of an over-actuated vehicle are effectively coordinated and that MPC may be used to perform combined path tracking and longitudinal velocity control. In some studies, Ackermann steering geometry is not considered, which may result in unecessary wheel slip, and the lateral controllers do not consider the effect of their actions on the longitudinal dynamics of the vehicle.

The design of the agent-based controller presented in this thesis is informed by the preceding work that is outlined in this chapter. An integrated reference model with wheel torque inputs that considers nonlinear combined slip lateral tire force is constructed; these features enhance the prediction accuracy by representing the complete relationship between wheel torque and the motion of the vehicle. However, a combined slip longitudinal tire force model is not used; to reduce computational complexity, these forces are assumed to be proportional to the torque at each wheel. The vehicle parameters used to construct the prediction model are determined using computer aided design software. It is assumed that, given these features, the model accuracy is sufficient and that adaptive and learning-based methods are not needed. Furthermore, it is assumed that an upstream trajectory planner will provide a safe reference trajectory, so integrated collision avoidance features are not

needed. The agent-based nature of the controller allows it to easily consider the effects of hardware faults. Moreover, Ackermann geometry is used to both lessen the challenges associated with over-actuated vehicle control and minimize unnecessary wheel slip.

Chapter 3

Vehicle dynamics prediction model

3.1 WATonoTruck hardware platform

The vehicle motion controller presented in this work was created for use with WATonoTruck, an autonomous flatbed truck developed by the MVS Lab at the University of Waterloo. The development of WATonoTruck is motivated by the benefits provided by autonomous mobility in domains such as mining, shipping, agriculture and warehousing that involve the frequent transportation of heavy material. The assembled truck is displayed in Figure 3.1.

Figure 3.1: The assembled WATonoTruck.

The truck body was built to incorporate CMs, which are individual wheel modules that contain independent steering, drive, brake, and suspension systems, along with a microcontroller (MCU). A primary advantage of the CM-based modular vehicle design

approach is the ease with which the CMs can be installed on any vehicle chassis, in any configuration, to suit the needs of a particular application. CMs are space efficient, can support large payloads, and provide independent drive-, steer-, and brake-by-wire control capabilities at each wheel. CM specifications are given in Table 3.1. A diagram of the CM is shown in Figure 3.2. For cost reasons, the two rear CMs on WATonoTruck do not contain electric motors. To enable autonomous driving cababilities, WATonoTruck is further equipped with a suite of sensors and computers used to perceive obstacles in the environment and implement the autonomous driving stack. The sensors used include two Robosense RS-LiDAR-16 LIDARs, which are installed at the front and rear of the truck and provide a detection range of up to 150 m across a 180 degree horizontal field of view; two Robosense RS-Bpearl LIDARs, installed at the sides to enable blind spot dectection; and four Basler Dart daA1920-160uc S-Mount cameras, each equipped with 101 degree horizontal field of view lenses, that are installed in pairs at the front and rear. This combination of sensors enables the detection of all obstacles in the vicinity of the truck. A Nvidia Jetson AGX Xavier is used to process the camera data and perform 2D object detection, while a Simply NUC Ruby PC is used to run all other processes.

Parameter	Value
Maximum payload	3000 kgf
Rated motor torque output	$\pm$ 220 N m
Motor-to-wheel gear reduction	8:1
Maximum steering angle	$\pm$ 32 deg
Maximum steering angle rate	$\pm$ 6.8 deg/s

Table 3.1: Corner module specifications.

The modular nature of WATonoTruck makes it suitable to demonstrate the benefits of the AMPC framework when applied to vehicle motion control problems. Each CM can be represented as a subsystem, or agent, that operates as part of a larger distributed system. Each agent uses its own MPC controller to select optimal control actions; the optimization procedure is performed in collaboration with the other agents. The Simply NUC Ruby PC serves as the central computer used to facilitate the agent collaboration, while the MCUs at each corner are used to apply the optimal control actions.

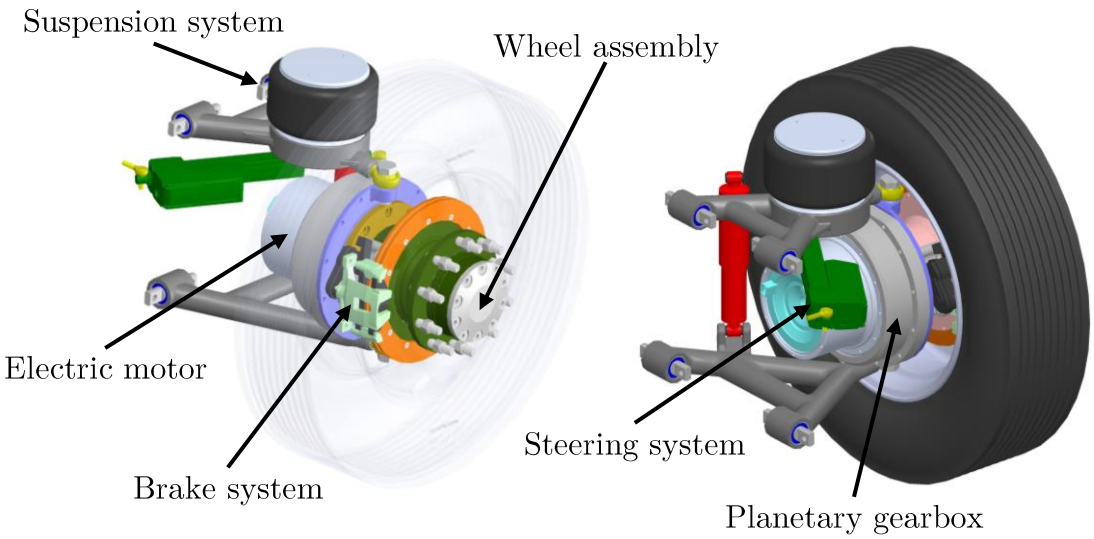


Figure 3.2: Corner module hardware overview. Diagram originally presented in an MVS Lab internal report.

3.2 Double-track vehicle dynamics modelling

MPC, and therefore AMPC, require a prediction model that accurately represents the system dynamics to provide optimal control performance. In this section, a double-track longitudinal and lateral vehicle dynamics model is presented. The model consists of a point mass at the vehicle's CG location and the position of the vehicle's wheels, which generate forces at the tire-road interface that act upon the CG. The vehicle's longitudinal velocity u, lateral velocity v, and yaw rate r serve as the model's states. The evolution of these states is characterized by the differential equations

$$m(\dot{u} - vr) = F_x, \tag{3.1}$$

$$m(\dot{v} + ur) = F_y, \tag{3.2}$$

$$I_z\dot{r} = M_z, \tag{3.3}$$

where m represents the vehicle mass; I_z, the yaw moment of inertia; F_x and F_y, the total longitudinal and lateral forces that act on the CG along the vehicle's x- and y-axes, respectively; and M_z, the total yaw moment that acts on the CG about the vehicle's z

axis. These equations can be rearranged as

$$\dot{u} = \frac{1}{m} F_x + vr, \tag{3.4}$$

$$\dot{v} = \frac{1}{m} F_y - ur, \tag{3.5}$$

$$\dot{r} = \frac{1}{I_z} M_z, \tag{3.6}$$

to model the state time derivatives. The total forces and moment F_x, F_y, and M_z that act upon the vehicle CG are summations of the corresponding forces and moments

$$F_x = \sum_i F_{xi}, \tag{3.7}$$

$$F_y = \sum_i F_{yi}, \tag{3.8}$$

$$M_z = \sum_i M_{zi}, \tag{3.9}$$

generated at the wheels. Here, $i \in \{fl, fr, rl, rr\}$ indexes the vehicle corners and F_{xi}, F_{yi}, and M_{zi} are the forces and moments generated at the wheels expressed in the vehicle body frame. These quantities equivalent to

$$F_{xi} = \cos(\delta_i) f_{xi} - \sin(\delta_i) f_{yi}, \tag{3.10}$$

$$F_{yi} = sin(\delta_i) f_{xi} + \cos(\delta_i) f_{yi}, \tag{3.11}$$

$$M_{zi} = T_i F_{xi} + a_i F_{yi}, \tag{3.12}$$

where f_{xi} and f_{yi} are the tire forces expressed in the frame of wheel i, δ_i is the steering angle of wheel i, a_i is the longitudinal distance between the CG and wheel i, and T_i is the lateral distance between the CG and wheel i. a_i and T_i are given as

$$a_i = \begin{cases} a & i = \{fl, fr\} \\ -b & i = \{rl, rr\} \end{cases}, \tag{3.13}$$

$$w = a + b, \tag{3.14}$$

$$T_i = \begin{cases} -T_f/2 & i = fl \\ T_f/2 & i = fr \\ -T_r/2 & i = rl \\ T_r/2 & i = rr \end{cases}, \tag{3.15}$$

$$T = T_f = T_r. \tag{3.16}$$

Here, a and b are the longitudinal distances between the CG and the front and rear axles, respectively; w is the wheelbase; and T is the track width, which is equivalent for both the front and rear axles. Figure 3.3 displays the vehicle states, geometry, and coordinate frame definitions. The relationship between the tire forces expressed in the vehicle body and wheel frames is outlined in Figure 3.4.

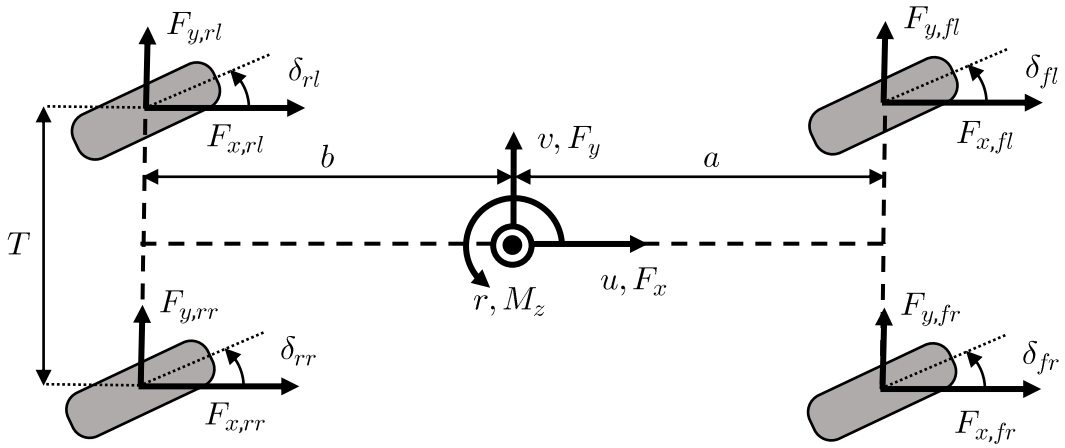


Figure 3.3: Double-track vehicle dynamics model.

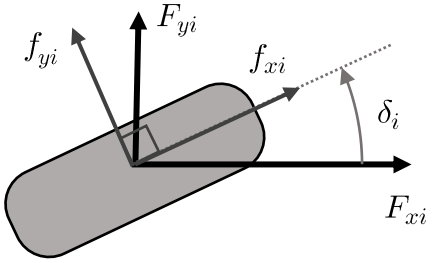
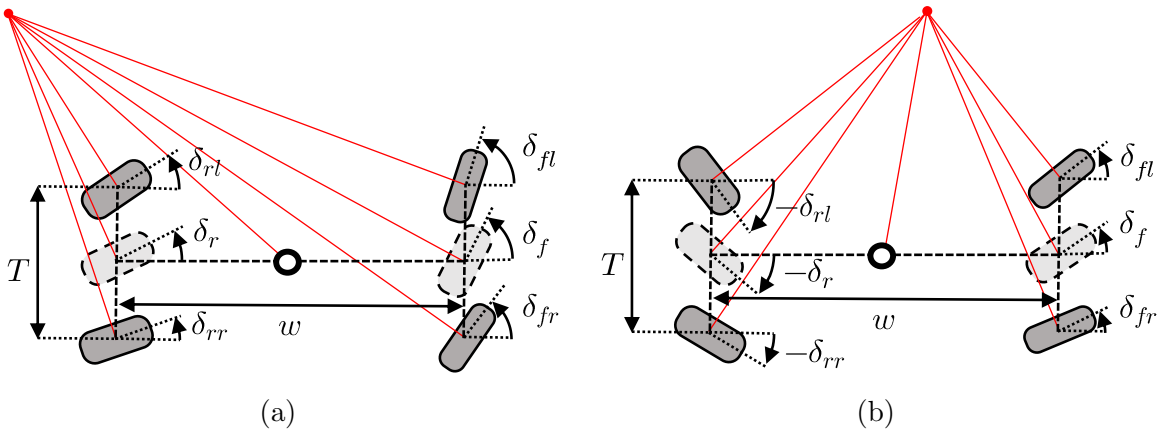


Figure 3.4: Tire forces expressed in the vehicle body and wheel frames.

Ackermann steering geometry is used to reduce the number of inputs that the controller must optimize and to prevent unnecessary wheel slip. Two steering angles, δ_f and δ_r, are defined; these angles correspond to the imaginary center wheels shown in Figure 3.5. Once



(a) (b)

Figure 3.5: Example steering angle combinations where all wheels rotate about the same point to maintain the Ackermann steering geometry. In (a), the point about which the vehicle is rotating is behind the rear axle; all steering angles have the same sign. In (b), the point about which the vehicle is rotating is between the axles; the front and rear steering angles have opposite signs.

δ_f and δ_r are selected, a point about which the vehicle should smoothly revolve is defined. Subsequently, the corner steering angles $\delta_i \ \forall i$ are selected using [13]

$$\delta_{fl} = \arctan\left(\frac{\tan(\delta_f)}{1 - \frac{T}{2w}(\tan(\delta_f) - \tan(\delta_r))}\right) \approx \frac{\delta_f}{1 - \frac{T}{2w}(\delta_f - \delta_r)}, \tag{3.17}$$

$$\delta_{fr} = \arctan\left(\frac{\tan(\delta_f)}{1 + \frac{T}{2w}(\tan(\delta_f) - \tan(\delta_r))}\right) \approx \frac{\delta_f}{1 + \frac{T}{2w}(\delta_f - \delta_r)}, \tag{3.18}$$

$$\delta_{rl} = \arctan\left(\frac{\tan(\delta_r)}{1 - \frac{T}{2w}(\tan(\delta_f) - \tan(\delta_r))}\right) \approx \frac{\delta_r}{1 - \frac{T}{2w}(\delta_f - \delta_r)}, \tag{3.19}$$

$$\delta_{rr} = \arctan\left(\frac{\tan(\delta_r)}{1 + \frac{T}{2w}(\tan(\delta_f) - \tan(\delta_r))}\right) \approx \frac{\delta_r}{1 + \frac{T}{2w}(\delta_f - \delta_r)}, \tag{3.20}$$

such that all wheels revolve around this point with minimal sideslip. Small angle approximations are used to simplify this relationship and the overall model.

States that relate the vehicle's pose to the path must also be included in the prediction model to enable path tracking capabilities. These states include the lateral error e and

heading error $\Delta\psi$ at the vehicle's position, given as the arc length s, along the path. Changes in the lateral and heading error values are governed by

$$\Delta\dot{\psi} = r - r_{des} = r - K\dot{s} \approx r - Ku \tag{3.21}$$

$$\dot{e} = u\sin(\Delta\psi) + v\cos(\Delta\psi) \approx u\Delta\psi + v \tag{3.22}$$

where r_{des} is the yaw rate that must be maintained for the vehicle to follow the path and K is the path curvature at s [27, 47, 48]. e is defined as positive when the vehicle is to the left of the path. K is defined as positive when the path is curving in the counterclockwise direction. The path state definitions are shown in Figure 3.6.

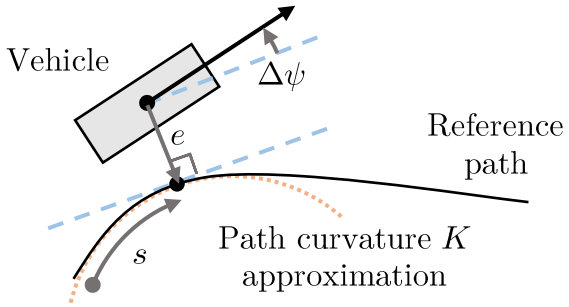


Figure 3.6: Definitions of path states and related quantities.

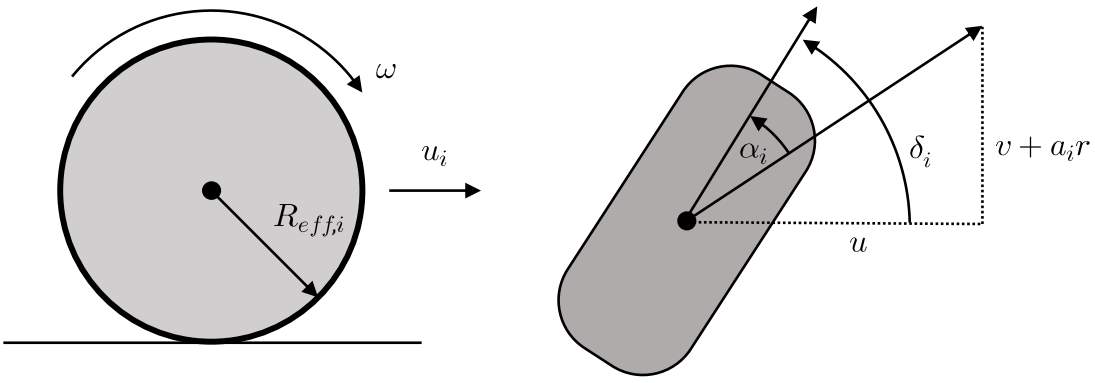
3.3 Tire force modelling

Many nonlinear combined slip lateral tire force models establish a relationship between the longitudinal slip ratio λ_i, the sideslip angle α_i, and the generated lateral force f_{yi}. λ_i and α_i are defined as

$$\lambda_i = \frac{R_{\text{eff},i}\omega_i - u_i}{\max(R_{\text{eff},i}\omega_i, u_i)}, \tag{3.23}$$

$$\alpha_i = \delta_i - \arctan\left(\frac{v + a_i r}{u}\right) \approx \delta_i - \frac{v + a_i r}{u}, \tag{3.24}$$

where $R_{\text{eff},i}$ and ω_i are the effective radius and angular velocity of wheel i, respectively. Equation (3.24) is simplified using a small angle approximation. The definitions of the quantities required to estimate λ_i and α_i are depicted in Figure 3.7.



(a) (b)

Figure 3.7: (a) Definitions of the variables needed to compute λ_i. (b) Definitions of the variables needed to compute α_i.

In this work, the Burckhardt combined slip lateral tire force model is used to estimate the lateral tire forces generated at the tire-road interface [49]. This model is selected over other popular nonlinear combined slip tire force models, such as the magic formula [50], for its reduced complexity, which will simplify the vehicle dynamics model and improve the optimal controller efficiency. The Burckhardt model is defined as

$$S_{\text{res},i} = \sqrt{\lambda_i^2 + \alpha_i^2}, \tag{3.25}$$

$$\mu_{\text{B},yi} = \frac{\alpha_i}{S_{\text{res},i}}[C_{1y}(1 - e^{-C_{2y}S_{\text{res},i}}) - C_{3y}S_{\text{res},i}], \tag{3.26}$$

$$f_{\text{B},yi} = F_{zi}\mu_{\text{B},yi}, \tag{3.27}$$

where $f_{\text{B},yi}$ represents the Burckhardt model's lateral force estimation, $\mu_{\text{B},yi}$ is the normalized lateral force, $S_{\text{res},i}$ is the combined slip ratio, and F_{zi} is the normal load on the wheel. C_{1y}, C_{2y}, and C_{3y} are the model parameters to be fitted. Therefore, in this work,

$$f_{yi} = f_{\text{B},yi}. \tag{3.28}$$

When the prediction model described in this chapter is used to validate the motion controller in simulation, the simulated plant, described in Section 5.1, provides the normal force acting on each tire throughout the simulation; therefore, F_{zi} estimation is not required. However, when performing experimental testing, the hardware platform, described

in 5.3, does not have tire force sensors. To account for this, when using the controller experimentally, F_{zi} is estimated using

$$F_{zfl} = \frac{1}{2w}(mgb - ma_x h) + \frac{ma_y h}{2T}, \tag{3.29}$$

$$F_{zfr} = \frac{1}{2w}(mgb - ma_x h) - \frac{ma_y h}{2T}, \tag{3.30}$$

$$F_{zrl} = \frac{1}{2w}(mgb + ma_x h) + \frac{ma_y h}{2T}, \tag{3.31}$$

$$F_{zrr} = \frac{1}{2w}(mgb + ma_x h) - \frac{ma_y h}{2T}, \tag{3.32}$$

where g is the gravitational acceleration constant, a_x is the longitudinal acceleration, a_y is the lateral acceleration, and h is the CG height with respect to the ground.

The combined slip Burckhardt model can also be used to estimate the longitudinal tire forces. However, to further reduce model complexity, an affine relationship between force and torque is instead used. Ignoring wheel dynamics, the longitudinal force is given as

$$f_{xi} = Q_i / R_{\text{eff},i}, \tag{3.33}$$

where Q_i represents the torque applied at wheel i. While WATonoTruck is FWD, wheel torque inputs Q_i are included for all four corners to improve model generality and enable different drivetrain configurations to be evaluated in Simulation. However, using the agent-based framework, the model can be reconfigured to incorporate only the front two Q_i when used for experimental testing with WATonoTruck; this is discussed further in Section 3.4.2.

3.4 State-space system representation

3.4.1 State-space model linearization and discretization

Linear MPC relies on the definition of a discrete, linear state-space model that represents the dynamics of the system. In this work, the state-space model used to represent the behaviour of WATonoTruck is constructed using the equations presented in Sections 3.2 and 3.3. A nonlinear relationship between the system states and inputs,

$$\dot{x} = f_{\text{SS}}(x, U), \tag{3.34}$$

$$x = [u, v, r, e, \Delta\psi]^T, \tag{3.35}$$

$$U = [\delta_f, \delta_r, Q_{fl}, Q_{fr}, Q_{rl}, Q_{rr}]^T, \tag{3.36}$$

is established, where $x \in \mathbb{R}^{n_x}$ is the state vector, $U \in \mathbb{R}^{n_U}$ is the input vector, and f_{SS} represents the nonlinear relationship. To further reduce model complexity, ω_i is not included as a state in x, and wheel dynamics are therefore not incorporated into the reference model. At each sampling instant, the measured ω_i is used with eq. (3.23) to estimate λ_i; this value of λ_i is used to construct the continuous reference model f_{SS}.

The nonlinear model given in eq. (3.34) can be linearized by performing a first order Taylor expansion of f_{SS}; the result is a linearized model of the form

$$\dot{x} \approx Ax + BU + W, \tag{3.37}$$

where A is the state matrix, B is the input matrix, and W can be treated as a disturbance term. A, B, and W are defined as

$$A = \left[\begin{matrix} \frac{\partial \dot{u}}{\partial u} & \frac{\partial \dot{u}}{\partial v} & \cdots & \frac{\partial \dot{u}}{\partial \Delta\psi} \\ \frac{\partial \dot{v}}{\partial u} & \frac{\partial \dot{v}}{\partial v} & \cdots & \frac{\partial \dot{v}}{\partial \Delta\psi} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \Delta\dot{\psi}}{\partial u} & \frac{\partial \Delta\dot{\psi}}{\partial v} & \cdots & \frac{\partial \Delta\dot{\psi}}{\partial \Delta\psi} \end{matrix}\right]\Bigg|_{x=x(k),\ U=U(k-1)}, \tag{3.38}$$

$$B = \left[\begin{matrix} \frac{\partial \dot{u}}{\partial \delta_f} & \frac{\partial \dot{u}}{\partial \delta_r} & \cdots & \frac{\partial \dot{u}}{\partial Q_{rr}} \\ \frac{\partial \dot{v}}{\partial \delta_f} & \frac{\partial \dot{v}}{\partial \delta_r} & \cdots & \frac{\partial \dot{v}}{\partial Q_{rr}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \Delta\dot{\psi}}{\partial \delta_f} & \frac{\partial \Delta\dot{\psi}}{\partial \delta_r} & \cdots & \frac{\partial \Delta\dot{\psi}}{\partial Q_{rr}} \end{matrix}\right]\Bigg|_{x=x(k),\ U=U(k-1)}, \tag{3.39}$$

$$W = f_{\text{SS}}(x(k), U(k-1)) - Ax(k) - BU(k-1), \tag{3.40}$$

and are evaluated at the operating point $x(k)$, $U(k-1)$ at each sampling instant k. The linearized system is then discretized, resulting in a model of the form

$$x(k+1) = A_d x(k) + B_d U(k) + W_d, \tag{3.41}$$

where A_d, B_d, and W_d are the discretized forms of the matrices A, B, and W, respectively. They are discretized according to

$$A_d = I_{n_x} + \Delta t A, \tag{3.42}$$

$$B_d = \Delta t B, \tag{3.43}$$

$$W_d = \Delta t W, \tag{3.44}$$

where I_{n_x} is the $n_x \times n_x$ identity matrix and Δt is the sampling time. The system output vector $y \in \mathbb{R}^{n_y}$ is defined as

$$y(k) = C_y x(k) = [u, e, \Delta\psi]^T, \tag{3.45}$$

$$C_y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.46}$$

where C_y is the output matrix.

In certain cases, it is desirable to control or constrain additional quantities that do not comprise y. The controlled output $z \in \mathbb{R}^{n_z}$, given as

$$z(k) = C_z x(k) + D_z U(k) + W_z, \tag{3.47}$$

represents the vector of states to be controlled. In this application, the only controlled outputs are u, e, and $\Delta\psi$; therefore, $z = y$, $C_z = C_y$, $D_z = \mathbf{0}_{n_x \times n_U}$, and $W_z = \mathbf{0}_{n_x \times 1}$. Here, $\mathbf{0}_{n_1 \times n_2}$ represents the $n_1 \times n_2$ zero matrix for any $n_1, n_2 \in \mathbb{Z}^+$. Similarly, the constrained output $z_c \in \mathbb{R}^{n_{z_c}}$ is a vector of states to be constrained. It is defined as

$$z_c(k) = C_{z_c} x(k) + D_{z_c} U(k) + W_{z_c}. \tag{3.48}$$

In this application, the states to be constrained are r, α_{fl}, α_{fr}, α_{rl}, and α_{rr}. Therefore, unlike z, $z_c \neq y$; instead, $z_c = [y_c^T, y_{z_c}^T]^T$, where $y_c \in \mathbb{R}^{n_{y_c}}$ represents any elements in x to be constrained, and $y_{z_c} \in \mathbb{R}^{n_{y_{z_c}}}$ represents the states to be constrained that are not elements of x. y_c and y_{z_c} are defined as

$$y_c = [r(k)] = C_{y_c} x, \tag{3.49}$$

$$C_{y_c} = [0, 0, 1, 0, 0], \tag{3.50}$$

$$y_{z_c}(k) = [\alpha_{fl}(k), \alpha_{fr}(k), \alpha_{rl}(k), \alpha_{rr}(k)]^T = A_{y_{z_c}} x(k) + B_{y_{z_c}} U(k) + W_{y_{z_c}}. \tag{3.51}$$

The relationship between α_i, x, and U described in eq. (3.24) is nonlinear. The form of the matrices $A_{y_{z_c}}$, $B_{y_{z_c}}$, and $W_{y_{z_c}}$ are therefore determined by performing a first order Taylor expansion to linearize this relationship; they are defined as

$$A_{y_{z_c}} = \left.\begin{bmatrix} \frac{\partial \alpha_{fl}}{\partial u} & \frac{\partial \alpha_{fl}}{\partial v} & \cdots & \frac{\partial \alpha_{fl}}{\partial \Delta\psi} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \alpha_{rr}}{\partial u} & \frac{\partial \alpha_{rr}}{\partial v} & \cdots & \frac{\partial \alpha_{rr}}{\partial \Delta\psi} \end{bmatrix}\right|_{x=x(k),\ U=U(k-1)}, \tag{3.52}$$

$$B_{y_{z_c}} = \left[\begin{matrix} \frac{\partial \alpha_{fl}}{\partial \delta_f} & \frac{\partial \alpha_{fl}}{\partial \delta_r} & \cdots & \frac{\partial \alpha_{fl}}{\partial Q_{rr}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \alpha_{rr}}{\partial \delta_f} & \frac{\partial \alpha_{rr}}{\partial \delta_r} & \cdots & \frac{\partial \alpha_{rr}}{\partial Q_{rr}}, \end{matrix}\right]\Bigg|_{x=x(k),\ U=U(k-1)} \tag{3.53}$$

$$W_{y_{z_c}} = y_{z_c}(k) - A_{y_{z_c}} x(k) - B_{y_{z_c}} U(k-1). \tag{3.54}$$

Given this relationship, the matrices used to compute z_c as shown in eq. (3.48) are defined as

$$C_{z_c} = \begin{bmatrix} C_{y_c} \\ A_{y_{z_c}} \end{bmatrix}, \tag{3.55}$$

$$D_{z_c} = \begin{bmatrix} \mathbf{0}_{n_{y_c} \times n_U} \\ B_{y_{z_c}} \end{bmatrix}, \tag{3.56}$$

$$W_{z_c} = \begin{bmatrix} \mathbf{0}_{n_{y_c} \times 1} \\ W_{y_{z_c}} \end{bmatrix}. \tag{3.57}$$

3.4.2 State-space model for individual agents

The model presented in Subsection 3.4.1 is formulated for use with a single, centralized controller, and must be augmented before it is used within the agent-based framework. The agent-based control strategy involves representing the system as a set of n_a subsystems, or agents, that each regulate a subset of the system's inputs and collaborate to provide optimal control performance. For any agent j, the system state evolution is described by an extension of eq. (3.41),

$$x(k+1) = A_d x(k) + B_{jd} U_j(k) + B_{jd\mathrm{OC}} U_{j\mathrm{OC}}(k) + B_{d\mathrm{NC}} U_{\mathrm{NC}}(k) + B_{d\mathrm{DA}} U_{\mathrm{DA}}(k) + W_d. \tag{3.58}$$

Here, $U_j \in \mathbb{R}^{n_{U_j}}$ represents the vector of inputs controlled by agent j, $U_{j\mathrm{OC}} \in \mathbb{R}^{n_{U_{j\mathrm{OC}}}}$ represents the vector of inputs controlled by all controllable agents other than j (i.e., all "other controllable" agents, OC), $U_{\mathrm{NC}} \in \mathbb{R}^{n_{U_{j\mathrm{NC}}}}$ represents the vector of inputs controlled by all non-controllable (NC) agents, and $U_{\mathrm{DA}} \in \mathbb{R}^{n_{U_{j\mathrm{DA}}}}$ represents the vector of inputs for all disabled actuators (DA) that are not in use. The controllable agents are those whose optimal input vector is selected by the AMPC controller. The NC agents are those whose inputs are selected using control logic external to the AMPC controller. The DA agent is used to account for actuators that are not in use. For example, since the reference model includes Q_i inputs for all four wheels while the experimental platform WATonoTruck only has electric motors at the front two corners, the DA agent can be used to disable the inputs Q_{rl} and Q_{rr} when the controller is used with WATonoTruck for experimental testing.

Furthermore, the DA agent can be used to account for actuator faults that have occurred; this is further described in Section 3.6. The advantage of this model formulation is that it allows agent j to consider the contribution of all other agent's actions on the overall system dynamics while it optimizes its own, including those of the NC and DA agents.

The input matrices B_{jd}, $B_{jd\text{OC}}$, $B_{d\text{NC}}$, and $B_{d\text{DA}}$ characterize the effect of the corresponding agents' inputs on the system. These matrices are constructed using elements from the matrix B_d. The columns that comprise the matrix B_d may be denoted as $B_{\delta_f d} \in \mathbb{R}^{n_x}$, $B_{\delta_r d} \in \mathbb{R}^{n_x}$, $B_{Q_{fl} d} \in \mathbb{R}^{n_x}$, $B_{Q_{fr} d} \in \mathbb{R}^{n_x}$, $B_{Q_{rl} d} \in \mathbb{R}^{n_x}$, and $B_{Q_{rr} d} \in \mathbb{R}^{n_x}$, such that

$$B_d = [B_{\delta_f d}, B_{\delta_r d}, B_{Q_{fl} d}, B_{Q_{fr} d}, B_{Q_{rl} d}, B_{Q_{rr} d}]. \tag{3.59}$$

Each column characterizes the effect of the corresponding element of U on the system dynamics. The set of columns that comprise the matrices B_{jd}, $B_{jd\text{OC}}$, $B_{d\text{NC}}$, and $B_{d\text{DA}}$ are the same as the columns that comprise the B_d. For example, if some controllable agent 1 handles $U_1 = [\delta_f, \delta_r]^T$, some controllable agent 2 handles $U_2 = Q_{fl}$, some controllable agent 3 handles $U_3 = Q_{fr}$, a non-controllable agent controls $U_{\text{NC}} = [Q_{rl}, Q_{rr}]^T$, and no actuators are disabled, then input matrices used to construct the prediction model for agent 1 will consist of $B_{1d} = [B_{\delta_f d}, B_{\delta_r d}]$, $B_{1d\text{OC}} = [B_{Q_{fl} d}, B_{Q_{fr} d}]$, $B_{d\text{NC}} = [B_{Q_{rl} d}, B_{Q_{rr} d}]$. In this example, $B_{d\text{DA}}$ is empty as U_{DA} does not exist. Figure 3.8 displays a representation of the system using this example agent configuration.

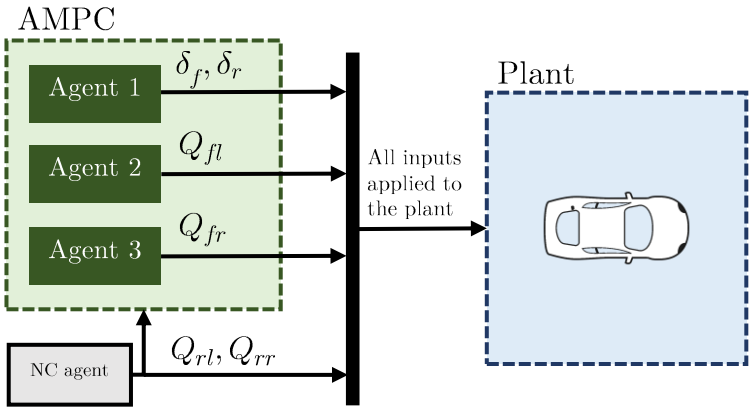


Figure 3.8: Example agent configuration.

Equation (3.58) can be extended so that each agent j can estimate z and z_c. In this case, z and z_c are defined as

$$z(k) = C_z x(k) + D_{zj} U_j(k) + D_{zj\text{OC}} U_{j\text{OC}}(k) + D_{z\text{NC}} U_{\text{NC}}(k) + D_{z\text{DA}} U_{\text{DA}}(k) + W_z, \tag{3.60}$$

$$z_c(k) = C_{z_c}x(k) + D_{z_cj}U_j(k) + D_{z_cj\text{OC}}U_{j\text{OC}}(k) + D_{z_c\text{NC}}U_{\text{NC}}(k) + D_{z_c\text{DA}}U_{\text{DA}}(k) + W_{z_c}. \quad (3.61)$$

Since $z(k) = y(k)$, it follows that $D_{zj} = \mathbf{0}_{n_z \times n_{U_j}}$, $D_{zj\text{OC}} = \mathbf{0}_{n_z \times n_{U_{j\text{OC}}}}$, $D_{z\text{NC}} = \mathbf{0}_{n_z \times n_{U_{\text{NC}}}}$, and $D_{z\text{DA}} = \mathbf{0}_{n_z \times n_{U_{\text{DA}}}}$. Conversely, since $z_c = [y_c^T, y_{z_c}^T]^T$,

$$D_{z_cj} = \begin{bmatrix} \mathbf{0}_{n_{y_c} \times n_{U_j}} \\ B_{y_{z_cj}} \end{bmatrix}, \quad (3.62)$$

$$D_{z_cj\text{OC}} = \begin{bmatrix} \mathbf{0}_{n_{y_c} \times n_{U_{j\text{OC}}}} \\ B_{y_{z_cj\text{OC}}} \end{bmatrix}, \quad (3.63)$$

$$D_{z_c\text{NC}} = \begin{bmatrix} \mathbf{0}_{n_{y_c} \times n_{U_{\text{NC}}}} \\ B_{y_{z_c\text{NC}}} \end{bmatrix}, \quad (3.64)$$

$$D_{z_c\text{DA}} = \begin{bmatrix} \mathbf{0}_{n_{y_c} \times n_{U_{\text{DA}}}} \\ B_{y_{z_c\text{DA}}} \end{bmatrix}, \quad (3.65)$$

where $B_{y_{z_cj}}$, $B_{y_{z_cj\text{OC}}}$, $B_{y_{z_c\text{NC}}}$, and $B_{y_{z_c\text{DA}}}$ are constructed using the columns of $B_{y_{z_c}}$ analogously to how B_{jd}, $B_{jd\text{OC}}$, $B_{d\text{NC}}$, and $B_{d\text{DA}}$ are constructed using the columns of B_d.

3.5 Predicting the system state over a receding horizon

MPC controllers use the system dynamics reference model at each sampling instant throughout runtime to predict the system state over a finite, receding prediction horizon of N_p time steps into the future. These predictions are used to select optimal control actions along an N_c step control horizon, where $N_c \leq N_p$; the control actions are held constant for the remaining $N_p - N_c$ time steps that the state evolution is predicted. In the centralized case, the controlled states are predicted over the horizon using

$$\bar{z} = S_x x + S_U \bar{U} + S_{W_d} \bar{W}_d + S_{W_z}, \quad (3.66)$$

where $\bar{z} = [z(k+1)^T, ..., z(k+N_p)^T]^T \in \mathbb{R}^{n_z N_p}$ represents the predicted states over the horizon, $\bar{U} = [U(k)^T, ..., U(k+N_c-1)^T]^T \in \mathbb{R}^{n_U N_c}$ represents the control actions applied over the control horizon, and $\bar{W}_d = [W_d^T, W_d^T, ..., W_d^T]^T \in \mathbb{R}^{n_x N_p}$. The definitions of the matrices S_x, S_U, S_{W_d}, and S_{W_z} are given in Appendix A. The constrained state $\bar{z_c} = [z_c(k+1)^T, ..., z_c(k+N_p)^T]^T \in \mathbb{R}^{n_z N_p}$ predicted over the horizon is similarly defined as

$$\bar{z_c} = S_{cx} x + S_{cU} \bar{U} + S_{cW_d} \bar{W}_d + S_{cW_z}. \quad (3.67)$$

The definitions of the matrices S_{cx}, S_{cU}, S_{cW_d}, and S_{cW_z} are also give in Appendix A. In the non-centralized case, the model used by agent j to predict the states $\bar{z}$ and $\bar{z_c}$ is

$$\bar{z} = S_x x + S_{U_j}\bar{U}_j + S_{U_{j\text{OC}}}\bar{U}_{j\text{OC}} + S_{U_{\text{NC}}}\bar{U}_{\text{NC}} + S_{U_{\text{DA}}}\bar{U}_{\text{DA}} + S_{W_d}\bar{W}_d + S_{W_z}, \tag{3.68}$$

$$\bar{z}_c = S_{cx} x + S_{cU_j}\bar{U}_j + S_{cU_{j\text{OC}}}\bar{U}_{j\text{OC}} + S_{cU_{\text{NC}}}\bar{U}_{\text{NC}} + S_{cU_{\text{DA}}}\bar{U}_{\text{DA}} + S_{cW_d}\bar{W}_d + S_{cW_z}. \tag{3.69}$$

Again, the definitions of the matrices S_{U_j}, $S_{U_{j\text{OC}}}$, $S_{U_{\text{NC}}}$, $S_{U_{\text{DA}}}$, S_{cU_j}, $S_{cU_{j\text{OC}}}$, $S_{cU_{\text{NC}}}$, and $S_{cU_{\text{DA}}}$ are given in Appendix A.

MPC controllers can also minimize the rate of change of the control inputs when using the prediction model to satisfy all other control objectives. To enable this capability in the centralized case, the relationship

$$\Delta\bar{U} = S_{\Delta UU}\bar{U} + S_{\Delta U} \tag{3.70}$$

is established. Here, $\Delta U(k) = U(k) - U(k-1)$ represents the change in U between sampling instants and $\Delta\bar{U} = [\Delta U(k)^T, ..., \Delta U(k+N_c-1)^T]^T \in \mathbb{R}^{n_U N_c}$ represents the change in U between each sampling instant over the control horizon. In the non-centralized case,

$$\Delta\bar{U}_j = S_{\Delta U_j U_j}\bar{U}_j + S_{\Delta U_j}, \tag{3.71}$$

where $\Delta\bar{U}_j = [\Delta U_j(k)^T, ..., \Delta U_j(k+N_c-1)^T]^T \in \mathbb{R}^{n_{U_j} N_c}$ and $\Delta U_j(k) = U_j(k) - U_j(k-1)$ represents the change in U_j between sampling instants. The definitions of the matrices $S_{\Delta UU}$, $S_{\Delta U}$, $S_{\Delta U_j U_j}$, and $S_{\Delta U_j}$ are given in Appendix A.

3.6 Incorporating hardware faults into the prediction model

One benefit of the AMPC framework is its ability to consider the effects of actuator faults on the dynamics of the vehicle. In this work, three types of faults are considered. Wheel torque faults are the first type; these faults occur when an electric motor driving a wheel fails and no torque can be produced. To accommodate for this fault type, the agent configuration is changed; the input Q_i corresponding to the failed motor, previously handled by a controllable or non-controllable agent, is instead assigned to the DA agent. The value of Q_i is assumed to be zero for the remainder of the runtime. The second type of fault includes cases where the agent selecting one or both of the imaginary center steering angles δ_f and δ_r fails. In this case, it is assumed that the agent controlling the faulted imaginary

center steering angle will continuously output a constant value. To account for this fault type, the imaginary center steering angle is again assigned to the DA agent, and the value of this input is held constant. The third fault type includes cases where a CM's steering actuator fails. As discussed in Section 3.4.2, due to the incorporation of Ackermann geometry within the reference model, the imaginary center steering angles δ_f and δ_r serve as inputs to the model, while the corner steering angles δ_i do not. Because the variables δ_i are not treated as model inputs, the DA agent cannot be used to account for this type of fault. Faults of this type can be considered by incorporating

$$\delta_i(k) = \sigma_i \delta_i(k) + \delta_{i,\text{fault}} \tag{3.72}$$

into the prediction model for all corners. If no fault has occurred, $\sigma_i = 1$ and $\delta_{i,\text{fault}} = 0$ so that $\delta_i(k) = \delta_i(k)$. However, at the sampling instant when a fault occurs, and for the remainder of runtime, $\sigma_i = 0$ and $\delta_{i,\text{fault}} = \delta_i(k-1)$ so that $\delta_i(k) = \delta_i(k-1)$. This ensures that if no fault has occurred, the controller can utilize the reference model to correctly consider the steering angles δ_i that will be applied as a result of the selected δ_f and δ_r, but when a fault at some corner i does occur, the model indicates to the controller that the current steering angle at corner i will be held constant for the remainder of runtime regardless of the values of δ_f and δ_r.

3.7 Preventing model singularities

As presented thus far, an outstanding issue with the model is the possibility for singularities to occur when u approaches zero. This issue results from the division by u in eqs. (3.23) and (3.24). Furthermore, as λ_i, α_i, and therefore $S_{\text{res},i}$ go to zero with u, the same problem arises with eq. (3.26). To prevent this issue, eqs. (3.24) and (3.26) may be augmented as

$$\alpha_i = \delta_i - \arctan\left(\frac{v + a_i r}{u + \epsilon_\alpha}\right), \tag{3.73}$$

$$\mu_{\text{B},yi} = \frac{\alpha_i}{S_{\text{res},i} + \epsilon_{\mu_y}}[C_{1y}(1 - e^{-C_{2y}S_{\text{res},i}}) - C_{3y}S_{\text{res},i}], \tag{3.74}$$

where ϵ_α and ϵ_{μ_y} are small scalar values that prevent divisions by 0. Similarly, λ_i may be bounded by

$$|\lambda_i| \geq \epsilon_\lambda \tag{3.75}$$

where ϵ_λ is a small scalar. Rather than including ϵ_λ in the denominator of eq. (3.23), it is acceptable to bound λ_i in this way because, as mentioned in Section 3.4.1 and Appendix A,

eq. (3.23) is not included directly within the continuous reference model f_{SS}; λ_i is estimated before the reference model is constructed at each sampling instant, and is held constant over the prediction horizon.

To improve model accuracy, it is desirable for ϵ_λ, ϵ_α, and ϵ_{μ_y} to be large when the vehicle is travelling at low speeds, and much smaller when the vehicle is travelling at higher speeds. To achieve this, the values of ϵ_λ, ϵ_α, and ϵ_{μ_y} are varied with u using an arctangent relationship. The relationship between these quantities and u, displayed in Figure 3.9, is represented by the equation

$$\epsilon_\ell = \frac{-2}{\pi}\frac{\epsilon_{\ell,\text{low}} - \epsilon_{\ell,\text{high}}}{2}\arctan\left(\frac{1}{\text{scale}}(u - u_{\text{threshold}})\right) + \frac{\epsilon_{\ell,\text{low}} + \epsilon_{\ell,\text{high}}}{2} \;\forall \ell \in \{\lambda, \alpha, \mu_y\}, \tag{3.76}$$

where $\epsilon_{\ell,\text{low}}$ represents the value of ϵ_ℓ as $u \to 0$, $\epsilon_{\ell,\text{high}}$ represents the value of ϵ_ℓ as $u \to \infty$, scale is a scale factor that determines the slope of the linear transition region, and $u_{\text{threshold}}$ is a threshold that determines the location of the transition region. The parameters values used in this work are given in Table 3.2.

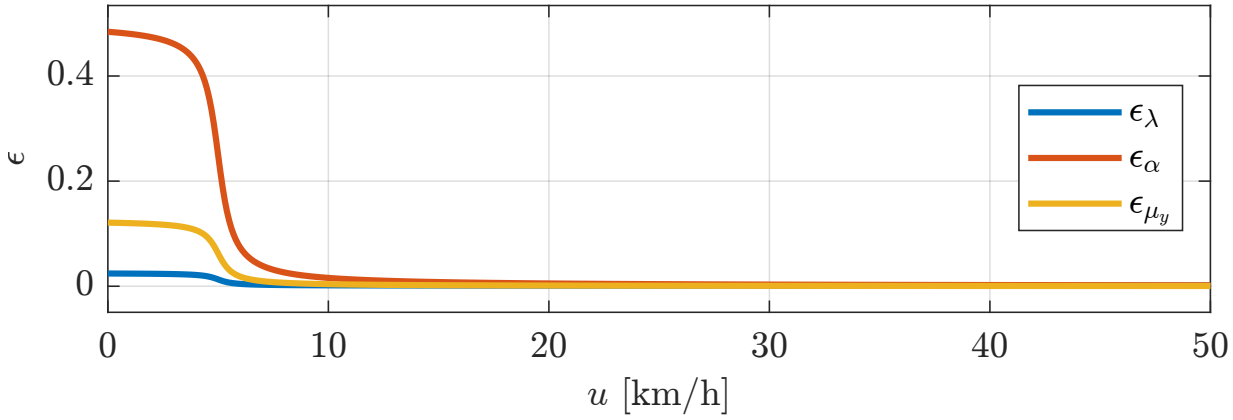


Figure 3.9: ϵ_λ, ϵ_α, and ϵ_{μ_y} plotted as a function of u.

	Value		
Parameter	ϵ_λ	ϵ_α	ϵ_{μ_y}
$\epsilon_{\ell,\text{low}}$	0.025	0.5	0.125
$\epsilon_{\ell,\text{high}}$	0.001	0	0
scale	0.5 km/h	0.5 km/h	0.5 km/h
$u_{\text{threshold}}$	5 km/h	5 km/h	5 km/h

Table 3.2: Parameters used to prevent model singularities.

Chapter 4

Development of a vehicle path tracking controller using agent-based model predictive control

4.1 Constraints

4.1.1 State constraints

The optimal controller presented in this work must achieve the control objective while adhering to a set of constraints on the states, inputs, and input rates. As discussed in Subsection 3.4.1, the states to be constrained are $z_c = [r, \alpha_{fl}, \alpha_{fr}, \alpha_{rl}, \alpha_{rr}]^T$. The desired bounds on z_c are given as

$$z_{c,\min}(k) = [r_{\min}(k), \alpha_{fl,\min}(k), \alpha_{fr,\min}(k), \alpha_{rl,\min}(k), \alpha_{rr,\min}(k)]^T, \tag{4.1}$$

$$z_{c,\max}(k) = [r_{\max}(k), \alpha_{fl,\max}(k), \alpha_{fr,\max}(k), \alpha_{rl,\max}(k), \alpha_{rr,\max}(k)]^T, \tag{4.2}$$

where $z_{c,\min}(k)$ and $z_{c,\max}(k)$ represent the desired minimum and maximum bounds, respectively, on the controlled states z_c. Soft constraints of the form

$$\bar{z}_c(k) \geq \bar{z}_{c,\min}(k) - \bar{\epsilon}_{\min}, \tag{4.3}$$

$$\bar{z}_c(k) \leq \bar{z}_{c,\max}(k) + \bar{\epsilon}_{\max}, \tag{4.4}$$

are used to enforce the desired constraints over the prediction horizon while preventing optimization infeasibility. Here, $\bar{z}_{c,\min}(k) = [z_{c,\min}^T(k), ..., z_{c,\min}^T(k + N_p)]^T$ and $\bar{z}_{c,\max}(k) =$

$[z_{c,\max}^T(k), ..., z_{c,\max}^T(k+N_p)]^T$ represent the constraints applied over the prediction horizon. The slack variable vectors $\bar{\epsilon}_{\min} \in \mathbb{R}^{n_{\bar{\epsilon}_{\min}}}$ and $\bar{\epsilon}_{\max} \in \mathbb{R}^{n_{\bar{\epsilon}_{\max}}}$ are defined as

$$\bar{\epsilon}_{\min} = [\epsilon_{\min}^T, ..., \epsilon_{\min}^T]^T, \tag{4.5}$$

$$\bar{\epsilon}_{\max} = [\epsilon_{\max}^T, ..., \epsilon_{\max}^T]^T, \tag{4.6}$$

$$\epsilon_{\min} = [\epsilon_{r_{\min}}, \epsilon_{\alpha_{fl,\min}}, \epsilon_{\alpha_{fr,\min}}, \epsilon_{\alpha_{rl,\min}}, \epsilon_{\alpha_{rr,\min}}]^T, \tag{4.7}$$

$$\epsilon_{\max} = [\epsilon_{r_{\max}}, \epsilon_{\alpha_{fl,\max}}, \epsilon_{\alpha_{fr,\max}}, \epsilon_{\alpha_{rl,\max}}, \epsilon_{\alpha_{rr,\max}}]^T, \tag{4.8}$$

where $\epsilon_{\min} \in \mathbb{R}^{n_{\epsilon_{\min}}}$ and $\epsilon_{\max} \in \mathbb{R}^{n_{\epsilon_{\max}}}$ are used to enforce the soft constraints at each sampling instant along the prediction horizon; $\epsilon_{r_{\min}}$, $\epsilon_{\alpha_{fl,\min}}$, $\epsilon_{\alpha_{fr,\min}}$, $\epsilon_{\alpha_{rl,\min}}$, $\epsilon_{\alpha_{rr,\min}}$, $\epsilon_{r_{\max}}$, $\epsilon_{\alpha_{fl,\max}}$, $\epsilon_{\alpha_{fr,\max}}$, $\epsilon_{\alpha_{rl,\max}}$, and $\epsilon_{\alpha_{rr,\max}}$ are the individual scalar slack variables used to relax the constraints on individual states; and $n_{\bar{\epsilon}_{\min}} = n_{\bar{\epsilon}_{\max}} = N_p n_{z_c}$. In total, there are $n_{\bar{\epsilon}} = n_{\bar{\epsilon}_{\min}} + n_{\bar{\epsilon}_{\max}}$ slack variables. The bounds on r are

$$r_{\min} = \frac{-\mu g}{u}, \tag{4.9}$$

$$r_{\max} = \frac{\mu g}{u}, \tag{4.10}$$

where μ is the TRFC. At each sampling instant, these bounds calculated using the current u and μ and are held constant over the prediction horizon. The TRFC at each wheel i is determined using the longitudinal TRFC μ_{xi} and lateral TRFC μ_{yi} at wheel i as [51]

$$\mu_i = \frac{\mu_{xi}\mu_{yi}}{\sqrt{\mu_{xi}^2 \sin^2(\alpha_i) + \mu_{yi}^2 \cos^2(\alpha_i)}}. \tag{4.11}$$

In this work, μ is then taken to be the average of all μ_i. The bounds on α_i correspond to the sideslip angle that results in the maximum F_{yi}. For standard passenger AV applications, it is common to establish a relationship between α_i and the vehicle body sideslip angle, β, and use this relationship to limit β such that the bounds on α_i are not violated. However, the standard relationship between α_i and β for passenger vehicles does not apply to WATonoTruck, since all wheels are independently steered. For example, there may be a scenario where the steering angle for all wheels is set to the same, non-zero value; in this case, β would not be zero as the heading of the vehicle would not align with the direction of its velocity vector, but all α_i would be approximately zero. Therefore, for this application, all α_i are constrained instead of β.

4.1.2 Input and input rate constraints

It is also necessary to constraint both the inputs and input rates. In the centralized case, these constraints are of the form

$$\bar{U}(k) \geq \bar{U}_{\min}(k), \tag{4.12}$$

$$\bar{U}(k) \leq \bar{U}_{\max}(k), \tag{4.13}$$

$$\Delta\bar{U}(k) \geq \Delta\bar{U}_{\min}(k), \tag{4.14}$$

$$\Delta\bar{U}(k) \leq \Delta\bar{U}_{\max}(k), \tag{4.15}$$

where $\Delta\bar{U}$ is the difference in $\bar{U}$ between sampling instants and $\bar{U}_{\min}$, $\bar{U}_{\max}$, $\Delta\bar{U}_{\min}$, and $\Delta\bar{U}_{\max}$, represent the bounds on the inputs and input rates over the control horizon. These constraint vectors are defined as

$$\bar{U}_{\min}(k) = [U_{\min}(k)^T, ..., U_{\min}(k + N_c - 1)^T]^T, \tag{4.16}$$

$$\bar{U}_{\max}(k) = [U_{\max}(k)^T, ..., U_{\max}(k + N_c - 1)^T]^T, \tag{4.17}$$

$$\Delta\bar{U}_{\min}(k) = [\Delta U_{\min}(k)^T, ..., \Delta U_{\min}(k + N_c - 1)^T]^T, \tag{4.18}$$

$$\Delta\bar{U}_{\max}(k) = [\Delta U_{\max}(k)^T, ..., \Delta U_{\max}(k + N_c - 1)^T]^T, \tag{4.19}$$

$$U_{\min} = [\delta_{f,\min}, \delta_{r,\min}, Q_{fl,\min}, Q_{fr,\min}, Q_{rl,\min}, Q_{rr,\min}]^T, \tag{4.20}$$

$$U_{\max} = [\delta_{f,\max}, \delta_{r,\max}, Q_{fl,\max}, Q_{fr,\max}, Q_{rl,\max}, Q_{rr,\max}]^T, \tag{4.21}$$

$$\Delta U_{\min} = [\Delta\delta_{f,\min}, \Delta\delta_{r,\min}, \Delta Q_{fl,\min}, \Delta Q_{fr,\min}, \Delta Q_{rl,\min}, \Delta Q_{rr,\min}]^T, \tag{4.22}$$

$$\Delta U_{\max} = [\Delta\delta_{f,\max}, \Delta\delta_{r,\max}, \Delta Q_{fl,\max}, \Delta Q_{fr,\max}, \Delta Q_{rl,\max}, \Delta Q_{rr,\max}]^T. \tag{4.23}$$

In the non-centralized case, the constraints on the actions for some agent j are of the form

$$\bar{U}_j \geq \bar{U}_{j,\min}, \tag{4.24}$$

$$\bar{U}_j \leq \bar{U}_{j,\max}, \tag{4.25}$$

$$\Delta\bar{U}_j \geq \Delta\bar{U}_{j,\min}, \tag{4.26}$$

$$\Delta\bar{U}_j \leq \Delta\bar{U}_{j,\max}, \tag{4.27}$$

where $\Delta\bar{U}_j$ is the rate of change of $\bar{U}_j$ and $\bar{U}_{j,\min}$, $\bar{U}_{j,\max}$, $\Delta\bar{U}_{j,\min}$, and $\Delta\bar{U}_{j,\max}$, represent the bounds on the inputs and input rates for agent j. These constraint vectors are defined as

$$\bar{U}_{j,\min}(k) = [U_{j,\min}(k)^T, ..., U_{j,\min}(k + N_c - 1)^T]^T, \tag{4.28}$$

$$\bar{U}_{j,\max}(k) = [U_{j,\max}(k)^T, ..., U_{j,\max}(k+N_c-1)^T]^T, \tag{4.29}$$

$$\Delta\bar{U}_{j,\min}(k) = [\Delta U_{j,\min}(k)^T, ..., \Delta U_{j,\min}(k+N_c-1)^T]^T, \tag{4.30}$$

$$\Delta\bar{U}_{j,\max}(k) = [\Delta U_{j,\max}(k)^T, ..., \Delta U_{j,textmax}(k+N_c-1)^T]^T. \tag{4.31}$$

The elements of $U_{j,\min}$, $U_{j,\max}$, $\Delta U_{j,\min}$, and $\Delta U_{j,\max}$, are dependent on the inputs selected by agent j.

The bounds on δ_f and δ_r are

$$\delta_{f,\min} = \arctan\left(\left(\frac{1}{\tan(\delta_{\min})} - \frac{T}{2w} + \frac{T\tan(\delta_{\max})}{2w\tan(\delta_{\min})}\right)^{-1}\right), \tag{4.32}$$

$$\delta_{f,\max} = \arctan\left(\left(\frac{1}{\tan(\delta_{\max})} + \frac{T}{2w} - \frac{T\tan(\delta_{\min})}{2w\tan(\delta_{\max})}\right)^{-1}\right), \tag{4.33}$$

$$\delta_{r,\min} = \arctan\left(\frac{\tan(\delta_{\min})\tan(\delta_{f,\max})}{\tan(\delta_{\max})}\right), \tag{4.34}$$

$$\delta_{r,\max} = \arctan\left(\frac{\tan(\delta_{\max})\tan(\delta_{f,\min})}{\tan(\delta_{\min})}\right), \tag{4.35}$$

where $\delta_{\min}$ and $\delta_{\max}$ represent the steering actuator limits at each wheel. So long as $\delta_{f,\min} \leq \delta_f \leq \delta_{f,\max}$ and $\delta_{r,\min} \leq \delta_r \leq \delta_{r,\max}$, these constraints ensure that the Ackermann geometry is maintained without violating the actuator limits for any combination of δ_f and δ_r. These constraints were derived using eqs. (3.17) to (3.20). The rate constraints on the steering rates $\Delta\delta_f$ and $\Delta\delta_r$ are the actuator steering rate limits $\Delta\delta_{\min}$ and $\Delta\delta_{\max}$. The bounds on Q_i are

$$Q_{i,\min} = \max(Q_{\min,\text{act}}, Q_{\min,f_{xi}}), \tag{4.36}$$

$$Q_{i,\max} = \min(Q_{\max,\text{act}}, Q_{\max,f_{xi}}), \tag{4.37}$$

where $Q_{\min,\text{act}}$ and $Q_{\max,\text{act}}$ represent the actuator torque limits and $Q_{\min,f_{xi}}$ and $Q_{\max,f_{xi}}$ represent the longitudinal force limits. $Q_{\min,f_{xi}}$ and $Q_{\max,f_{xi}}$ are given as

$$Q_{\min,f_{xi}} = -f_{xi,\max}R_{\text{eff},i}, \tag{4.38}$$

$$Q_{\max,f_{xi}} = f_{xi,\max}R_{\text{eff},i}, \tag{4.39}$$

where the maximum tire force $f_{xi,\max}$ is computed using the friction ellipse,

$$f_{xi,\max} = \mu_{xi}\sqrt{F_{zi}^2 - (f_{yi}/\mu_y)^2}. \tag{4.40}$$

The rate constraints on the wheel torque rate ΔQ_i are set according to the motor limits.

To reduce the controller optimization complexity and increase the stability of the vehicle, a relationship between δ_f and δ_r of the form

$$\delta_r = k_\delta \delta_f \tag{4.41}$$

is established. The coefficient k_δ is adjusted according to the vehicle velocity u to improve both mauenverability at low speeds and stability at high speeds. The vehicle's yaw rate gain, which describes the relationship between the steering angle and yaw rate, can be used to determine the effect of k_δ on stability. Using a bicycle model with both front and rear wheel steering, the yaw rate gain is defined as [52]

$$G = \frac{r}{\delta_f} = \frac{u}{w + K_{us} u^2} \left(1 - \frac{\delta_r}{\delta_f}\right), \tag{4.42}$$

$$K_{us} = \frac{m}{w} \left(\frac{b}{C_f} - \frac{a}{C_r}\right), \tag{4.43}$$

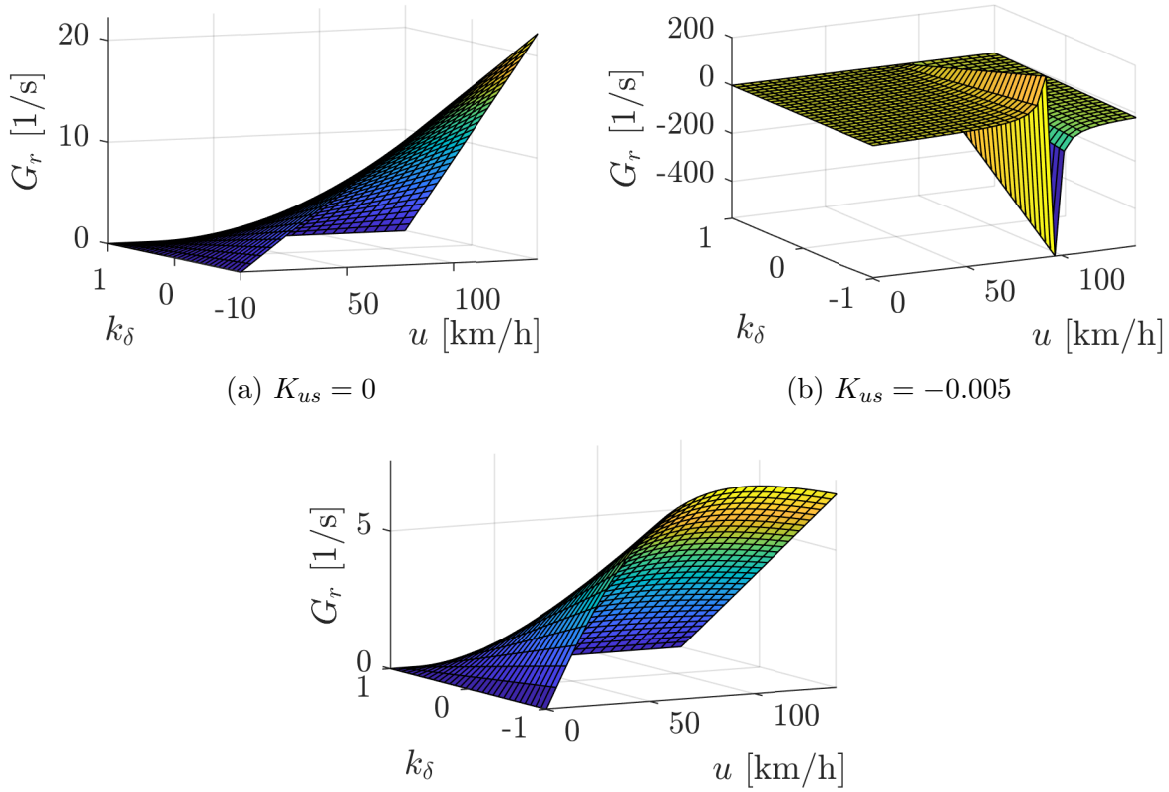
where G is the yaw rate gain, K_{us} is the understeer gradient, and C_f and C_r are respectively the front and rear cornering stiffnesses. Given the relationship eq. (4.41),

$$G = \frac{r}{\delta_f} = \frac{u}{w + K_{us} u^2}(1 - k_\delta). \tag{4.44}$$

Equation (4.44) demonstrates that the yaw rate gain, and therefore the stability of the vehicle, is influenced by K_{us}, u, and k_δ. The relationship between these quantities is given in Figure 4.1. $K_{us} > 0$, $K_{us} < 0$, and $K_{us} = 0$ correspond to oversteer, understeer, and neutral steer vehicles, respectively. For the purposes of this analysis, it is assumed that all tires of WATonoTruck have the same properties and are subjected to the same normal load; thus, $C_f = C_r$. Furthermore, for WATonoTruck, $a = b$. These properties indicate that WATonoTruck is a neutral steer vehicle. Therefore, to reduce the yaw rate gain and improve stability at high speeds, k_δ should be reduced. Conversely, k_δ can be increased at low speeds to improve the vehicle maneuverability. Figure 4.2 displays the relationship between k_δ and u that is used in this work. Note that this constraint may only be imposed if both δ_f and δ_r are controlled by the same agent, because optimal control performance will not be provided if there are coupled input constraints between agents [53].

4.2 Model predictive control formulation

This section is used to describe the formulation of standard, centralized MPC prior to the presentation of AMPC; the AMPC algorithm, an extension of MPC, is outlined in the



(a) $K_{us} = 0$

(b) $K_{us} = -0.005$

(c) $K_{us} = 0.005$

Figure 4.1: Relationship between yaw rate gain, u, and k_δ, in (a) neutral steer, (b) oversteer, and (c) understeer cases.

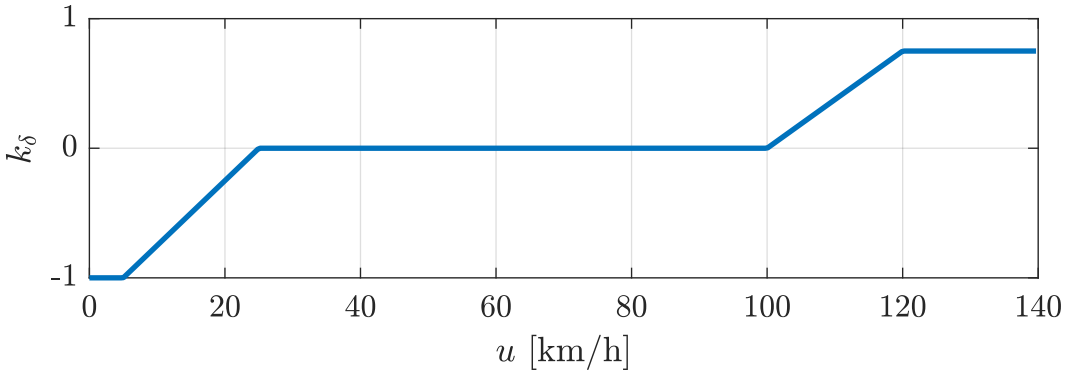


Figure 4.2: Relationship between k_δ and u used in this thesis.

next section. The objective of an MPC controller is to minimize the deviation between the system state and reference trajectories while also minimizing the control input effort, control input rate of change, and violation of the soft constraints. To achieve this, at each sampling instant k, the controller selects an optimal sequence of input vectors $\bar{U}$ that, when applied to the system over a receding prediction horizon of N_p time steps into the future, is predicted to best satisfy the control objectives over the horizon. The optimal inputs may vary over the control horizon, i.e., the first N_c steps along the prediction horizon, where $N_c \leq N_p$; the inputs are held constant over the remainder of the prediction horizon. At each time step k, The first input vector $U(k)$ in the sequence $\bar{U}(k)$ is applied to the system, while the remainder are discarded and the optimization procedure is performed again at the following sampling instant. The set of control objectives are described by the objective function

$$\begin{aligned} J = \min_{\bar{U},\bar{\epsilon}_{\min},\bar{\epsilon}_{\max}} \sum_{\ell=1}^{N_p} \Big[&\|z(k+\ell) - z_{\text{ref}}(k+\ell)\|_Q^2 + \|U(k+\ell-1)\|_R^2 \\ &+ \|\Delta U(k+\ell)\|_P^2 + \|\epsilon_{\min}(k+\ell)\|_{Q_\epsilon}^2 + \|\epsilon_{\max}(k+\ell)\|_{Q_\epsilon}^2 \Big], \end{aligned} \tag{4.45}$$

where z_{ref} is the reference trajectory to be tracked by the controller, and $Q \in \mathbb{R}^{n_z \times n_z}$, $R \in \mathbb{R}^{n_U \times n_U}$, $P \in \mathbb{R}^{n_U \times n_U}$, and $Q_\epsilon \in \mathbb{R}^{n_{\epsilon_{\min}} \times n_{\epsilon_{\min}}}$ are positive definite diagonal weight matrices whose diagonal elements are used to weigh the relative importance of each objective. In this work, to enable path tracking and longitudinal velocity control objectives, the reference trajectory over the prediction horizon $\bar{z}_{\text{ref}}(k)$ at time step k is defined as

$$\bar{z}_{\text{ref}}(k) = [z_{\text{ref}}(k+1)^T, ..., z_{\text{ref}}(k+N_p)^T]^T, \tag{4.46}$$

$$z_{\text{ref}}(k+\ell) = [u_{\text{ref}}(k+\ell), e_{\text{ref}}(k+\ell), \Delta\psi_{\text{ref}}(k+\ell)]^T, \tag{4.47}$$

where u_{ref} is the reference longitudinal velocity, e_{ref} is the reference lateral error, $\Delta\psi_{\text{ref}}$ is the reference heading error, and z_{ref} is the controlled state reference vector at a single sampling instant. It is assumed that u_{ref} is provided to the controller by an upstream trajectory planner. To achieve the path tracking objectives, $e_{\text{ref}} = \Delta\psi_{\text{ref}} = 0$ for all time. The matrices Q, R, P, and Q_ϵ are defined as

$$Q = \begin{bmatrix} Q_u & 0 & 0 \\ 0 & Q_e & 0 \\ 0 & 0 & Q_{\Delta\psi} \end{bmatrix}, \tag{4.48}$$

$$R = \begin{bmatrix} R_{\delta_f} & 0 & \dots & 0 \\ 0 & R_{\delta_r} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R_{Q_{rr}} \end{bmatrix}, \tag{4.49}$$

$$P = \begin{bmatrix} P_{\Delta\delta_f} & 0 & \dots & 0 \\ 0 & P_{\Delta\delta_r} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & P_{\Delta Q_{rr}} \end{bmatrix}, \tag{4.50}$$

$$Q_\epsilon = \begin{bmatrix} Q_{\epsilon_r} & 0 & \dots & 0 \\ 0 & Q_{\epsilon_{\alpha_{fl}}} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & Q_{\epsilon_{\alpha_{fr}}} \end{bmatrix}. \tag{4.51}$$

The diagonal elements of each matrix Q, R, P, and Q_ϵ are scalar weight values used to establish the relative importance of the set of all objectives characterized by J.

Since the reference model presented in Chapter 3 is linear, the cost function J is quadratic and can be rewritten in quadratic programming (QP) form as

$$J = \min_{\bar{U}_\epsilon} \frac{1}{2} \bar{U}_\epsilon^T H_\epsilon \bar{U}_\epsilon + f_\epsilon^T \bar{U}_\epsilon. \tag{4.52}$$

The MATLAB function `quadprog` is used to solve the QP problem; the objective function form given in eq. (4.52) is the form accepted by `quadprog`. With J in this form, a convex optimization may be performed by the MPC controller at each time step to determine the optimal control input sequence. Here, $\bar{U}_\epsilon = [\bar{U}^T, \bar{\epsilon}_{\min}^T, \bar{\epsilon}_{\max}^T]^T \in \mathbb{R}^{n_{\bar{U}_\epsilon}}$ is an augmented version of the optimal sequence of input vectors that also includes the slack variables. The Hessian matrix H_ϵ and gradient vector f_ϵ are defined as

$$H_\epsilon = \begin{bmatrix} H & \mathbf{0} \\ \mathbf{0} & \bar{Q}_\epsilon \end{bmatrix}, \tag{4.53}$$

$$f_\epsilon = \begin{bmatrix} f \\ \mathbf{0} \end{bmatrix}, \tag{4.54}$$

$$H = 2(S_U^T \bar{Q} S_U + \bar{R} + S_{\Delta UU}^T \bar{P} S_{\Delta UU}), \tag{4.55}$$

$$f = 2 S_U^T \bar{Q} E + 2 S_{\Delta U}^T \bar{P} S_{\Delta UU}, \tag{4.56}$$

$$E = S_x x(k) + S_{W_d} \bar{W}_d + S_{W_z} - \bar{z}_{\text{ref}}. \tag{4.57}$$

Here, H and f represent the Hessian matrix and gradient vector, respectively, that would be used to construct the QP form of the cost function J if hard rather than soft constraints were used, and therefore the slack variables were not. E represents the predicted reference tracking error over the prediction horizon if all control inputs were zero. $\bar{z}_{\text{ref}}$ is the reference

trajectory over the prediction horizon. $\bar{Q}$ and $\bar{Q}_\epsilon$ represent block diagonal matrices with N_p row and column partitions; Q and Q_ϵ serve as the blocks along their diagonals, respectively. $\bar{R}$ and $\bar{P}$ are block diagonal matrices with N_c row and column partitions; the blocks along their diagonals are R and P.

The MATLAB function `quadprog` accepts constraints of the form

$$A_{\text{QP}}\bar{U}_\epsilon \leq b_{\text{QP}}, \tag{4.58}$$

$$A_{\text{QPE}}\bar{U}_\epsilon = b_{\text{QPE}}, \tag{4.59}$$

$$\text{lb} \leq \bar{U}_\epsilon \leq \text{ub}, \tag{4.60}$$

where A_{QP} and b_{QP} are used to impose inequality constraints, and A_{QPE} and b_{QPE} are used to impose equality constraints. In this work, the inequality constraints are used to enforce constraints on the input rates and soft constraints on the constrained states, while the equality constraints are used to impose the relationship between the steering angles δ_f and δ_r given in eq. (4.41). The matrices A_{QP}, b_{QP}, A_{QPE}, and b_{QPE} are defined in Appendix B. lb and ub are simply used to enforce the bounds on inputs $\bar{U}$ given in eqs. (4.12) and (4.13) and the bounds on the slack variables $\bar{\epsilon}_{\min}$ and $\bar{\epsilon}_{\max}$. All elements of $\bar{\epsilon}_{\min}$ and $\bar{\epsilon}_{\max}$ must be greater than zero, but can be arbitrarily large.

4.3 Agent-based model predictive control formulation

The formulation of AMPC is derived from centralized MPC. However, in the agent-based case, rather than utilizing a single, centralized controller to select all optimal control inputs for the system, each agent uses its own MPC controller to select its optimal control inputs at each sampling instant. Furthermore, rather than performing a single optimization at each sampling instant, each agent uses their MPC controller to iteratively optimize their control actions in collaboration with all others; the agents act cooperatively to achieve the control objectives described in Section 4.2. The objective function that is used by all agents at each iteration of the cooperative optimization procedure is of the form

$$\begin{aligned} J_j = \min_{\bar{U}_j, \bar{\epsilon}_{\min}, \bar{\epsilon}_{\max}} \sum_{\ell=1}^{N_p} \Big[&\|z(k+\ell) - z_{\text{ref}}(k+\ell)\|_Q^2 + \|U_j(k+\ell-1)\|_{R_j}^2 \\ &+ \|\Delta U_j(k+\ell)\|_{P_j}^2 + \|\epsilon_{\min}(k+\ell)\|_{Q_\epsilon}^2 + \|\epsilon_{\max}(k+\ell)\|_{Q_\epsilon}^2 \Big], \end{aligned} \tag{4.61}$$

where R_j and P_j are diagonal matrices whose diagonal elements are those from R and P that correspond to the inputs U_j and input changes ΔU_j. The objective function can be rewritten in QP form as

$$J_j = \min_{\bar{U}_{j\epsilon}} \frac{1}{2} \bar{U}_{j\epsilon}^T H_{j\epsilon} \bar{U}_{j\epsilon} + f_{j\epsilon}^T \bar{U}_{j\epsilon}. \tag{4.62}$$

Like the centralized case, this objective function is convex and will result in a globally optimal solution. Here, $\bar{U}_{j\epsilon} = [\bar{U}_j^T, \bar{\epsilon}_{\min}^T, \bar{\epsilon}_{\max}^T]^T \in \mathbb{R}^{n_{\bar{U}_{j\epsilon}}}$ is an augmented version of the optimal sequence of input vectors for agent j that also includes the slack variables. The Hessian matrix $H_{j\epsilon}$ and gradient vector $f_{j\epsilon}$ are defined as

$$H_{j\epsilon} = \begin{bmatrix} H_j & \mathbf{0} \\ \mathbf{0} & \bar{Q}_\epsilon \end{bmatrix}, \tag{4.63}$$

$$f_{j\epsilon} = \begin{bmatrix} f_j \\ \mathbf{0} \end{bmatrix}, \tag{4.64}$$

$$H_j = 2(S_{U_j}^T \bar{Q} S_{U_j} + \bar{R}_j + S_{\Delta U_j U_j}^T \bar{P}_j S_{\Delta U_j U_j}), \tag{4.65}$$

$$f_j = 2 S_{U_j}^T \bar{Q} E_j + 2 S_{\Delta U_j}^T \bar{P}_j S_{\Delta U_j U_j}, \tag{4.66}$$

$$E_j = S_x x(k) + S_{U_{j\mathrm{OC}}} \bar{U}_{j\mathrm{OC}} + S_{U_{\mathrm{NC}}} \bar{U}_{\mathrm{NC}} + S_{U_{\mathrm{DA}}} \bar{U}_{\mathrm{DA}} + S_{W_d} \bar{W}_d + S_{W_z} - \bar{z}_{\mathrm{ref}}. \tag{4.67}$$

The interpretation of H_j, f_j, and E_j are analagous to H, f, and E used in the centralized case. $\bar{R}_j$ and $\bar{P}_j$ are block diagonal matrices with N_c row and column partitions whose blocks on the diagonal are all R_j and P_j, respectively.

At each sampling instant k, each agent uses its prediction model, i.e., eq. (3.68), with inputs $\bar{U}_j$ to formulate the objective function given in eq. (4.62). To achieve this, they use the current $\bar{U}_{\mathrm{NC}}(k)$ and $\bar{U}_{\mathrm{DA}}(k)$ provided by the NC and DA agents and the inputs of the other controllable agents from the previous time step $\bar{U}_{j\mathrm{OC}}(k-1)$; it is initially assumed that $\bar{U}_{j\mathrm{OC}}(k)$ will be approximately equal to $\bar{U}_{j\mathrm{OC}}(k-1)$. The resulting J_j is used to determine a tentative optimal $\bar{U}_j(k)$. Agent j broadcasts this $\bar{U}_j(k)$ to all other controllable agents $p \neq j$ and receives a tentative optimal $\bar{U}_p(k)$ from all other controllable agents. These $\bar{U}_p(k)$ $\forall p$ are used to construct a new estimation of $\bar{U}_{j\mathrm{OC}}(k)$ that is used by agent j to update its objective function J_j. This procedure is then repeated iteratively by all agents until each converge on an optimal $\bar{U}_j(k)$. This algorithm, which can be interpreted as a cooperative game played by all controllable agents and is known as the cooperative control routine (CCR) [15,43] is summarized in Algorithm 1. In this algorithm, c denotes the current CCR iteration, $c_{\max}$ the maximum number of iterations, $\bar{U}_j^c(k)$ the tentative $\bar{U}_j(k)$ for agent j determined using J_j at iteration c, $\bar{U}_j^*(k)$ the converged $\bar{U}_j(k)$ to

be applied by agent j at time k, α_{CCR} the CCR update rate, and $\epsilon_{\text{CCR}j} \in \mathbb{R}^{N_c n_{U_j}}$ the CCR convergence threshold vector. The elements of $\epsilon_{\text{CCR}j}$ are threshold values corresponding to each input in $\bar{U}_j$ that define the amount of change in $\bar{U}_j^c(k)$ between iterations that indicates convergence. The elements that correspond to steering inputs are denoted $\epsilon_{\text{CCR}\delta}$ and those that correspond to torque inputs are denoted $\epsilon_{\text{CCR}Q}$.

Algorithm 1: Cooperative control routine.

At each time step k, receive state $x(k)$ measurements;
$c \leftarrow 1$; `// Initialize CCR iteration counter`
$\bar{U}_j^{c-1}(k) \leftarrow \bar{U}_j^*(k-1)\ \forall j = \{1, ..., n_a\}$; `// Initialize inputs`
while any $|\bar{U}_j^c(k) - \bar{U}_j^{c-1}(k)| \geq \epsilon_{\text{CCR}j}\ \forall i$ **do** `// Loop until convergence`
  for all controllable agents j whose actions have not converged **do**
    Compute $\bar{U}_j(k)$ using eq. (4.62), $\bar{U}_p^{c-1}(k)\ \forall p \neq j$, $\bar{U}_{\text{NC}}$, and $\bar{U}_{\text{DA}}$;
    $\bar{U}_j^c(k) \leftarrow \alpha_{\text{CCR}}\bar{U}_j^{c-1}(k) + (1 - \alpha_{\text{CCR}})\bar{U}_j(k)$; `// Update inputs`
    Broadcast $\bar{U}_j^c(k)$, receive $\bar{U}_p^c(k)\ \forall p \neq j$;
  end
  if $c == c_{\max}$ **then**
    break; `// Enforce maximum number of while loop iterations`
  else
    $c \leftarrow c + 1$; `// Update iteration counter`
  end
end
$\bar{U}_j^*(k) \leftarrow \bar{U}_j^c(k)$; `// Agent j's optimal control input`
Apply the first input in the sequence $\bar{U}_j^*(k)\ \forall j$;
Repeat at next time step $k \leftarrow k + 1$;

When optimizing their control inputs, each agent adheres to the set of constraints on the states, inputs, and input rates considered in Section 4.2. In the form accepted by MATLAB's `quadprog`, these constraints are represented as

$$A_{\text{QP}j}\bar{U}_{j\epsilon} \leq b_{\text{QP}j}, \tag{4.68}$$

$$A_{\text{QPE}j}\bar{U}_{j\epsilon} = b_{\text{QPE}j}, \tag{4.69}$$

$$\text{lb}_j \leq \bar{U}_{j\epsilon} \leq \text{ub}_j. \tag{4.70}$$

Similar to the centralized case, $A_{\text{QP}j}$ and $b_{\text{QP}j}$ are used to impose inequality constraints and $A_{\text{QPE}j}$ and $b_{\text{QPE}j}$ are used to impose equality constraints In this work, the inequality

constraints are used to enforce constraints on the input rates and soft constraints on the constrained states, while the equality constraints are used to impose the relationship between the steering angles δ_f and δ_r given in eq. (4.41). The matrices $A_{\mathrm{QP}j}$, $b_{\mathrm{QP}j}$, $A_{\mathrm{QPE}j}$, and $b_{\mathrm{QPE}j}$ are defined in Appendix B. lb_j and ub_j are simply used to enforce the bounds on inputs $\bar{U}_j$ given in eqs. (4.24) and (4.25) and the bounds on the slack variables $\bar{\epsilon}_{\min}$ and $\bar{\epsilon}_{\max}$. All elements of $\bar{\epsilon}_{\min}$ and $\bar{\epsilon}_{\max}$ must be greater than zero, but can be arbitrarily large.

4.4 Longitudinal velocity controller

To further reduce the complexity of the AMPC optimization problem, an external NC agent can be used to perform longitudinal velocity control. When in use, this agent selects all Q_i inputs not assigned to the DA agent in place of the AMPC controller; in this case, the AMPC controller is only responsible for the path tracking objectives. A proportional (P) controller with the transfer function

$$H = K_p \tag{4.71}$$

is used by this agent. The error signal $u_{\mathrm{ref}} - u$ is used to select the value of all non-controllable Q_i inputs. Here, K_p is the proportional gain. A block diagram of the NC longitudinal control agent is given in Figure 4.3.

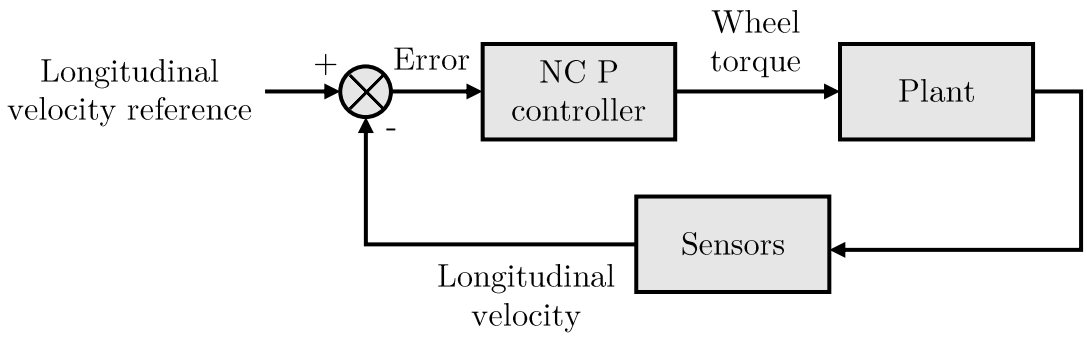


Figure 4.3: NC longitudinal control agent block diagram.

When the NC longitudinal control agent is in use, then the matrix C_y used to predict the controlled output z may be redefined as

$$C_y = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.72}$$

since the AMPC controller is no longer responsible for the longitudinal velocity control objective. In this case, $\bar{z}_{\text{ref}}$ must also be modified by removing the longitudinal velocity reference such that $\bar{z}_{\text{ref}}(k) = [e_{\text{ref}}(k+1), \Delta\psi_{\text{ref}}(k+1), ..., e_{\text{ref}}(k+N_p), \Delta\psi_{\text{ref}}(k+N_p)]^T$.

Chapter 5

Results and discussion

5.1 Simulation environment

MATLAB and Simulink are used to implement the AMPC controller described in Section 4.3 and validate its performance throughout a variety of simulated test scenarios. The vehicle plant, WATonoTruck, is represented by two blocks within Simulink's Vehicle Dynamics Blockset: Vehicle Body 3DOF, which characterizes the longitudinal, lateral, and yaw characteristics of the vehicle by considering its mass, geometry, and forces exerted by the tires, Combined Slip Wheel 2DOF, which characterizes the longitudinal and lateral behaviour of all four wheels by utilizing the nonlinear combined slip magic formula tire force model [50]. Since factors such as roll stability, altitude changes, and non-zero grade and bank angles are not considered in this study, the vehicle's vertical, pitch, and roll motion is not significant, and does not need to be modelled. To create an accurate representation of the system dynamics, both blocks are configured to use vehicle body and tire model parameters that correspond to WATonoTruck. The vehicle parameters used in both the plant and controller reference models are given in Table 5.1; the values of these parameters correspond to the physical properties of WATonoTruck. Unless otherwise mentioned, a TRFC of $\mu_{xi} = \mu_{yi} = 1 \ \forall i$ is used throughout all scenarios. A TRFC of $\mu_{xi} = \mu_{yi} = 0.5$ will be used when evaluating the controller performance in harsh conditions. Tire force data provided by MSC Adams for a tire of size 285/80 R22.5 is used to create both the plant tire force model and to determine the Burckhardt lateral tire force model parameters when both $\mu_{xi} = \mu_{yi} = 1$ and $\mu_{xi} = \mu_{yi} = 0.5$. This tire size closely resembles the tires installed on WATonoTruck, which are 11 R24.5; MSC Adams does not provide data for the 11 R24.5 tire size and empirical tire force data collection is outside of the scope of

this work, so the 285/80 R22.5 tire size serves as a similar substitute. The friction coefficients used are adjusted in the model and plant using the longitudinal and lateral peak friction coefficients in the magic formula model [50]. The Burckhardt model parameters, fitted using nonlinear least squares regression, are displayed in Table 5.2. It is assumed that the properties of each wheel are identical, and therefore the fitted coefficients are the same. The use of the high fidelity Vehicle Dynamics Blockset blocks in combination with WATonoTruck-specific body and tire force parameters results in a plant model that characterizes the behaviour of the system with high accuracy.

Parameter	Value
m	2880.1 kg
a	1.75 m
b	1.75 m
I_z	11241.67 kg m^2
T	2.00001 m
$R_{\text{eff},i}\ \forall i$	0.65 m

Table 5.1: Vehicle parameters.

Parameter	$\mu_x,\ \mu_y$	Value
C_{1y}	1	0.7990
C_{2y}	1	7.4327
C_{3y}	1	0.1858
C_{1y}	0.5	0.3364
C_{2y}	0.5	38.6918
C_{3y}	0.5	0.0518

Table 5.2: Burckhardt model parameters.

Beyond the vehicle body and tire force modelling, a second order steering model is used to represent the dynamics of the plant's steering actuators. Each CM contains a linear actuator that is used to set the steering angle; the relationship between the linear actuator travel speed and the wheel steering rate is represented by the empirical linear model

$$\dot{\delta}_i = m_{\delta x_{\text{la}}} \dot{x}_{\text{la},i} + b_{\delta x_{\text{la}}}, \tag{5.1}$$

where, for some wheel i, $\dot{\delta}_i$ is the steering angle rate, $\dot{x}_{\text{la},i}$ is the linear actuator travel speed, and $m_{\delta x_{\text{la}}}$ and $b_{\delta x_{\text{la}}}$ are respectively the slope and intercept of the linear relationship. Given this, the steering dynamics are represented by the transfer function

$$\frac{\dot{\delta}_{i,\max}}{\tau s^2 + s}, \tag{5.2}$$

where $\dot{\delta}_{i,\max}$ represents the maximum steering rate determined using eq. (5.1) and the maximum linear actuator travel speed, and τ is the system time constant. Here, s represents the complex variable resulting from a Laplace transform. A time constant of $\tau = 0.05s$ is used. PD controllers with transfer functions

$$H(s) = K_p + K_d s \tag{5.3}$$

are used by the steering systems to track the desired steering angles at each corner output from the optimal controller. The proportional gain K_p and derivative gain K_d are set to 2000 and 200, respectively. Again, the s represents a complex variable resulting from the Laplace transform in this case. A block diagram representing the steering system is given in Figure 5.1. A block diagram outlining the overall simulation environment is given in Figure 5.2.

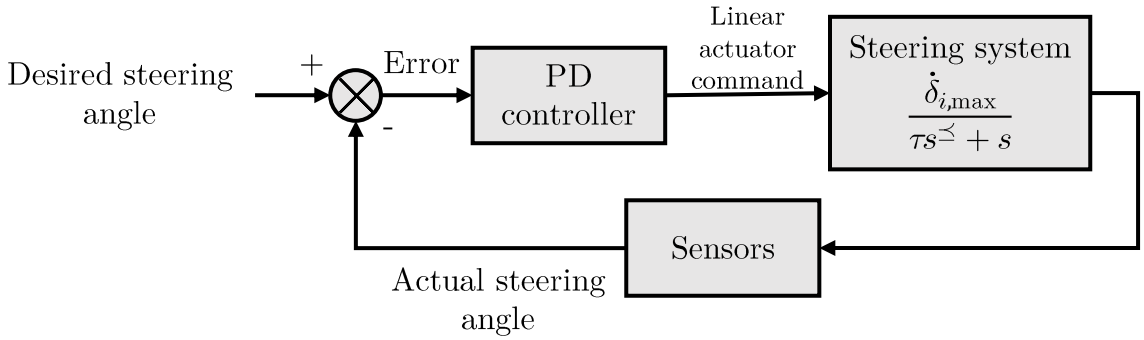


Figure 5.1: Steering system block diagram.

5.2 Simulation results

The results presented in the following subsections demonstrate the capability of the AMPC controller to control the vehicle throughout a varied set of path tracking scenarios. These

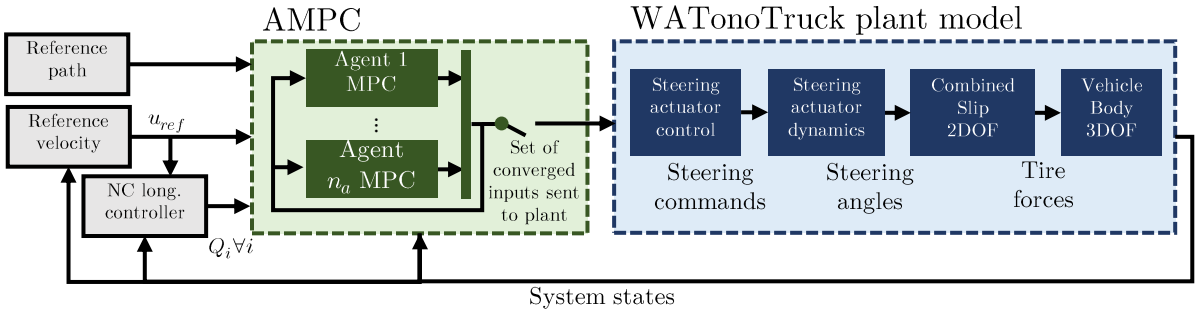


Figure 5.2: Simulation environment overview, implemented using MATLAB and Simulink.

include cases where the actuator topology is varied, where hardware failures occur in the system, and when the vehicle is required to make maneuvers in harsh scenarios. The set of controller parameters used throughout all simulations is given in Table 5.3.

5.2.1 Varied actuator topologies

One benefit of the AMPC framework is its ability to control the vehicle using a wide variety of different actuator topologies. In this subsection, the AMPC path tracking performance is compared over multiple trials; in each trial, the reference path and desired velocity profile remain the same, while the actuator configuration is changed. The chosen reference path represents a standard single lane change (SLC) maneuver. Due to the large weight and low maximum steering rate of the vehicle, a relatively low maximum desired velocity of 20 km/h is used. The actuator configurations include all-wheel steer (AWS) and all-wheel drive (AWD), AWS and front-wheel drive (FWD), AWS and rear-wheel drive (RWD), front-wheel steer (FWS) and AWD, FWS and FWD, and FWS and RWD. In the AWS cases, both δ_f and δ_r are selected by a single controllable agent. It is necessary for one controllable agent to select both δ_f and δ_r so that the constraint eq. (4.41) can be used. In the FWS cases, δ_r is disabled. In each trial, the torque at the driven wheels is selected by the NC longitudinal control agent. More specifically, the NC longitudinal control agent selects Q_{fl}, Q_{fr}, Q_{rl}, and Q_{rr} in the AWD trials; Q_{fl} and Q_{fr} in the FWD trials; and Q_{rl} and Q_{rr} in the RWD trials. Q_{rl} and Q_{rr} are disabled in the FWD trials, and Q_{fl} and Q_{fr} are disabled in the RWD trials. The disabled actuators are assigned to the DA agent. Over all of the agent configurations considered in this section, the inputs are grouped by actuator type rather than by their CM.

Figure 5.3 displays a comparison between the vehicle's trajectory, its controlled states,

Parameter	Value
Δt	0.05 s
N_p	20
N_c	10
Q_u	1e5
Q_e	5e5
$Q_{\Delta\psi}$	1e6
R_{δ_f}	5e5
R_{δ_r}	5e5
$R_{Q_{fl}}$	2.5e–3
$R_{Q_{fr}}$	2.5e–3
$R_{Q_{rl}}$	2.5e–3
$R_{Q_{rr}}$	2.5e–3
$P_{\Delta\delta_f}$	2e6
$P_{\Delta\delta_r}$	2e6
$P_{\Delta Q_{fl}}$	0
$P_{\Delta Q_{fr}}$	0
$P_{\Delta Q_{rl}}$	0
$P_{\Delta Q_{rr}}$	0
Q_{ϵ_r}	1e5
$Q_{\epsilon_{\alpha_{fl}}}$	1e8
$Q_{\epsilon_{\alpha_{fr}}}$	1e8
$Q_{\epsilon_{\alpha_{rl}}}$	1e8
$Q_{\epsilon_{\alpha_{rr}}}$	1e8
Maximum CCR iterations	10
α_{CCR}	0.5
$\epsilon_{\text{CCR}\delta}$	$0.1\pi/180$
$\epsilon_{\text{CCR}Q}$	10
K_p (NC longitudinal controller)	3500

Table 5.3: Controller parameters used in simulation.

and its control inputs across all trials. The results demonstrate that regardless of the actuator topology, the AMPC scheme is able to consider the available steering actuators while accounting for the torque applied by NC longitudinal velocity control agent and the absence of any disabled actuators to provide similar control performance across all trials.

In each case, the vehicle is able to closely follow the path at the desired velocity with minimal lateral or heading error while the steering and torque inputs remain smooth.

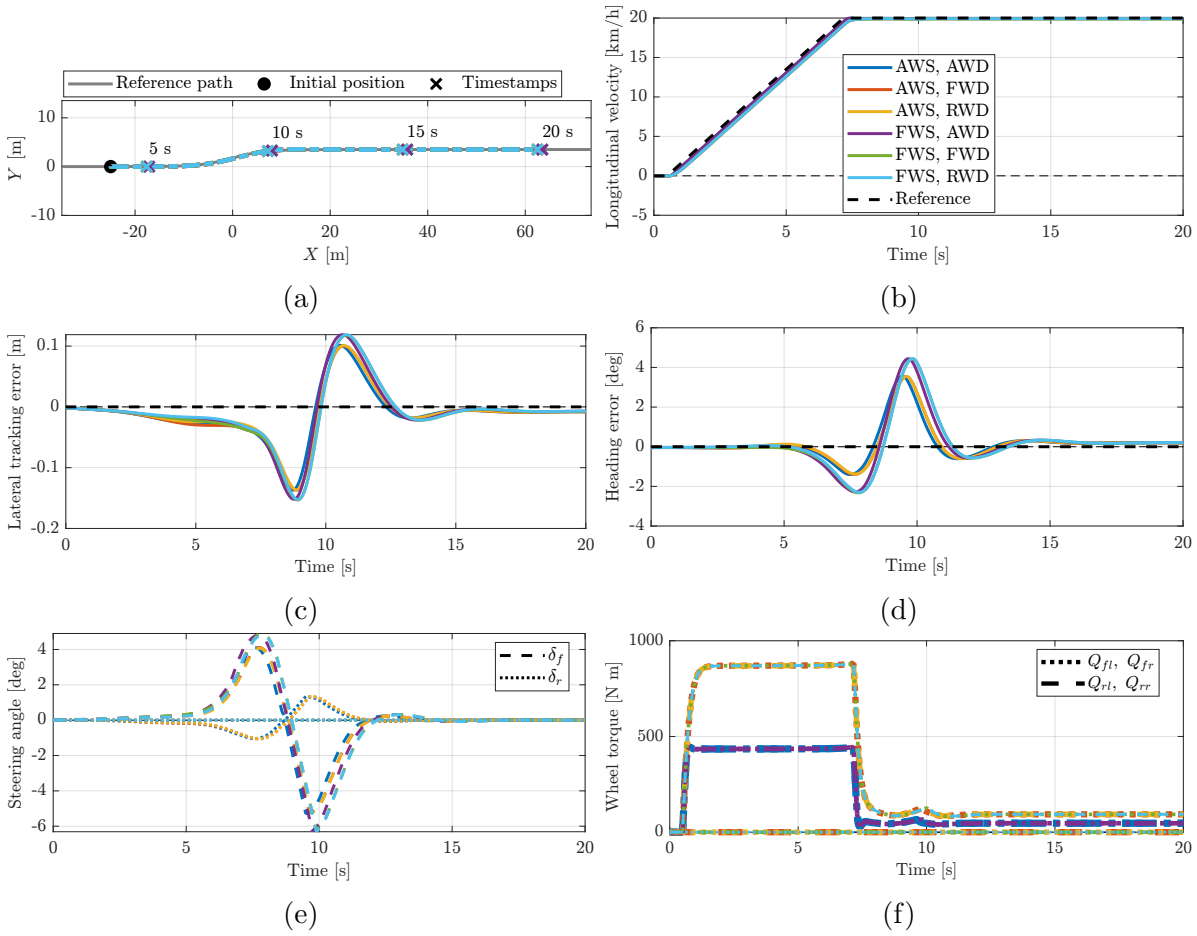


Figure 5.3: Comparison between SLC maneuvers performed by the vehicle across multiple trials using different actuator configurations. Across each trial, the reference path and desired velocity profile were the same. (a) displays the vehicle trajectory in each case, (b) the longitudinal velocity, (c) the lateral tracking error, (d) the heading error, (e) the imaginary center steering angles, and (f) the wheel torque.

5.2.2 Actuator faults

An additional benefit of the proposed AMPC scheme is its fault-tolerant control capabilities. In this subsection, the ability of the AMPC to control the vehicle in the presence of hardware faults is presented. For each scenario, the cases where the faults are both detected and not detected will be compared. When a fault is detected, the agents update their system prediction models as described in Section 3.6 to account for the change in the dynamics of the system. In the scenarios where the faults are not detected, no changes are made to the prediction models and the controller attempts to control the vehicle as if no fault has occurred.

In the first fault scenario, the vehicle is performing a 180 degree turn when the two right motors simultaneously fail. At the start of this scenario, one controllable agent is used to select both δ_f and δ_r, which allows the constraint eq. (4.41) to be enforced. Four additional controllable agents are used to optimize the wheel torque inputs, each selecting the torque Q_i for one wheel i. Therefore, at the start of runtime, five controllable agents are in use. This configuration, where the wheel torque inputs are grouped by their CM, highlights how each torque input can be individually adjusted to account for the others that have faulted. The failure occurs 10 s into runtime, and results in $Q_{rl} = Q_{rr} = 0$ for the remainder of the simulation. At the time of the fault, in the case where the fault is detected, these actuators are assigned to the DA agent; the number of controllable agents thus reduces by two. Figure 5.4 displays the path tracking performance and control inputs throughout the scenario in the cases where the faults both are and are not detected. In the case where the fault is detected, a small, immediate change occurs in the steering angles to compensate for it. In contrast, this change does not immediately occur in the case where the fault is not detected. There is also an observable difference in the torque inputs between both cases after the fault occurs. As a result, there is a noticeable difference in the lateral tracking and heading error values after the fault occurs; the error is generally lower in the case where the fault is detected. Further, the longitudinal velocity slightly exceeds the reference after the fault occurs in the case where the fault is not detected.

In the second fault scenario, the vehicle performs a 90 degree turn. Throughout this trial, multiple motors fail in sequence; Q_{rl} fails 4 seconds into runtime, Q_{fr} fails at 6 seconds, and Q_{fl} fails at 8 seconds. At the start of this scenario, one controllable agent is used to select both δ_f and δ_r, which allows the constraint eq. (4.41) to be enforced. Four additional controllable agents are used to optimize the wheel torque inputs, each selecting the torque Q_i for one wheel i. Therefore, at the start of runtime, five controllable agents are in use. This configuration, where the wheel torque inputs are grouped by their CM, highlights how each torque input can be individually adjusted to account for the others

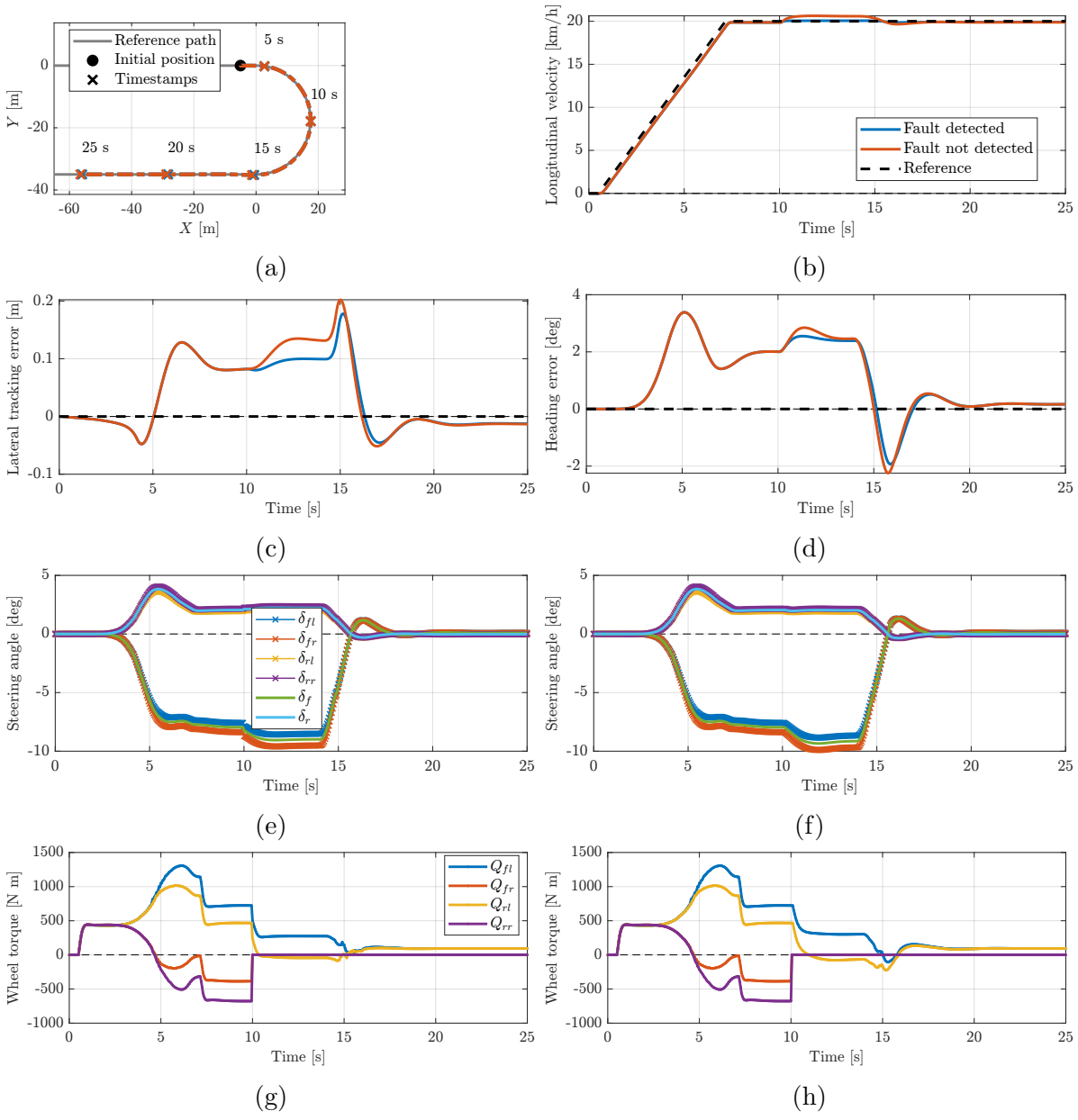


Figure 5.4: Right motors simultaneously fail while the vehicle performs a 180 degree turn. Cases where fault both is and is not detected are compared. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e, g) Steering angle, torque inputs in case where fault is detected. (f, h) Steering angle, torque inputs in case where fault is not detected.

that have faulted. At the time of each fault, the corresponding actuator is assigned to the DA agent, and the number of controllable agents reduces by one. These actuators are assigned to the DA agent at the times that their corresponding fault occurs. For the remainder of the 15 second simulation, the only motor available is Q_{rr}. Figure 5.5 displays the path tracking performance and control inputs throughout the scenario in the cases where the faults both are and are not detected. In this example, while not as noticeable as the previous scenario, there is an observable difference in the lateral tracking and heading error values between the cases where the fault was and was not detected. There is also an observable difference in all inputs between both cases after the faults occur.

In the third fault scenario, the vehicle is tracking an alternative curved path when an issue occurs with the steering actuators. At the start of this scenario, one controllable agent is used to select δ_f and another δ_r. Separate agents are used to select δ_f and δ_r in this scenario to demonstrate how one can compensate for the other when one of them fails. All Q_i are selected by the NC longitudinal control agent. Therefore, at the start of runtime, two controllable agents are in use. 7 seconds after the start of runtime, an issue occurs with the agent controlling δ_r that results in δ_r remaining constant for the remainder of the simulation. At this time, δ_r is assigned to the DA agent and the number of controllable agents reduces to one. Figure 5.6 displays the path tracking performance and control inputs throughout the scenario in the cases where the faults both are and are not detected. In the case where the fault is detected, the front steering angle inputs remain smooth and the path is tracked effectively. However, in the case where the fault is not detected, the front steering inputs become unreliable and the vehicle deviates significantly from the path.

In the fourth fault scenario, the vehicle is following a sinusoidal path when two steering actuators fail in sequence; δ_{rl} first fails 4 seconds after the start of the simulation, and δ_{fl} fails 4 seconds later. After δ_{rl} and δ_{fl} fault, they remain constant for the remainder of the simulation. In this case, one controllable agent selects both δ_f and δ_r, while the NC longitudinal control agent selects all Q_i. The failures of the δ_{rl} and δ_{fl} actuators are accounted for by the AMPC controller using eq. (3.72). Since δ_{rl} and δ_{fl} are not inputs to the reference model, there are no updates to the model's agent configuration when the faults occur; δ_f and δ_r are selected and used as normal to determine δ_{fr} and δ_{rr}. Figure 5.7 displays the path tracking performance and control inputs throughout the scenario in the cases where the faults both are and are not detected. In this scenario, there is again a noticeable difference in the lateral and heading errors between the cases where the faults were and were not detected; when the faults are detected and considered by the AMPC controller, the tracking errors are much lower.

Overall, the results in this subsection indicate that the AMPC scheme can utilize the

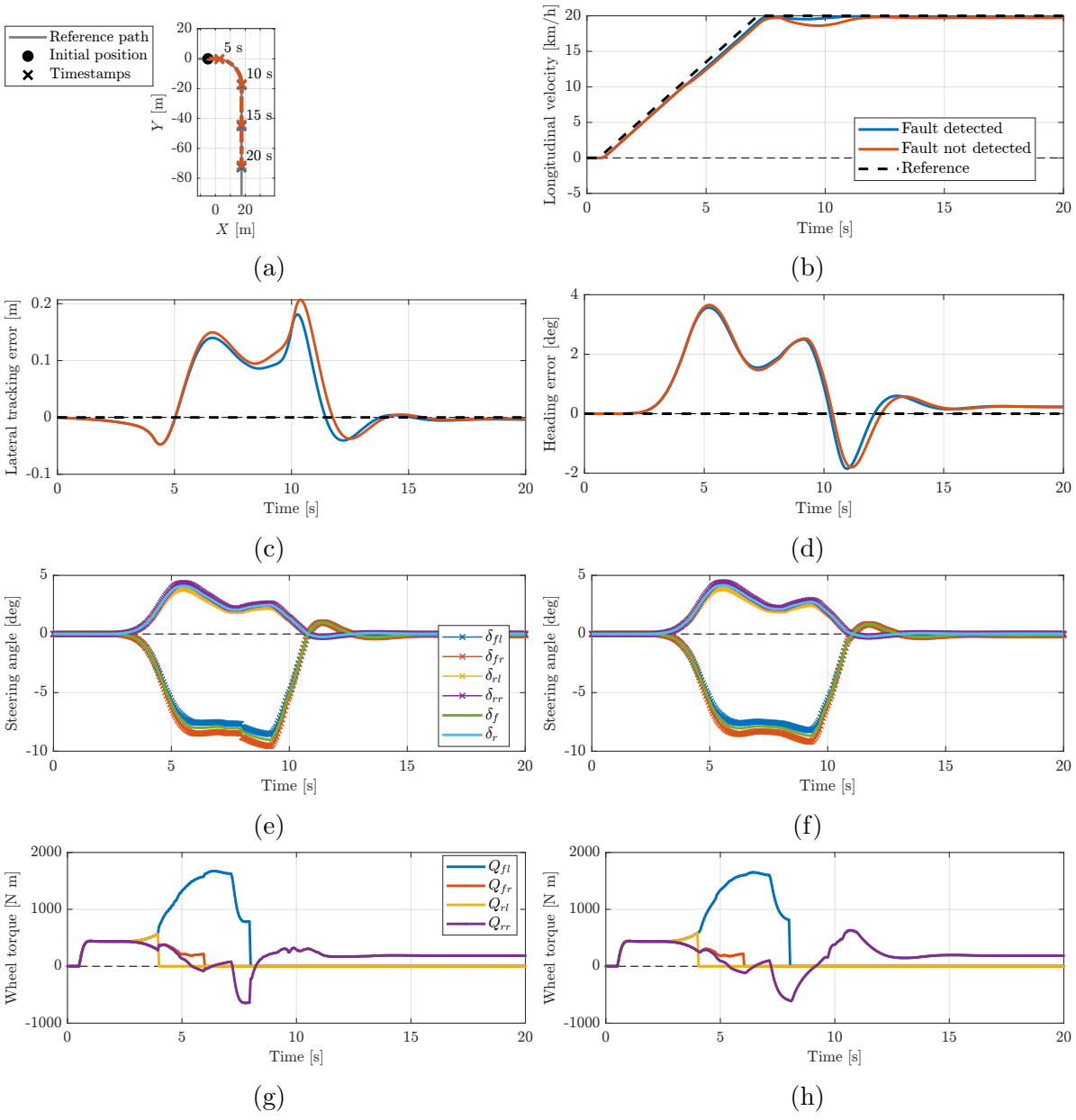


Figure 5.5: Three motors fail in sequence while the vehicle performs a 90 degree turn. Cases where faults both are and are not detected are compared. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e, g) Steering angle, torque inputs in case where faults are detected. (f, h) Steering angle, torque inputs in case where faults are not detected.

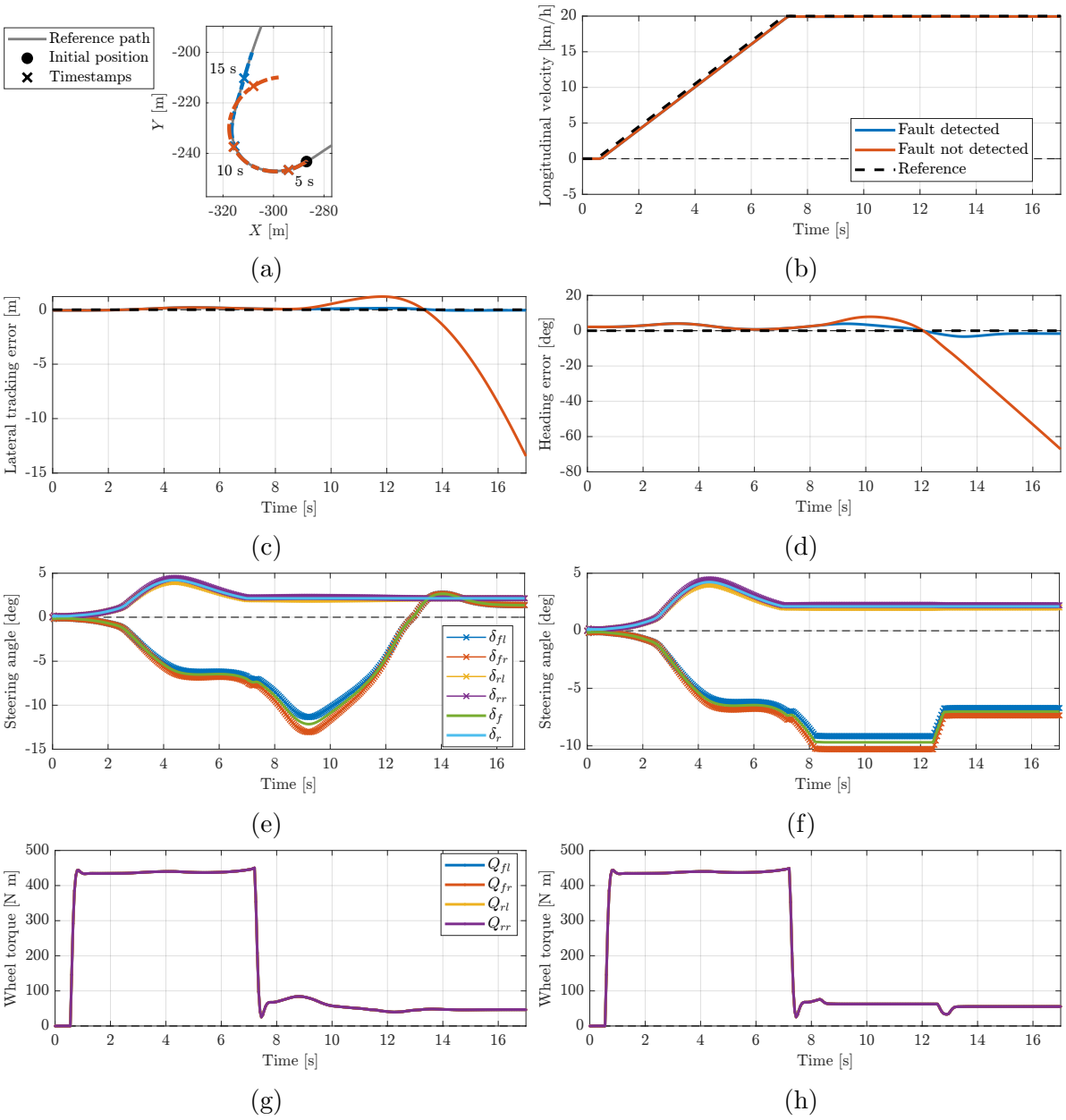


Figure 5.6: δ_r fails while the vehicle follows a curved path. Cases where the fault both is and is not detected are compared. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e, g) Steering angle, torque inputs in case where fault is detected. (f, h) Steering angle, torque inputs in case where fault is not detected.

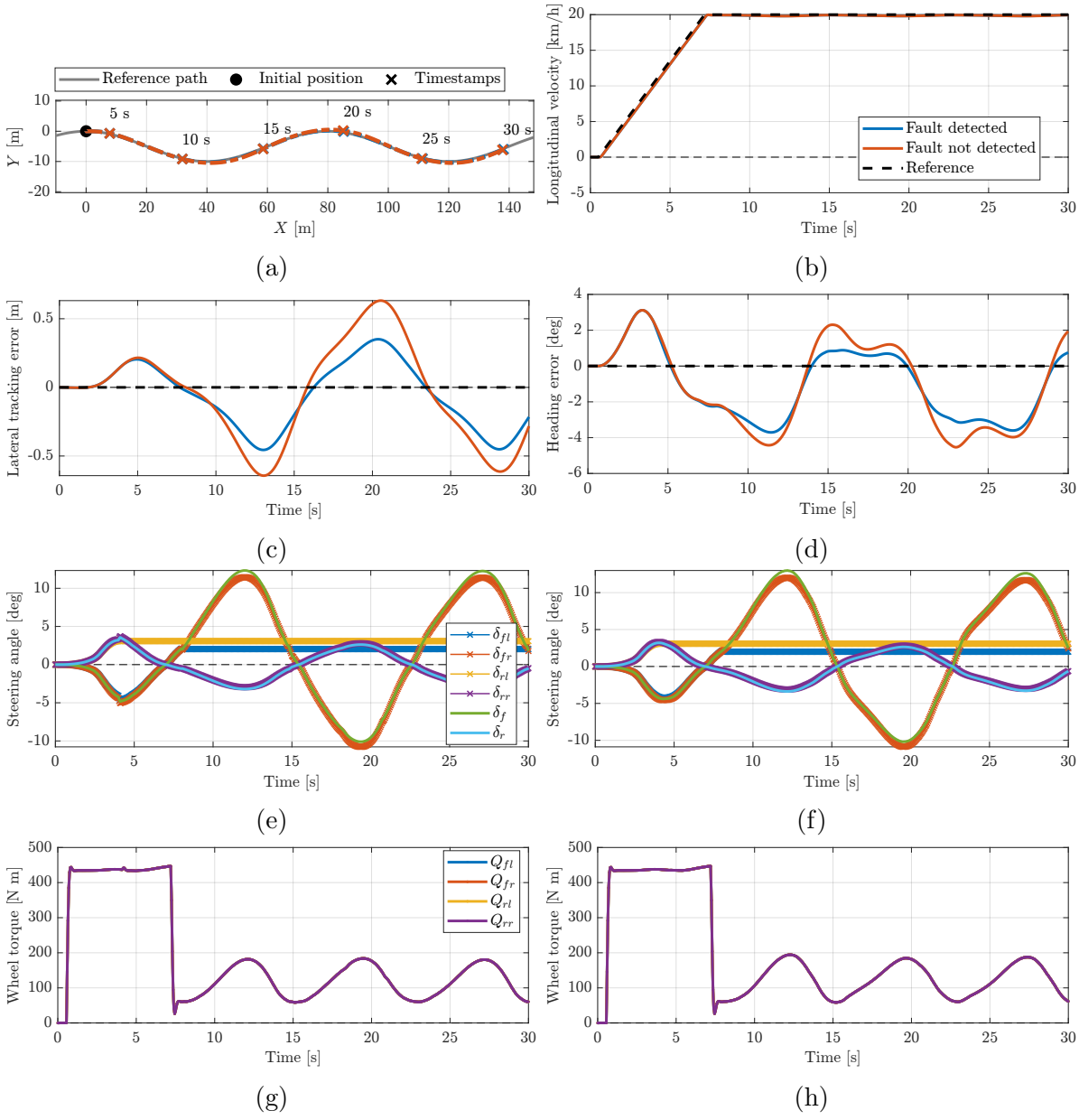


Figure 5.7: The left steering actuators fail in sequence while the vehicle follows a sinusoidal path. Cases where faults both are and are not detected are compared. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e, g) Steering angle, torque inputs in case where faults are detected. (f, h) Steering angle, torque inputs in case where faults are not detected.

DA agent to account for faults that are detected throughout runtime. In doing so, the controller can provide much better control performance compared to cases where faults are not detected or considered and the prediction model continues to assume that the system has not changed. Furthermore, as shown in the previous subsection, the controller can also account for the inputs selected by the NC longitudinal control agent. In certain cases, the fault detection capabilities result in minor improvements in the lateral and heading error values, while in others, these capabilities prevent the vehicle from significantly deviating from the path.

5.2.3 Harsh scenarios

In this subsection, the ability of the AMPC scheme to effectively control the vehicle in harsh path tracking scenarios is investigated. Furthermore, the simulation results in this section show that changing the topology of the actuators available to the controller, which can be easily done using the AMPC framework, can result in path tracking improvements. To demonstrate this, for each harsh scenario, two simulations are compared: in one case, the NC longitudinal control agent selects a single torque value to apply to each wheel, and in the second, torque vectoring is enabled by utilizing four controllable agents in place of the NC agent that each select one of the wheel torque values Q_i.

In the first harsh scenario, the vehicle attempts to perform a double lane change (DLC) maneuver on a slippery road with a TRFC of $\mu_x = \mu_y = 0.5$ at a high speed of 80 km/h. One controllable agent controls both δ_f and δ_r, which enables the use of the constraint given in eq. (4.41). However, given the vehicle speed and the relationship between k_δ and u described in Figure 4.2, δ_r is constrained to be zero throughout the entire maneuver. Figure 5.8 displays the path tracking performance and control inputs throughout the scenario in the cases where torque vectoring both is and is not used. To demonstrate the ability of the controller to satisfy the constraints in harsh scenarios, the yaw rate and sideslip angles throughout these simulations are shown in Figure 5.9. The results demonstrate multiple benefits of augmenting the actuator configuration to enable torque vectoring, which is easily achieved using the AMPC framework. Firstly, the lateral and heading error are greatly minimized when torque vectoring is used; when the NC agent selects the wheel torque, the vehicle becomes unstable and deviates significantly from the path when performing the second lane change. Second, to account for the large deviation from the path, the steering angle becomes large and irregular when torque vectoring is not used in an unsuccessful attempt to return to the path; the use of torque vectoring allows the steering angles to remain comparatively small and smooth. Third, the use of torque vectoring prevents the yaw rate and sideslip angles from significantly exceeding the constraints, while these states become undesirably large when torque vectoring is not used.

In the second harsh scenario, the vehicle performs an acceleration-in-turn (AIT) maneuver around a circular path. Again, a TRFC of $\mu_x = \mu_y = 0.5$ is used to increase the difficulty of the control task. While the maximum speed used here is lower than the harsh DLC case, the acceleration and top speed of the vehicle is larger in this scenario in comparison to the results presented in Subsections 5.2.1 and 5.2.2. One controllable agent controls both δ_f and δ_r, which enables the use of the constraint given in eq. (4.41). Figure 5.10 displays the path tracking performance and control inputs throughout the scenario in the cases where torque vectoring both is and is not used. To demonstrate the ability of the

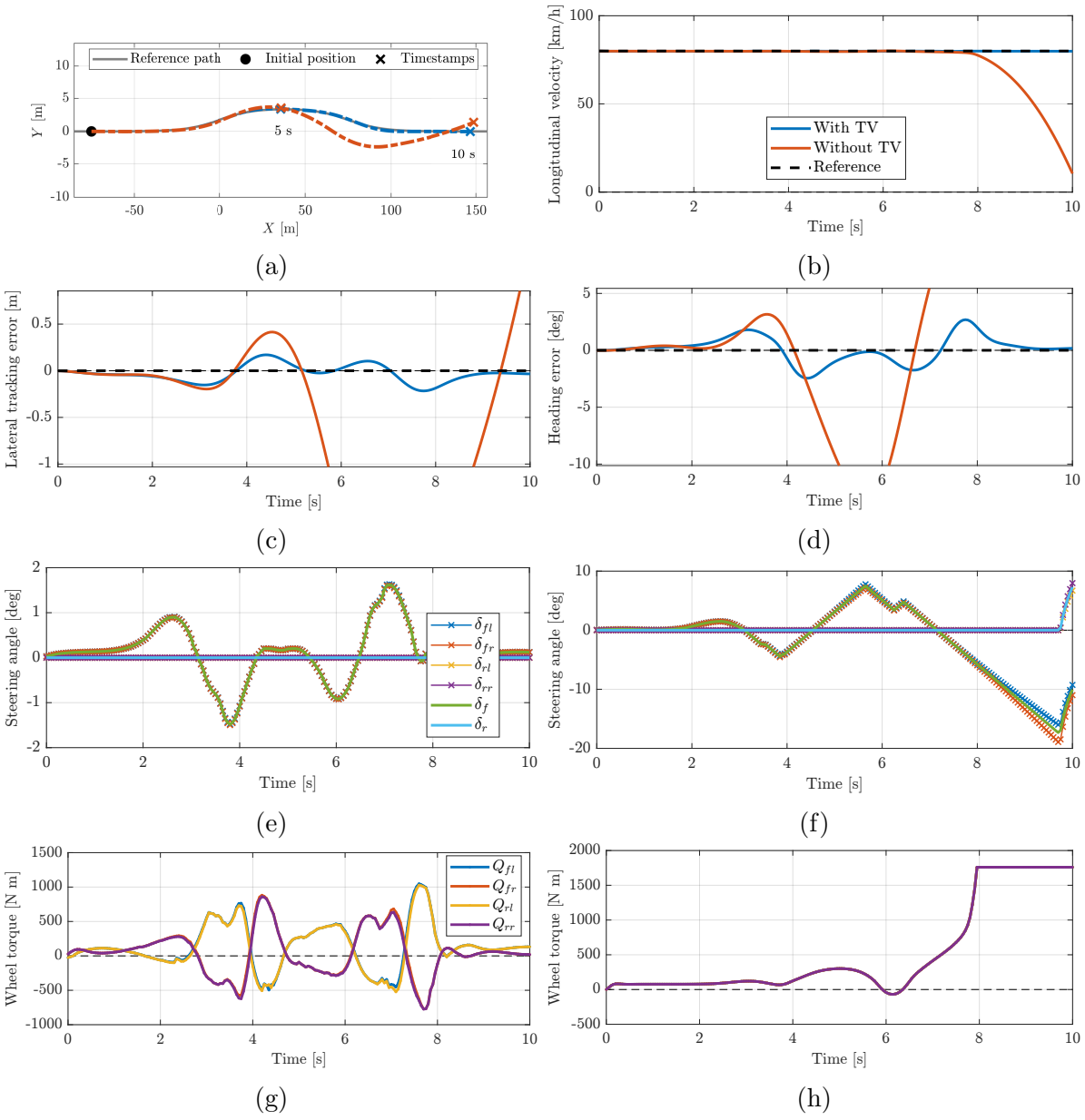


Figure 5.8: Harsh DLC scenario. Results in cases where torque vectoring both is and is not used is compared. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e, g) Steering angle, torque inputs in case where torque vectoring is used. (f, h) Steering angle, torque inputs in case where torque vectoring is not used.

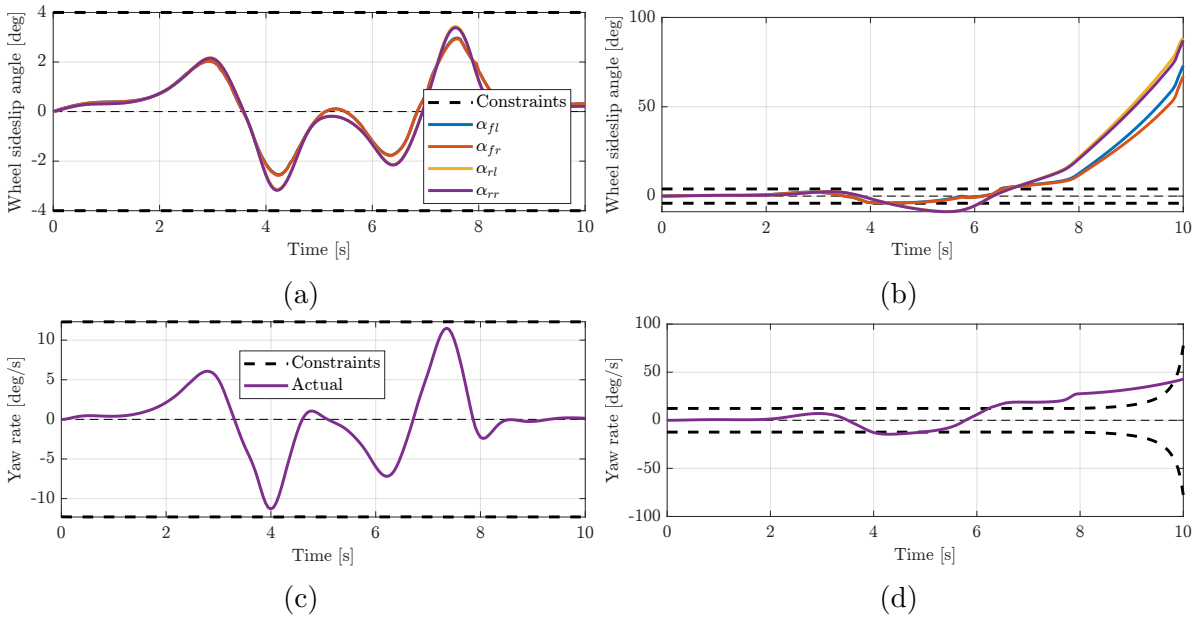


Figure 5.9: Sideslip angles and yaw rate throughout the harsh DLC scenario in the cases where torque vectoring both is and is not used. (a, c) Sideslip angles and yaw rate, respectively, in the case where torque vectoring is used. (b, d) Sideslip angles and yaw rate, respectively, in the case where torque vectoring is not used.

controller to satisfy the constraints in harsh scenarios, the yaw rate and sideslip angles throughout these simulations are shown in Figure 5.11. In this scenario, it is again seen that when torque vectoring is utilized, the lateral and heading error responses are smaller and smoother, the steering inputs are smoother, and the sideslip angles remain closer to bounds.

The results in this subsection demonstrate that reconfiguring the actuator topology, which can be easily done using the AMPC framework, can result in a large improvement in path tracking performance. In these examples, the configurations used did not change throughout their respective simulations. However, in future work, the capability to switch between actuator topologies online depending on the state of the system can be explored. For example, it would be possible to use the NC longitudinal control agent to select the wheel torque values when the vehicle is travelling in a stable condition, but when the vehicle stability worsens as indicated, for instance, by some pre-defined sideslip angle or lateral error thresholds, the actuator topology could be reconfigured to enable torque vectoring and improve the system stability. A technique of this nature would further utilize the benefits

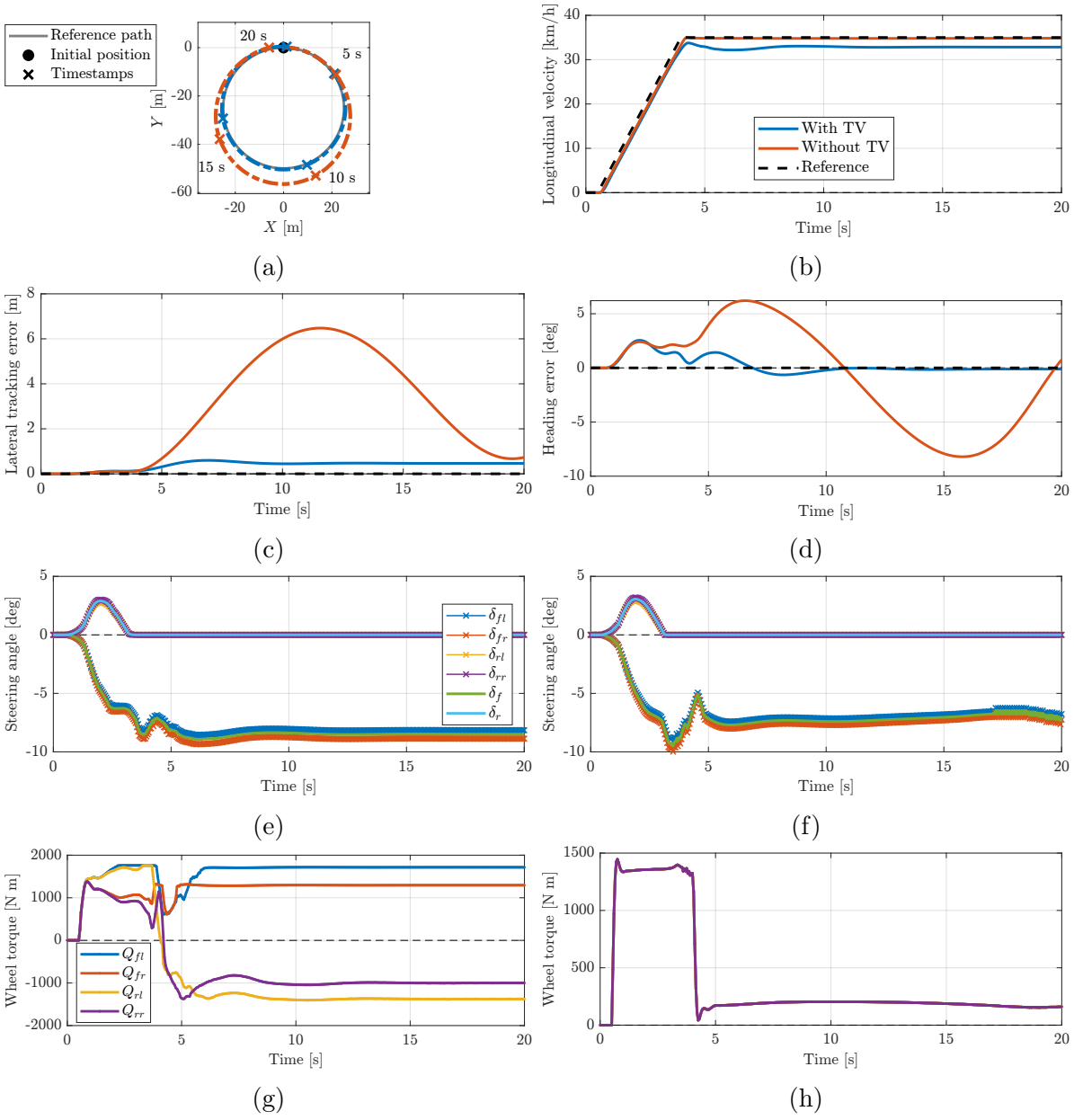


Figure 5.10: Harsh AIT scenario. Results in cases where torque vectoring both is and is not used is compared. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e, g) Steering angle, torque inputs in case where torque vectoring is used. (f, h) Steering angle, torque inputs in case where torque vectoring is not used.

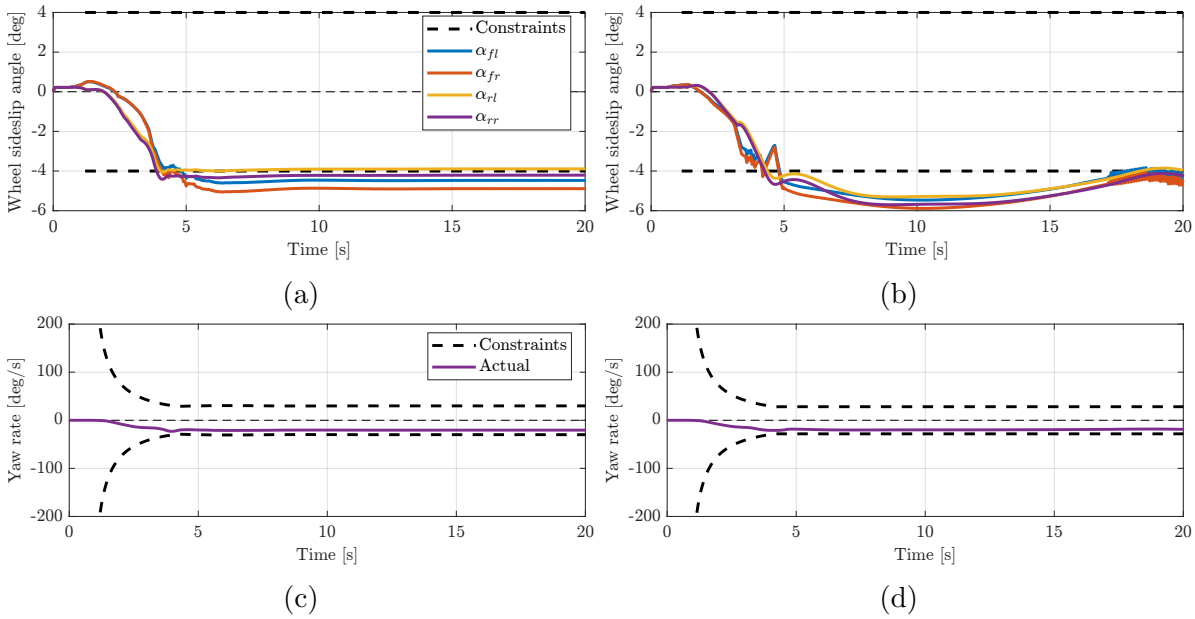


Figure 5.11: Sideslip angles and yaw rate throughout the harsh AIT scenario in the cases where torque vectoring both is and is not used. (a, c) Sideslip angles and yaw rate, respectively, in the case where torque vectoring is used. (b, d) Sideslip angles and yaw rate, respectively, in the case where torque vectoring is not used.

of the capability to easily switch actuator configurations within the AMPC framework.

5.3 Experimental vehicle platform

The AMPC controller, outlined in Section 4.3, is evaluated experimentally using WATonoTruck. The values of the parameters used to construct the AMPC prediction model are the same as those given in Table 5.1, as these correspond to the physical properties of WATonoTruck. It is assumed that the values listed in this table correspond perfectly to the properties of the plant; no additional parameter estimation techniques are used. For safety reasons, all experimental results were collected using low-speed, high-friction driving scenarios. The road surface was dry throughout all experimental testing and had an estimated TRFC of $\mu_x = \mu_y = 1$. Because of this, it was not necessary to use torque vectoring during any experimental testing to ensure vehicle stability. Figure 5.12 displays images of WATonoTruck captured during experimental testing.

(a) (b)

Figure 5.12: Displays the truck throughout an autonomous cornering maneuver performed experimentally. (a) Entering the corner. (b) Exiting the corner.

As mentioned in Section 3.1, WATonoTruck has one independent steering system at all four corners, and one independent electric motor at the two front corners. For each of the experimental tests conducted, unless otherwise specified, one controllable agent is used to select δ_f and δ_r, the NC longitudinal control agent is used to select Q_{fl} and Q_{fr}, and the Q_{rl} and Q_{rr} inputs are assigned to the DA agent as the actuators corresponding to these inputs do not exist on the hardware platform.

The AMPC controller is run using an augmented version of the Simulink model used to perform simulation testing. In this Simulink model, the plant dynamics are replaced with a Simulink-Robot Operating System (ROS) interface that enables the controller to receive sensor measurements and publish the desired control input requests. The ROS interface is connected to the vehicle's CAN bus, which is used to retrieve the sensor measurements from and send the control requests to the hardware. Wheel speed sensors report the wheel angular velocity measurements while the vehicle position, velocity, acceleration, heading,

and yaw rate are reported by an inertial navigation system (INS) that combines global navigation satellite system (GNSS) and inertial measurement unit (IMU) sensors. The acceleration is reported directly from the IMU while the INS uses extended Kalman filters (EKFs) to determine the position, velocity, heading, and yaw rate by fusing the GNSS and IMU readings. A block diagram outlining the overall hardware platform is given in Figure 5.13.

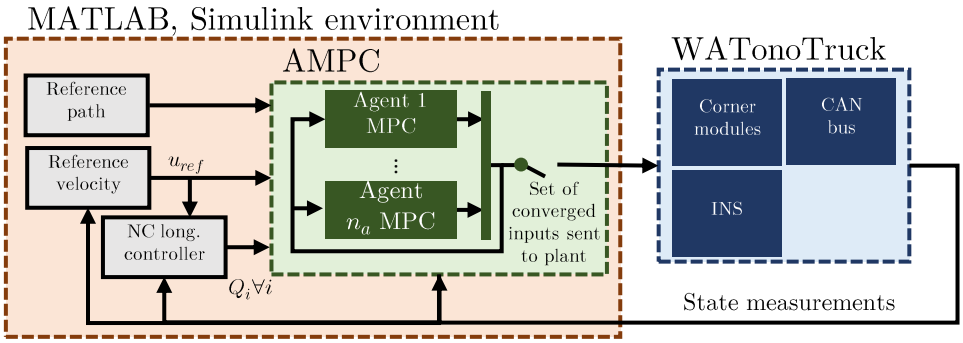


Figure 5.13: Vehicle hardware platform overview.

5.4 Experimental results

The results presented in the following subsections demonstrate the capability of the AMPC controller to control the vehicle throughout a varied set of experimental path tracking scenarios. The set of controller parameters used throughout all experiments is largely the same as those given in Table 5.3; all differences are highlighted in Table 5.4.

Parameter	Value
Q_e	5e6
$Q_{\Delta\psi}$	5e6

Table 5.4: Controller parameters used to generate experimental results that differ from those used in simulation.

5.4.1 Path tracking

The first set of experimental scenarios serve to validate the ability of the AMPC scheme to provide effective path tracking performance using the physical WATonoTruck platform as it performs two standard path following maneuvers: an SLC and a 90 degree turn. Due to the large size and weight of the vehicle, its low maximum steering rate, and a limited amount of physical space available to perform experimental testing, a low reference longitudinal velocity of 2.5 km/h was used in both scenarios.

Figure 5.14 displays the results of the experimental scenario where the vehicle performs an SLC maneuver. The vehicle trajectory with respect to the reference path, its controlled states, and the steering and torque inputs selected by the AMPC scheme and the NC longitudinal control agent, respectfully, are shown. Figure 5.16 displays this information for the experimental scenario where the vehicle performs a 90 degree turn. In both cases, the AMPC controller enables the vehicle to effectively follow the path, maintaining low lateral and heading error values, while the longitudinal velocity remains close to the reference. The optimal controller can consider both the inputs selected by the NC longitudinal control agent and the unused inputs assigned to the DA agent when selecting the steering angles. The non-zero lateral and heading errors present at the beginning of the scenario results from the difficulty of perfectly manually aligning the vehicle with the path before its autonomous path tracking capabilities are evaluated. Despite the presence of state measurement noise, the steering profile remains smooth.

5.4.2 Actuator faults

An experimental test in which an actuator fault occurs is performed to determine the ability of the AMPC framework to handle faults that occur on the physical hardware. For safety reasons and to prevent damage to the wheels caused by unecessary wheel slip, steering faults are not analyzed experimentally. Furthermore, since WATonoTruck does not have electric motors at the rear wheels, only faults to the front motors can be considered.

Figure 5.16 displays the results of the experimental actuator fault test. In this scenario, the vehicle is performing an SLC maneuver. Shortly after the beginning of the maneuver, a wheel torque fault is simulated at the front-right corner by assigning Q_{fr} to the DA agent such that Q_{fr} remains zero for the remainder of the maneuver, and the SLC must be completed using only the front-left motor. In spite of the fault, the AMPC controller enables the vehicle to smoothly track the path and desired velocity while maintaining low lateral and heading error values. The optimal controller effectively considers the inputs

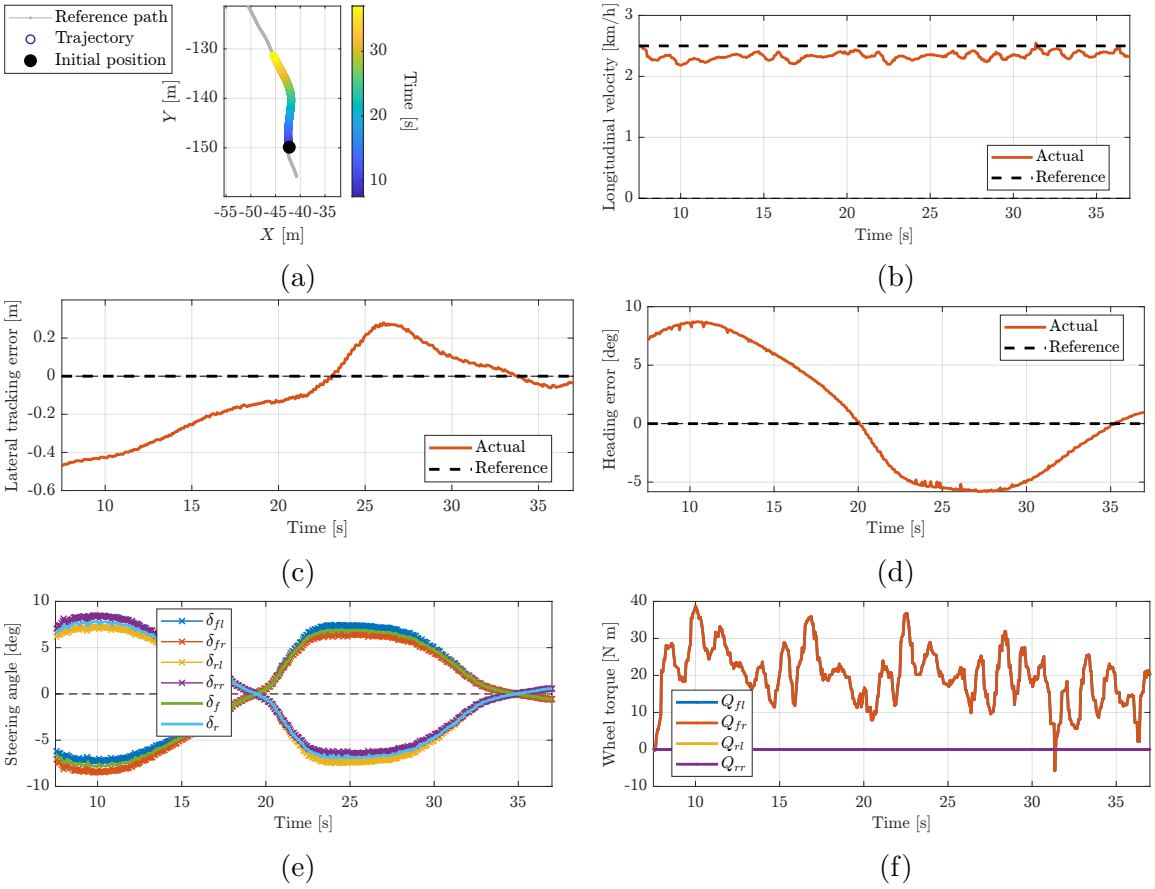


Figure 5.14: Experimental SLC path tracking scenario. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e) Steering angle. (f) Wheel torque.

of both the NC and DA agents. As in the previous subsection, the non-zero lateral and heading errors present at the beginning of the scenario results from the difficulty of perfectly manually aligning the vehicle with the path before its autonomous path tracking capabilities are evaluated. Further, despite the presence of state measurement noise and the occurrence of the fault, the steering profile remains smooth.

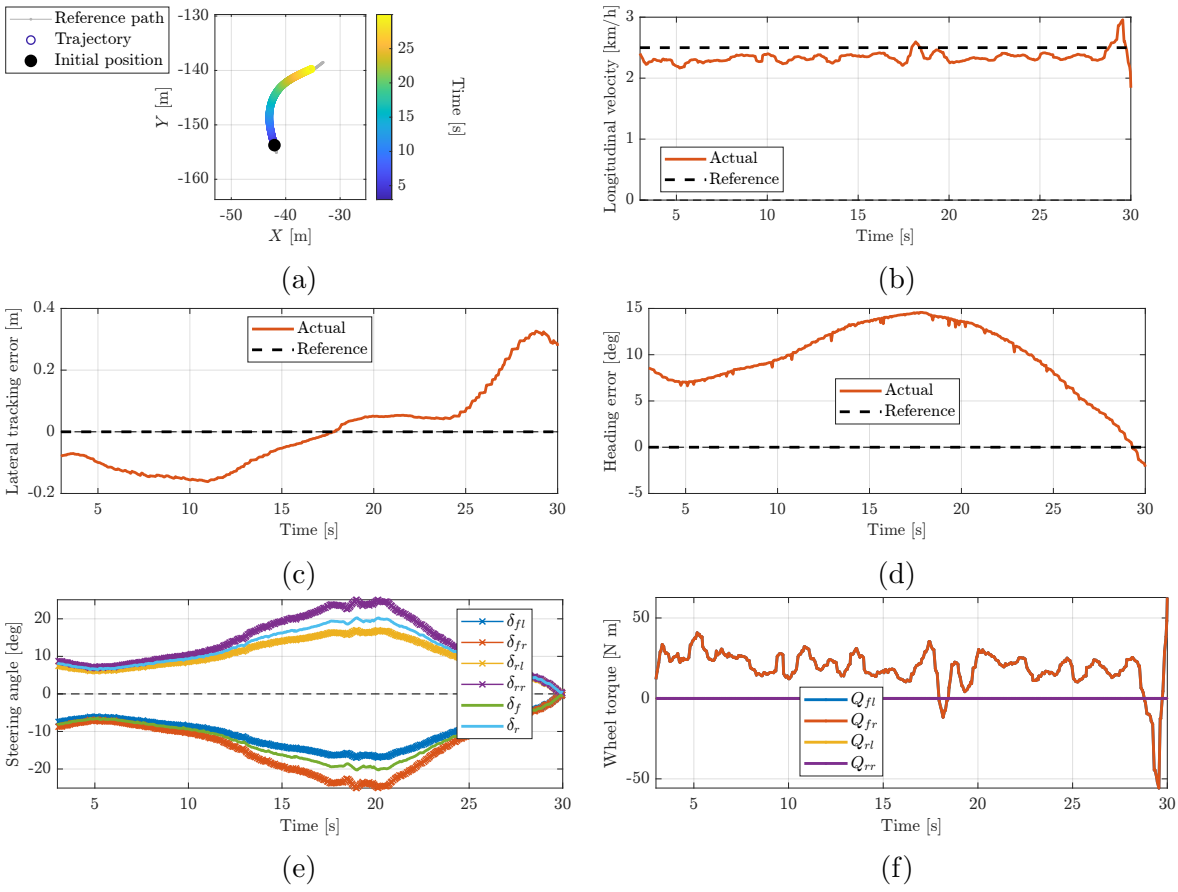


Figure 5.15: Experimental 90 degree turn path tracking scenario. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e) Steering angle. (f) Wheel torque.

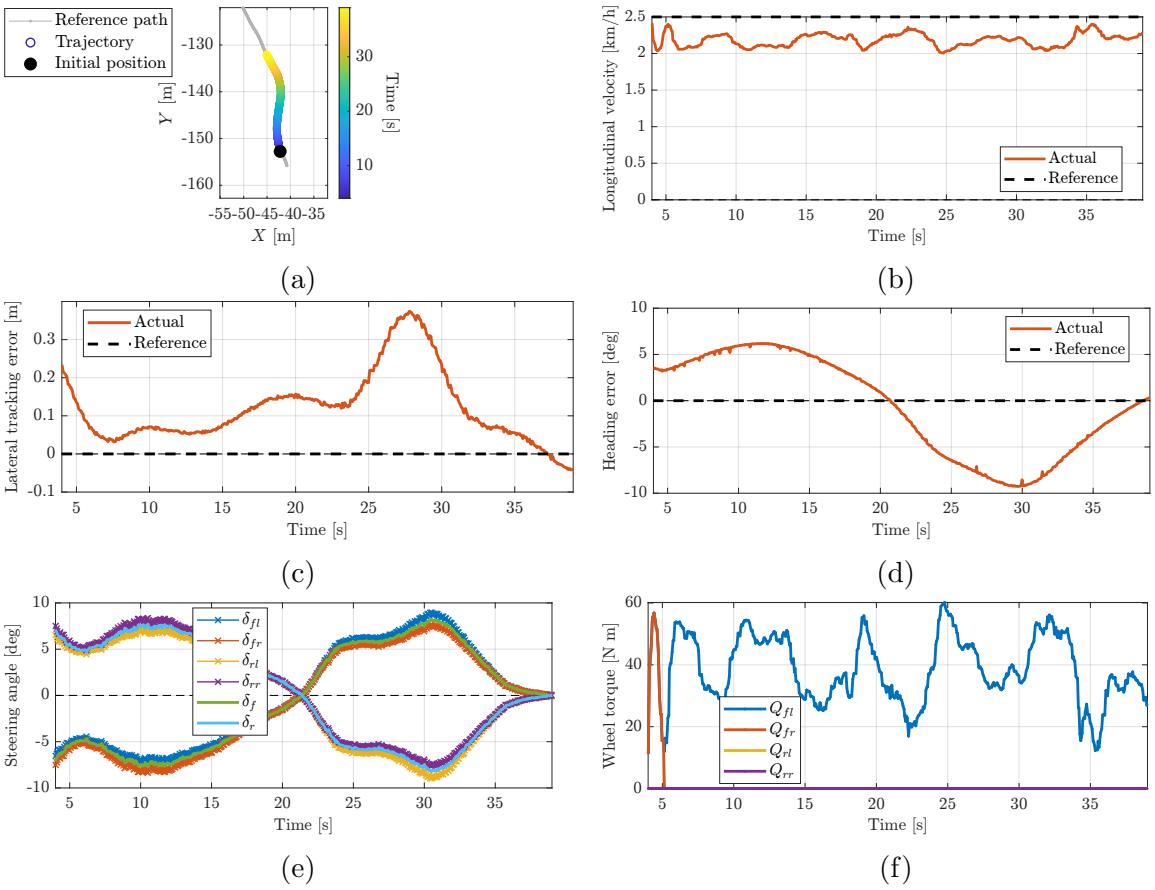


Figure 5.16: Experimental SLC path tracking scenario where a fault at the front-left motor is simulated, causing the torque output of this motor to go to zero. (a) Vehicle trajectory. (b) Longitudinal velocity. (c) Lateral tracking error. (d) Heading error. (e) Steering angle. (f) Wheel torque.

Chapter 6

Conclusions and future work

6.1 Conclusions

The purpose of this thesis was to outline the formulation and features of the AMPC framework; detail the application of AMPC to the path tracking and velocity control task for WATonoTruck, an over-actuated, 4WIS, two-wheel independent drive autonomous flatbed truck; and present simulation and experimental results that validate the performance of the controller. While past studies have separately explored the use of MPC for vehicle path tracking control and utilizing AMPC to achieve vehicle stability objectives, using AMPC to perform combined path tracking and longitudinal velocity control is a novel approach.

The hardware platform WATonoTruck, in particular, is a suitable for use with a distributed control approach given its modular, CM-based design. The prediction model presented in this work, which serves to characterize the behaviour of a vehicle with independent steering and torque actuators at each corner, includes features such as a nonlinear tire force model that accurately describes the complex relationship between the tires and road surface and the Ackermann steering geometry, which lessens the complexity of the control optimization problem by reducing the number of inputs that the controller must select. To utilize the reference model for agent-based control design, a method to augment the model for use with each controllable agent is presented. Furthermore, the AMPC framework can handle varied agent configurations and allows the controllable agents to incorporate the contribution of NC agents on the system dynamics along, consider the effects of actuator faults, and disable inputs if the corresponding actuator is not in use or does not exist on the hardware platform. Beyond this, a set of constraints to impose on the system states, inputs, and input rates are outlined; these constraints serve to ensure sys-

tem stability, enforce the actuator hardware limitations, and reduce the complexity of the optimization problem by introducing an additional steering relationship. Slack variables are incorporated into the optimal control scheme to prevent optimization infeasibility in cases where the state constraints cannot be satisfied. A proportional controller designed to perform longitudinal velocity control is presented; the AMPC framework can be configured to utilize this external controller as an NC agent, or handle both the path tracking and velocity control tasks on its own.

The AMPC controller is validated in both simulation and experimental test scenarios. The simulation results demonstrate that the AMPC controller can perform both path tracking and velocity control simultaneously; provide path tracking capabilities in cooperation with an NC agent that handles velocity control; be easily reconfigured to control the vehicle using different actuator topologies; enable effective path tracking in the presence of hardware faults; and work effectively even in harsh scenarios where the desired longitudinal velocity is high and the TRFC is low. The success of the controller throughout the experimental tests in both standard path tracking scenarios and in the presence of actuator faults further reinforces the conclusions drawn from the simulation results and indicates that the designed prediction model effectively characterizes the behaviour of the vehicle.

6.2 Future work

While the presented AMPC controller is successful in meeting the objectives of this thesis, there are different means through which the algorithm can be further improved that can be explored in future work. A key component of AMPC is the prediction model that represents the dynamics of the system; to maximize the control performance, it is desirable that the prediction model represents reflects the characteristics of the system with high accuracy. One method to improve the model accuracy involves the incorporation of parameter estimation techniques to estimate the values of the parameters used to construct the model. Given the intention to utilize WATonoTruck in applications where large amounts of material must be transported, this feature is particularly relevant as the mass and CG location of the vehicle will frequently change. Incorporating online parameter estimation techniques will enable the controller to consider the change in these parameters without the need to manually remeasure them whenever the payload changes. Beyond this, the model accuracy could be further improved by incorporating a combined slip tire force model to estimate the longitudinal tire forces in addition to the lateral forces. While a simpler longitudinal tire force model was used in this work to reduce the model complexity and improve the optimization efficiency, the trade-off between these quantities and the

model accuracy can be explored. Furthermore, the inclusion of a machine learning-based component to the prediction model could also help to improve its accuracy. A data-driven model could be used, for example, to estimate the error between the actual vehicle motion and the behaviour predicted by the physics-based model, or to represent the complex tire force behaviour or estimate tire force limits.

An additional vehicle motion control-related challenge that is particularly prevalent in experimental scenarios is the task of accurately determining the longitudinal slip ratio and sideslip angle at each wheel when the vehicle speed is low. Along with the methods that will generally improve the accuracy of the prediction model, there are multiple techniques that may improve slip estimation accuracy or mitigate the effects of inaccuracies that were not explored in this work. One source of error is measurement noise, which is particularly detrimental at low speeds as the magnitude of the velocity, yaw rate, and wheel angular velocity noise becomes closer to the magnitude of the true underlying signal. The use of predictive filters such as the Kalman filter (KF) to reduce measurement noise could be explored; unlike other low-pass filters, the KF does not incur a phase delay in the filtered signal, and the reference model that the KF requires is already known. Furthermore, to mitigate the issue entirely, the use of a kinematic AMPC reference model at low speeds could be explored; when the vehicle speed exceeds a certain threshold where the slip estimations become accurate, the presented dynamic model could then be used as the AMPC reference.

A third avenue for future work relates to actuator prioritization. In this thesis, the same actuator topology was used throughout all test scenarios, except for those in which a fault occurred. Nonetheless, using the AMPC framework presented here, it is possible to change the actuator topology online at any time. This capability could be used to prioritize which actuators are in use. For example, in standard driving conditions where torque vectoring is not needed to ensure the stability of the system, the NC longitudinal control agent could be used to select the wheel torque that will be applied to all wheels; however, if the vehicle detects that it is entering an unstable operating condition, the actuator topology could be reconfigured such that the NC longitudinal control agent is replaced with four controllable agents, each controlling one of the Q_i inputs, which will then perform torque vectoring to improve the vehicle stability.

References

[1] S. Singh. Critical reasons for crashes investigated in the national motor vehicle crash causation survey. National Center for Statistics and Analysis, March, 2018.

[2] U.S. Department of Transportation. Preparing for the future of transportation: Automated vehicles 3.0. U.S. Department of Transportation, October 4 2018.

[3] Peter Davidson and Anabelle Spinoulas. Autonomous vehicles - what could this mean for the future of transport? 2015.

[4] Bin Li, Avesta Goodarzi, Amir Khajepour, Shih ken Chen, and Baktiar Litkouhi. An optimal torque distribution control strategy for four-independent wheel drive electric vehicles. *Vehicle System Dynamics*, 53(8):1172–1189.

[5] Zheming Chen, Bohan Zhang, Bao Chen, and Jianghua Fu. Development and evaluation of torque distribution methods for four-wheel independent-drive electric vehicle based on parameter estimation. *Advances in Mechanical Engineering*, 11(5):1687814019847405, 2019.

[6] Seongjin Yim. Integrated chassis control with four-wheel independent steering under constraint on front slip angles. *IEEE Access*, 9:10338–10347, 2021.

[7] Stefano Di Cairano, Hongtei Eric Tseng, Daniele Bernardini, and Alberto Bemporad. Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain. *IEEE Transactions on Control Systems Technology*, 21(4):1236–1248, 2013.

[8] Asal Nahidi, Alireza Kasaiezadeh, Saeid Khosravani, Amir Khajepour, Shih-Ken Chen, and Bakhtiar Litkouhi. Modular integrated longitudinal and lateral vehicle stability control for electric vehicles. *Mechatronics*, 44:60–70, 2017.

[9] Milad Jalali, Amir Khajepour, Shih ken Chen, and Bakhtiar Litkouhi. Integrated stability and traction control for electric vehicles using model predictive control. *Control Engineering Practice*, 54:256–266, 2016.

[10] Mansour Ataei, Amir Khajepour, and Soo Jeon. Model predictive control for integrated lateral stability, traction/braking control, and rollover prevention of electric vehicles. *Vehicle system dynamics*, 58(1):49–73, 2020.

[11] Qifan Tan, Xu Wang, Javad Taghia, and Jayantha Katupitiya. Force control of two-wheel-steer four-wheel-drive vehicles using model predictive control and sequential quadratic programming for improved path tracking. *International Journal of Advanced Robotic Systems*, 14(6):1729881417746295, 2017.

[12] Peng Hang, Fengmei Luo, Shude Fang, and Xinbo Chen. Path tracking control of a four-wheel-independent-steering electric vehicle based on model predictive control. In *2017 36th Chinese Control Conference (CCC)*, pages 9360–9366, 2017.

[13] Xiaojun Tan, Deliang Liu, and Huiyuan Xiong. Optimal control method of path tracking for four-wheel steering vehicles. *Actuators*, 11(2), 2022.

[14] Waters, Allison. A novel universal corner module for urban electric vehicles: Design, prototype, and experiment. Master's thesis, 2017.

[15] Chen Tang and Amir Khajepour. Wheel modules with distributed controllers: A multi-agent approach to vehicular control. *IEEE Transactions on Vehicular Technology*, 69(10):10879–10888, 2020.

[16] Chen Tang and Amir Khajepour. Agent-based model predictive controller (ampc) for flexible and efficient vehicular control. *IEEE Transactions on Vehicular Technology*, 70(10):9877–9885, 2021.

[17] Chen Tang, Mehdi Abroshan, and Amir Khajepour. Agent-based model predictive controller (ampc) for vehicular stability with experimental results. *IEEE Transactions on Vehicular Technology*, 71(7):7104–7112, 2022.

[18] Giovanni Palmieri, Osvaldo Barbarisi, Stefano Scala, and Luigi Glielmo. A preliminary study to integrate ltv-mpc lateral vehicle dynamics control with a slip control. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 4625–4630, 2009.

[19] Craig Earl Beal and J. Christian Gerdes. Model predictive control for vehicle stabilization at the limits of handling. *IEEE Transactions on Control Systems Technology*, 21(4):1258–1269, 2013.

[20] D. Bernardini, S. Di Cairanoz, A. Bemporad, and H.E. Tsengz. Drive-by-wire vehicle stabilization and yaw regulation: a hybrid model predictive control design. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 7621–7626, 2009.

[21] Mansour Ataei, Amir Khajepour, and Soo Jeon. Reconfigurable integrated stability control for four- and three-wheeled urban vehicles with flexible combinations of actuation systems. *IEEE/ASME Transactions on Mechatronics*, 23(5):2031–2041, 2018.

[22] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on Control Systems Technology*, 15(3):566–580, 2007.

[23] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat. Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation. *International Journal of Robust and Nonlinear Control*, 18(8):862–875, 2008.

[24] Qingjia Cui, Rongjun Ding, Bing Zhou, and Xiaojian Wu. Path-tracking of an autonomous vehicle via model predictive control and nonlinear filtering. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 232(9):1237–1252, 2018.

[25] Yihuai Zhang, Baijun Shi, Xizhi Hu, and Wandong Ai. Low-Speed Vehicle Path-Tracking Algorithm Based on Model Predictive Control Using QPKWIK Solver. *Journal of Dynamic Systems, Measurement, and Control*, 143(12), 09 2021. 121003.

[26] Keke Geng, Nikolai Alexandrovich Chulin, and Ziwei Wang. Fault-tolerant model predictive control algorithm for path tracking of autonomous vehicle. *Sensors*, 20(15), 2020.

[27] Matthew Brown, Joseph Funke, Stephen Erlien, and J. Christian Gerdes. Safe driving envelopes for path tracking in autonomous vehicles. *Control Engineering Practice*, 61:307–316, 2017.

[28] John P. Alsterda, Matthew Brown, and J. Christian Gerdes. Contingency model predictive control for automated vehicles. In *2019 American Control Conference (ACC)*, pages 717–722, 2019.

[29] Reza Hajiloo, Mehdi Abroshan, Amir Khajepour, Alireza Kasaiezadeh, and Shih-Ken Chen. Integrated steering and differential braking for emergency collision avoidance in autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 22(5):3167–3178, 2021.

[30] Jie Ji, Amir Khajepour, Wael William Melek, and Yanjun Huang. Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. *IEEE Transactions on Vehicular Technology*, 66(2):952–964, 2017.

[31] Yadollah Rasekhipour, Amir Khajepour, Shih-Ken Chen, and Bakhtiar Litkouhi. A potential field-based model predictive path-planning controller for autonomous road vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 18(5):1255–1267, 2017.

[32] Zejiang Wang, Yunhao Bai, Junmin Wang, and Xiaorui Wang. Vehicle Path-Tracking Linear-Time-Varying Model Predictive Control Controller Parameter Selection Considering Central Process Unit Computational Load. *Journal of Dynamic Systems, Measurement, and Control*, 141(5), 01 2019. 051004.

[33] Bing Zhang, Changfu Zong, Guoying Chen, and Bangcheng Zhang. Electrical vehicle path tracking based model predictive control with a laguerre function and exponential weight. *IEEE Access*, 7:17082–17097, 2019.

[34] Yoonsuk Choi, Wonwoo Lee, Jeesu Kim, and Jinwoo Yoo. A variable-sampling time model predictive control algorithm for improving path-tracking performance of a vehicle. *Sensors*, 21(20), 2021.

[35] Fen Lin, Yuke Chen, Youqun Zhao, and Shaobo Wang. Path tracking of autonomous vehicle based on adaptive model predictive control. *International Journal of Advanced Robotic Systems*, 16(5):1729881419880089, 2019.

[36] Yixiao Liang, Yinong Li, Amir Khajepour, Yanjun Huang, Yechen Qin, and Ling Zheng. A novel combined decision and control scheme for autonomous vehicle in structured road based on adaptive model predictive control. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):16083–16097, 2022.

[37] Anil Aswani, Humberto Gonzalez, S. Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.

[38] Chris J. Ostafew, Angela P. Schoellig, and Timothy D. Barfoot. Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4029–4036, 2014.

[39] Chris J. Ostafew, Angela P. Schoellig, and Timothy D. Barfoot. Robust constrained learning-based nmpc enabling reliable mobile robot path tracking. *The International Journal of Robotics Research*, 35(13):1547–1563, 2016.

[40] Mohammad Rokonuzzaman, Navid Mohajer, Saeid Nahavandi, and Shady Mohamed. Model predictive control with learned vehicle dynamics for autonomous vehicle path tracking. *IEEE Access*, 9:128233–128249, 2021.

[41] A.N. Venkat, J.B. Rawlings, and S.J. Wright. Stability and optimality of distributed model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 6680–6685, 2005.

[42] James B. Rawlings and Brett T. Stewart. Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9):839–845, 2008. Selected Papers From Two Joint Conferences: 8th International Symposium on Dynamics and Control of Process Systems and the 10th Conference Applications in Biotechnology.

[43] Panagiotis D. Christofides, Riccardo Scattolini, David Muñoz de la Peña, and Jinfeng Liu. Distributed model predictive control: A tutorial review and future research directions. *Computers & Chemical Engineering*, 51:21–41, 2013. CPC VIII.

[44] Yue Ren, Ling Zheng, and Amir Khajepour. Integrated model predictive and torque vectoring control for path tracking of 4-wheel-driven autonomous vehicles. *IET Intelligent Transport Systems*, 13(1):98–107, 2019.

[45] Haidong Wu, Zhenli Si, and Zihan Li. *IEEE Access*.

[46] Yonghwan Jeong and Seongjin Yim. Model predictive control-based integrated path tracking and velocity control for autonomous vehicle with four-wheel independent steering and driving. *Electronics*, 10(22), 2021.

[47] R. Rajamani. *Vehicle Dynamics and Control*. Springer, 2012.

[48] Joseph Funke, Matthew Brown, Stephen M. Erlien, and J. Christian Gerdes. Collision avoidance and stabilization for autonomous vehicles in emergency scenarios. *IEEE Transactions on Control Systems Technology*, 25(4):1204–1216, 2017.

[49] Merijn Floren, Amir Khajepour, and Ehsan Hashemi. An integrated control approach for the combined longitudinal and lateral vehicle following problem. In *2021 American Control Conference (ACC)*, pages 436–441, 2021.

[50] Hans B. Pacejka. *Tyre and Vehicle Dynamics*. Butterworth-Heinemann, 2005.

[51] Raymond Brach and Matthew Brach. The tire-force ellipse (friction ellipse) and tire characteristics. In *SAE 2011 World Congress & Exhibition*. SAE International, apr 2011.

[52] Kwanwoo Park, Eunhyek Joa, Kyongsu Yi, and Youngsik Yoon. Rear-wheel steering control for enhanced steady-state and transient vehicle handling characteristics. *IEEE Access*, 8:149282–149300, 2020.

[53] J. Rawlings, D. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2017.

APPENDICES

Appendix A

State prediction matrix definitions

All matrix definitions given in this section assume that the matrices A_d, B_d, and so forth remain constant over the prediction horizon; their constant values are determined by considering the operating condition of the system at each time step. This implies that the quantities ω_i, λ_i, and F_{zi}, measured or estimated at each sampling instant, are also assumed to be constant over the prediction horizon. The matrices S_x, S_U, S_{W_d}, and S_{W_z} that are used to predict the controlled state over the prediction horizon $\bar{z}$ in the centralized case are defined as

$$S_x = \begin{bmatrix} C_z A_d \\ \vdots \\ C_z A_d^{N_p} \end{bmatrix}, \tag{A.1}$$

$$S_U = \begin{bmatrix} C_z B_d & D_z & \mathbf{0}_{n_z \times n_U} & \dots & \mathbf{0}_{n_z \times n_U} \\ C_z A_d B_d & C_z B_d & D_z & \dots & \mathbf{0}_{n_z \times n_U} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_c-2} B_d & C_z A_d^{N_c-3} B_d & C_z A_d^{N_c-4} B_d & \dots & D_z \\ C_z A_d^{N_c-1} B_d & C_z A_d^{N_c-2} B_d & C_z A_d^{N_c-3} B_d & \dots & C_z B_d + D_z \\ C_z A_d^{N_c} B_d & C_z A_d^{N_c-1} B_d & C_z A_d^{N_c-2} B_d & \dots & C_z A_d B_d + C_z B_d + D_z \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_p-1} B_d & C_z A_d^{N_p-2} B_d & C_z A_d^{N_p-3} B_d & \dots & C_z \sum_{\ell=0}^{N_p-N_c} [A_d^{\ell}] B_d + D_z \end{bmatrix}, \tag{A.2}$$

$$S_{W_d} = \begin{bmatrix} C_z & \mathbf{0}_{n_z \times n_{W_d}} & \dots & \mathbf{0}_{n_z \times n_{W_d}} \\ C_z A_d & C_z & \dots & \mathbf{0}_{n_z \times n_{W_d}} \\ \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_p-1} & C_z A_d^{N_p-2} & \dots & C_z \end{bmatrix}, \tag{A.3}$$

$$S_{W_z} = \begin{bmatrix} W_z \\ \vdots \\ W_z \end{bmatrix}. \tag{A.4}$$

The matrices S_{cx}, S_{cU}, S_{cW_d}, and S_{cW_z} used to compute $\bar{z}_c$ are defined analogously to S_x, S_U, S_{W_d}, and S_{W_z}, respectively, but with C_{z_c}, D_{z_c}, W_{z_c}, $\mathbf{0}_{n_z \times n_U}$, and $\mathbf{0}_{n_z \times n_{W_d}}$ respectively substituted for C_z, D_z, W_z, $\mathbf{0}_{n_{z_c} \times n_U}$, and $\mathbf{0}_{n_{z_c} \times n_{W_d}}$. The matrices S_{U_j}, $S_{U_{j\mathrm{OC}}}$, $S_{U_{\mathrm{NC}}}$, and $S_{U_{\mathrm{DA}}}$ used to predict $\bar{z}$ in the non-centralized case are defined as

$$S_{U_j} = \begin{bmatrix} C_z B_{jd} & D_{zj} & \mathbf{0}_{n_z \times n_{U_j}} & \dots & \mathbf{0}_{n_z \times n_{U_j}} \\ C_z A_d B_{jd} & C_z B_{jd} & D_{zj} & \dots & \mathbf{0}_{n_z \times n_{U_j}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_c-2} B_{jd} & C_z A_d^{N_c-3} B_{jd} & C_z A_d^{N_c-4} B_{jd} & \dots & D_{zj} \\ C_z A_d^{N_c-1} B_{jd} & C_z A_d^{N_c-2} B_{jd} & C_z A_d^{N_c-3} B_{jd} & \dots & C_z B_{jd} + D_{zj} \\ C_z A_d^{N_c} B_{jd} & C_z A_d^{N_c-1} B_{jd} & C_z A_d^{N_c-2} B_{jd} & \dots & C_z A_d B_{jd} + C_z B_{jd} + D_{zj} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_p-1} B_{jd} & C_z A_d^{N_p-2} B_{jd} & C_z A_d^{N_p-3} B_{jd} & \dots & C_z \sum_{\ell=0}^{N_p-N_c} [A_d^\ell] B_{jd} + D_{zj} \end{bmatrix}, \tag{A.5}$$

$$S_{U_{j\mathrm{OC}}} = \begin{bmatrix} C_z B_{jd\mathrm{OC}} & D_{zj\mathrm{OC}} & \mathbf{0}_{n_z \times n_{U_{j\mathrm{OC}}}} & \dots & \mathbf{0}_{n_z \times n_{U_{j\mathrm{OC}}}} \\ C_z A_d B_{jd\mathrm{OC}} & C_z B_{jd\mathrm{OC}} & D_{zj\mathrm{OC}} & \dots & \mathbf{0}_{n_z \times n_{U_{j\mathrm{OC}}}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_c-2} B_{jd\mathrm{OC}} & C_z A_d^{N_c-3} B_{jd\mathrm{OC}} & C_z A_d^{N_c-4} B_{jd\mathrm{OC}} & \dots & D_{zj\mathrm{OC}} \\ C_z A_d^{N_c-1} B_{jd\mathrm{OC}} & C_z A_d^{N_c-2} B_{jd\mathrm{OC}} & C_z A_d^{N_c-3} B_{jd\mathrm{OC}} & \dots & C_z B_{jd\mathrm{OC}} + D_{zj\mathrm{OC}} \\ C_z A_d^{N_c} B_{jd\mathrm{OC}} & C_z A_d^{N_c-1} B_{jd\mathrm{OC}} & C_z A_d^{N_c-2} B_{jd\mathrm{OC}} & \dots & C_z A_d B_{jd\mathrm{OC}} + C_z B_{jd\mathrm{OC}} + D_{zj\mathrm{OC}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_p-1} B_{jd\mathrm{OC}} & C_z A_d^{N_p-2} B_{jd\mathrm{OC}} & C_z A_d^{N_p-3} B_{jd\mathrm{OC}} & \dots & C_z \sum_{\ell=0}^{N_p-N_c} [A_d^\ell] B_{jd\mathrm{OC}} + D_{zj\mathrm{OC}} \end{bmatrix}, \tag{A.6}$$

$$S_{U_{j\mathrm{NC}}} =$$
$$\begin{bmatrix} C_z B_{jd\mathrm{NC}} & D_{zj\mathrm{NC}} & \mathbf{0}_{n_z \times n_{U_{j\mathrm{NC}}}} & \dots & \mathbf{0}_{n_z \times n_{U_{j\mathrm{NC}}}} \\ C_z A_d B_{jd\mathrm{NC}} & C_z B_{jd\mathrm{NC}} & D_{zj\mathrm{NC}} & \dots & \mathbf{0}_{n_z \times n_{U_{j\mathrm{NC}}}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_c-2} B_{jd\mathrm{NC}} & C_z A_d^{N_c-3} B_{jd\mathrm{NC}} & C_z A_d^{N_c-4} B_{jd\mathrm{NC}} & \dots & D_{zj\mathrm{NC}} \\ C_z A_d^{N_c-1} B_{jd\mathrm{NC}} & C_z A_d^{N_c-2} B_{jd\mathrm{NC}} & C_z A_d^{N_c-3} B_{jd\mathrm{NC}} & \dots & C_z B_{jd\mathrm{NC}} + D_{zj\mathrm{NC}} \\ C_z A_d^{N_c} B_{jd\mathrm{NC}} & C_z A_d^{N_c-1} B_{jd\mathrm{NC}} & C_z A_d^{N_c-2} B_{jd\mathrm{NC}} & \dots & C_z A_d B_{jd\mathrm{NC}} + C_z B_{jd\mathrm{NC}} + D_{zj\mathrm{NC}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_p-1} B_{jd\mathrm{NC}} & C_z A_d^{N_p-2} B_{jd\mathrm{NC}} & C_z A_d^{N_p-3} B_{jd\mathrm{NC}} & \dots & C_z \sum_{\ell=0}^{N_p-N_c} [A_d^\ell] B_{jd\mathrm{NC}} + D_{zj\mathrm{NC}} \end{bmatrix}, \tag{A.7}$$

$$S_{U_{j\mathrm{DA}}} =$$
$$\begin{bmatrix} C_z B_{jd\mathrm{DA}} & D_{zj\mathrm{DA}} & \mathbf{0}_{n_z \times n_{U_{j\mathrm{DA}}}} & \dots & \mathbf{0}_{n_z \times n_{U_{j\mathrm{DA}}}} \\ C_z A_d B_{jd\mathrm{DA}} & C_z B_{jd\mathrm{DA}} & D_{zj\mathrm{DA}} & \dots & \mathbf{0}_{n_z \times n_{U_{j\mathrm{DA}}}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_c-2} B_{jd\mathrm{DA}} & C_z A_d^{N_c-3} B_{jd\mathrm{DA}} & C_z A_d^{N_c-4} B_{jd\mathrm{DA}} & \dots & D_{zj\mathrm{DA}} \\ C_z A_d^{N_c-1} B_{jd\mathrm{DA}} & C_z A_d^{N_c-2} B_{jd\mathrm{DA}} & C_z A_d^{N_c-3} B_{jd\mathrm{DA}} & \dots & C_z B_{jd\mathrm{DA}} + D_{zj\mathrm{DA}} \\ C_z A_d^{N_c} B_{jd\mathrm{DA}} & C_z A_d^{N_c-1} B_{jd\mathrm{DA}} & C_z A_d^{N_c-2} B_{jd\mathrm{DA}} & \dots & C_z A_d B_{jd\mathrm{DA}} + C_z B_{jd\mathrm{DA}} + D_{zj\mathrm{DA}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_z A_d^{N_p-1} B_{jd\mathrm{DA}} & C_z A_d^{N_p-2} B_{jd\mathrm{DA}} & C_z A_d^{N_p-3} B_{jd\mathrm{DA}} & \dots & C_z \sum_{\ell=0}^{N_p-N_c} [A_d^\ell] B_{jd\mathrm{DA}} + D_{zj\mathrm{DA}} \end{bmatrix}. \tag{A.8}$$

The matrices S_{cU_j}, $S_{cU_{j\mathrm{OC}}}$, $S_{cU_{\mathrm{NC}}}$, and $S_{cU_{\mathrm{DA}}}$ used to predict $\bar{z}_c$ are defined analogously to S_{U_j}, $S_{U_{j\mathrm{OC}}}$, $S_{U_{\mathrm{NC}}}$, and $S_{U_{\mathrm{DA}}}$, respectively, but with C_{z_c}, $D_{z_c j}$, $D_{z_c j\mathrm{OC}}$, $D_{z_c j\mathrm{NC}}$, $D_{z_c j\mathrm{DA}}$, $\mathbf{0}_{n_z \times n_{U_j}}$, $\mathbf{0}_{n_z \times n_{U_{j\mathrm{OC}}}}$, $\mathbf{0}_{n_z \times n_{U_{j\mathrm{NC}}}}$, and $\mathbf{0}_{n_z \times n_{U_{j\mathrm{DA}}}}$ respectively substituted for C_z, D_{zj}, $D_{zj\mathrm{OC}}$, $D_{zj\mathrm{NC}}$, $D_{zj\mathrm{DA}}$, $\mathbf{0}_{n_{z_c} \times n_{U_j}}$, $\mathbf{0}_{n_{z_c} \times n_{U_{j\mathrm{OC}}}}$, $\mathbf{0}_{n_{z_c} \times n_{U_{j\mathrm{NC}}}}$, and $\mathbf{0}_{n_{z_c} \times n_{U_{j\mathrm{DA}}}}$.

The matrices $S_{\Delta UU}$, $S_{\Delta U}$, $S_{\Delta U_j U_j}$, and $S_{\Delta U_j}$ that are used to determine $\Delta\bar{U}$ and $\Delta\bar{U}_j$are defined as

$$S_{\Delta UU} = \begin{bmatrix} -I_{n_U} & \mathbf{0}_{n_U \times n_U} & \dots & \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} \\ I_{n_U} & -I_{n_U} & \dots & \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} & \dots & -I_{n_U} & \mathbf{0}_{n_U \times n_U} \\ \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} & \dots & I_{n_U} & -I_{n_U} \end{bmatrix}, \tag{A.9}$$

$$S_{\Delta U} = \begin{bmatrix} -U(k-1) \\ \mathbf{0}_{n_U \times n_U} \\ \vdots \\ \mathbf{0}_{n_U \times n_U} \end{bmatrix}, \tag{A.10}$$

$$S_{\Delta U_j U_j} = \begin{bmatrix} -I_{n_{U_j}} & \mathbf{0}_{n_{U_j} \times n_{U_j}} & \dots & \mathbf{0}_{n_{U_j} \times n_{U_j}} & \mathbf{0}_{n_{U_j} \times n_{U_j}} \\ I_{n_{U_j}} & -I_{n_{U_j}} & \dots & \mathbf{0}_{n_{U_j} \times n_{U_j}} & \mathbf{0}_{n_{U_j} \times n_{U_j}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0}_{n_{U_j} \times n_{U_j}} & \mathbf{0}_{n_{U_j} \times n_{U_j}} & \dots & -I_{n_{U_j}} & \mathbf{0}_{n_{U_j} \times n_{U_j}} \\ \mathbf{0}_{n_{U_j} \times n_{U_j}} & \mathbf{0}_{n_{U_j} \times n_{U_j}} & \dots & I_{n_{U_j}} & -I_{n_{U_j}} \end{bmatrix}, \tag{A.11}$$

$$S_{\Delta U_j} = \begin{bmatrix} -U_j(k-1) \\ \mathbf{0}_{n_{U_j} \times n_{U_j}} \\ \vdots \\ \mathbf{0}_{n_{U_j} \times n_{U_j}} \end{bmatrix}, \tag{A.12}$$

where $U(k-1)$ and $U_j(k-1)$ represents control actions selected at the previous time step.

Appendix B

Constraint matrix definitions

In the centralized case, the input rate constraints are enforced using

$$A_{\mathrm{QP}\Delta U_{\min}} \bar{U} \leq b_{\mathrm{QP}\Delta U_{\min}}, \tag{B.1}$$

$$A_{\mathrm{QP}\Delta U_{\max}} \bar{U} \leq b_{\mathrm{QP}\Delta U_{\max}}, \tag{B.2}$$

$$A_{\mathrm{QP}\Delta U_{\min}} = \begin{bmatrix} -I_{n_U} & \mathbf{0}_{n_U \times n_U} & \dots & \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} \\ I_{n_U} & -I_{n_U} & \dots & \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} & \dots & -I_{n_U} & \mathbf{0}_{n_U \times n_U} \\ \mathbf{0}_{n_U \times n_U} & \mathbf{0}_{n_U \times n_U} & \dots & I_{n_U} & -I_{n_U} \end{bmatrix}, \tag{B.3}$$

$$b_{\mathrm{QP}\Delta U_{\min}} = \begin{bmatrix} -\Delta t\ \Delta U_{\min} - U(k-1) \\ -\Delta t\ \Delta U_{\min} \\ -\Delta t\ \Delta U_{\min} \\ \vdots \\ -\Delta t\ \Delta U_{\min} \end{bmatrix}, \tag{B.4}$$

$$A_{\mathrm{QP}\Delta U_{\max}} = -A_{\mathrm{QP}\Delta U_{\min}}, \tag{B.5}$$

$$b_{\mathrm{QP}\Delta U_{\max}} = \begin{bmatrix} \Delta t\ \Delta U_{\max} + U(k-1) \\ \Delta t\ \Delta U_{\max} \\ \Delta t\ \Delta U_{\max} \\ \vdots \\ \Delta t\ \Delta U_{\max} \end{bmatrix}. \tag{B.6}$$

Note that $A_{\mathrm{QP}\Delta U_{\max}} = -A_{\mathrm{QP}\Delta U_{\min}} = S_{\Delta UU}$. Soft constraints on the constrained states are enforced using

$$A_{\mathrm{QP}z_c}\bar{U}_\epsilon \leq b_{\mathrm{QP}z_c}, \tag{B.7}$$

$$A_{\mathrm{QP}z_c} = \begin{bmatrix} -S_{cU} & I_{n_{\bar{\epsilon}_{\min}}\times n_{\bar{\epsilon}_{\min}}} & \mathbf{0}_{n_{\bar{\epsilon}_{\max}}\times n_{\bar{\epsilon}_{\max}}} \\ S_{cU} & \mathbf{0}_{n_{\bar{\epsilon}_{\min}}\times n_{\bar{\epsilon}_{\min}}} & I_{n_{\bar{\epsilon}_{\max}}\times n_{\bar{\epsilon}_{\max}}} \end{bmatrix}, \tag{B.8}$$

$$b_{\mathrm{QP}z_c} = \begin{bmatrix} -\bar{z}_{c,\min} + S_{cx}x(k) + S_{cW_d}\bar{W}_d + S_{cW_z} \\ \bar{z}_{c,\max} - S_{cx}x(k) - S_{cW_d}\bar{W}_d - S_{cW_z} \end{bmatrix}. \tag{B.9}$$

The input rate and constrained states are combined as,

$$A_{\mathrm{QP}} = \begin{bmatrix} A_{\mathrm{QP}\Delta U_{\min}} & \mathbf{0}_{N_c n_U \times n_{\bar{\epsilon}}} \\ A_{\mathrm{QP}\Delta U_{\max}} & \mathbf{0}_{N_c n_U \times n_{\bar{\epsilon}}} \\ \multicolumn{2}{c}{A_{\mathrm{QP}z_c}} \end{bmatrix}, \tag{B.10}$$

$$b_{\mathrm{QP}} = \begin{bmatrix} b_{\mathrm{QP}\Delta U_{\min}} \\ b_{\mathrm{QP}\Delta U_{\max}} \\ b_{\mathrm{QP}z_c} \end{bmatrix}, \tag{B.11}$$

to form a single set of inequality constraints. The matrices that define the equality constraints used to impose the relationship between the steering angles given in eq. (4.41) are defined as

$$A_{\mathrm{QP}\delta} = \begin{bmatrix} k_\delta & -1 & \mathbf{0}_{1\times(n_U-2)} & & \mathbf{0}_{1\times n_U} & & \dots & & \mathbf{0}_{1\times n_U} \\ & \mathbf{0}_{1\times n_U} & & k_\delta & -1 & \mathbf{0}_{1\times(n_U-2)} & \dots & & \mathbf{0}_{1\times n_U} \\ & \vdots & & & \vdots & & \ddots & & \vdots \\ & \mathbf{0}_{1\times n_U} & & & \mathbf{0}_{1\times n_U} & & \dots & k_\delta & -1 & \mathbf{0}_{1\times(n_U-2)} \end{bmatrix}, \tag{B.12}$$

$$A_{\mathrm{QPE}} = \begin{bmatrix} A_{\mathrm{QP}\delta} & \mathbf{0}_{N_c\times n_{\bar{\epsilon}}} \end{bmatrix}, \tag{B.13}$$

$$b_{\mathrm{QPE}} = [\mathbf{0}_{N_p\times 1}]. \tag{B.14}$$

In the non-centralized case, the input rates are enforced using

$$A_{\mathrm{QP}j\Delta U_{\min}}\bar{U}_j \leq b_{\mathrm{QP}j\Delta U_{\min}}, \tag{B.15}$$

$$A_{\mathrm{QP}j\Delta U_{\max}}\bar{U}_j \leq b_{\mathrm{QP}j\Delta U_{\max}}, \tag{B.16}$$

$$A_{\mathrm{QP}j\Delta U_{\min}} = \begin{bmatrix} -I_{n_U} & \mathbf{0}_{n_{U_j}\times n_{U_j}} & \dots & \mathbf{0}_{n_{U_j}\times n_{U_j}} & \mathbf{0}_{n_{U_j}\times n_{U_j}} \\ I_{n_U} & -I_{n_U} & \dots & \mathbf{0}_{n_{U_j}\times n_{U_j}} & \mathbf{0}_{n_{U_j}\times n_{U_j}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0}_{n_{U_j}\times n_{U_j}} & \mathbf{0}_{n_{U_j}\times n_{U_j}} & \dots & -I_{n_U} & \mathbf{0}_{n_{U_j}\times n_{U_j}} \\ \mathbf{0}_{n_{U_j}\times n_{U_j}} & \mathbf{0}_{n_{U_j}\times n_{U_j}} & \dots & I_{n_U} & -I_{n_U} \end{bmatrix}, \tag{B.17}$$

$$b_{\mathrm{QP}j\Delta U_{\min}} = \begin{bmatrix} -\Delta t\ \Delta U_{j,\min} - U_j(k-1) \\ -\Delta t\ \Delta U_{j,\min} \\ -\Delta t\ \Delta U_{j,\min} \\ \vdots \\ -\Delta t\ \Delta U_{j,\min} \end{bmatrix}, \tag{B.18}$$

$$A_{\mathrm{QP}j\Delta U_{\max}} = -A_{\mathrm{QP}j\Delta U_{\min}}, \tag{B.19}$$

$$b_{\mathrm{QP}j\Delta U_{\max}} = \begin{bmatrix} \Delta t\ \Delta U_{j,\max} + U_j(k-1) \\ \Delta t\ \Delta U_{j,\max} \\ \Delta t\ \Delta U_{j,\max} \\ \vdots \\ \Delta t\ \Delta U_{j,\max} \end{bmatrix}. \tag{B.20}$$

Note that $A_{\mathrm{QP}j\Delta U_{\max}} = -A_{\mathrm{QP}j\Delta U_{\min}} = S_{\Delta U_j U_j}$. The matrices used to enforce the soft state constraints are defined as

$$A_{\mathrm{QP}jz_c}\bar{U}_{j\epsilon} \le b_{\mathrm{QP}jz_c}, \tag{B.21}$$

$$A_{\mathrm{QP}jz_c} = \begin{bmatrix} -S_{cU_j} & I_{n_{\bar{\epsilon}_{\min}}\times n_{\bar{\epsilon}_{\min}}} & \mathbf{0}_{n_{\bar{\epsilon}_{\max}}\times n_{\bar{\epsilon}_{\max}}} \\ S_{cU_j} & \mathbf{0}_{n_{\bar{\epsilon}_{\min}}\times n_{\bar{\epsilon}_{\min}}} & I_{n_{\bar{\epsilon}_{\max}}\times n_{\bar{\epsilon}_{\max}}} \end{bmatrix}, \tag{B.22}$$

$$b_{\mathrm{QP}jz_c} = \begin{bmatrix} -\bar{z}_{c,\min} + S_{cx}x(k) + S_{U_{j\mathrm{OC}}}\bar{U}_{j\mathrm{OC}} + S_{U_{\mathrm{NC}}}\bar{U}_{\mathrm{NC}} + S_{U_{\mathrm{DA}}}\bar{U}_{\mathrm{DA}} + S_{cW_d}\bar{W}_d + S_{cW_z} \\ \bar{z}_{c,\max} - S_{cx}x(k) - S_{U_{j\mathrm{OC}}}\bar{U}_{j\mathrm{OC}} - S_{U_{\mathrm{NC}}}\bar{U}_{\mathrm{NC}} - S_{U_{\mathrm{DA}}}\bar{U}_{\mathrm{DA}} - S_{cW_d}\bar{W}_d - S_{cW_z} \end{bmatrix}. \tag{B.23}$$

The matrices used to implement the full set of inequality constraints are

$$A_{\mathrm{QP}j} = \begin{bmatrix} A_{\mathrm{QP}j\Delta U_{\min}} & \mathbf{0}_{N_c n_{U_j}\times n_{\bar{\epsilon}}} \\ A_{\mathrm{QP}j\Delta U_{\max}} & \mathbf{0}_{N_c n_{U_j}\times n_{\bar{\epsilon}}} \\ A_{\mathrm{QP}z_c} & \end{bmatrix}, \tag{B.24}$$

$$b_{\mathrm{QP}j} = \begin{bmatrix} b_{\mathrm{QP}j\Delta U_{\min}} \\ b_{\mathrm{QP}j\Delta U_{\max}} \\ b_{\mathrm{QP}jz_c} \end{bmatrix}. \tag{B.25}$$

The matrices used to construct the equality constraints that establish a relationship between δ_f and δ_r are defined as

$$A_{\mathrm{QP}j\delta} = \begin{bmatrix} k_\delta & -1 & \mathbf{0}_{1\times(n_{U_j}-2)} & & \mathbf{0}_{1\times n_{U_j}} & \dots & & \mathbf{0}_{1\times n_{U_j}} \\ & \mathbf{0}_{1\times n_{U_j}} & & k_\delta & -1 & \mathbf{0}_{1\times(n_{U_j}-2)} & \dots & \mathbf{0}_{1\times n_{U_j}} \\ & \vdots & & & \vdots & \ddots & & \vdots \\ & \mathbf{0}_{1\times n_{U_j}} & & & \mathbf{0}_{1\times n_{U_j}} & \dots & k_\delta & -1 & \mathbf{0}_{1\times(n_{U_j}-2)} \end{bmatrix}, \tag{B.26}$$

$$A_{\mathrm{QPE}j} = \begin{bmatrix} A_{\mathrm{QP}j\delta} & \mathbf{0}_{N_c\times n_{\bar{\epsilon}}} \end{bmatrix}, \tag{B.27}$$

$$b_{\mathrm{QPE}j} = [\mathbf{0}_{N_p\times 1}]. \tag{B.28}$$

Note that this constraint relationship may only be used if some agent j handles both δ_f and δ_r.