High Speed Obstacle Avoidance at the Dynamic Limits for Autonomous Ground Vehicles

by

Jiechao Liu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in the University of Michigan
2017

Doctoral Committee:

Professor Jeffrey L. Stein, Co-Chair
Assistant Research Scientist Tulga Ersal, Co-Chair
Doctor Paramsothy Jayakumar, TARDEC
Professor Ilya V. Kolmanovsky
Doctor Steve M. Rohde, Quantum Signal
Professor Dawn M. Tilbury

2017

This dissertation is dedicated to my family.

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincerest appreciation to my advisors, Prof. Jeffrey L. Stein and Dr. Tulga Ersal, for their continuous guidance and support through my graduate study. Their invaluable mentoring and encouragement helped me overcome many challenges and difficulties in my research.

Prof. Stein has been very helpful with big picture details. He has provided a lot of constructive advices on how to make the research work complete and cohesive and how to present the work with concise and precise messages. Dr. Ersal's hands-on help at the beginning of the project and continuous guidance to the end of the project are all-important. I have greatly benefited from his brilliance, dedication, and kindness. My writings has greatly benefited from his excellent critiquing and review. It has been my great fortune to have worked with both of my advisors. I have also worked closely with Dr. Paramsothy Jayakumar. I deeply appreciate all the inputs and resources that he has provided. His critical comments and insightful suggestions greatly improved the quality of the work.

I also would like to thank my other committee members, Prof. Dawn Tilbury, Prof. Ilya Kolmanovsky, and Dr. Steve Rohde, for their enormously helpful advice and comments. I truly appreciate their time and effort they volunteered to guide this dissertation work.

The experimental tests are conducted with the support of Quantum Signal, LLC. Dr. Mitchell Rohde, and Dr. Steve Rohde provided the access to the vehicle platform. James M. Walker and Chris Showers helped with integrating the interfaces of the DRIVER system and all sensors in the program. James Walker also helped with scheduling the test space and operating the platform. Kevin Melotti provided the vehicle data set and helped with installing the LIDAR sensor. Yingshi Zheng and Huckleberry Febbo from University of Michigan also provided valuable help during the tests. I am extremely thankful of them for their help.

I am also deep thankful of Brian Rapp and Timothy Mattox from Army Research Laboratory and Longhan Li and Kun Zhang from the University of Michigan to help with improving the computational performance of the algorithm.

As for my funding, I am grateful for the support from the Automotive Research Center at the University of Michigan. I would like to recognise support from the Rackham Graduate School, the College of Engineering, and the Department of Mechanical Engineering.

Most of the chapters of this dissertation are heavily based on manuscripts that have been published or have been submitted for publication. Specifically, Chapter 3

is based on [1, 4], Chapter 5 on [2, 5], Chapter 6 on [3, 6], and Chapter 7 on [7, 8]. I gratefully acknowledge the contributions of my co-authors on those publications.

Many thanks are given to my colleagues in Automated Modeling Lab for their company and ideas - Xinyi Ge, Xin Zhou, Huckleberry Febbo, Yingshi Zheng, Alireza Goshtasbi, and Hossein Mirinejad.

Great appreciation should be given to my friends Jinhong Qu, Peihong Wu, Kai Chen, Lei Yu, Mingjie Xu, Jiachen Chen, and many others. I also greatly appreciate Xingzhi Qiu for listening to my complaints in my rough time.

Finally, I would like to thank my parents, Jianxing Liu and Hongyan Shi, and all my family members, for their unconditional and perennial love and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF ACRONYMS

AGV Autonomous ground vehicle

BFCF body-fixed coordinate frame

CoG center of gravity

DoF degrees-of-freedom

LGR Legendre-Gauss-Radau

LHD Latin Hypercube Design

LIDAR light detection and ranging

MPC model predictive control

NLMPC nonlinear MPC

NLP nonlinear programming

OCP optimal control problem

ODE ordinary differential equation

LIST OF SYMBOLS

Model Parameters and Variables

$()_\bullet$	Front left(fl), front right (fr), rear left (rr), or rear right (rr)	[-]
$()_\star$	Front (f), or rear(r)	[-]
C_q	Brake effective flow coefficient	[$\text{cm}^3/\text{s}\sqrt{\text{kPa}}$]
K_b	Brake gain	[-]
P_{mc}	Pressure in master cylinder	[Pa]
P_{po}	Push-out pressure	[kPa]
P_w	Pressure in wheel cylinder	[kPa]
$\tau_\bullet$	Driving/brake torque	[N-m]
τ_{brake}	Brake torque	[N-m]
V_{brake}	Brake cylinder volume	[cc]
$e_{\text{differential}}$	Differential efficiency	[-]
$r_{\text{differential}}$	Differential speed ratio	[-]
I_{engine}	Engine inertia	[kg-m^2]
I_{turbine}	Turbine inertia	[kg-m^2]
ω_{driving}	Driving speed	[rad/s]
ω_{engine}	Engine speed	[rad/s]
ω_{pump}	Pump speed	[rad/s]
ω_{shaft}	Shaft speed	[rad/s]
ω_{turbine}	Turbine speed	[rad/s]
τ_{driving}	Driving torque	[N-m]
τ_{engine}	Engine torque	[N-m]
τ_{pump}	Pump/load torque	[N-m]
τ_{shaft}	Shaft torque	[N-m]
τ_{turbine}	Turbine torque	[N-m]
F_{aero}	Aerodynamic resistance	[N]
$F_{dz,\bullet}$	Jacking force	[N]
$F_{sx,\bullet}/F_{sy,\bullet}/F_{sz,\bullet}$	Longitudinal/lateral/vertical forces transferred to	

	the sprung mass in the BFCF	[N]
$F_{y,f}$	Tire lateral force generated at the front axle	[N]
$F_{y,r}$	Tire lateral force generated at the rear axle	[N]
$F_{z,f0}$	Static tire vertical load at the front axle	[N]
$F_{z,f}$	Dynamic tire vertical load at the front axle	[N]
$F_{z,r0}$	Static tire vertical load at the rear axle	[N]
$F_{z,r}$	Dynamic tire vertical load at the rear axle	[N]
$F_{tx,\bullet}/F_{ty,\bullet}/F_{tz,\bullet}$	Longitudinal/lateral/vertical forces generated at the tire-ground contact patch in the tire-fixed coordinate frame	[N]
$F_{uz,\bullet}$	Vertical force transferred to the unsprung mass in BFCF	[N]
h_{CG}	Height of vehicle CoG location above the ground	[m]
$h_{u,f}$	Height of front unsprung mass CoG location above the ground	[m]
$h_{u,r}$	Height of rear unsprung mass CoG location above the ground	[m]
I_w	Wheel rotational inertial	[kg-m^2]
I_{xx}	Roll inertia of sprung Mass	[kg-m^2]
I_{yy}	Pitch inertia of sprung Mass	[kg-m^2]
I_{zz}	Yaw inertia of sprung Mass	[kg-m^2]
C_α	Cornering stiffness	[N/rad]
$k_{\text{susp},\bullet}$	Suspension spring stiffness	[N/m]
$b_{\text{susp},\bullet}$	Suspension damping coefficient	[N-s/m]
$K_{z,x}$	Longitudinal load transfer coefficient	[N/(m/s^2)]
$K_{z,yf}$	Front axle lateral load transfer coefficient	[N/(m/s^2)]
$K_{z,yr}$	Rear axle lateral load transfer coefficient	[N/(m/s^2)]
L_f	Distance between the CoG and the front axle	[m]
L_r	Distance between the CoG and the rear axle	[m]
L_t	Track width	[m]
L_v	Distance between front and rear axles	[m]
R_t	Tire radius	[m]
M_s	Sprung mass	[kg]
M_u	Unsprung mass	[kg]
$m_\bullet$, $m_\star$	Unsprung mass	[kg]
M_v	Vehicle total Mass	[kg]
ω_x	Roll angular velocity	[rad/s]
ω_y	Pitch angular velocity	[rad/s]
ω_z	Yaw angular velocity	[rad/s]

ϕ	Roll angle	[rad]
θ	Pitch angle	[rad]
ψ	Yaw angle	[rad]
P_c	Shape factor for lateral force	[-]
$P_{d,z}$	Variation of lateral friction with load	[-]
P_d	Lateral friction	[-]
$P_{e,z}$	Variation of lateral curvature with load	[-]
P_e	Lateral curvature	[-]
$P_{k,z}$	Load at which stiffness reaches maximum value	[-]
P_k	Maximum value of stiffness	[-]
α_f	Front tire slip angle	[rad]
α_r	Rear tire slip angle	[rad]
$\alpha_\bullet$	Slip angle of the tire	[rad]
$\kappa_\bullet$	Slip ratio	[-]
δ_f	Front wheel steering angle	[rad]
$\delta_\star$	Steering angle	[rad]
η_f	Front wheel steering acceleration	[rad/s^2]
γ_f	Front wheel steering rate	[rad/s]
a_x	Longitudinal acceleration	[m/s^2]
J_x	Longitudinal jerk	[m/s^3]
U_0	Constant reference longitudinal velocity	[m/s]
U	Longitudinal velocity in BFCF	[m/s]
V	Lateral velocity in BFCF	[m/s]
W	Vertical velocity in BFCF	[m/s]
$w_{s,\bullet}$	Vertical velocities at the suspension corner in BFCF	[m/s]
$u_{u,\bullet}/v_{u,\bullet}/w_{u,\bullet}$	Longitudinal/lateral/vertical velocities of the unsprung mass in BFCF	[m/s]
$u_{t,\bullet}$	Longitudinal velocity at tire contact patch	[m/s]
$v_{t,\bullet}$	Lateral velocity at tire contact patch	[m/s]
$x_{\text{susp},\bullet}$	Suspension spring compression	[m]
(x, y)	Vehicle front center location in inertial coordinates	[m]
X	Longitudinal location of the sprung mass in inertial frame	[m]
Y	Lateral location of the sprung mass in inertial frame	[m]
Z	Vertical location of the sprung mass in inertial frame	[m]

Algorithm Parameters and Variables

$()^{(i)}$	Functions or variables of i-th sub-region, $i = 1, \ldots, N$	[-]
$()_{\max}$	Upper bound value	[-]
$()_{\min}$	Lower bound value	[-]
$()_{\text{load}}$	Most likely smallest vertical load scenario	[-]
$()_{\text{traj}}$	Most likely stiffest trajectory scenario	[-]
$\boldsymbol{\mathcal{F}}(\cdot)$	Generic terminal constraint	[-]
$\boldsymbol{\mathcal{G}}(\cdot)$	Generic path constraint	[-]
$\mathcal{I}(\cdot)$	Integrand of generic integral cost	[-]
$\mathcal{J}(\cdot)$	Generic cost function	[-]
$\boldsymbol{\mathcal{M}}(\cdot)$	Generic model function	[-]
$\mathcal{P}(\cdot)$	Pacejka tire model function	[-]
$\boldsymbol{\mathcal{R}}(\cdot)$	Generic safe region constraint	[-]
$\boldsymbol{\mathcal{S}}(\cdot)$	Generic dynamical safety constraint	[-]
$\mathcal{T}(\cdot)$	Generic terminal cost	[-]
$\boldsymbol{\mathcal{V}}(\cdot)$	Generic vehicle model function	[-]
C_{traj}	Curvature of trajectory	[1/m]
E_{control}	Control effort	[^{circ}-s]
$I_{\text{curvature}}$	Integral of curvature of trajectory	[s/m]
T_{total}	Total navigation time to reach the target	[s]
$F_{z,a}$	Small vertical load penalty parameter	[-]
$F_{z,b}$	Small vertical load penalty parameter	[-]
ψ_{diff}	Difference between the vehicle final heading angle and the desired heading angle	[rad]
s_0	Distance between the vehicle current position and the target	[m]
s_f	Distance between the end of the prediction and the target	[m]
v_{cf}	Control effort metric value	[-]
v_{dt}	Heading direction metric value	[-]
w_ψ	Cost function weighting term	[-]
w_{cf}	Cost function weighting term	[-]
w_{dt}	Cost function weighting term	[-]
w_δ	Cost function weighting term	[-]
w_t	Cost function weighting term	[-]
w_{fz}	Cost function weighting term	[-]
w_γ	Cost function weighting term	[-]
w_J	Cost function weighting term	[-]
$c_1, \ \ldots, c_8$	Polynomial fitting coefficients for acceleration bounds	[-]
$F_{z,\text{threshold}}$	The vertical load threshold	[N]
R_σ	Prediction distance relaxation constant	[m]

σ	Target arrival margin	[m]
$U_{\text{threshold}}$	The maximum terminal velocity	[m/s]
ϵ	LIDAR angular resolution	[°]
$L_{\max}$	Maximum obstacle size	[m]
R_{LIDAR}	LIDAR maximum detection range	[m]
n_p	Number of intervals over the prediction horizon	[-]
n_s	Number of elements in the steering angle pool	[-]
$\boldsymbol{A}^{(i)}_{L^{(i)}\times 1}$	Vectors used for the representation of the i-th sub-region	[-]
$\boldsymbol{B}^{(i)}_{L^{(i)}\times 1}$	Vectors used for the representation of the i-th sub-region	[-]
$\boldsymbol{C}^{(i)}_{L^{(i)}\times 1}$	Vectors used for the representation of the i-th sub-region	[-]
$L^{(i)}$	Total number of lines bounding the i-th sub-region	[-]
$\mathcal{M}_{\text{adjacency}}$	Adjacency matrix	[-]
N	Total number of sub-regions to transverse	[-]
$\boldsymbol{\xi}^0$	State vector initial value	[-]
$\boldsymbol{\xi}^*$	Optimal state vector	[-]
$\boldsymbol{\xi}$	State vector	[-]
$\boldsymbol{\zeta}^*$	Optimal control vector	[-]
$\boldsymbol{\zeta}$	Control vector	[-]
t_0	Initial time	[s]
T_e	Execution horizon	[s]
T_p	Prediction horizon	[s]
t	time	[s]
T^i	Time for transitioning from one sub-region to the next	[s]
ψ_t	Desired final heading angle	[deg]
$(x_t,\, y_t)$	Target location in inertial coordinates	[m]

ABSTRACT

High Speed Obstacle Avoidance at the Dynamic Limits for Autonomous Ground Vehicles

by

Jiechao Liu

Chair: Jeffrey L. Stein and Tulga Ersal

Enabling autonomy of passenger-size and larger vehicles is becoming increasingly important in both military and commercial applications. For large autonomous ground vehicles (AGVs), the vehicle dynamics are critical to consider to ensure vehicle safety during obstacle avoidance maneuvers especially at high speeds. This research is concerned with large-size high-speed AGVs with high center of gravity (CoG) that operate in unstructured environments. The term 'unstructured' in this context denotes that there are no lanes or traffic rules to follow. No map of the environment is available *a priori*. The environment is perceived through a planar light detection and ranging sensor. The mission of the AGV is to move from its initial position to a given target position safely and as fast as possible.

In this dissertation, a model predictive control (MPC)-based obstacle avoidance algorithm is developed to achieve the objectives through an iterative simultaneous optimization of the path and the corresponding control commands. MPC is chosen because it offers a rigorous and systematic approach for taking vehicle dynamics and safety constraints into account.

Firstly, this thesis investigates the level of model fidelity needed in order for a MPC-based obstacle avoidance algorithm to be able to safely and quickly avoid obstacles even when the vehicle is close to its dynamic limits. Five different representations of vehicle dynamics models are considered: four variations of the two Degrees of-Freedom (DoF) representation as lower fidelity models and a fourteen DoF representation with combined-slip Magic Formula tire model as a higher fidelity model. It is concluded that the two DoF representation that accounts for tire nonlinearities and longitudinal load transfer is necessary for the MPC-based obstacle avoidance algorithm in order to operate the vehicle at its limits within an environment that includes large obstacles. For less challenging environments, however, the two DoF representation with linear tire model and constant axle loads is sufficient.

Secondly, existing MPC formulations for passenger vehicles in structured environments do not readily apply to this context. Thus, a novel nonlinear MPC formulation is developed. First, a new cost function formulation is used that aims to find the shortest path to the target position, since no reference trajectory exists in unstructured environments. Second, a region partitioning approach is used in conjunction with a multi-phase optimal control formulation to accommodate the complicated forms the obstacle-free region can assume due to the presence of multiple obstacles along the prediction horizon in an unstructured environment. Third, the no-wheel-lift-off condition, which is the major dynamical safety concern for high-speed, high-CoG AGVs, is established offline using a fourteen DoF vehicle dynamics model and is included in the MPC formulation. A formulation is first developed by assuming a constant-speed operation. It is then extended with the capability of simultaneous optimization of both steering angle and reference longitudinal speed commands. Simulation results show that the proposed algorithm is capable of safely exploiting the dynamic limits of the vehicle while navigating the vehicle through sensed obstacles of different size and number.

Thirdly, in the algorithm, a model of the vehicle is used explicitly to predict and optimize future actions, but in practice the model parameter values are not exactly known. Thus, the robustness of the algorithm to parametric uncertainty is also evaluated. It is demonstrated that using nominal parameter values in the algorithm leads to safety issues in about one fourth of the evaluated scenarios with the considered parametric uncertainty distributions. To improve the robustness of the algorithm, a novel double-worst-case formulation is developed that simultaneously accounts for the robust satisfaction of the two safety requirements of high-speed obstacle avoidance: collision-free and no-wheel-lift-off. Results from simulations with stratified random scenarios and worst-case scenarios show that the double-worst-case formulation considering both the most likely worst-case scenarios and the less likely worst-case scenarios renders the algorithm robust to all uncertainty realizations tested. The trade-off between the robustness and the task completion performance of the algorithm is also quantified.

Finally, in addition to simulation-based validation, preliminary experimental validation is also performed. These results demonstrate that the developed algorithm is promising in terms of its capability of avoiding obstacles. Limitations and potential improvements of the algorithm are discussed.

CHAPTER 1

Introduction

1.1 Motivation

Autonomous ground vehicles (AGVs) hold great potential for increased safety, performance, and convenience and thus have attracted interest for both military and commercial applications. In the military field, the extensive use of remotely controlled ground robots over the past decade has demonstrated the benefits of unmanned ground vehicles [9] and motivated the transition to fully autonomous large vehicles as a natural extension. As an example, the Autonomous Platform Demonstrator is a 9-ton vehicle that has been used to develop, integrate, and test many next generation unmanned ground vehicle mobility technologies. It is capable of performing maneuvers at speeds up to 22.4 m/s [10]. In the commercial field, almost all major automobile manufactures and some technology companies are pursuing autonomous passenger vehicles to achieve improved safety, reduced congestion, increased convenience and many other potential advantages [11].

The development of AGVs for military versus commercial applications poses different research challenges due to the contextual differences. Military AGVs need the capability to operate in unstructured environments and maneuver at high speeds to respond rapidly to changing threats and priorities [12]. Thus, operating the AGVs at their dynamic limits is a critical research problem for military applications [13]. Commercial AGVs, on the other hand, need to be able to navigate in urban and highway environments safely. Within that context, operating the AGV at its dynamic limits is less relevant; instead, the traffic and its rules become critical considerations. Hence, the main research focus for commercial AGVs is on perception and localization [14].

Obstacle avoidance is a critical capability for both military and commercial AGVs.

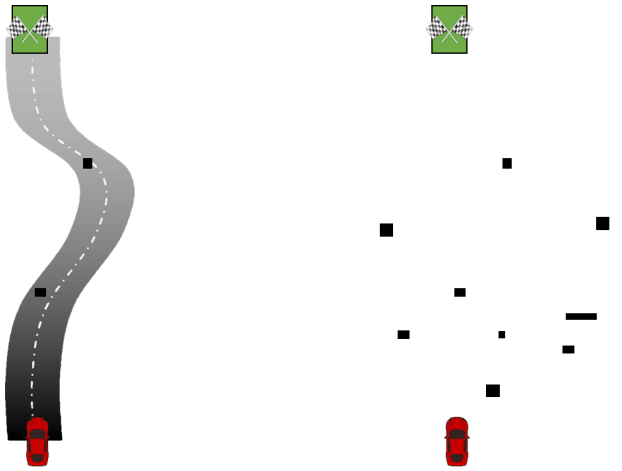
It refers to the task of sensing the vehicle's surroundings and generating control commands to navigate the vehicle safely around the obstacles [15] as demonstrated in Figure 1.1. Obstacle avoidance for small ground robots is a relatively more mature research area. Many obstacle avoidance algorithms have been developed in the literature that allow for fast, continuous, and smooth motion of small AGVs among unexpected obstacles. Some early examples include the Artificial Potential Field method [16, 17], the Vector Field Histogram method [18, 19], the Dynamic Window approach [20, 21], and the Curvature-Velocity method [22]. Reviews and comparisons of these algorithms can be found in [23, 24, 25]. These algorithms were originally developed for small ground robots and mainly focus on finding collision-free paths.

However, for large vehicles at high speed, stability or handling issues such as excessive side slip, wheel lift-off or rollover may be induced by the obstacle avoidance maneuver if the controller does not take into account the dynamic constraints properly [26]. One intuitive approach to avoid these failures is to operate at speeds that minimize the impact of the vehicle dynamics. However, it is not good from a mission performance perspective because mission completion time and mobility are key performance issues for military applications. It is preferred to operate the vehicle at its dynamical limits during emergency situations, which means that the vehicle should travel as fast as possible and steer as sharp as possible if needed, while still maintaining its dynamic safety. Thus, for military AGVs, obstacle avoidance algorithms are needed that can exploit the vehicle dynamics to avoid collisions even when the vehicle is operating at its handling limits.

Figure 1.1: Graphical illustration of obstacle avoidance.

1.2 Objective and Scope of the Study

The objective of this thesis is to develop and test an obstacle avoidance algorithm for large AGVs at high speeds that avoids sensed obstacles in unstructured, unknown environments. The term 'unstructured environment' in this context denotes that there are no roadways to follow and no traffic rules to obey as shown in Figure 1.2. The term 'unknown environment' is used in the sense that no map of the environment is available. Thus, the location, size, and shape of obstacles are unknown *a priori* but rather are detected when they come into range of the planar light detection and ranging (LIDAR) sensor. To limit the scope of the problem, it is assumed that the vehicle moves on flat terrain and all the obstacles are static. The obstacles are assumed to be detected through a LIDAR sensor as shown in Figure 1.3.



(a) Structured environment (b) Unstructured environment

Figure 1.2: Comparison of structured and unstructured environments.

The mission of the AGV is to move from its initial position to a given target position safely and as quickly as possible. The safety requirement is not only to prevent collisions but also to ensure the vehicle's dynamical safety. Thus the vehicle is expected to be traveling at high speed with no unnecessary deceleration.

A typical military truck is considered as a representative large-size, high-speed AGV with significant vehicle dynamics. For the particular vehicle of interest, the vehicle dynamical safety requirement is defined as avoiding single-wheel lift-off, because wheel lift-off occurs before excessive sideslip [27]. This is due to the fact that the truck has a high center of gravity (CoG). In addition, single-wheel lift-off is a conservative criterion for preventing rollover as illustrated in Figure 1.4.

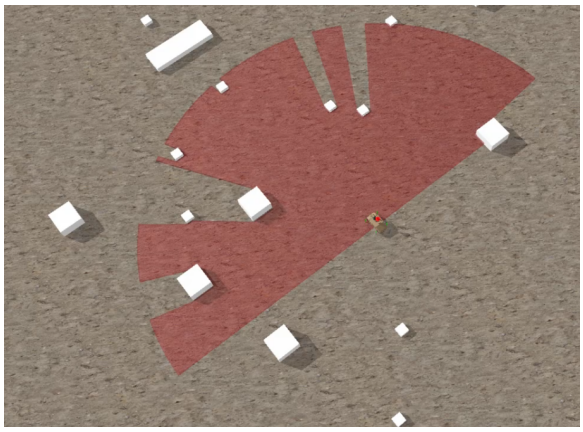
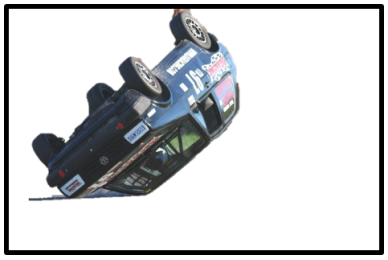


Figure 1.3: Example scene illustrating how the vehicle perceives the environment using a LIDAR sensor.

(a) Single-wheel lift-off [1]

(b) Two-wheel lift-off [2]



(c) Rollover [3]

Figure 1.4: Violations of vehicle dynamical safety.

[1]Single-wheel lift-off. http://www.ft86club.com/forums/showthread.php?t=40003&page=2

[2]Two-wheel lift-off. http://www.jk-forum.com/forums/jeep-xj-cherokee-grand-cherokees-35/2012-jeep-grand-cherokee-rollover-risks-chrysler-engineers-invited-new-testing-241707/

[3]Rollover. http://turbozens.com/safetytechnologyautomobiles/

1.3 Overview of Motion Generation Algorithms

The online obstacle avoidance algorithm is often used in complement to a global path planning algorithm in motion generation for autonomous vehicles. The global path planning algorithm is used to generate some coarse way-points and online obstacle avoidance algorithm is used to navigate the vehicle between the selected way-points safely. Table 1.1 summarizes the differences between the two categories of the algorithms [24, 28].

Table 1.1: Comparison of global path planning and online obstacle avoidance algorithms

	Global Path Planning	**Online Obstacle Avoidance**
Primary task	Allows the vehicle to avoid being trapped with large obstacles, undesirable terrain or dead ends	Allows the vehicle to travel stably and safely
Obstacles concerned	Large scale obstacles: hills, rivers, buildings, etc.	Small scale obstacles and obstacles that are not in the original map
Information	Complete map	Real-time sensing data
Plan range	Plans for long distance and time period	Plans for immediate vicinity for a short time ahead
Computation time	No requirement to run in real-time	Must run in real-time
Vehicle dynamics considered	None / Non-holonomic and kinodynamic constraints	Non-holonomic and kinodynamic constraints / Complex dynamics model

The algorithm developed in this thesis is an online obstacle avoidance algorithm. However, because a complete map of the environment is not available, a reference trajectory that accounts for large obstacles cannot be generated. Thus, relatively large obstacles, such as buildings, need to be avoided using the online obstacle avoidance algorithm and a long distance and time period planning is required.

Numerous motion planning algorithms for different types of AGVs have been developed in the literature. They can be classified into the following four categories [29].

The first category is methods based on geometrical model searching [15, 30]. A geometrical model of the environment including the obstacles is first constructed. Then, based on the resulting graph or tree model, these methods leverage the mature graph search algorithms from the computer science to search for a feasible or near-

optimal solution. Some of the techniques include cell decomposition [31], visibility graph [32], Voronoi diagram [33], probabilistic roadmaps [34]. However, this type of algorithms is focused only on finding a collision-free path and it is difficult to incorporate the dynamic safety constraints.

The second category is methods based on potential field and navigation function [16, 26]. The fundamental idea of this approach is to use repulsive potential fields around the obstacles and forbidden regions to force the vehicle away and use an attractive potential field around target to attract the vehicle. Then, the vehicle experiences a generalized force equal to the negative of the total potential gradient. This force drives the robot towards the target position until it reaches a minimum point [35]. The advantage is that they require very low computation time. However, the algorithm only generates instantaneous commands, thus, the actions are not anticipative. Some of the techniques in this category include artificial potential field [16], harmonic function field [36], and flow field [37].

The third category is methods based on meta-heuristic algorithms [38]. A meta-heuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space. Learning strategies are used to structure information in order to find efficiently near-optimal solutions [39]. Thus, the algorithms combine heuristics and randomness to find efficiently near-optimal solutions with the presence of complex non-structured constraints. However, because the search space is discretized, the generated trajectories are not smooth and the computational loads are high. Some of the techniques include particle swarm [40], ant colony [41], and genetic algorithm [42].

The last category is methods based on mathematical optimization [43]. Mathematical optimization deals with the problem of finding numerically a minimum of a function, which is called the cost function, or objective function. For the application of path planning, it could be used to generate optimal solutions in a rigorous way. Also, dynamic constraints can be included, although a differentiable mathematical representation of all constraints is required. The computation time required is higher than the potential field and navigation function-based methods. But it is typically lower than the computation time required in the meta-heuristic algorithms. However, a collection of methods exist to improve the computational performance [44].

Among these categories, mathematical optimization based methods are particularly attractive for this application because they offer a rigorous and systematic approach for taking vehicle dynamics and safety constraints into account.

A mathematical optimization approach can be used either in open-loop, planning an optimal path from an initial point to a target point offline and following the path with a feedback controller online, or in closed loop, performing the optimization online with regular vehicle state information updates and environmental information updates from sensors. Regarding the latter, the model predictive control (MPC) approach is the most widely adopted technique [45]. In this approach, an optimal control problem (OCP) is solved repeatedly over a receding finite horizon. The resulting control strategy from the OCP solution at the current step is executed until a new OCP is solved with updated information. The concept of MPC is reviewed in the next Section.

1.4 Basic Principle of MPC

MPC is an advanced control technique that utilizes a model of the system to predict and optimize future system behavior over a receding finite time horizon. It is an optimal control based state-feedback controller. Only the initial portion of the control command optimized over the prediction horizon is applied to the plant. A new OCP is solved over a shifted finite time horizon at the next time step using the state feedback received from the plant. This process is explained in detail below.

The basic principle of MPC is illustrated in Figure 1.5. At time t_0, starting from the current state measurements, an optimal control sequence $\boldsymbol{\zeta}^*(t)$, $t \in [t_0, t_0 + T_p]$, is computed by solving an open-loop, constrained, finite-time OCP over the prediction horizon T_p. Thus, the control command generation is based on both future predictions and current measurements. The general form of the OCP formulation is given by

$$\underset{\boldsymbol{\xi},\, \boldsymbol{\zeta},\, T_p}{\text{minimize}} \quad \mathcal{J}\left[\boldsymbol{\xi}\left(t\right),\ \boldsymbol{\zeta}\left(t\right),\ \boldsymbol{\xi}\left(t_0\right),\ \boldsymbol{\xi}\left(t_0 + T_p\right), T_p\right] \tag{1.1}$$

$$\text{subject to} \quad \dot{\boldsymbol{\xi}}\left(t\right) = \boldsymbol{\mathcal{M}}\left[\boldsymbol{\xi}\left(t\right),\ \boldsymbol{\zeta}\left(t\right)\right] \tag{1.2}$$

$$\boldsymbol{\mathcal{G}}\left[\boldsymbol{\xi}\left(t\right),\ \boldsymbol{\zeta}\left(t\right)\right] \leq 0 \tag{1.3}$$

$$\boldsymbol{\mathcal{F}}\left[\boldsymbol{\xi}\left(t_0\right),\ \boldsymbol{\xi}\left(t_0 + T_p\right)\right] \leq 0 \tag{1.4}$$

$$\boldsymbol{\xi}_{\min}\left(t\right) \leq \boldsymbol{\xi}\left(t\right) \leq \boldsymbol{\xi}_{\max}\left(t\right) \tag{1.5}$$

$$\boldsymbol{\zeta}_{\min}\left(t\right) \leq \boldsymbol{\zeta}\left(t\right) \leq \boldsymbol{\zeta}_{\max}\left(t\right) \tag{1.6}$$

$$t \in [0, T_p],\ 0 < T_{p,\min} \leq T_p \leq T_{p,\max} \tag{1.7}$$

By minimizing the cost function Equation (1.1) subject to constraints Equation (1.2) - Equation (1.7), the optimal control sequence $\boldsymbol{\zeta}^*(t)$ is bounded by the control in-

put saturation, and the resulting estimated optimal states $\boldsymbol{\xi}^*(t)$ satisfy the state constraints. Regarding the constraints, Equation (1.2) is the model of the system represented by a set of first-order ordinary differential equations. Equation (1.3) is the path constraint, which limits the state and control variables over the entire prediction horizon, whereas Equation (1.4) is the terminal constraint, which is applied to only the initial and final values of the state variables. Equation (1.5) and Equation (1.6) denote the set of feasible state and control variable values, respectively. Equation (1.7) limits the prediction horizon.

The optimal control sequence $\boldsymbol{\zeta}^*(t)$ is bounded by input saturations, and the resulting estimated optimal state sequence $\boldsymbol{\xi}^*(t)$ satisfies the state constraints. A cost function is also minimized. Although the optimal control sequence is calculated over the horizon $t \in [t_0, t_0 + T_p]$, only the initial portion of the computed control sequence $\boldsymbol{\zeta}^*(t)$, $t \in [t_0, t_0 + T_e]$, is sent to the plant and executed, where T_e is called the execution horizon. Due to model simplification, model parametric uncertainty, measurement noise, and other disturbances, the actual values of the system states $\boldsymbol{\xi}(t)$, $t \in [t_0, t_0 + T_e]$, are likely to be different from their predicted values $\boldsymbol{\xi}^*(t)$, $t \in [t_0, t_0 + T_e]$. Therefore, in the next time step $t_0 + T_e$ the OCP is solved again over a shifted horizon $[t_0 + T_e, t_0 + T_e + T_p]$ based on the new state measurements at $t_0 + T_e$. Thus, the control law is updated every T_e seconds. The feedback of the measured state values to the optimization endows the procedure with a robustness typical of feedback systems. This process is thus repeated at each step until the terminal requirements are satisfied.

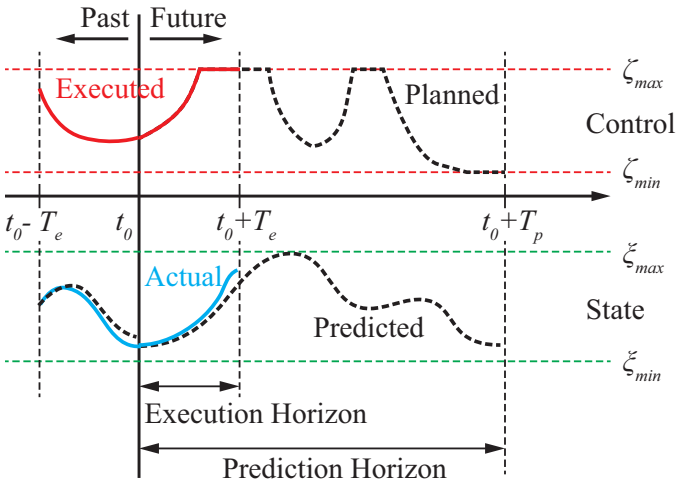


Figure 1.5: Basic principle of MPC.

1.5 Literature Review on MPC-based Obstacle Avoidance Algorithms

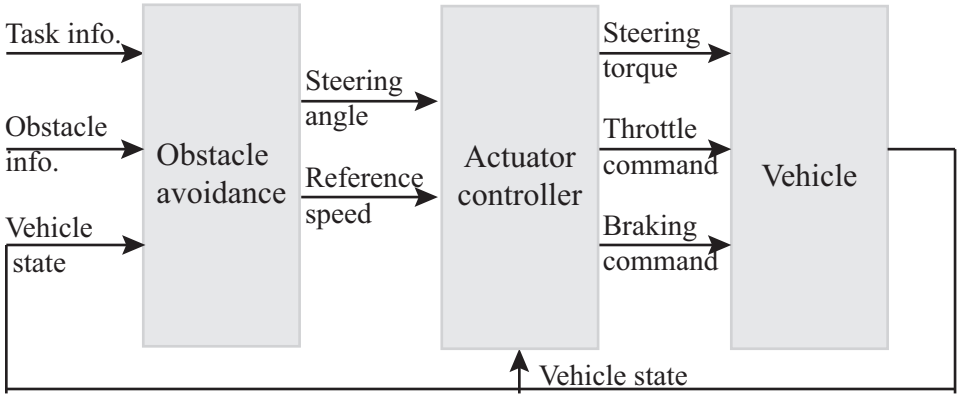
One of the early applications of the MPC framework in the vehicle motion control domain is path following. The goal is to stabilize a vehicle along a desired path while fulfilling its physical constraints. Obstacle avoidance is not explicitly considered in the control design. For example, Borrelli et. al. [46] used an MPC-based approach to develop a feedback controller to compute the front steering angle in order to follow a double lane change maneuver on slippery surfaces such as snow covered road with the highest possible entry speed. Simulation results showed that complex steering maneuvers are relatively easily obtained as a result of the MPC feedback policy. They later developed a computationally efficient suboptimal MPC controller based on successive online linearization of the nonlinear vehicle model [47]. The vehicle model is linearized around the current operating point at each time step and a linear MPC controller is designed for the resulting linear time-varying system. In another work [48], they also extended the formulation to simultaneously optimize both the front steering angle and the tire slip ratios at the four wheels. Performance enhancement is observed when combined braking and steering are used instead of steering only. Moreover the capability of the controller of slowing down the vehicle in order to perfectly follow the desired path has been shown. Some other examples are [49] and [50].

Two types of architectures can be used for taking into account the obstacle information. The first scheme directly computes the commands for the actuator controller based on the available information as shown in Figure 1.6a, which is referred to as the one-level controller. Discussions on the variations of the actuator controller inputs can be found in [51]. The second scheme decomposes the problem into two subproblems, which is referred to as the two-level controller. Thus, two controllers are used as shown in Figure 1.6b. The first one computes a reference trajectory taking into account the obstacle information, which is referred to as the higher-level controller or the path planner. The second one is designed to compute the commands for actuator controller to track the generated trajectory, which is referred to as the lower-level controller or path follower.

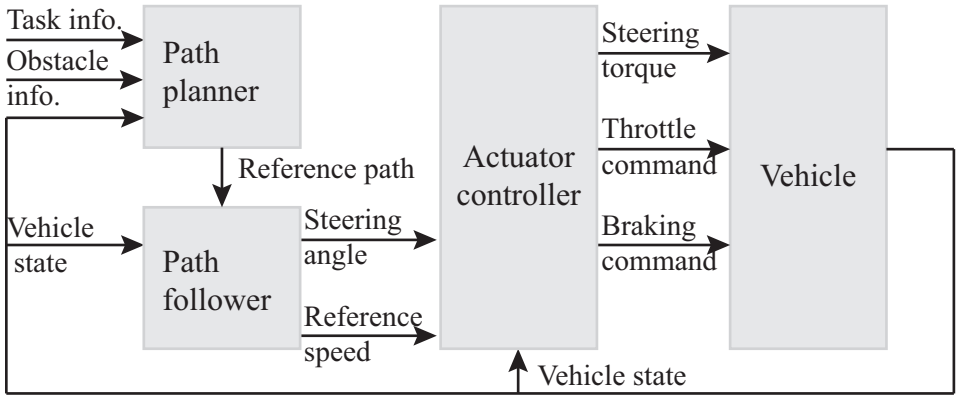
When all the controllers are implemented in the MPC framework, the two-level MPC is more computationally efficient [52]. However, when it becomes necessary to push the vehicle to its dynamical limits by either minimizing time-to-target or maximizing progress-on-track, as in racing situations [53, 54] or with military applications,

the higher-level path planner may create dynamically infeasible trajectories because only simple vehicle dynamics are considered. Additionally, collisions with obstacles may occur if the vehicle deviates from the reference trajectory, since lower-level path follower does not generally constrain the vehicle to avoid obstacles.

Next, several representative works are reviewed.



(a) One-level architecture



(b) Two-level architecture

Figure 1.6: Two architectures for obstacle avoidance.

Earlier research on MPC-based obstacle avoidance for AGVs include Ogren and Leonard [55], and Tahirovic and Magnani [56, 57]. They use the one-level formulation and guarantee the completion of the task using different techniques. However, the first method is limited to a unicycle robot and the second method requires the complete knowledge of the environment.

In particular, Ogren and Leonard [55] proposed a way to combine the convergence-oriented potential field method [17] and the global dynamic window approach [21] by casting these two approaches in an MPC and control Lyapunov function framework [58]. They proved the algorithm's convergence properties. They worked on a unicycle robot and its dynamics model is simple. However, with this framework, a control

action that decreases the value of Lyapunov function has to be found in advance, which is difficult, if not impossible for complex vehicle models [57].

Tahirovic and Magnani [56, 57] then provided a framework incorporating a passivity-based constraint to obtain a MPC scheme with guaranteed closed-loop stability for nonlinear systems [59]. It guarantees task completion and has the potential to work with a broad class of vehicles and terrains. A virtual vehicle model is generated by energy-shaping technique using a navigation function, which is constructed for the obstacle field to be transversed. This virtual vehicle model is then used to find the optimal trajectory along the prediction horizon. In this framework, the information on the obstacles and target position are included into the optimization setup and the target position is a stable equilibrium point. However, they assume that the environment is completely known, that is, the locations of all the obstacles that the vehicle may ever encounter are known *a priori*. In reality, only local information from onboard sensors will be available and the vehicle will not be aware of the environment beyond the sensor range.

Yoon et. al. [60] developed a system for obstacle avoidance of autonomous ground vehicles as an active safety procedure in an unknown environment. They used the two-level scheme and used only the local obstacle information. Safe trajectories are generated using the nonlinear MPC (NLMPC) framework, in which the simplified dynamics of the vehicle are used. For the local path regeneration upon detecting new obstacles, the cost function is augmented using the obstacle information using two methods. The first method uses the distance from the AGV to the nearest detected obstacle, and the second method uses the parallax information from the vehicle about the detected obstacles. Simulation results in cluttered and dynamic environments show that the modified parallax method is more effective. Park et. al. [61] continued this work and developed controllers to track the generated trajectory to compensate for the dissimilarity between the simplified model and the actual vehicle. In particular, the longitudinal dynamics of the vehicle are controlled using the inverse dynamics of the vehicle powertrain model, and the lateral dynamics are controlled using a linear quadratic regulator. However, these works [60, 61] use soft constraints for taking into account the obstacle information. Thus, even when a feasible solution is obtained, obstacle avoidance is not guaranteed.

Nanao and Ohtsuka [62] developed a one-level MPC-based collision avoidance method. In this work, to consider whether it is possible to avoid an obstacle physically, an unavoidable region (UR) is constructed as the region in which the vehicle cannot avoid the obstacle owing to physical limitations. A barrier function is added to the

cost function for satisfying the constraints of avoiding the UR and staying within the road bounds. The UR is constructed approximately by finding a trajectory with the shortest avoidable distance considering the vehicle's current speed and assuming that the theoretical maximum force is applied. However, the approach of finding the UR is limited to considering only one obstacle within the vicinity of the vehicle.

Gao et. al. [52] designed a two-level MPC scheme for autonomous obstacle avoidance and lane keeping. The path planner uses an MPC with a point mass vehicle model to plan obstacle free paths with a long prediction horizon. The path follower uses an MPC with a higher fidelity vehicle model to follow the planned path with a shorter prediction horizon. The decomposition of the problem reduces the computational complexity compared to a single level MPC approach. However, the trajectories generated by the point-mass vehicle model are not always feasible for the actual vehicle due to oversimplification. Gray et. al. [63] follow the same hierarchical MPC scheme. Differently, for the higher level path planner, they use a clothoid to plan obstacle avoiding maneuvers. The use of clothoid makes it possible to plan feasible trajectories with few parameters and thus with low computational speed. However, the vehicle maneuverability is limited to only a subset of motions since the path planner selects a sequence of primitives from an offline computed look-up table. Another update to the scheme is presented in [64]. The higher level uses a nonlinear bicycle vehicle model and utilizes a coordinate transformation that uses the vehicle position along a path as the independent variable. The resulting trajectories are more trackable for the real vehicles. However, in these works, it is assumed that a reference trajectory, longitudinal speed, and yaw rate exist.

Frasch et. al. [65] addressed the problem of real-time obstacle avoidance on low-friction road surfaces using single-level spatial NLMPC. Different from the previously introduced work, they used a nonlinear four-wheel vehicle dynamics model that includes wheel dynamics and load transfer. The formulation of the load transfer used in this work leads to an algebraic loop through the tire model, which is relaxed by introducing first order models with a time constant. Similar to [64], the time-dependent vehicle dynamics is transformed into a track-dependent (spatial) dynamics. This allows a natural formulation of obstacles and general road bounds under varying vehicle speed. However, this work requires reference values of all states.

Table A.1 summarizes some main features of the five representative MPC-based obstacle avoidance algorithms that are reviewed above. These features include the vehicle model and tire model used, the control inputs to be optimized, the way of accounting obstacle information, maximum vehicle speed, the optimizer used and

other relevant information for the discussion of this thesis.

It would be impossible to cover all existing work in this field in detail here. It is worth noting several other relevant works. A summary to distinguish the scope of this work from the literature can then be given.

Katriniok and Abel [66] developed a linear time-varying MPC approach for lateral vehicle guidance through successive linearizations over the prediction horizon. Ali et. al. [67] developed a multi-objective predictive controller for prevention of unintended vehicle collisions, which is capable of identifying and avoiding unintended collisions with stationary and moving road obstacles, vehicle control loss as well as unintended roadway departures. Jeon et. al. [68] developed a RRT*-based fast motion planner that incorporates the half-car dynamical model for wheeled vehicles. They introduced a fast local steering algorithm. Beal and Gerdes [69] developed a driver assistant algorithm used affine force input model to account for nonlinear tire model and phase portrait based envelope boundaries to prevent vehicle from becoming unstable. Liniger et. al. [70] developed a two-level NLMPC-based algorithm for autonomous racing, where the objective is to maximum progress-on-track subject to the requirement of staying on the track and avoiding opponents. Obstacle avoidance is incorporated by means of a high-level corridor planner based on dynamic programming, which generates convex constraints for the controllers according to the current position of opponents and the track layout.

However, these MPC based obstacle avoidance algorithms are for on-road application, where the environment is structured. In contrast, this thesis is concerned with AGVs in unstructured environments, such as military applications.

There are three considerations that distinguish the scope of this work from the literature and motivate a new MPC formulation. First, in a structured environment, the lane provides a reference trajectory and obstacle avoidance is achieved by perturbing the given reference path, for example, as in [71, 72]. In an unstructured environment, besides avoiding obstacles, finding an optimal path from the initial position to the target position is also part of the OCP, because no reference trajectory exists. This requires a longer prediction horizon to achieve successful navigation in directions that may deviate significantly from the 'current' direction. A new cost function formulation is also necessary, which includes terms to navigate the vehicle to the target as fast as possible.

Second, for on-road applications it is sufficient to use box constraints or constant bounds on position variables to represent the obstacle-free region; e.g., as in [73, 74]. However, in an unstructured environment, the obstacle-free region typically has a

more complicated form especially when multiple obstacles are present within the prediction horizon. This obstacle-free region cannot be simply represented using the type of constraints utilized in on-road applications. Thus, a systematic approach is required to include complex obstacle-free regions in the MPC formulation.

The third consideration is related to the vehicle's dynamical safety, which is an important constraint during obstacle avoidance at high speed. Existing algorithms focus on vehicle platforms where the major dynamical safety concern is excessive sideslip [75], such as passenger vehicles on slippery road or race cars [52]. Instead, this thesis focuses on vehicles with higher CoG location, such as military vehicles, in which the major dynamical safety concern is wheel lift-off. Current obstacle avoidance algorithms do not take wheel lift-off into consideration explicitly. Therefore, it is important to develop a new formulation to ensure the dynamical safety for AGVs with high CoG in terms of avoiding wheel lift-off.

1.6 Problem Statement

This work aims to develop and test a novel NLMPC-based obstacle avoidance algorithm for large-size high-speed AGVs to address the aforementioned considerations. The MPC framework relies on the iterative solution of an OCP. The cost function and constraints are formulated to reflect the following requirements. The resulting optimal trajectory should be 1) dynamically feasible; 2) collision free; 3) dynamically safe in terms of no wheel lift-off; and 4) shortest in terms of traveling time.

In particular, this thesis achieves its goal by addressing the following tasks.

1. A model of the platform is required to evaluate the obstacle avoidance algorithm in a closed-loop manner. The model should be with enough details so that it can predict the vehicle's responses accurately.

2. The level of fidelity of the model that needs to be used in the MPC formulation for best performance is to be addressed.

3. A new cost function formulation is needed that aims to find the shortest path to the target position in addition to approaching the target from a desired direction and minimizing the control effort.

4. An obstacle-free region (safe region) can be established using the LIDAR data. To accommodate the complicated form of the obstacle-free region in the OCP formulation, systematically partitioning of the safe region is necessary, enabling

a differentiable mathematical representation of the obstacle-free region and its inclusion in the OCP through a multi-phase approach.

5. Explicit consideration of the vehicle dynamical safety in terms of avoiding single-wheel lift-off is required for the vehicle with relatively high CoG location.

6. The prediction-correction characteristic allows the MPC to tolerate uncertainties to a certain level. However, to further improve the robustness of the algorithm, parametric uncertainty needs to be considered explicitly in the algorithm.

7. Experimental validation is essential in demonstrating the effectiveness of the proposed algorithm.

1.7 Closed-Loop System Schematic

Fig. 1.7 shows the schematic of the NLMPC obstacle avoidance algorithm with the AGV in closed loop. This section gives an overview of this framework and explains its basic flow at a high level.

In this simulation-based study, the AGV is represented using a fourteen degrees-of-freedom (DoF) vehicle dynamics model that includes suspension dynamics, combined-slip nonlinear tire dynamics, powertrain dynamics, and brake dynamics; the vehicle model is introduced in detail in Chapter 2. The inputs to the AGV are the steering, throttle, and braking commands. This model represents the real system and is referred as the plant model in this paper. The development of a high-fidelity model is not the focus of this paper. Thus, models that are developed and validated in the literature are directly used without additional validation.

The NLMPC obstacle avoidance algorithm consists of two parts: the LIDAR data processor and the control commands generator.

The LIDAR data processor first simplifies the obstacle shape given by the raw LIDAR data, adds a safety margin, and partitions the obstacle-free region to aid with the mathematical formulation of constraints. It is discussed in detail in Chapter 4. The outputs of the LIDAR data processor, the task information, and the estimated vehicle states are used in the formulation of the OCPs.

Different formulations of OCPs are then introduced under different assumptions. In Chapter 5, an OCP formulation is presented that can optimize the steering angle sequence under the assumption that the longitudinal speed is maintained to be a constant. In Chapter 6, the OCP formulation is extended to enable simultaneous optimization of steering angle and reference longitudinal speed commands. In

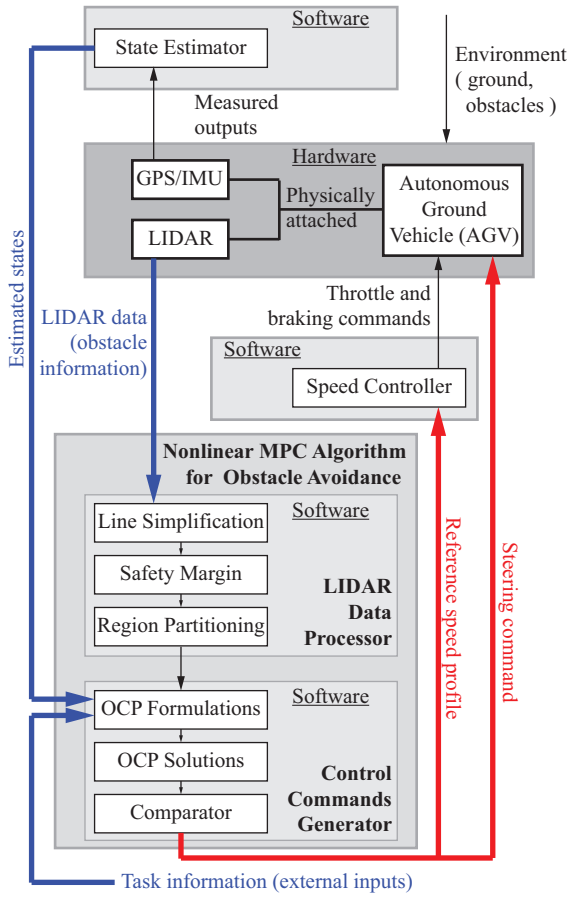


Figure 1.7: Schematic of an AGV with the NLMPC-based obstacle avoidance algorithm.

Chapter 7, the formulation is further extended to explicitly take into account the parametric uncertainty to further improve the robustness of the algorithm. The formulated OCPs are then solved and the control commands associated with the lowest cost solution are executed by the AGV.

The algorithm presented in this thesis is within the NLMPC framework. At each step, one or more multi-phase OCPs are solved. Thus, throughout the thesis, the term 'nonlinear MPC algorithm' and the term 'multi-phase optimal control algorithm' are used interchangeably.

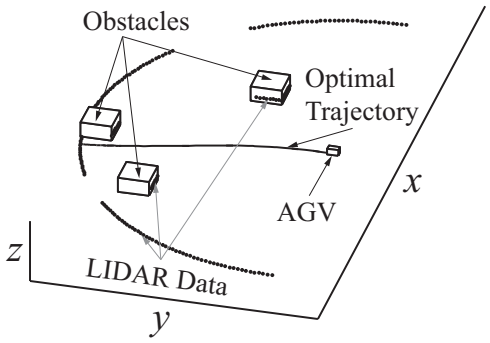
Optimality in this work refers to the optimality of the solution within the prediction horizon, taking into account all the information available at that moment in time, and not to the optimal solution that would have resulted if all environmental information was available for all times. Because the formulated OCP is non-convex, it is not guaranteed that the solution from the OCP solver is the unique global optimal solution over the prediction horizon. Thus, the terms 'optimal trajectory', 'optimal states', and 'optimal control' in this thesis refer to the local optimal solution generated by the OCP solver, which is the first minimum it finds.

1.7.1 Algorithm Inputs and Outputs

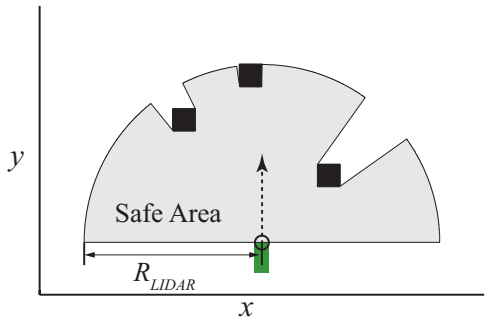
Three external inputs to obstacle avoidance algorithm are required: task information, obstacle information, and estimated states.

Within the scope of this paper, the task information is the specified target location, final heading angle requirement, and the desired final vehicle speed at the target position.

The obstacle information is obtained through a planar LIDAR sensor that is mounted in front of the vehicle. It provides information about the range and geometrical characteristics of the closest objects to the vehicle. The LIDAR returns the distance to the closest obstacle boundary in each radial direction at an angular resolution of ϵ. The angular range is [0°, 180°], with the vehicle heading direction being the 90° direction. For a direction without obstacles within the detection range, the LIDAR returns the maximum detection range R_{LIDAR}. Figure 1.8 shows an obstacle field with three obstacles and the output of the LIDAR for the particular vehicle position and orientation. It is assumed that all obstacles of interest are static and at least the height of where the LIDAR is mounted. The vehicle is assumed to travel on a planar surface; hence, a 2D representation as shown in Figure 1.8b is sufficient for this study. The sensor data is not a complete description of the obstacles because



(a) A sample obstacle field with the LIDAR detection data shown in the 3D space.



(b) The top view of the LIDAR data and the safe area. For the rest of this thesis, it suffices to show only the top view because a planar LIDAR sensor is used.

Figure 1.8: LIDAR detection data.

not all boundaries of the obstacle can be detected. This fact limits the region that can be considered as the safe area as illustrated in Figure 1.8b.

Vehicle states are needed to properly initialize the vehicle model used in the algorithm. In a physical implementation of the proposed system, a state estimator is needed to estimate the states with the available sensor measurements, since not all states can be directly measured. However, in this thesis, the AGV is simulated and, therefore, implementation of the state estimator is not warranted.

1.8 Dissertation Organization

This thesis is organized as follows. Chapter 2 presents the details of the vehicle model used as the plant in the simulation-based study and the vehicle models that are used in the MPC formulation. The study on model fidelity for MPC-based obstacle avoidance in high-speed AGVs is discussed in Chapter 3. Chapter 4 describes the LIDAR data processor in detail. In particular, the procedure of partitioning the obstacle-free region and formulating position constraints is discussed. It is first assumed that the vehicle longitudinal speed is maintained constant. Chapter 5 presents the OCP formulation and the simulation results under this assumption. Chapter 6 extends the formulation to simultaneously optimize both the steering angle command and reference longitudinal speed. A variety of simulation results are used to demonstrate the effectiveness of the algorithm. Chapter 7 improves the robustness of the algorithm to parametric uncertainty by introducing a double-worst-case formulation. Chapter 8 shows the experiment setup and results. Chapter 9 concludes the dissertation with a

summary, and a list of contributions and a list of possible extensions of this work.

CHAPTER 2

Vehicle Modeling

2.1 Introduction

A vehicle is a complex multi-DoF nonlinear system. For a kinematic and dynamic analysis of the vehicle, an appropriate mathematical model that can reflect the actual operating conditions is needed. For different purposes, a different vehicle model with adequate complexity for the respective application should be used for a balance between accuracy and efficiency [76].

The modeling of vehicle dynamics has been extensively studied in the literature. A variety of vehicle models has been developed [77, 78]. For the work of this thesis, first of all, a model of the vehicle is required to represent the plant (AGV in Figure 1.7) in the simulation-based study. To conduct successful dynamic simulations of vehicle with extreme maneuvers, six DoF [79], eight DoF [80, 81], fourteen DoF [82, 83, 84, 85], and eighteen DoF [86] dynamic models have been considered to better represent the vehicle lateral and yaw dynamics and to take into account the coupling of yaw-roll motion due to the transient lateral load transfer. A six DoF model is the simplest full-car model, which neglects the effects of wheels and suspension systems. In an eight DoF model, pitch and heave motions are ignored but the wheel spin dynamics at each wheel are considered compared to a six DoF model. A fourteen DoF model considers the suspension at each corner. The effects of the suspension longitudinal elasticity on the vehicle dynamic performance is taken into account in an eighteen DoF model. In addition, more complex and accurate representations of vehicle can be constructed in multi-body dynamic software packages. For example, the vehicle model presented in [87] is built in ADAMS/Car and has 76 bodies and 101 DoF, which results in a set of 1895 equations that are solved at every integration step. Thus, the computational time required is significantly longer than the models with only tens of

ordinary differential equations (ODEs). According to [85], the fourteen DoF model is sufficient to represent the vehicle motions that are important in most active chassis control systems, even though that it has lower number of degrees of freedom and consequently certain limitations when compared with a multi-body dynamics model. Thus, the fourteen DoF developed in [83, 84, 85] is adopted in this thesis to represent the real system and is referred to as the plant model. In Section 2.2, the dynamic equations of the fourteen DoF vehicle model are given.

Secondly, in an MPC-based algorithm, a model of the system is used to predict and optimize future actions. In the literature, a wide range of vehicle dynamic models has been used [88], for example, a relatively complex four wheel model with nonlinear tire model [65], a nonlinear bicycle model [52], a linear bicycle model [61, 62, 89], and the simplest point mass model with no tire model but only a friction circle constraint on tire forces [52]. The prediction accuracies and computational times of the models vary with respect to their complexities. Thus, there exists a trade-off in the model based controller design [88]. When a higher fidelity model is used in the MPC controller, even though the prediction accuracy is higher, the update rate could be slow, which may result in lack of correction and hence limit the controller performance. When a lower fidelity model is used, the algorithm can be updated more frequently. However, the large model mismatch may also lead to poor controller performance. Thus, the selection of model to be used in the MPC is critical. For certain applications, the differences between the prediction accuracies of different models can be small, for example, in small and low-speed ground robots. In this case, a simpler model could be sufficient for the control purpose. For large and high-speed platforms, the differences of the dynamic responses could be dramatic between the models and a more complex model might be required to provide satisfactory controller performance despite the increased computational load. For the context of this thesis, a two DoF or a three DoF nonlinear bicycle model with proper details is used in the MPC. In this chapter, the possible variations of these models are discussed in detail in Section 2.3. The study of the model fidelity requirement is presented later in Chapter 3.

2.2 Full Model

In this section, the vehicle full model that represents the plant is presented. The block diagram is given by Figure 2.1. The components are described in detail in the rest of this section.

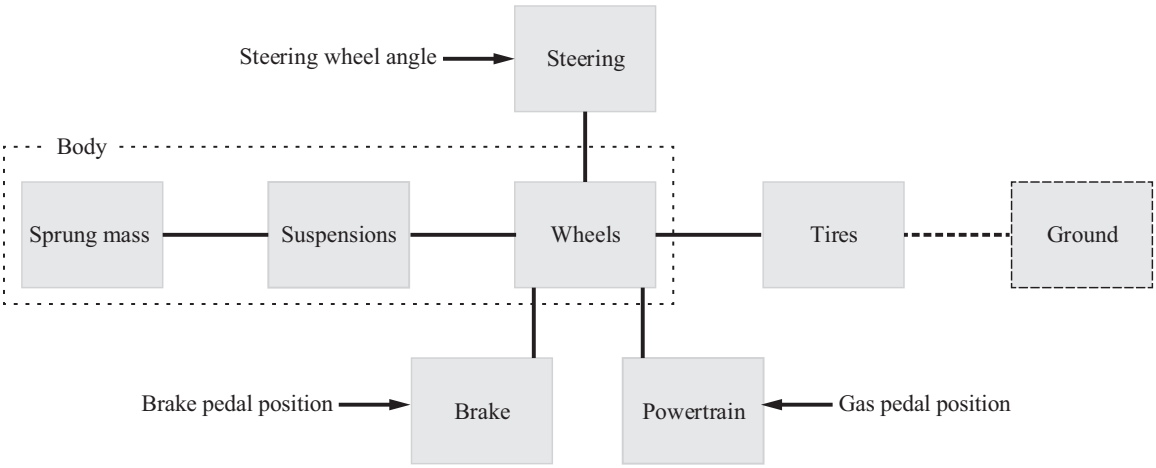


Figure 2.1: Block diagram of the full vehicle model.

Suspension systems play an important role in vehicle handling behavior. To predict wheel lift-off accounting for suspension systems, at least a fourteen DoF vehicle (body) model is necessary, which consists of a single sprung mass connected to four unsprung masses [85] and is illustrated in Figure 2.2. The sprung mass is represented as a plane and is allowed to pitch, roll, and yaw, as well as displace in longitudinal, lateral, and vertical directions. The suspensions, which are modeled as passive spring and damper elements, are allowed to bounce vertically with respect to the sprung mass. Each wheel is also allowed to rotate about its horizontal axis and only the two front wheels are free to steer. In summary, this model consists of six DoF at the vehicle CoG, and two DoF at each of the four wheels, including vertical tire stiffness and wheel spin. The dynamic equations are presented in Section 2.2.1.

Without a good tire model, the simulated handling responses will not be realistic. Of the many different tire models that are available today, the Pacejka Magic Formula tire model [90] is one of the most widely used and has proven to be very accurate when compared to experimental data. Thus, the version that takes into account the coupling of slip ratio and slip angle is used with the fourteen DoF vehicle model to predict the longitudinal and lateral behaviors of the tires. The details of the tire model are discussed in Section 2.2.2.

In addition, powertrain and brake dynamics are also modeled to simulate the vehicle speed and acceleration capabilities. In particular, powertrain dynamics are modeled according to [91, 92, 93, 94, 95]. They include the flywheel, engine, torque converter, transmission, drive shafts, and differential. The dynamics of the hydraulically actuated brake is modeled using the single state model in [96] that produces a good representation of the dynamics from the master cylinder to vehicle deceleration.

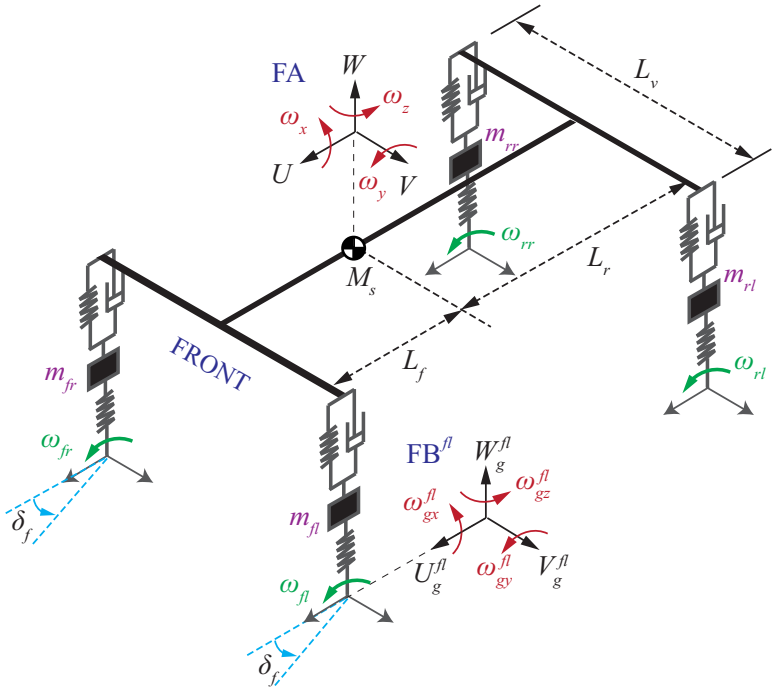


Figure 2.2: Schematic of the fourteen DoF vehicle (body) model.

These components are introduced with more details in Section 2.2.3.

Note that the powertrain and brake dynamics are considered only when the vehicle longitudinal speed is varying. Thus, they are not included in the plant model that is used to generate the constant-speed obstacle avoidance results presented in Chapter 3 and Chapter 5. The driving torques and brake torques are used directly as the inputs to the full vehicle model. When the reference longitudinal speed is also optimized, the simulations are conducted with powertrain and brake dynamics taken into account as in Chapter 6 and Chapter 7.

Finally, the steering system is modeled as a constant ratio and the effect of steering inertia is thus ignored. It is assumed that only the steering angle at the front wheels can be controlled. In addition, the front left and front right steering angles are assumed to be the same. Rolling resistance due to passive stabilizer bar and body flexibility are also neglected. No electronic stability control systems are considered.

2.2.1 Dynamic Equations

In this section, the dynamic equations of the fourteen DoF vehicle model is presented, which includes the dynamics of roll center and nonlinear effects due to vehicle geometry changes. This model is adopted from [83, 84, 85]. Thus, the detailed derivations and explanations are omitted and only the final equations are included.

The sprung mass is represented by a rigid plane with a body-fixed coordinate frame (BFCF) attached to the center of gravity and aligned with the principal directions as shown in Figure 2.2. U, V, W are the longitudinal, lateral and vertical velocities of the sprung mass, respectively. ω_x, ω_y, and ω_z are the roll, pitch, and yaw angular velocities, respectively. The equations of motion for the sprung mass can be derived by applying the Newton's laws to the system, which are given by

$$M_s\left(\dot{U} + \omega_y W - \omega_z V\right) = \sum\left(F_{sx,\bullet}\right) + M_s g \sin\theta - F_{\text{aero}} \tag{2.1}$$

$$M_s\left(\dot{V} + \omega_z U - \omega_x W\right) = \sum\left(F_{sy,\bullet}\right) - M_s g \sin\phi\cos\theta \tag{2.2}$$

$$M_s\left(\dot{W} + \omega_x V - \omega_y U\right) = \sum\left(F_{sz,\bullet} + F_{dz,\bullet}\right) - M_s g \cos\phi\cos\theta \tag{2.3}$$

$$I_{xx}\dot{\omega}_x = \frac{1}{2}\left(F_{sz,fl} + F_{sz,rl} - F_{sz,fr} - F_{sz,rr}\right) L_t \tag{2.4}$$

$$I_{yy}\dot{\omega}_y = \left(F_{sz,rl} + F_{sz,rr}\right) L_r - \left(F_{sz,fl} + F_{sz,fr}\right) L_f \tag{2.5}$$

$$\begin{aligned} I_{zz}\dot{\omega}_z = {} & \left(F_{sy,fl} + F_{sy,fr}\right) L_f - \left(F_{sy,rl} + F_{sy,rr}\right) L_r \\ & + \frac{1}{2}\left(F_{sx,fr} + F_{sx,rr} - F_{sx,fl} - F_{sx,rl}\right) L_t \end{aligned} \tag{2.6}$$

where M_s is the sprung mass. I_{xx}, I_{yy}, and I_{zz} are the roll, pitch, and roll inertia of the sprung mass, respectively. L_f is the distance between the front axle and the CoG location. L_r is the distance between the rear axle and the CoG location. L_t is the track width. $F_{sx,\bullet}/F_{sy,\bullet}/F_{sz,\bullet}$ are the longitudinal/lateral/vertical forces transferred to the sprung mass in the BFCF and the subscript $()_\bullet$ denotes front left (fl), front right (fr), rear left (rl), and rear right (rr) corners. F_{aero} is the aerodynamic resistance.

The forces transmitted to the sprung mass, $F_{sx,\bullet}/F_{sy,\bullet}/F_{sz,\bullet}$, are obtained by considering coordinate transformations of the forces generated at the tire-ground contact path, $F_{tx,\bullet}/F_{ty,\bullet}/F_{tz,\bullet}$, and taking into account the unsprung mass weight and inertial forces. $F_{tx,\bullet}$ and $F_{ty,\bullet}$ are calculated using the tire model discussed in Section 2.2.2. $F_{tz,\bullet}$ is calculated using the tire stiffness and the instantaneous tire deflection.

The Cardan angles roll (ϕ), pitch (θ), and yaw (ψ) are obtained by

$$\dot{\phi} = \frac{\omega_y \sin\phi}{\cos\theta} + \frac{\omega_z \cos\phi}{\cos\theta} \tag{2.7}$$

$$\dot{\theta} = \omega_y \cos\phi - \omega_z \sin\phi \tag{2.8}$$

$$\dot{\psi} = \omega_x + \omega_y \sin\phi\tan\theta + \omega_z \cos\phi\tan\theta \tag{2.9}$$

The attitude (Z) and position (X, Y) of the body with respect to the inertial frame can be derived by performing coordinate transformation between the inertial

frame and the BFCF.

$$\begin{aligned}\dot{X} &= U\left(\cos\theta\cos\psi\right) + V\left(-\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi\right) \\ &\quad + W\left(\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi\right)\end{aligned} \tag{2.10}$$

$$\begin{aligned}\dot{Y} &= U\left(\cos\theta\sin\psi\right) + V\left(\cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi\right) \\ &\quad + W\left(-\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi\right)\end{aligned} \tag{2.11}$$

$$\dot{Z} = U\left(-\sin\theta\right) + V\left(\sin\phi\cos\theta\right) + W\left(\cos\phi\cos\theta\right) \tag{2.12}$$

The dynamics of the unsprung mass vertical velocity $w_{u,\bullet}$ are expresses as

$$\begin{aligned}m_\bullet\left(\dot{w}_{u,\bullet} + \omega_x v_{u,\bullet} - \omega_y u_{u,\bullet}\right) = F_{uz,\bullet} - F_{dz,\bullet} - k_{\text{susp},\bullet}x_{\text{susp},\bullet} - b_{\text{susp},\bullet}\dot{x}_{\text{susp},\bullet} \\ - m_\bullet g\cos\phi\cos\theta\end{aligned} \tag{2.13}$$

$$\dot{x}_{\text{susp},\bullet} = w_{u,\bullet} - w_{s,\bullet} \tag{2.14}$$

where $m_\bullet$ is the unsprung mass. $k_{\text{susp},\bullet}$ and $b_{\text{susp},\bullet}$ are the suspension spring stiffness and damping coefficient, respectively. The following forces and velocities are defined in the BFCF. $u_{u,\bullet}/v_{u,\bullet}/w_{u,\bullet}$ are the longitudinal/lateral/vertical velocities of the unsprung mass. $w_{s,\bullet}$ is the vertical velocity at the suspension corner. $x_{\text{susp},\bullet}$ is the suspension spring compression. $F_{uz,\bullet}$ is the vertical force transmitted to the unsprung mass, and $F_{dz,\bullet}$ is the jacking force, which is the link load transfer force when the roll center is considered.

The wheel rotational speeds are calculated by

$$I_w\dot{\omega}_\bullet = \tau_\bullet - F_{tx,\bullet}R_t \tag{2.15}$$

where I_w is the wheel rotational inertial. R_t is the radius of the tire. Positive $\tau_\bullet$ denotes driving torque while negative $\tau_\bullet$ denotes braking torque.

2.2.2 Tire Modeling

The longitudinal and lateral forces acting from road to tire, as shown in Figure 2.3, for the condition of combined slip are calculated using the Pacejka Magic Formula tire model, which is a nonlinear, semi-empirical model and is compactly written as:

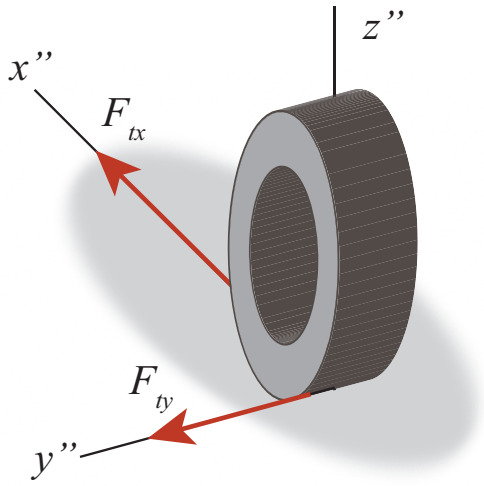


Figure 2.3: Illustration of tire forces generated at the tire-ground contact patch.

$$F_{tx,\bullet} = \mathcal{P}_{c,x}\left(\alpha_\bullet, \kappa_\bullet, F_{tz,\bullet}\right) \tag{2.16}$$

$$F_{ty,\bullet} = \mathcal{P}_{c,y}\left(\alpha_\bullet, \kappa_\bullet, F_{tz,\bullet}\right) \tag{2.17}$$

where $\alpha_\bullet$ is the slip angle and $\kappa_\bullet$ is the slip ratio.

Figure 2.4 defines the notions related to the front left tire. The tire slip angle $\alpha_\bullet$ is the angle between the tire velocity direction and its longitudinal direction. Thus, it is given by

$$\alpha_\bullet = \arctan\frac{-u_{t,\bullet}\sin\delta_\star + v_{t,\bullet}\cos\delta_\star}{u_{t,\bullet}\cos\delta_\star + v_{t,\bullet}\sin\delta_\star} \tag{2.18}$$

where $u_{t,\bullet}$ and $v_{t,\bullet}$ are the longitudinal and lateral velocities at the tire contact patch, respectively. $\delta_\star$ is the steering angle and the subscript $()_\star$ denotes front (f) or rear (r). In this work, the rear wheels are not steerable. Thus, $\delta_r = 0$.

The slip ratio is expressed as:

$$\kappa_\bullet = \frac{r_w\omega_\bullet - u_{t,\bullet}\cos\delta_\star + v_{t,\bullet}\sin\delta_\star}{\left|u_{t,\bullet}\cos\delta_\star + v_{t,\bullet}\sin\delta_\star\right|} \tag{2.19}$$

The tire normal force is calculated using the tire stiffness and instantaneous tire deflection.

The detailed equations of the combined-slip Pacejka Magic Formula tire model can be found in [97]. Figure 2.5 shows the tire forces generated using the model when the normal force is 5000 N.

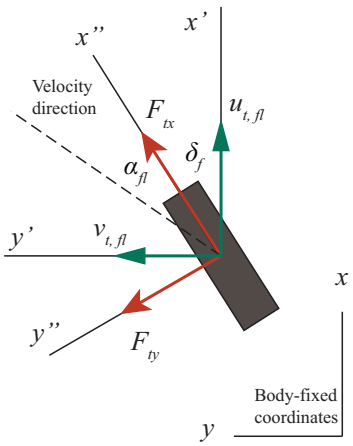
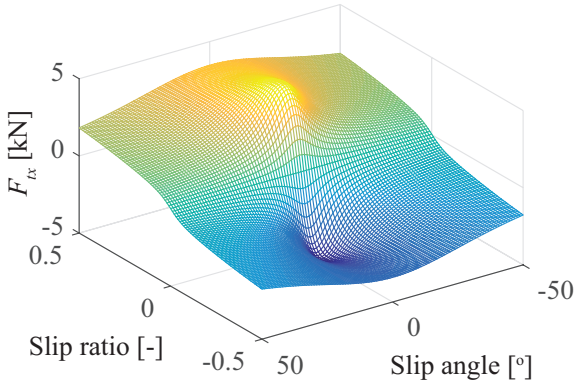
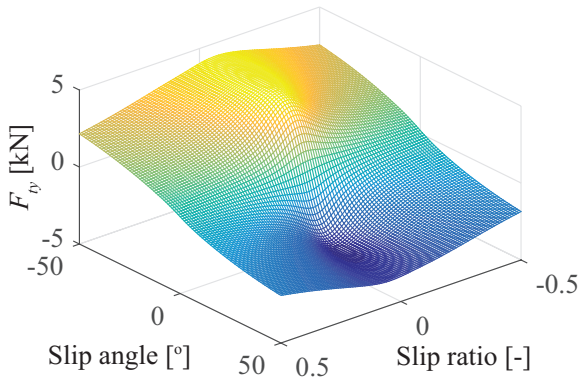


Figure 2.4: Coordinates and notations defined for the front left tire.



(a) Longitudinal tire force for a vertical load of 5 kN.



(b) Lateral tire force for a vertical load of 5 kN.

Figure 2.5: Example longitudinal and lateral tire forces as functions of slip ratio and slip angle in combined cornering and braking/driving condition, respectively, for a vertical load of 5 kN. The functions are described by the Pacejka Magic Formula tire model.

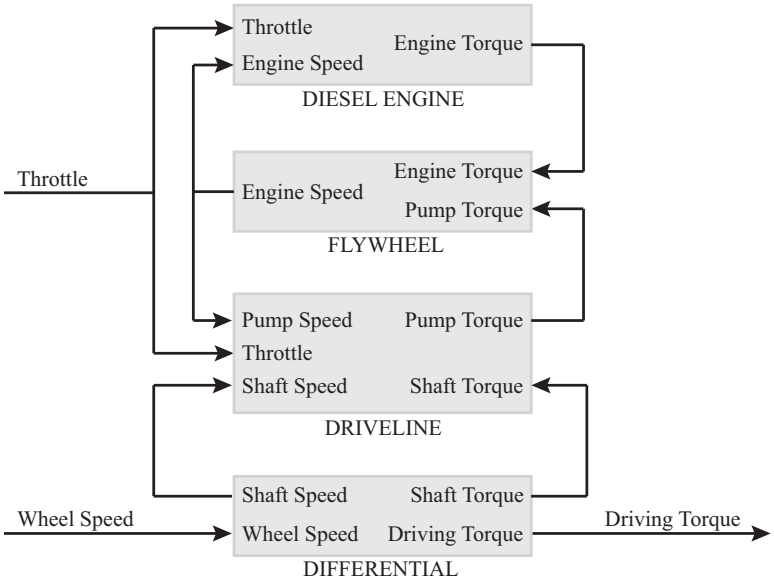


Figure 2.6: Block diagram of the powertrain model.

2.2.3 Powertrain and Brake Modeling

The powertrain generates power and delivers it to the road surface. A high fidelity modeling of the powertrain dynamics is important for the simulation of vehicle longitudinal dynamics, because it determines the vehicle's maximum speed and acceleration performance. The powertrain dynamics model describes the applied torques and angular speeds transmitted through all components, which consists of the flywheel, engine, torque converter, transmission, and differential. This model is adopted from [91, 92, 93, 94, 95] and thus only a brief description is presented below.

Figure 2.6 shows the inputs and output of the powertrain system. The inputs are the average wheel speed and the throttle command. The output of the powertrain system is the driving torque, which is then distributed evenly to the four wheels.

The flywheel is modeled as an inertia element that takes the load torque τ_{pump} and the engine torque τ_{engine} as inputs and determines the engine speed ω_{engine} as the output. This speed signal is fed to the driveline model as the pump speed and to the engine as the engine speed.

$$\dot{\omega}_{\text{engine}} = \frac{\tau_{\text{engine}} - \tau_{\text{pump}}}{I_{\text{engine}}} \tag{2.20}$$

where I_{engine} is the engine inertia.

A map-based engine model is used as shown in Figure 2.7. The input to the map is the fuel rate and the engine speed, and the output is the engine torque. The fuel rate

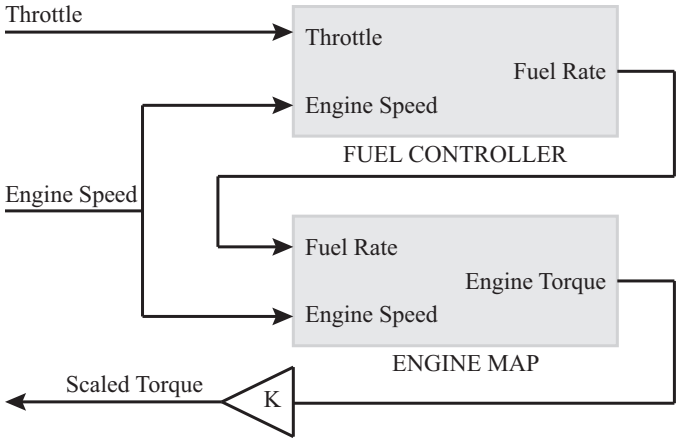


Figure 2.7: Block diagram of the engine model.

is given by the fuel controller, which takes into account the first order turbo-charge delay and a linear cutoff logic which controls the amount of fuel injected to be above the idle speed. Fig. 2.8 shows the engine map.

The driveline model includes the torque converter, transmission gear set, and driveshaft as shown in Figure 2.9.

The torque converter is a fluid clutch by which the engine is coupled to the transmission. The typical three element converter consists of pump, stator, and turbine. The pump is connected rigidly to the engine flywheel. Thus, pump speed ω_{pump} and engine speed ω_{engine} are the same. The turbine side, on the other hand, is connected to the transmission. The torque converter is approximated by a static model that

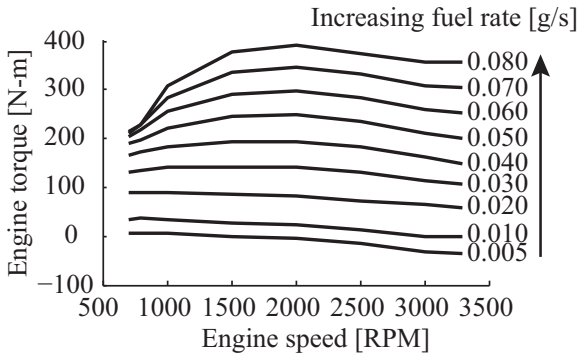


Figure 2.8: Static engine map.

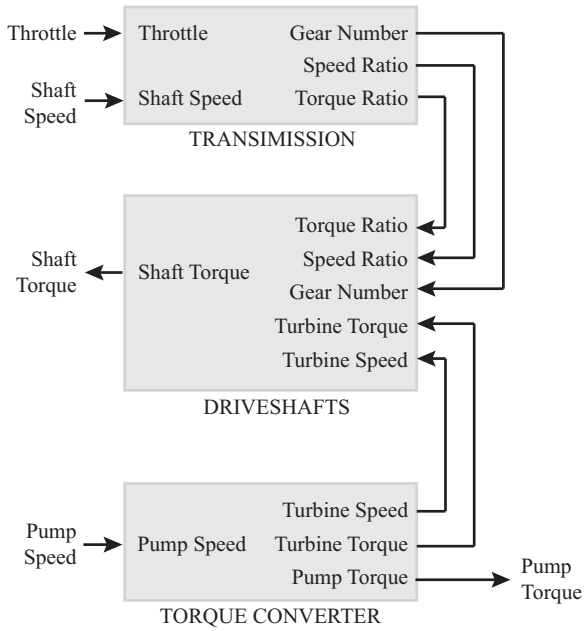


Figure 2.9: Block diagram of the drivetrain model.

takes pump and turbine speeds as inputs and generates pump and turbine torques.

$$\tau_{\text{pump}} = \left[\frac{\omega_{\text{pump}}}{\kappa(\omega_r)}\right]^2 \text{sign}(1-\omega_r) \tag{2.21}$$

$$\tau_{\text{turbine}} = \alpha(\omega_r)\tau_{\text{pump}} \tag{2.22}$$

$$\omega_r = \frac{\omega_{\text{turbine}}}{\omega_{\text{pump}}} \tag{2.23}$$

$$\dot{\omega}_{\text{turbine}} = \frac{\tau_{\text{turbine}}}{I_{\text{turbine}}} \tag{2.24}$$

where $\kappa(\omega_r)$ is a piecewise function approximating a desired capacity factor curve, and $\alpha(\omega_r)$ is a piecewise linear function approximating a desired torque ratio curve. I_{turbine} is the turbine inertia.

The transmission is represented by a set of shift logic, which determines whether or not an upshift or downshift is to be initiated based on the shaft speed and throttle demand. It is assumed that the speed reduction in each gear is ideal, while the torque multiplication is assumed to be scaled by an efficiency factor.

In addition, the shaft inertia, stiffness, and damping, as well as the gear inefficiencies and torque losses due to fluid churning are modeled to calculate the shaft torque transmitted to the differential. The schematic of the components considered is shown in Figure 2.10.

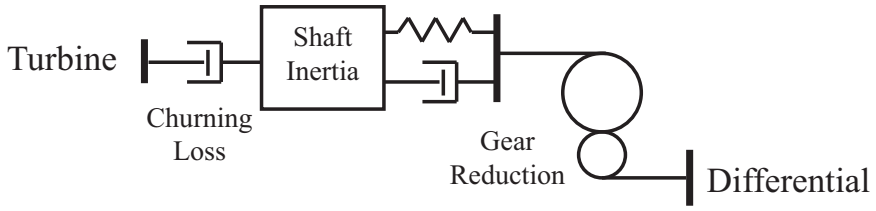


Figure 2.10: Schematic of the driveshaft model.

Finally, the differential is modelled using the following equations.

$$\tau_{\text{driving}} = \tau_{\text{shaft}} r_{\text{differential}} e_{\text{differential}} \tag{2.25}$$

$$\omega_{\text{shaft}} = \omega_{\text{driving}} r_{\text{differential}} \tag{2.26}$$

where τ_{shaft} and τ_{driving} are the shaft and driving torques, respectively. ω_{shaft} and ω_{driving} are the corresponding rotational speeds. $r_{\text{differential}}$ is the differential speed ratio and $e_{\text{differential}}$ is the differential efficiency.

A schematic of the hydraulically actuated brake model is given by Fig. 2.11. A single state model that produces a good representation of the dynamics from the master cylinder to vehicle deceleration is used [96]. The vacuum booster is a pneumatic servo mechanism used to amplify the input force from the brake pedal. The pressure in the master cylinder, P_{mc}, is calculated by subtracting the master cylinder pre-load and the seal friction from the output of the vacuum booster. The state equation for the cylinder volume V_{brake} is modeled using the Bernoulli's equation:

$$\dot{V}_{\text{brake}} = C_q \text{sgn}(P_{mc} - P_w) \sqrt{|P_{mc} - P_w|} \tag{2.27}$$

where C_q is the effective flow coefficient, and the wheel cylinder pressure is modeled as a cubic polynomial function, $P_w = P_w(V_{\text{brake}})$, which is obtained from experimental data. The braking torque τ_{brake} is then assumed to follow from this pressure

$$\tau_b = K_b \left(P_w - P_{po} \right) \tag{2.28}$$

where P_{po} is the push-out pressure and K_b denotes the brake gain.

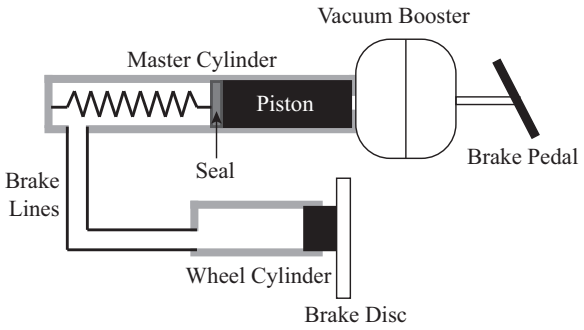


Figure 2.11: Schematic of the hydraulically actuated brake system.

2.3 Bicycle Model

In this section, the single-track vehicle model or the “bicycle" model is presented, which is widely used for the purpose of vehicle controller design. It is assumed that the left and right wheels can be lumped together for both the vehicle front and rear axles. Figure 2.12 shows the schematic of the bicycle model. The variations of the bicycle model is discussed in detail below.

2.3.1 Two DoF Vehicle Model

It is first assumed that the longitudinal velocity U of the vehicle is constant during the simulation. Thus, a two DoF is sufficient for this application, which is given by the following first-order ODEs.

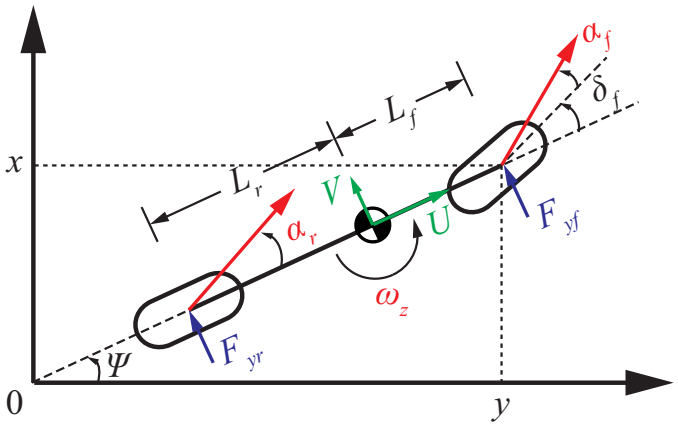


Figure 2.12: Schematic of the bicycle model.

$$\dot{V} = \left(F_{y,f} + F_{y,r}\right)/M_v - U\omega_z \tag{2.29}$$

$$\dot{\omega}_z = \left(F_{y,f}L_f - F_{y,r}L_r\right)/I_{zz} \tag{2.30}$$

$$\dot{x} = U\cos\psi - (V + L_f\omega_z)\sin\psi \tag{2.31}$$

$$\dot{y} = U\sin\psi + (V + L_f\omega_z)\cos\psi \tag{2.32}$$

$$\dot{\psi} = \omega_z \tag{2.33}$$

where M_v is the vehicle mass. (x, y) is the vehicle front center location in global coordinates. U is the longitudinal velocity in the BFCF and is constant. Thus, $U = U_0$ and $\dot{U} = 0$. $F_{y,f}$ and $F_{y,r}$ are the tire lateral forces generated at the front and rear axles, respectively. The model used for the calculations of the lateral tire forces is the key component that has a significant impact on the prediction accuracy. Thus, it is discussed in detail in Section 2.3.3. Several possibilities are presented.

Thus, the five states of the two DoF vehicle model are the lateral velocity in the BFCF V, the yaw rate ω_z, the vehicle front center location in inertia frame (x, y), and the yaw angle ψ. The input to the system is the front steering angle δ_f.

2.3.2 Three DoF Vehicle Model

When the vehicle longitudinal velocity is varying, one more ODE and one more control input are added to the previous two DoF vehicle model to account for the longitudinal dynamic. In addition to Equations (2.29) to (2.33), the following ODE is also considered.

$$\dot{U} = a_x \tag{2.34}$$

where a_x is the longitudinal acceleration, which is another control input in addition to the front wheel steering angle.

Thus, the six states of the three DoF vehicle model are the longitudinal and lateral velocities in the BFCF V, the yaw rate ω_z, the vehicle front center location in inertia frame (x, y), and the yaw angle ψ. The inputs to the system are the front steering angle δ_f and the longitudinal acceleration a_x.

2.3.3 Tire Modeling

The tire lateral forces can be calculated as a function of the tire slip angle and the tire vertical load. Two different tire models are considered in this thesis: the pure-

slip Pacejka Magic Formula tire model, which is also referred to as the nonlinear tire model, and the linear tire model.

With a Pacejka Magic Formula tire model, the lateral forces used in Equations (2.29) and (2.30) are calculated using the following equations

$$X_f = B\left(F_{z,f}\right)\alpha_f \tag{2.35}$$

$$\begin{aligned} F_{y,f} &= D\left(F_{z,f}\right)\sin\left\{C\left(F_{z,f}\right)\operatorname{atan}\left[X_f - E\left(F_{z,f}\right)X_f + E\left(F_{z,f}\right)\operatorname{atan}\left(X_f\right)\right]\right\} \\ &= \mathcal{P}\left(F_{z,f},\ \alpha_f\right) \end{aligned} \tag{2.36}$$

$$X_r = B\left(F_{z,r}\right)\alpha_r \tag{2.37}$$

$$\begin{aligned} F_{y,r} &= D\left(F_{z,r}\right)\sin\left\{C\left(F_{z,r}\right)\operatorname{atan}\left[X_r - E\left(F_{z,r}\right)X_r + E\left(F_{z,r}\right)\operatorname{atan}\left(X_r\right)\right]\right\} \\ &= \mathcal{P}\left(F_{z,r},\ \alpha_r\right) \end{aligned} \tag{2.38}$$

where $F_{z,f}$, and $F_{z,r}$ are the vertical loads at the front axle and rear axle, respectively. α_f and α_r are the tire slip angles. $B(\cdot), C(\cdot), D(\cdot)$, and $E(\cdot)$ are functions of the tire vertical load. These functions define the tire-ground interaction properties. The details of these relationships can be found in [97]. Figure 2.13 shows the relationship between the tire lateral forces and the slip angle at different vertical loads described by the Pacejka Magic Formula tire model.

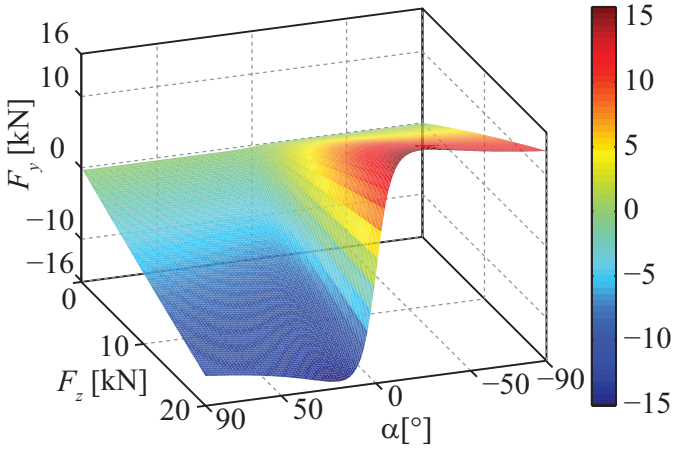


Figure 2.13: Lateral tire force described by the Pacejka Magic Formula tire model in pure cornering condition.

For a single-track model, the following two assumptions are made in calculating the tire lateral forces in Equations (2.35) to (2.38).

Assumption 1:

$$\alpha_{fl} \approx \alpha_{fr} \triangleq \alpha_f,\ \alpha_{rl} \approx \alpha_{rr} \triangleq \alpha_r$$

which means that the slip angles on the left and right tires are assumed to the same.

Assumption 2:

$$\mathcal{P}\left(F_a, \alpha\right) + \mathcal{P}\left(F_b, \alpha\right) \approx \mathcal{P}\left(F_a + F_b, \alpha\right)$$

which means the tire lateral force is approximately a linear function with respect to tire vertical load. This assumption allows for neglecting the lateral load transfer in the calculation of the lateral force.

Under the *Assumption 1*, the slip angles of the front and rear tires are obtained using

$$\alpha_f = \tan^{-1}\left(\frac{V + L_f\omega_z}{U}\right) - \delta_f \tag{2.39}$$

$$\alpha_r = \tan^{-1}\left(\frac{V - L_r\omega_z}{U}\right) \tag{2.40}$$

As for the linear tire model, the constant cornering stiffness $C_\alpha\left(F_z\right)$ is found as the slope at the origin for different tire vertical loads.

$$F_{y,f} = C_\alpha\left(F_{z,f}\right) \cdot \alpha_f \tag{2.41}$$

$$F_{y,r} = C_\alpha\left(F_{z,r}\right) \cdot \alpha_r \tag{2.42}$$

Figure 2.14 shows the comparison between the linear tire model and the Magic Formula tire model at a fixed tire vertical load. The two models agree well when the slip angle is less than 5 degrees.

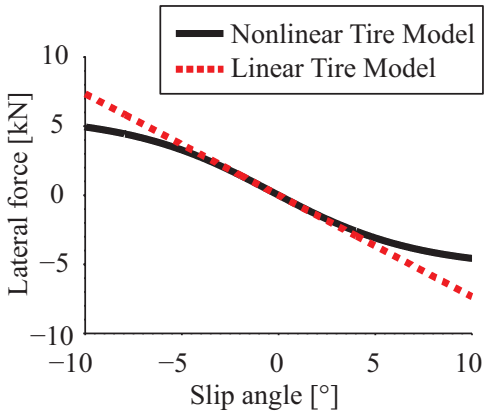


Figure 2.14: Comparison of the linear and nonlinear tire models for a vertical load of 7 kN.

For a single-track vehicle model, a typical assumption is that there is no longitudinal load transfer, so that the vertical loads on each axle are constant, which are

calculated using the following equations.

$$F_{z,f} = F_{z,f0} = \frac{M_s g L_r}{L_f + L_r} + m_f g \tag{2.43}$$

$$F_{z,r} = F_{z,r0} = \frac{M_s g L_f}{L_f + L_r} + m_r g \tag{2.44}$$

When the vehicle travels at high speed, it is important to account for the longitudinal load transfer. Thus, the following relationships are used to calculate the vertical loads on the front and rear axles taking into account the longitudinal load transfer effects [85, 98].

$$F_{z,f} = F_{z,f0} - \left(\frac{M_s h_{CG} + m_f h_{u,f} + m_r h_{u,r}}{L_f + L_r} \right) \left(\dot{U} - V \omega_z \right) \tag{2.45}$$

$$F_{z,r} = F_{z,r0} + \left(\frac{M_s h_{CG} + m_f h_{u,f} + m_r h_{u,r}}{L_f + L_r} \right) \left(\dot{U} - V \omega_z \right) \tag{2.46}$$

where h_{CG} is the height of the vehicle CoG location above the ground. $h_{u,f}$ and $h_{u,r}$ are the CoG heights of the front and rear unsprung mass above the ground, respectively.

Furthermore, Equations (2.45) and (2.46) can be approximated using the following two equations

$$F_{z,f} \approx F_{z,f0} - K_{z,x} \left(\dot{U} - V \omega_z \right) \tag{2.47}$$

$$F_{z,r} \approx F_{z,r0} + K_{z,x} \left(\dot{U} - V \omega_z \right) \tag{2.48}$$

where $K_{z,x}$ is the longitudinal load transfer coefficient, which can be empirically estimated. More discussions are presented in Section 6.2.4.

CHAPTER 3

Model Fidelity Evaluation

3.1 Introduction

Based on the literature review, several research questions still remain to be addressed in terms of MPC-based obstacle avoidance. This chapter focuses on one of them; namely, the level of fidelity of the model that needs to be used in the MPC formulation for best performance. In most of the research, simple vehicle dynamics models such as the two DoF or three DoF representations are considered as the dynamic model included in the MPC formulation. The sole justification provided in the literature for this choice is the low computational load required by these models. As pointed out by Park et al. [61], only when the assumptions used to derive the simplified model are satisfied can the actual vehicle track the trajectory generated based on this simplified model. However, a justification does not yet exist in the literature for the level of detail included in the vehicle model from the point of view of providing a satisfactory predictive capability within an MPC framework.

Thus, the specific goal of this chapter is to determine the model fidelity requirement of the vehicle model used for predicting the vehicle trajectory online in an MPC-based obstacle avoidance algorithm. The answer to this research question depends on the context [99].

To pursue its specific goal, this chapter considers five different representations of vehicle dynamics models in the MPC-based obstacle avoidance algorithm: four variations of the two DoF representation as lower fidelity models and a fourteen DoF representation with combined-slip Magic Formula tire model as a higher fidelity model. In the two DoF representation, either a linear tire model or a pure-slip Magic Formula tire model can be used. For each tire model, the axle loads can be assumed constant or varying due to longitudinal load transfer. The details of these vehicle

dynamics models are presented in Chapter 2.

The performance of the obstacle avoidance algorithm is measured by several metrics including time to target, control effort, and integral of curvature. The best performance is achieved when the vehicle reaches the target safely in minimum time with smallest control effort and integral of curvature. Note that even though it is obvious that increasing the complexity of the model by adding, for example, tire nonlinearities or load transfer effects would increase its accuracy, it is not obvious whether those additional considerations would lead to a significant increase in MPC performance, because it is unknown how much robustness to unmodeled dynamics closing the loop would provide.

It is important to emphasize that the purpose of this chapter is not to present a new MPC algorithm. A very simplistic, exhaustive search-based MPC formulation is chosen for the purposes of this work to enable ease of implementation of vehicle dynamics models of various levels of complexity without worrying about convergence issues when higher fidelity models are used. The cost function and constraints formulation, as well as the dynamic optimizer components of the MPC-based obstacle avoidance algorithm used in this chapter are explained in Appendix B, since the algorithm itself is not the main focus of this chapter.

3.2 Open-Loop Comparisons

It has been assumed that the fourteen DoF vehicle model is an accurate representation of the AGV. Thus, in this section, the responses of the two DoF vehicle model with different tire models are checked against the fourteen DoF vehicle model in an open-loop simulation.

The following three combinations are compared:

1. *Combo.* 1: fourteen DoF vehicle model with combined-slip Magic Formula tire model

2. *Combo.* 2: two DoF vehicle model with pure-slip Magic Formula tire model

3. *Combo.* 3: two DoF vehicle model with linear tire model

When the axle vertical loads are assumed constant and given by Equations (2.43) and (2.43), using the steering profile in Figure 3.1a as the input and considering a vehicle speed of 30 m/s, the trajectories predicted by the two DoF vehicle model with the linear and Magic Formula tire models are compared to the trajectory of the

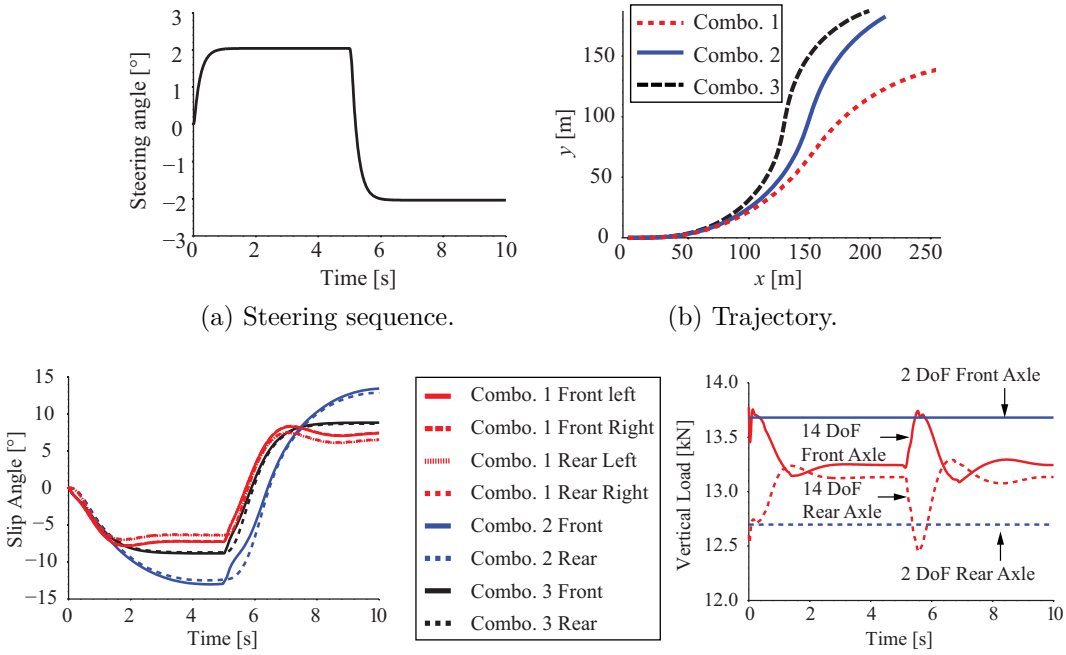
fourteen DoF vehicle model in Figure 3.1b. The responses generated by the three combinations are significantly different because of two reasons. First, the linear tire model is not a good approximation because the maximum slip angle is about 9 degrees as shown in Figure 3.1c. Second, there exists significant longitudinal load transfer when the vehicle moves at this speed as shown in Figure 3.1d.

After taking into account the longitudinal load transfer by incorporating Equations (2.45) and (2.46), the two DoF vehicle model with Magic Formula tire model becomes a very good approximation to the fourteen DoF vehicle model under this test condition as shown in Figure 3.2a. However, the prediction of the two DoF vehicle model with the linear tire model is still not as good, because the magnitudes of the slip angles are still larger than 5 degrees at some points in time as shown in Figure 3.2b and hence the differences between the linear and Magic Formula tire models are significant. Equations (2.45) and (2.46) can predict the axle vertical load well with either tire model as illustrated by Figure 3.2c.

In conclusion, tire nonlinearity and longitudinal load transfer are both important factors to be included in a two DoF vehicle model in order to predict vehicle trajectory accurately. Based on this set of comparisons, the two DoF vehicle model with the nonlinear tire model and longitudinal load transfer can be expected to yield better performance in terms of online vehicle trajectory predictions for obstacle avoidance. Note, however, that the comparisons provided in this section are open-loop only. In the following comparative study, all combinations of tire model and axle load equations are used to test the differences in the closed-loop obstacle avoidance performance, as well.

3.2.1 Computational Time

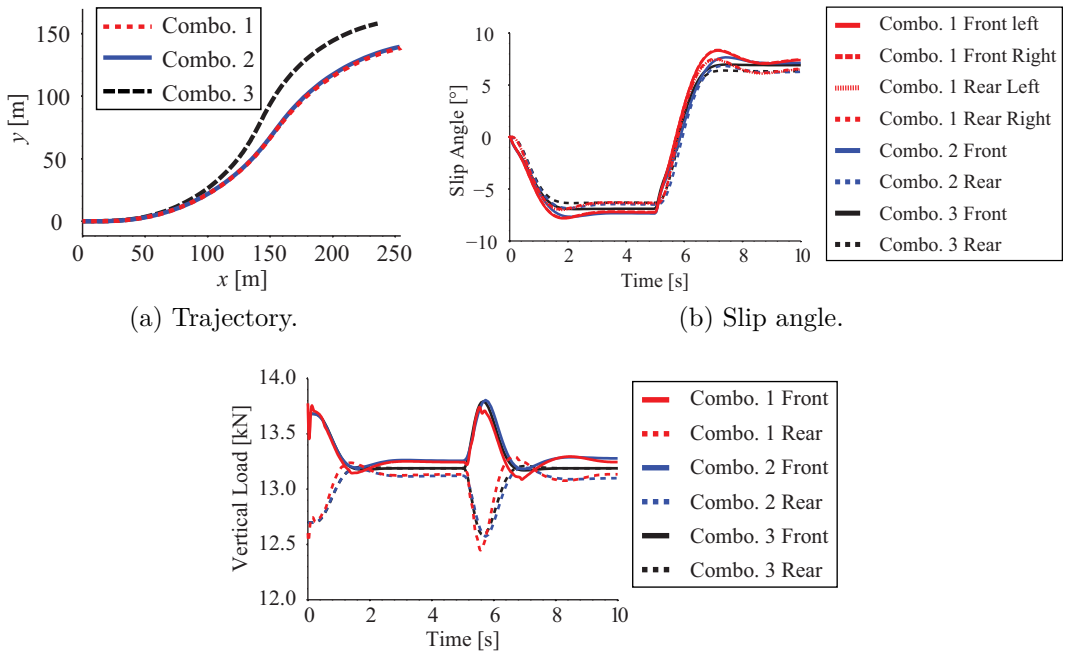
In this section, the open-loop simulation times for the five models are compared. The vehicle models are implemented in MATLAB®. The simulation times of these vehicle models are obtained using a 50 s sinusoidal steering maneuver on an Intel® Xeon® processor. Ten runs are performed for each model. Table 3.1 summarizes the average simulation time of different vehicle models. Using a nonlinear tire model or considering the longitudinal load transfer adds complexity to the model. Thus, simulation time is expected to increase, but the increment is less than 10%. The fourteen DoF vehicle model requires more than 20 times of the computational time of a two DoF vehicle model. Hence, the fourteen DoF model may not be feasible for a real-time implementation within MPC; nevertheless, it will still be considered in



(a) Steering sequence. (b) Trajectory.

(c) Slip angle. (d) Axle load.

Figure 3.1: Comparison of responses when the longitudinal load transfer effect is ignored in the two DoF vehicle model.



(a) Trajectory. (b) Slip angle.

(c) Axle load.

Figure 3.2: Comparison of responses when the longitudinal load transfer effect is considered in the two DoF vehicle model.

the closed-loop evaluations in the next section as a benchmark. On the other hand, the computational cost for all variations of the two DoF models is approximately the same. Therefore, closed-loop evaluations are performed in the next section to further elicit the performance differences between these models.

Table 3.1: Average simulation time of a 50 s sinusoidal steering maneuver.

Vehicle model	2 DoF	2 DoF	2 DoF	2 DoF	14 DoF
Tire model	Linear	Linear	Nonlinear	Nonlinear	Nonlinear
Load transfer	No	Yes	No	Yes	Yes
Average running time (s)	0.6048	0.6209	0.6492	0.6541	14.172
Standard deviation (s)	0.00379	0.00722	0.00712	0.0124	0.346

3.3 Simulation Results and Discussion

To study the effect of the fidelity of the vehicle model used in the MPC formulation, comparative simulation studies are conducted. The performance of the algorithm is characterized by several metrics including the time to target, control effort, and integral of curvature.

3.3.1 Evaluation Metrics

Three metrics are used to evaluate obstacle avoidance performance. First, time to target, T_{total}, is the total time required to navigate through the obstacle field and reach the target. A smaller T_{total} is desired.

Second, the control effort E_{control} is given by Equation (3.1) and a smaller value is preferred because it means less steering maneuver is performed.

$$E_{\text{control}} = \int_0^{T_p} |\delta_f(t)| \mathrm{d}t \tag{3.1}$$

Third, the integral of curvature $I_{\text{curvature}}$ is given by

$$I_{\text{curvature}} = \int_0^{T_p} C_{\text{traj}}(t) \mathrm{d}t \tag{3.2}$$

where C_{traj} is the curvature of the trajectory. A smaller C_{traj} means a relatively straighter trajectory, which is favored. The integral is used to evaluate the curvature over the entire maneuver.

3.3.2 Simulation Settings and Results

Obstacle avoidance simulations are performed using three obstacle fields denoted as MAP A, MAP B, and MAP C. MAP A is an obstacle field with two large obstacles. The vehicle needs to perform large turns to avoid obstacles on this map. MAP B is a small field with high density of obstacles. The obstacles have the same size, which is 2 m by 2 m. The vehicle speed is maintained at 20 m/s on these two maps. MAP C contains one long obstacle that is perpendicular to the vehicle's initial heading direction. The vehicle speed is maintained at 30 m/s on this map in order to push the vehicle to its dynamical limits. These simulations are used to study the effect of vehicle model and tire model combinations used in the MPC formulation to predict vehicle trajectory. The simulation settings and obtained results for MAP A and MAP B are given in Table 3.2.

Figure 3.3 shows the simulation results with MAP A. For this map, MPC with all five combinations are successful in terms of avoiding the obstacles safely. The performances are very close in terms of time to target. The two DoF model with Magic Formula tire model and varying axle loads requires the smallest control effort in this case and the largest control effort required is about 16% more. The largest difference in the integral of curvature is about 14%. Besides these statistical data, all trajectories are visually close to each other as shown in Figure 3.3a. This is within our expectations due to the basic idea of MPC that only part of the calculated optimal steering sequence is implemented even if the prediction is made over a longer time. More specifically, although the responses predicted by the first three models are significantly different from the responses predicted by the fourteen DoF vehicle model, the prediction error accumulates only with time. The predictions of all five models are closer to each other in the beginning of the prediction horizon than at the end. Hence, when the predicted horizon is not significantly long, the differences at the end of the prediction will not lead to different steering angle selections for the first step. The results show that the two DoF model with linear or Magic Formula tire model can perform comparable to the fourteen DoF model.

Figure 3.4 shows the simulation results with MAP B. For this map, MPC with all five combinations are again successful in terms of avoiding the obstacles safely. However, the two DoF vehicle model with nonlinear tire model and constant axle loads takes a different route than the other four models. This is not surprising on an obstacle field with multiple obstacles because only local information is available to the controller and the local information depends on the location and heading of the vehicle. With different vehicle models in the MPC, different steering commands can

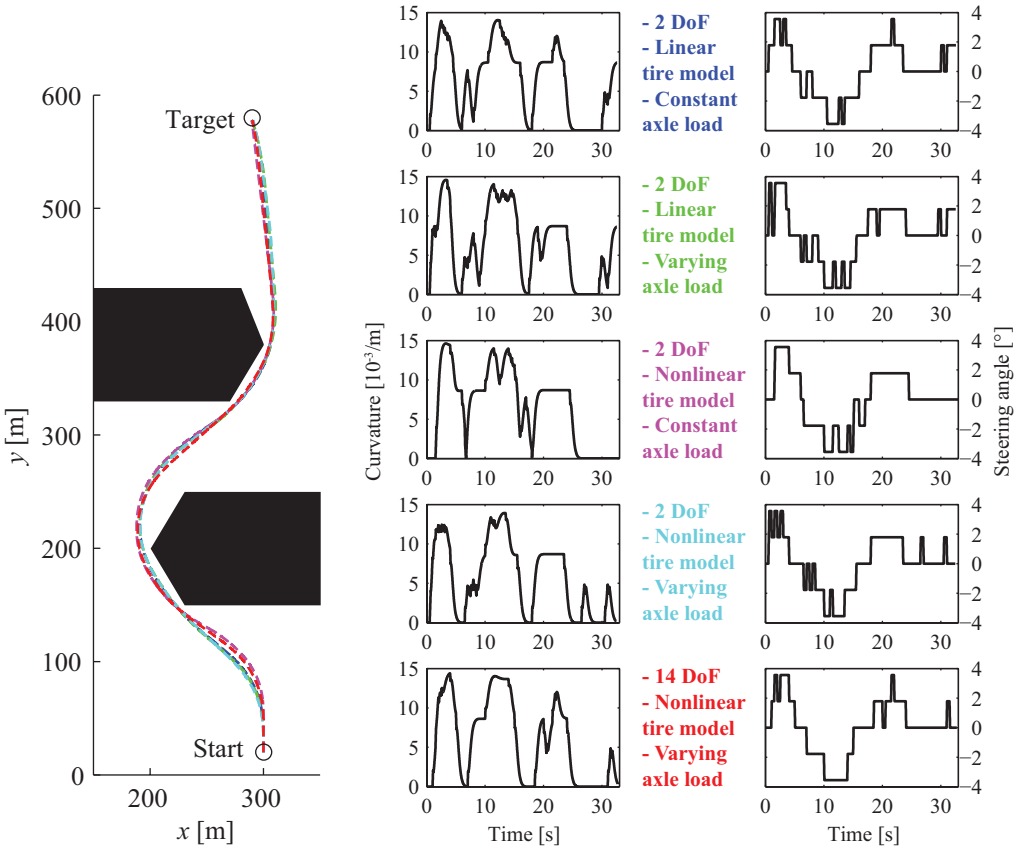
Table 3.2: Simulation settings and results for MAP A and MAP B

Parameter	**Value**				
U_0 (m/s)	20				
n_p	4				
n_s	5				
$T_{p,\min}$ (s), $L_{\max}$ unknown	5.98				
T_e (s)	0.5				
T_p (s)	6.98				
R_{LIDAR} (m)	129.6				
Vehicle model	2 DoF	2 DoF	2 DoF	2 DoF	14 DoF
Tire model	Linear	Linear	Nonlinear	Nonlinear	Nonlinear
Longitudinal load transfer	No	Yes	No	Yes	Yes
MAP A: T_{total} (s)	32.6	32.6	33.0	32.5	32.8
MAP A: E_{control} (°-s)	44.6	44.4	47.2	39.9	42.7
MAP A: $I_{\text{curvature}}$ (s/m)	0.2037	0.2006	0.2135	0.1850	0.1909
MAP B: T_{total} (s)	18.4	18.3	18.5	18.5	18.4
MAP B: E_{control} (°-s)	19.1	16.5	20.2	16.8	18.1
MAP B: $I_{\text{curvature}}$ (s/m)	0.0793	0.0671	0.0929	0.0812	0.0768

Note: numbers in red indicates largest value in the corresponding row, and numbers in blue indicates smallest value. Numbers in gray are excluded from the comparisons.

be generated, which leads the vehicle to different locations. This model is excluded from the following comparisons. For the four models taking approximately the same route, the performances are again very close in terms of time to target. The two DoF model with linear tire model and varying axle loads requires the smallest control effort and the largest control effort required is about 14% more. The integrals of curvature are also very close; the largest difference is less than 18%.

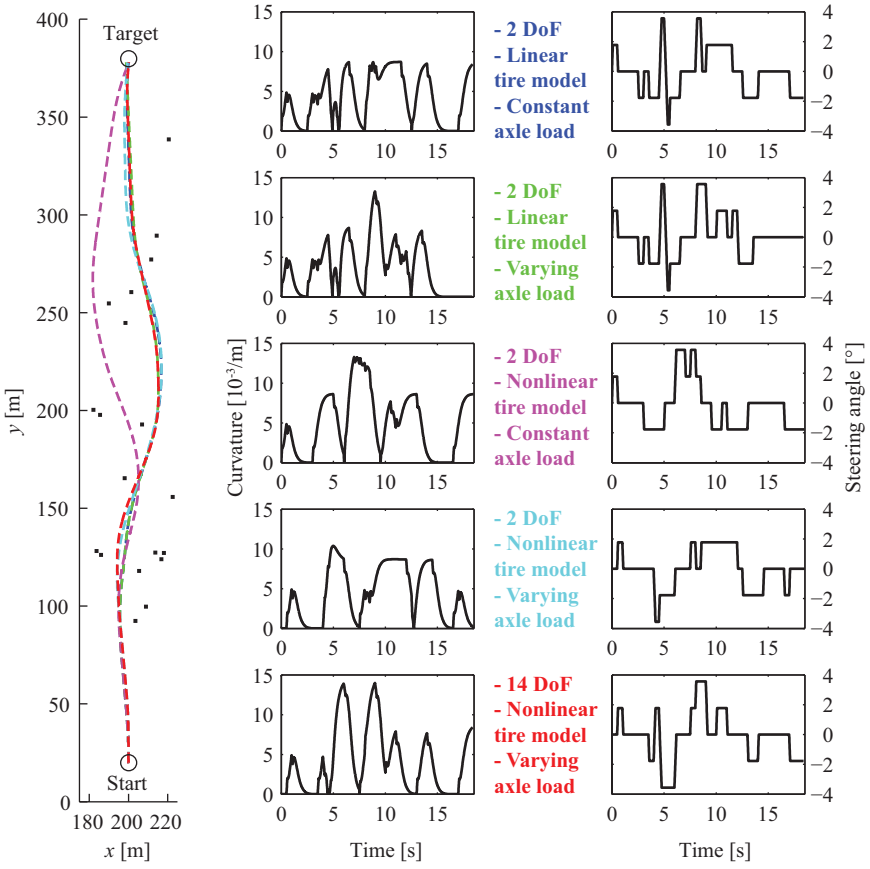
However, when the vehicle moves at the higher speed of 30 m/s, a vehicle model using linear tire model or ignoring longitudinal load transfer can fail to navigate the vehicle safely while a two DoF vehicle model with nonlinear tire model and varying axle loads still yields a satisfactory performance. The simulation setting for such a case is given in Table 3.3 and simulation results are shown in Figure 3.5. As shown in Figure 3.6, the two DoF vehicle model with linear tire model and constant axle loads turns at a smaller radius with the same steering angle. Thus, the algorithm decides not to turn initially and this results in the failure of this combination of models. The simulation using fourteen DoF vehicle model shows that it is capable of steering around this obstacle safely with the same simulation settings if the prediction within MPC is made more accurately. MPC with the vehicle model using linear tire model



(a) Trajectory.

(b) Trajectory curvature and steering angle.

Figure 3.3: Simulation results to study the effect of model fidelity with MAP A.



(a) Trajectory (b) Trajectory curvature and steering angle.

Figure 3.4: Simulation results to study the effect of model fidelity with MAP B.

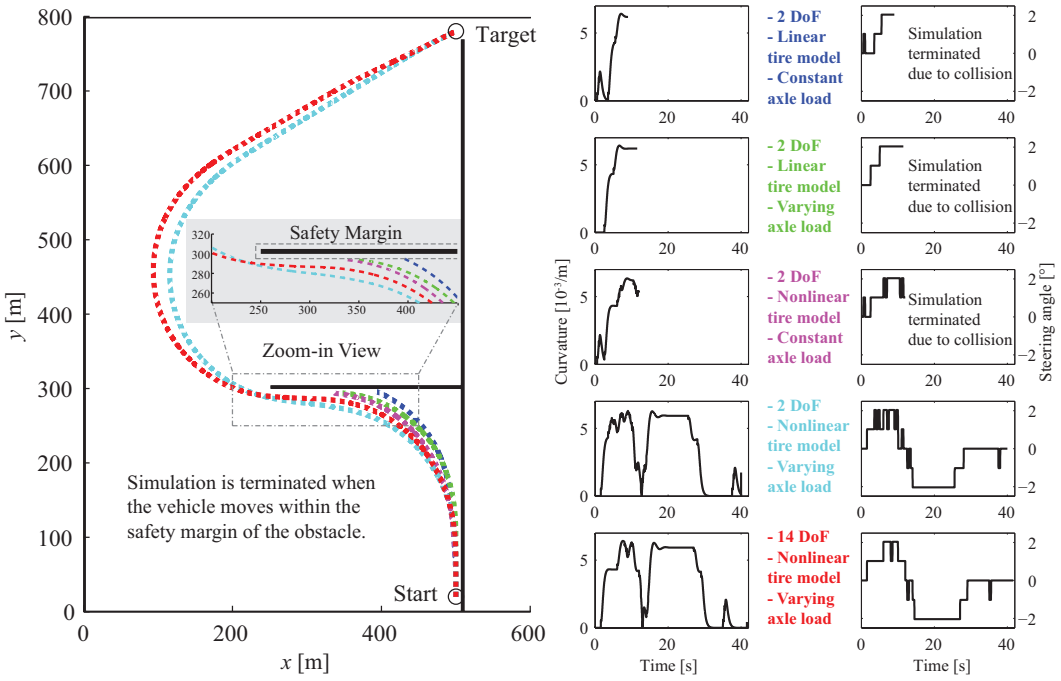
or ignoring longitudinal load transfer fails when the vehicle moves at a higher speed and the obstacle field includes significantly large obstacles. Hence, even though the feedback property of MPC provides certain robustness against prediction errors, a two DoF vehicle model with linear tire model or constant axle loads is not sufficient for extreme conditions, and at least a two DoF vehicle model with Magic Formula tire model and varying axle loads is needed.

Table 3.3: Simulation settings and results for MAP C

Parameter	**Value**				
U_0 (m/s)	30				
n_p	4				
n_s	5				
$T_{p,\min}$ (s), $L_{\max}$ unknown	8.69				
T_e (s)	0.5				
T_p (s)	9.19				
R_{LIDAR} (m)	275.7				
Vehicle model	2 DoF	2 DoF	2 DoF	2 DoF	14 DoF
Tire model	Linear	Linear	Nonlinear	Nonlinear	Nonlinear
Longitudinal load transfer	No	Yes	No	Yes	Yes
MAP C: T_{total} (s)	Fail	Fail	Fail	40.0	41.6
MAP C: E_{control} (°-s)				40.8	43.3
MAP C: $I_{\text{curvature}}$ (s/m)				0.1356	0.1422

3.4 Conclusion

This thesis considers MPC-based obstacle avoidance for large AGVs at high speed that maneuver within unknown, unstructured environments. The vehicle dynamics are critical to consider in this situation. Within this context, this chapter investigates the fidelity requirement of the vehicle model used for predicting vehicle trajectory in the MPC formulation. First, open-loop simulations suggest that tire nonlinearity and longitudinal load transfer are both important factors to be included in the two DoF vehicle model in order to predict vehicle trajectory accurately. Closed-loop simulations with the MPC-based obstacle avoidance algorithm are also conducted using various obstacle fields. The performance is characterized by several metrics including the time to target, control effort, and integral of curvature. The results show that a two DoF vehicle model with linear tire model and constant axle loads can perform comparable to the fourteen DoF model when the vehicle travels at lower speed. However, a two DoF model with linear tire model or constant axle loads can fail to



(a) Trajectory. (b) Trajectory curvature and steering angle.

Figure 3.5: Simulation results to study the effect of model fidelity with MAP C.

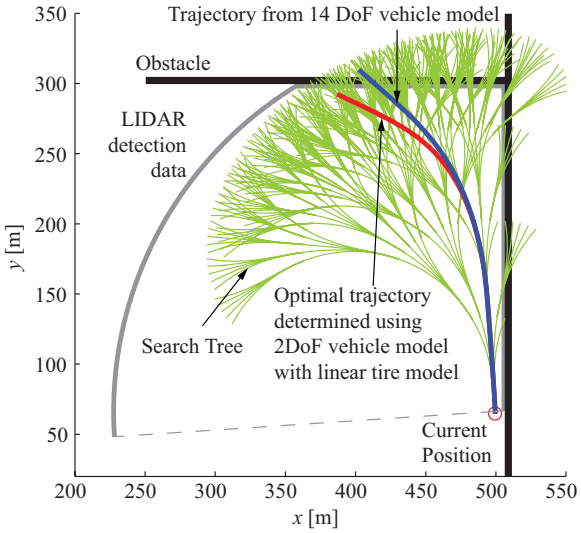


Figure 3.6: One step prediction made by the two DoF vehicle model with linear tire model when the obstacle enters the LIDAR range.

navigate the vehicle safely when the vehicle moves at higher speed within an obstacle field including large obstacles. It is concluded that the two DoF representation that accounts for tire nonlinearities and longitudinal load transfer is necessary for the MPC-based obstacle avoidance algorithm in order to operate the vehicle at its limits within an environment that includes large obstacles. For less challenging environments, however, the two DoF representation with linear tire model and constant axle loads is sufficient.

Future research opportunities are summarized as follows. The study pertains to the specific four-wheeled truck; for a different vehicle, the procedures presented in this chapter can be repeated. However, it would be beneficial if the model fidelity requirement can be determined easily for a series of platforms with different weight and CoG location. In addition, the model fidelity requirement also needs to be re-evaluated when the vehicle parameter and state measurements uncertainties are accounted for or when the environment is non-flat.

CHAPTER 4

LIDAR Data Processing

4.1 Introduction

The LIDAR data processor part of the obstacle avoidance algorithm shown in Figure 1.7 processes a sequence of points defining the obstacle-free region into specifications of constraints that can be used in the OCP formulation. The obstacle-free region is considered as the safe region in which the vehicle can operate, hence the terms 'obstacle-free region' and 'safe region' are used interchangeably in this context. In this thesis, a planar LIDAR sensor is used and the set of points defining the safe region is simply the points generated by the LIDAR. The instantaneous LIDAR data are used in this work for simulation simplicity. In real world, it is beneficial to utilize LIDAR data from multiple steps and maintain a continuously updated map of the environment to provide more accurate and more complete obstacle information. This exploitation of prior measurements can be achieved by augmenting the algorithm presented in this session with SLAM algorithms [100]. Nevertheless, this extension would not change the safe region partitioning approach presented below, as it takes as the input a set of points describing the obstacle boundaries regardless of how these points are obtained.

4.2 Line Simplification

The first step of the data processing is to reduce the number of points that defines the obstacle boundaries for further processing because the raw points obtained from the LIDAR can be noisy because the obstacle boundaries are not smooth and the detection results are not exact. In this work, the LIDAR noise is simulated by a

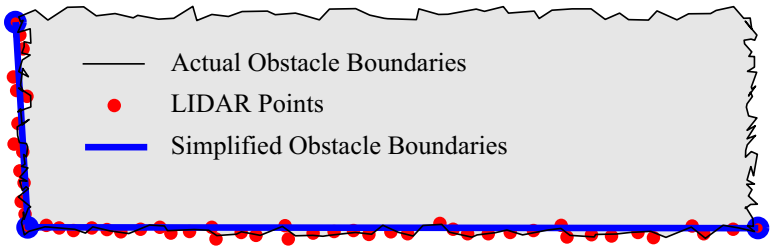


Figure 4.1: An illustration of the line simplification algorithm.

uniform distribution of range $[-0.1,\ 0.1]$ m. An algorithm is then required to identify the minimum number of lines for approximating the sequence of points considering the noise. The Ramer-Douglas-Peucker algorithm is used for reducing the number of points in a curve that is approximated by a series of points, which is widely used to perform simplification and denoising of range data acquired by a LIDAR sensor [101] and has been adopted in this work, as well. As illustrated in Figure 4.1, the numerous detected points, which are generated using a simulated LIDAR sensor with added noise, can be simplified into two line segments represented by three points, which provide a good approximation to the boundaries of the obstacle.

4.3 Safety Margin

A safety margin, l_{SM}, is added to the safe region to account for the size of the vehicle, detection noise, and differences between the predicted trajectory and the actual trajectory. Adding a safety margin allows for ignoring the vehicle size in the OCP formulation.

The safe area is a polygon in general and it is a simple polygon when the safe region boundary is from a planar LIDAR sensor. Thus, algorithms for performing polygon offsetting (inflating/deflating) in computer graphics can be adopted to add the safety margin to the safe region. Specifically, the Vatti's clipping algorithm [102] implemented in the Clipper library [103] is used in this work.

The safety margin needs to be added to only specific segments of the safe region boundary. As shown in Figure 4.2, which corresponds to the example in Figure 1.8, the boundary of the safe region consists of three types of segments:

- LIDAR data segments, which specify the boundaries of the obstacles and are called 'obstacle boundaries'.
- Maximum LIDAR detection range segments, which are directions free from obstacles and are called 'openings'.

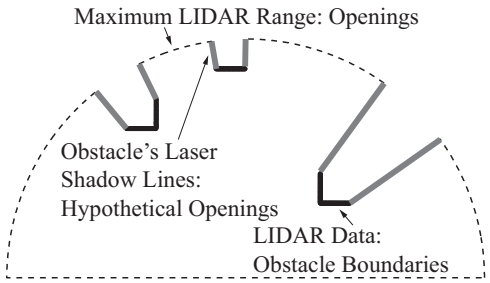


Figure 4.2: Three types of segments bounding the safe region.

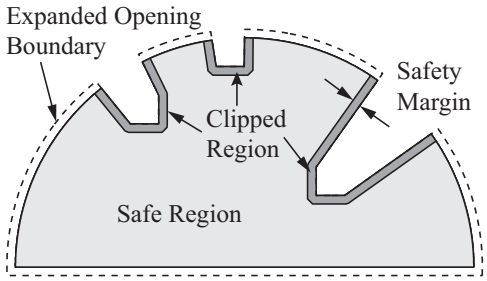


Figure 4.3: Safe region in Figure 4.2 with safety margin included.

- Obstacle's laser shadow lines, which are along the rays from the LIDAR and are called 'hypothetical openings'. These lines are called 'hypothetical openings', because in its current position and orientation the vehicle cannot know whether they are actual openings or not because of the blocked view by the obstacles.

The safety margin is to be applied to the obstacle boundaries and hypothetical openings only, but the Clipper library shrinks all the boundaries. Hence, the openings are first expanded by the same amount as the safety margin. The expanded region is then shrunk with the Clipper library, which effectively results in applying the safety margin only to the obstacle boundaries and hypothetical openings, and not to the openings. As an example, Figure 4.3 is the safe region with safety margin corresponding to Figure 4.2.

4.4 Region Partitioning

The safe region exemplified in Figure 4.3 is very difficult, if not impossible, to be represented mathematically using a single inequality. Even when the function exists, it is not differentiable at the connection points of the line segments from the first step. This would cause problems in the OCP solver that requires all functions to be twice continuously differentiable.

In order to represent the safe region in a mathematically suitable form for the OCP formulation, it is partitioned into several sub-regions, where each sub-region can be specified by a set of inequalities that are not piecewise functions and are differentiable. Two partitioning approaches are discussed below that can efficiently yield such sub-regions. Other partitioning approaches can be used, as well, as long as they provide efficient means for obtaining sub-regions that can be mathematically defined using non-piecewise and differentiable functions.

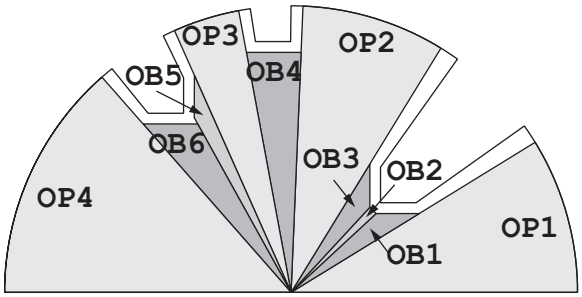
One approach is named the 'polar partitioning'. The safe area is divided into sectors and triangles, where sectors are regions including an opening and triangles are regions including an obstacle boundary. Figure 4.4a is the partitioning of the safe region shown in Figure 4.3 using this approach, where `OP` and `OB` denote the regions that terminate with an opening and obstacle boundary, respectively. As an example, Region `OB4` is a triangle, which can be specified using three linear inequalities, whereas Region `OP3` is a sector, which is bounded by two lines. The third boundary of Region `OP3` is an arc, which is specified by the prediction horizon. Thus, no additional inequality is required. Polar partitioning is easy to implement, because the original LIDAR data is in polar coordinates.

The second approach is called the 'optimal convex partitioning' or simply 'convex partitioning'. The interior of the safe area is decomposed into a minimum number of convex regions without introducing additional points inside the polygon. Specifically, the dynamic programming algorithm in [104] is incorporated into this work, which is very efficient in decomposing simple polygons. In this approach, `OP` denotes the regions that terminate with an opening, whereas `OB` denotes all other regions. Figure 4.4b is an example partitioning for the safe region in Figure 4.3 using the convex partitioning approach. Similar to the polar partitioning approach, all the regions can be specified using a set of linear inequalities after partitioning. This approach is capable of partitioning a safe region in a more general form. Thus, the algorithm is not limited to be used only with a planar LIDAR sensor. As long as a safe region can be deduced from the sensor data, the algorithm is applicable.

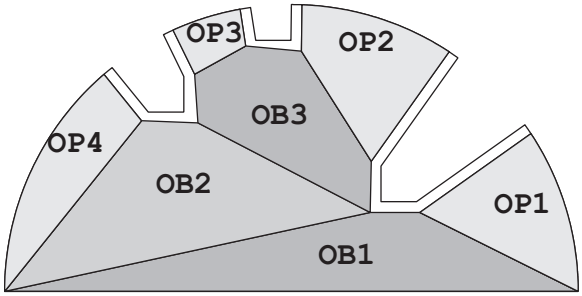
In either approach, a sub-region can be defined by

$$\begin{bmatrix} \vdots & \vdots \\ a_j & b_j \\ \vdots & \vdots \end{bmatrix}^{(i)} \begin{bmatrix} x^{(i)} \\ y^{(i)} \end{bmatrix} \leq \begin{bmatrix} \vdots \\ c_j \\ \vdots \end{bmatrix}^{(i)} , j = 1, \ldots, L^{(i)} \tag{4.1}$$

where i is the sub-region index and $L^{(i)}$ is the total number of line segments bounding



(a) Polar partitioning.



(b) Convex partitioning.

Figure 4.4: Example partitioning of the safe region in Figure 4.3 using two approaches.

that sub-region; a_j, b_j and c_j are coefficients calculated based on the two end points of the corresponding line segments; (x, y) is a position in Cartesian coordinates.

Equation (4.1) can be compacted in the following form:

$$\boldsymbol{A}^{(i)}_{L^{(i)}\times 1} x^{(i)} + \boldsymbol{B}^{(i)}_{L^{(i)}\times 1} y^{(i)} \leq \boldsymbol{C}^{(i)}_{L^{(i)}\times 1} \tag{4.2}$$

where $\boldsymbol{A}^{(i)}_{L^{(i)}\times 1}$ is a vector with the jth entry being a_j. The definitions of $\boldsymbol{B}^{(i)}_{L^{(i)}\times 1}$ and $\boldsymbol{C}^{(i)}_{L^{(i)}\times 1}$ are similar.

After partitioning, the entire safe region can be specified by a structure variable `SafeRegion`, whose definition is given by the following pseudo-code.

```
int N; //number of sub-regions
int L[N]; //vector of number of line segments

struct SafeRegion {
   double AM[N][N]; //adjacency matrix
   SubRegion SR[N]; //subregion specifications
};

struct SubRegion {
```

```
    char Type; //type of the subregion: `OP', `OB'
    int Index; //index of the subregion
    int HPNum; //number of hypothetical openings
    double EndPoints[L(Index)][4];
    //end points of line segments (xs, ys, xe, ye)
    double HPIndex[HPNum];
    //index of line segments representing hypothetical openings
};
```

An adjacency matrix $\mathcal{M}_{\text{adjacency}}$ can be used to represent the partitioning shown in Figure 4.4b. When two sub-regions have a common edge, the corresponding entry in the matrix is set to 1; otherwise, it is set to 0.

$$\mathcal{M}_{\text{adjacency}} = \begin{array}{c} \\ \text{OP1} \\ \text{OP2} \\ \text{OP3} \\ \text{OP4} \\ \text{OB1} \\ \text{OB2} \\ \text{OB3} \end{array} \begin{array}{c} \begin{array}{ccccccc} \text{OP1} & \text{OP2} & \text{OP3} & \text{OP4} & \text{OB1} & \text{OB2} & \text{OB3} \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Once the partitioning is performed, a mixed integer nonlinear programming approach [105, 106, 107] can be pursued for the OCP formulation. Using this approach, the optimal sequence of sub-regions that generates the optimal solution can be obtained. However, this approach is computationally intensive. Thus, an alternative strategy is pursued in this work. In particular, the feasible sequences of the sub-regions are first obtained. The number of the feasible sequences is often limited. Then, multiple OCPs are formulated using the multi-phase nonlinear programming approach [108, 109, 110]. The general form of the OCP formulation given by Equation (1.1) - Equation (1.7) has one set of constraints that are applied to the entire prediction horizon. In a multi-phase OCP formulation, the prediction horizon is divided in N-segments, where N is the number of phases. The constraints on the state and control variables over each segment can be different. Thus, this formulation is used because the position constraints of different sub-regions are different. Additional constraints are added to ensure the continuity of the state and control variables between the phases. These concepts are further illustrated in Figure 4.5.

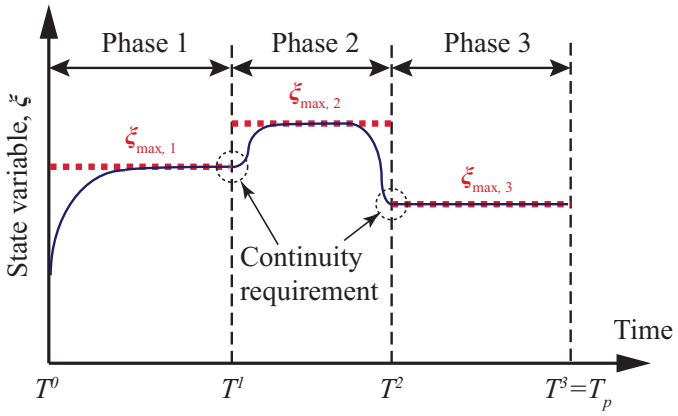


Figure 4.5: An illustration of multi-phase OCP formulation.

In particular, for each feasible opening, a sequence of sub-regions the vehicle needs to move through is obtained using the Dijkstra's algorithm. Thus, a multi-phase OCP is formulated for each of the feasible openings. These formulated problems are solved in parallel. The final solution is obtained by comparing the solutions of these problems and picking the one that gives the minimum cost. In summary, one or more OCPs are formulated and solved at each step. Moreover, it may be possible to use some heuristics to choose the best feasible opening based on the vehicle state and target position, however, a rigorous approach is used in this work.

To avoid collision with the obstacles and move towards the target, the trajectory should stay within the safe region and the last part of the predicted trajectory should lie within the sub-region of type `OP`. The following list presents the procedures of using the structure variable **`SafeRegion`** to form the constraints in the OCP formulation to meet the above requirement. The safe region partitioning shown in Figure 4.4b is used as an example to elaborate.

(a) Identify the first region to traverse, `SR` (starting region). Region `OB1` is the `SR` in this example, because the current position of the vehicle is located on one of its boundaries.

(b) Identify all regions with a feasible opening, `TRs` (terminal regions). A feasible opening is an arc segment, which can have an intersection with a feasible trajectory over a slightly longer period than the prediction horizon. When the longitudinal speed of the AGV is maintained constant, the `TRs` can be identified using the two extreme trajectories, which are shown in Figure 4.6. The two extreme trajectories are obtained by simulating a two DoF vehicle model using steering controls that are at the limits of handling at each step with the mea-

sured initial states. In the example, the arc segments in OP2 and OP3 are feasible openings and hence they are TRs, whereas, OP1 and OP4 are not TRs, because the vehicle cannot make a sharp enough turn to move into those partitions safely. When the longitudinal speed of the AGV is also to be optimized, all openings are considered as TRs.

(c) Find the sequence of regions from the SR to a TR for all TRs. This is a shortest path problem, where many algorithms can be used, including Dijkstra and A*. Dijkstra's algorithm [111] is chosen here, because it is one of the simplest solutions. For example, with OP3 being the TR, the region sequence is identified as OB1 → OB2 → OB3 → OP3. Figure 4.6 highlights this region sequence. With OP2 being the TR, the region sequence is identified as OB1 → OB2 → OB3 → OP2.

For a region sequence from the 'polar partitioning' as exemplified in Figure 4.7a, a different region partition as shown in Figure 4.7b can be obtained easily. This alternative partition approach is preferred when one of the boundaries separating two regions is almost along the vehicle heading direction.

The specifications of the regions with this partitioning approach are given by

$$\begin{aligned} R_{\min}^{(i)} &\leq \sqrt{[x^{(i)} - x_0^{(i)}]^2 + [y^{(i)} - y_0^{(i)}]^2} \leq R_{\max}^{(i)} \\ \Phi_{\min}^{(i)} &\leq \operatorname{atan2}(y^{(i)} - y_0^{(i)}, x^{(i)} - x_0^{(i)}) \leq \Phi_{\max}^{(i)} \end{aligned} \tag{4.3}$$

where $R_{\min}^{(i)}$, $R_{\max}^{(i)}$, $\Phi_{\min}^{(i)}$, and $\Phi_{\max}^{(i)}$ are bounds calculated from the coordinates of end points specifying a region.

In the simulations presented in this thesis, the convex partition approach is used primarily, and the polar partition approach is used secondarily as a failsafe approach in case the optimal solution to the problem formulated using the convex partition cannot be obtained.

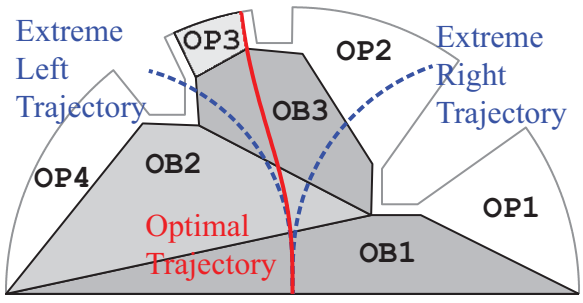
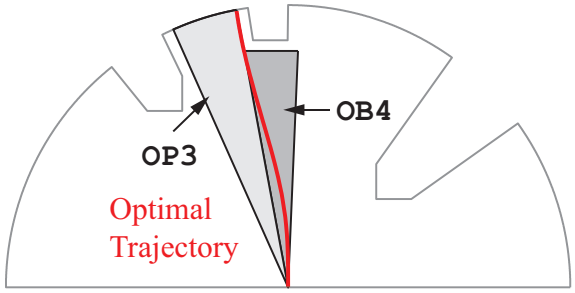
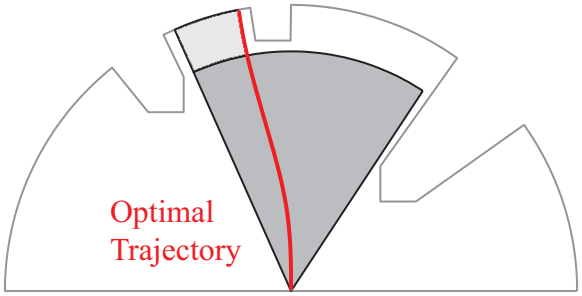


Figure 4.6: An example of extreme trajectories and region sequence.



(a) Basic polar partitioning.



(b) Modified polar partitioning.

Figure 4.7: Regions from the polar partitioning approach and its variance.

CHAPTER 5

Constant Speed Formulation and Results

5.1 Introduction

In this chapter, a novel nonlinear MPC-based algorithm is developed for obstacle avoidance of AGVs that can achieve an optimal and smooth operation of the vehicle through the obstacle field at high speed while ensuring vehicle safety; i.e., without wheel lift-off. As a starting point, the vehicle longitudinal speed is first maintained to be constant and is provided to the algorithm as an input in this chapter.

The novelty of the chapter compared to works such as [57, 61, 62, 63, 65, 68, 73, 89] is three-fold: (1) This work focuses on unstructured environments without a reference trajectory. A new cost function formulation is used that aims to find the shortest path to the target position in addition to approaching the target from a desired direction and minimizing the control effort. (2) The differentiable mathematical representation of the obstacle-free region obtained in Chapter 4 is included in the OCP formulation through a multi-phase approach. (3) The algorithm considers vehicles with relatively higher CoG and explicitly accounts for the vehicle dynamical safety in terms of avoiding single-wheel lift-off. This is achieved by limiting the steering angle within a range obtained offline using a fourteen DoF vehicle dynamics model. Simulations of an AGV in three different obstacle fields are given to demonstrate the effectiveness of the proposed algorithm.

5.2 OCP Formulation

As discussed in Chapter 4, one or more OCPs are formulated and solved at each step of the MPC. The formulation in general form is given by Equation (5.1) - Equation (5.8) below.

$$\underset{\xi,\,\zeta,\,T^1,\cdots,T^N}{\text{minimize}} \quad J = \mathcal{T}\left[\boldsymbol{\xi}^{(N)}\left(T^N\right), \boldsymbol{\xi}^{(1)}\left(T^0\right), T^N\right] + \sum_{i=1}^{N}\left\{\int_{T^{i-1}}^{T^i} \mathcal{I}^{(i)}\left[\boldsymbol{\xi}^{(i)}(t), \boldsymbol{\zeta}^{(i)}(t)\right] \mathrm{d}t\right\} \tag{5.1}$$

$$\text{subject to} \quad \dot{\boldsymbol{\xi}}^{(i)}(t) = \boldsymbol{\mathcal{V}}\left[\boldsymbol{\xi}^{(i)}(t), \boldsymbol{\zeta}^{(i)}(t)\right] \tag{5.2}$$

$$\forall i = 1, \cdots, N$$

$$\boldsymbol{\xi}^{(i)}\left(T^{i-1}\right) = \boldsymbol{\xi}^{(i-1)}\left(T^{i-1}\right) \tag{5.3}$$

$$\boldsymbol{\mathcal{R}}^{(i)}\left[x^{(i)}(t), y^{(i)}(t)\right] \leq 0 \tag{5.4}$$

$$\delta_{f,\min}(U_0) \leq \delta_f^{(i)}(t) \leq \delta_{f,\max}(U_0) \tag{5.5}$$

$$\gamma_{f,\min} \leq \gamma_f^{(i)}(t) \leq \gamma_{f,\max} \tag{5.6}$$

$$\text{subject to} \quad \boldsymbol{\xi}^{(0)}\left(T^0\right) = \boldsymbol{\xi}^0 \tag{5.7}$$

$$T^0 = 0, T^N = T_p, T_{p,\min} \leq T_p \leq T_{p,\max} \tag{5.8}$$

By minimizing the cost function specified in Equation (5.1), subject to constraints defined by Equations (5.2) - (5.6) for all phases and constraints given by Equations (5.7) and (5.8) for the initial and final state and time values, the optimal state trajectories $\boldsymbol{\xi}^{*(i)}(t), t \in [T^{i-1}, T^i]$, the optimal control trajectories $\boldsymbol{\zeta}^{*(i)}(t), t \in [T^{i-1}, T^i]$, and the time points T^{i-1}, T^i, $i = 1, \ldots, N$ for transitioning from one sub-region to the next are obtained, where N is the total number of phases, which is the number of regions from SR to the TR.

It is helpful to explain the constraints in the formulation at a high level before delving into the details of the formulation. Equation (5.2) is the dynamic model of the vehicle represented as a set of first order ODEs. Equation (5.3) sets the initial state values of each phase as the final state values of the previous phase. For the first phase, the initial states are the measured states, which is defined as Equation (5.7). All states are continuous at the boundaries of the phases. Equation (5.4) defines the position constraints due to the obstacles perceived by the LIDAR sensor. A general form is given here. Equation (5.5) and Equation (5.6) represent the bounds on the steering angle and the steering rate, respectively. Equation (5.8) specifies the prediction horizon, which is given by $t \in [0, T_p]$. The following sub-sections define the

variables and explain the problem formulation in detail. The constraints are discussed first, and the cost function formulation is explained next.

5.2.1 Equation (5.2): Vehicle Dynamics Model

The two DoF vehicle dynamics model with longitudinal load transfer and tire nonlinearity is used in the MPC to predict vehicle trajectories.

The steering rate γ_f is used as the control command to be optimized and the steering angle δ_f is set as an additional state variable of the system. The reason for choosing the steering rate as the control input instead of the steering angle is to obtain a smooth steering angle sequence and impose a limit on the steering rate.

By setting the state vector as $\boldsymbol{\xi} = [\ x \quad y \quad \psi \quad V \quad \omega_z \quad \delta_f\ ]^\top$ and the control vector as $\boldsymbol{\zeta} = \gamma_f$, the state-space equation for the two DoF nonlinear vehicle model can be written as

$$\dot{\boldsymbol{\xi}} = \boldsymbol{f}_{\text{2DoF}}(\boldsymbol{\xi}) + \boldsymbol{B}_{\text{2DoF}}\boldsymbol{\zeta} \tag{5.9}$$

where

$$\boldsymbol{f}_{\text{2DoF}}(\boldsymbol{\xi}) = \begin{bmatrix} U_0\cos\psi - (V + L_f\omega_z)\sin\psi \\ U_0\sin\psi + (V + L_f\omega_z)\cos\psi \\ \omega_z \\ (F_{y,f} + F_{y,r})/M - U_0\omega_z \\ (F_{y,f}L_f - F_{y,r}L_r)/I_{zz} \\ 0 \end{bmatrix}$$

$$\boldsymbol{B}_{\text{2DoF}}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

5.2.2 Equation (5.4): Safe Region Constraints

Obstacle avoidance is enforced through the constraint that the vehicle trajectory must lie within the safe region. For each of the phases in the multi-phase OCP, the vectors $\boldsymbol{A}^{(i)}_{L^{(i)}\times 1}$, $\boldsymbol{B}^{(i)}_{L^{(i)}\times 1}$, and $\boldsymbol{C}^{(i)}_{L^{(i)}\times 1}$ or the bounds $R^{(i)}_{\min}$, $R^{(i)}_{\max}$, $\Phi^{(i)}_{\min}$, and $\Phi^{(i)}_{\max}$ can be calculated using the values stored in the structure variable `SafeRegion` defined in Section 4.4. The specific form of Equation (5.4) is either given by Equation (4.2) or Equation (4.3) depending on the situation.

5.2.3 Equation (5.5): Dynamical Safety Constraints

In this study, ensuring the vehicle's dynamical safety is defined as avoiding single-wheel lift-off. This is a conservative criterion used to prevent rollover [27]. With

the specified dynamical safety requirement, all four tire vertical loads should be greater than a specified threshold value during the whole process as described in Equation 5.10.

$$F_{z,i}(t) \geq F_{z,\text{threshold}},\ \forall i = fl, fr, rl, rr, t \in [0,\ T_p] \tag{5.10}$$

Only when a vehicle model with the capability of predicting all four tire vertical loads is used in the OCP formulation, the criterion described by Equation (5.10) can be directly used. The simple vehicle representation used to predict the vehicle trajectory in the MPC does not offer this capability at its current form. In order to use Equation (5.10) as a constraint for ensuring dynamical safety, additional complexity needs to be added to the MPC model, which is not desired. Thus, another conservative approximation of the dynamical safety requirement is considered for the assumed constant-speed application; namely, an upper bound on the steering angle magnitude as expressed by the following inequality constraint

$$|\delta_f(t)| \leq \delta_{f,\max}(U_0) \tag{5.11}$$

where the maximum steering angle $\delta_{f,\max}$ is a function of the vehicle speed U_0, which has no analytical expression and is represented using a look-up table that is described below. In general, there exist other factors that can affect the maximum steering angle, such as the slope of the terrain, or the location of the CoG of the vehicle. However, in this work, the vehicle is assumed to move on a constant-friction surface and all parameter values remain constant. Therefore, the maximum steering angle is only a function of vehicle speed.

For all combinations of longitudinal speed ranging from 10 m/s to 30 m/s and maximum steering angle ranging from 0° to 14°, the corresponding minimum tire vertical loads are obtained using the fourteen DoF vehicle model. The relationship is shown in Figure 5.1. If a minimum vertical load threshold is set, the relationship between the maximum steering angle and longitudinal speed can be extracted. Figure 5.2 shows the relationship when $F_{z,\min}$ is set as 500 N.

5.2.4 Equation (5.1): Cost Function

The cost function defines the soft requirement, that is, in what sense the trajectory is optimal. Since a reference trajectory does not exist in this work, the cost function aims to find the shortest path. If the task is only to pass a target location without

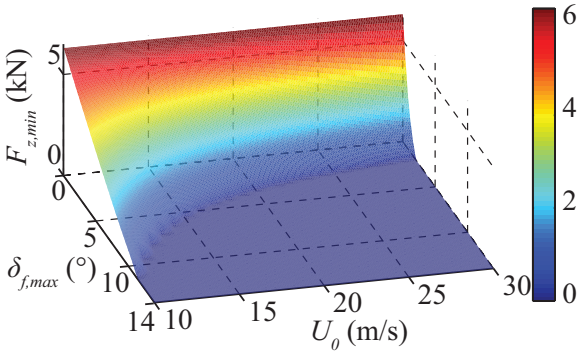


Figure 5.1: Minimum tire vertical load at different combinations of vehicle speed and maximum steering angle.

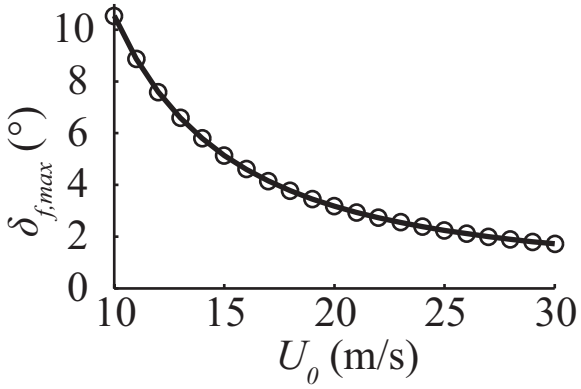


Figure 5.2: Maximum steering angle as a function of vehicle longitudinal speed when the minimum vertical load threshold is 500 N.

a specific direction requirement, the trajectory is optimal when the end point of the predicted trajectory is close to the target, and the final heading angle is pointing to the target, because a shorter distance-to-go is preferred to minimize travel time. The cost function for this case is defined as

$$J = \frac{s_f}{s_0} + w_\psi \psi_{\text{diff}}^2 + w_{\text{cf}} v_{\text{cf}} \tag{5.12}$$

where

$$s_0 = \sqrt{[x_t - x(0)]^2 + [y_t - y(0)]^2} \tag{5.13}$$

$$s_f = \sqrt{[x_t - x(T_p)]^2 + [y_t - y(T_p)]^2} \tag{5.14}$$

$$\psi_{\text{frg}} = \text{atan2}\left[y_t - y(T_p), x_t - x(T_p)\right] \tag{5.15}$$

$$\psi_{\text{diff}} = \text{atan2}\left[\sin(\psi(T_p) - \psi_{\text{frg}}), \cos(\psi(T_p) - \psi_{\text{frg}})\right] \tag{5.16}$$

$$v_{\text{cf}} = \int_0^{T_p} \left[\gamma_f^2(t) + w_\delta \delta_f^2(t)\right] \, \mathrm{d}t \tag{5.17}$$

Specifically, the cost function formulation includes three terms that are linearly combined using relative weights, w_ψ and w_{cf}.

The first term is a ratio between s_f and s_0, where s_0 is the distance between the initial position $[x(0), y(0)]$ and the target position $[x_t, y_t]$ as defined in Equation (5.13), and s_f is the distance between the end point of the predicted trajectory $[x(T_p), y(T_p)]$ and the target as defined in Equation (5.14). Visual representations of all variables are shown in Figure 5.3. The second term is the difference between the final heading angle $\psi(T_P)$ and the angle of the target relative to the end point of the predicted trajectory ψ_{frg} as defined in Equation (5.16). The weighted sum of the first two terms is used to lead the vehicle to the specified target position from its current position. This weighted sum captures the distance-to-go with the assumption that there exist no obstacles beyond the sensed horizon. The third term is a regularization term minimizing the control effort v_{cf} as defined in Equation (5.17), where γ_f is the steering rate, which is the control command to be optimized, δ_f is the front wheel steering angle, and w_δ is a weight.

If a particular direction of passing the target location in global coordinates is also required, the following cost function is used

$$J = \frac{s_f}{s_0} + w_\psi \psi_{\text{diff}}^2 + w_{\text{dt}} v_{\text{dt}} + w_{\text{cf}} v_{\text{cf}} \tag{5.18}$$

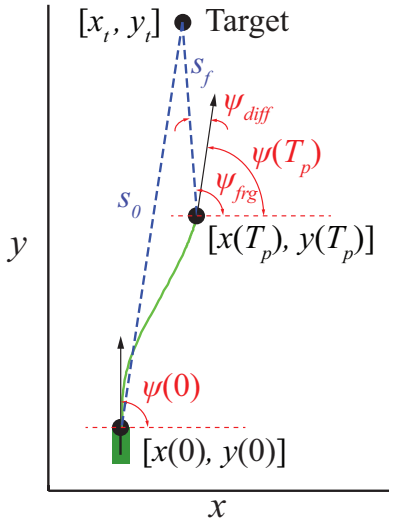


Figure 5.3: An illustration of the variables included in the terminal cost definitions.

where

$$v_{\mathrm{dt}} = \int_0^{T_p} [l_a x(t) + l_b y(t) + l_c]^2 \, \mathrm{d}t \tag{5.19}$$

$$\begin{aligned} l_a &= \sin(\psi_t), l_b = -\cos(\psi_t) \\ l_c &= -\sin(\psi_t) x_t + \cos(\psi_t) y_t \end{aligned} \tag{5.20}$$

The cost function specified by Equation (5.18) has one more term than Equation (5.12). This term represents the integral of the distance to the line given by $l_a x + l_b y + l_c = 0$ over the prediction horizon. This line passes through the target position $[x_t, y_t]$ along the desired direction ψ_t.

When the target position is within the sensor's detection range, the term s_f/s_0 and $w_\psi \psi^2_{\mathrm{diff}}$ are removed from the cost functions given in Equation (5.12) and Equation (5.18). Instead the following constraints are added to the OCP formulation.

$$\begin{aligned} x_t - \sigma \leq x(T_p) \leq x_t + \sigma \\ y_t - \sigma \leq y(T_p) \leq y_t + \sigma \end{aligned} \tag{5.21}$$

where σ is a small margin. If the vehicle is within this margin from the target position, then the target is considered to be reached.

5.2.5 Solution Techniques

The nonlinear multi-phase OCPs formulated specified by Equation (5.1) - Equation (5.8) are solved using a two-step procedure. First, the continuous-time OCP

is transcribed into to a nonlinear programming (NLP) problem using a direct method called *hp*-pseudospectral method [108, 109, 110]. Second, the resulting NLP problem is solved using the interior point method [112].

The *hp*-pseudospectral method discretizes a continuous-time OCP into an NLP problem by approximating the state and control variables using a variable number of approximating intervals and variable-degree polynomial approximations of them within each interval. The differential-algebraic constraints of the OCP are enforced at a finite set of collocation points, where the collocation points are Legendre-Gauss-Radau (LGR) quadrature points. This method has been shown to be able to accurately approximate the solution to a general continuous-time OCP in a computationally efficient manner [110].

After transforming from the time interval $t \in [0, T_p]$ to the time interval $\tau \in [-1, 1]$ via the following variable transformation

$$t = (T_p/2)(\tau + 1) \tag{5.22}$$

the state $\boldsymbol{\xi}$ is approximated by a polynomial of degree at most n as follows

$$\boldsymbol{\xi}(\tau) \approx \sum_{p=1}^{n} \boldsymbol{\Xi}_p L_p(\tau), \quad L_p(\tau) = \prod_{q=1, q\neq p}^{n} \frac{\tau - \tau_q}{\tau_p - \tau_q} \tag{5.23}$$

where $\tau_p (p = 1, \ldots, n)$ are the LGR collocation points, $L_p(\tau)(p = 1, \ldots, n)$ are the bases of the Lagrange polynomials, and $\boldsymbol{\Xi}_p$ is the state approximation at τ_p.

For example, after discretization, Equation (5.2) that represents the dynamic model of the vehicle using a set of first-order ODEs can be converted into the following sets of equality constraints to ensure the dynamical feasibility of the results

$$\boldsymbol{\Xi}_p^{(i)} - \boldsymbol{\Xi}_0^{(i)} - \frac{T^{(i-1)} - T^{(i)}}{2} \sum_{q=1}^{n} A_{pq}^{(i)} \boldsymbol{\mathcal{V}} \left[\boldsymbol{\Xi}_q^{(i)}, \boldsymbol{Z}_q^{(i)} \right] = 0, p = 1, \ldots, n \tag{5.24}$$

where A is called the integration matrix, which is defined by the selected LGR collocation points and the corresponding weights [110].

To solve the NLP problem, a primal-dual interior-point algorithm with a filter line search method implemented in IPOPT is used [112]. The basic idea of the interior point method is to decompose the NLP problem with both equality and inequality constraints into a sequence of equality constrained problems by introducing a barrier function and barrier parameter. The NLP problem with only equality constraints

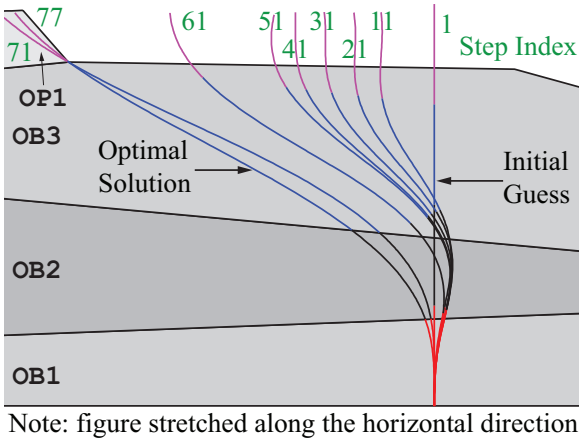


Figure 5.4: The trajectory iterations from the initial guess to the optimal solution for the example in Figure 4.6.

can then be solved iteratively. The search direction is determined using the Newton-Raphson method and the step size is obtained using the backtracking line search.

The interior point method converts the general NLP problem given by Equation (5.25) to a series of NLP problems with only equality constraints given by Equation (5.26).

$$\begin{aligned} &\underset{X\in\mathbb{R}^n}{\text{minimize}} && g(X) && (5.25)\\ &\text{subject to} && \boldsymbol{C}(X) = 0 \\ & && X \geq 0 \\ &\underset{X\in\mathbb{R}^n}{\text{minimize}} && g(X) + \mu_k B(X) && (5.26)\\ &\text{subject to} && \boldsymbol{C}(X) = 0 \end{aligned}$$

where μ is a small positive scalar called 'barrier parameter'. As μ converges to zero, the solution to Equation (5.26) should converge to a solution to Equation (5.25). $B(\boldsymbol{.})$ is a barrier function.

As an example, Figure 5.4 shows the trajectory iterations in solving the problem given in Figure 4.6. This is a four-phase problem. The initial guess is a straight line assuming equal length at each phase, which is not a feasible solution. Nevertheless, after 77 iterations, the solution converges to the optimal solution. Figure 5.5 shows the corresponding objective value and maximum constraint violation at all steps.

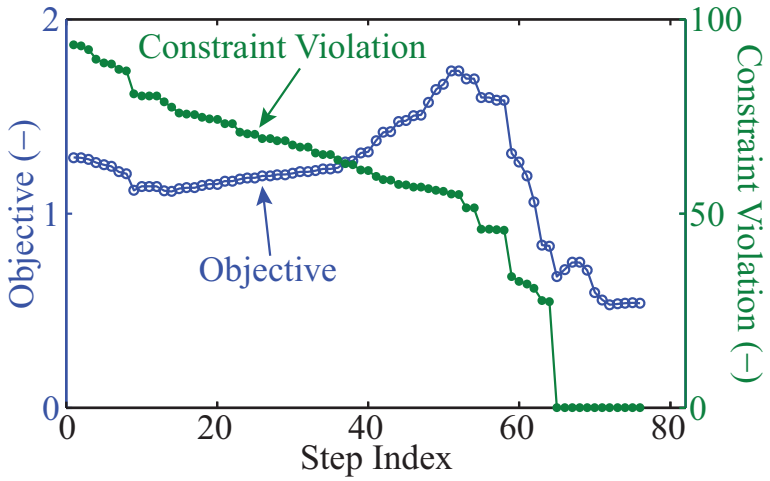


Figure 5.5: The objective and constraint violation at all steps during the iterations in Figure 5.4.

5.3 Simulation Results

This section presents numerical simulations of the developed nonlinear MPC obstacle avoidance algorithm with a fourteen DoF vehicle model as the plant. The algorithm is implemented in MATLAB® for proof-of-concept.

Three test cases are considered in this section. In the first test case, the vehicle is required to move from its initial position to a target position with the final heading angle required to be the same as the initial heading angle. Two obstacles are between the two locations. Vehicle speeds ranging from 10 m/s to 30 m/s are considered.

In the second test case, the vehicle has to traverse a dense obstacle field to reach the target position. There are 50 obstacles and each of them is 10 m $\times$ 10 m in size. The vehicle longitudinal speed is maintained at 20 m/s and there is no constraint on the final heading angle.

In the third test case, the vehicle performs a NATO double lane change maneuver at 15 m/s using the obstacle avoidance algorithm.

Table 5.1 summarizes the weighting parameters in the cost function used for all test cases. If there is no constraint on the final heading angle, the weight w_{dt} is not used. Thus, in the first and third test cases, all four weighting parameters are used. However, in the second test case, w_{dt} is omitted.

Table 5.1: Weighting Parameters

Parameter	Value
w_ψ	1
w_{cf}	10
w_δ	0.1
w_{dt}	10^{-4}

Table 5.2: Algorithm Parameters

U_0 (m/s)	10	15	20	25	30	30
l_{SM} (m)	3	3	3	3	3	3
R_{LIDAR} (m)	100	100	100	100	100	140
$T_{p,\max}$ (s)	10.0	6.7	5.0	4.0	3.3	4.7
T_e (s)	0.67	0.44	0.33	0.27	0.22	0.31
$\varsigma_{f,\max}$ ($^\circ/s$)	10	10	10	10	10	10
$\delta_{f,\max}$ ($^\circ$)	10.5	5.14	3.18	2.24	1.72	1.72

5.3.1 Test Case 1: Various Speeds

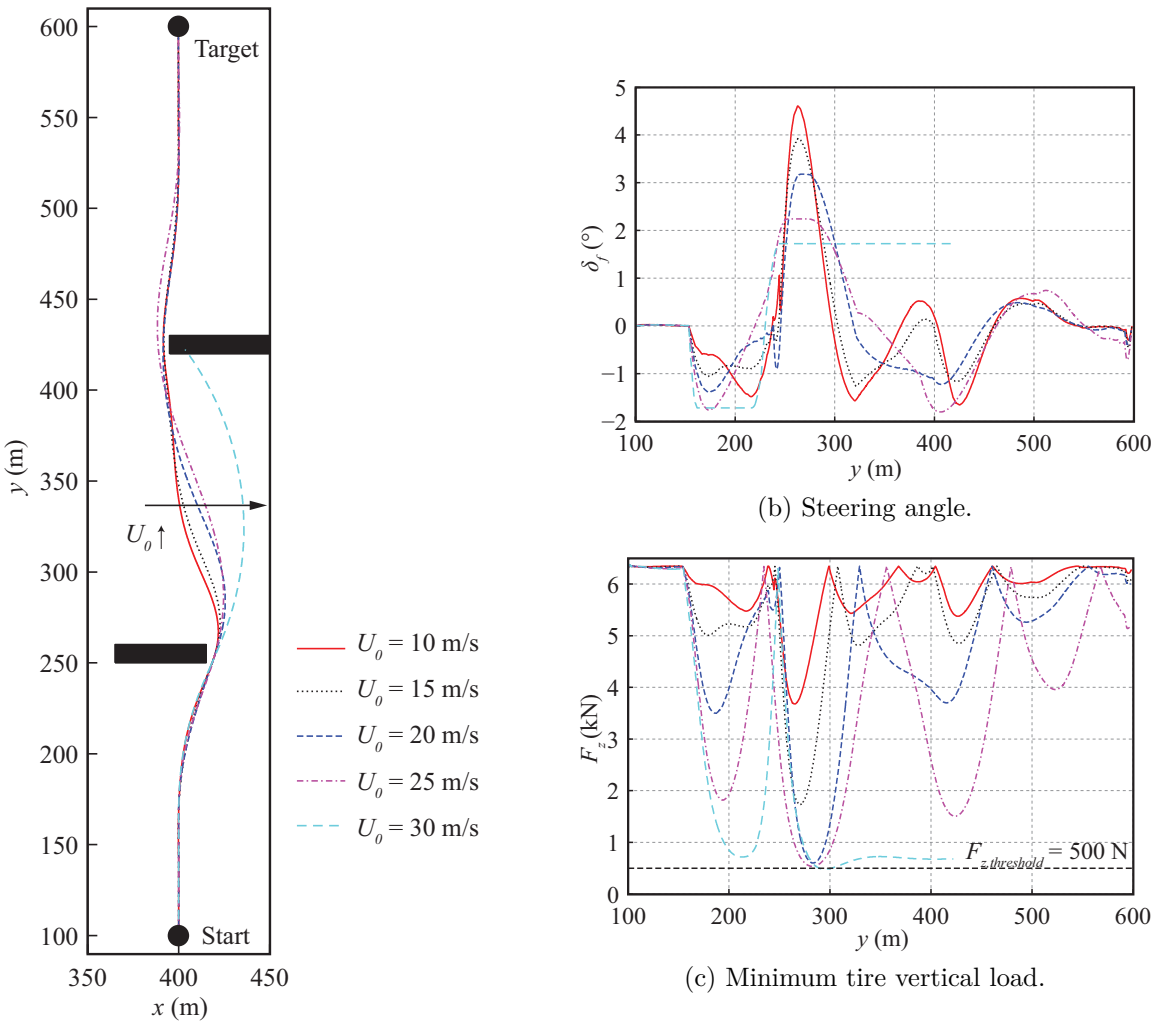
Table 5.2 summarizes the parameters of the nonlinear MPC algorithm, including the safety margin, LIDAR detection range, length of prediction horizon, length of execution horizon, and maximum steering angle. The cost function given by Equation (5.18) is used, because the angle of passing the target position is specified in this test case. In the settings, the following relationship is used, which ensures that all the predicted trajectories lie within the LIDAR detection range.

$$T_{p,\max} = R_{\text{LIDAR}}/U_0 \tag{5.27}$$

The execution horizon specifies the update rate of the algorithm. In this work, it is defined as one fifteenth of the prediction horizon. In Table 5.2, the values of the execution horizon in terms of time are different at different vehicle speeds. However, for a given sensor detection range, the execution horizon in terms of distance is fixed.

The first set of simulations uses a LIDAR with a detection range of 100 m. The results of the simulations are presented in Figure 5.6.

These results show that the developed algorithm can successfully navigate the vehicle through the specified obstacle field at 10 m/s, 15 m/s, 20 m/s, and 25 m/s. At these speeds, the vehicle avoids all obstacles, passes the target from the desired direction, and is dynamically safe as shown in Figure 5.6c. However, the vehicle



(a) Trajectory.

(b) Steering angle.

(c) Minimum tire vertical load.

Figure 5.6: Results of simulations with various longitudinal speed.

hits the second obstacle when the longitudinal speed is maintained at 30 m/s. This is because at this speed, the vehicle is not capable of making a turn at a smaller radius safely. A threshold of 500 N is set on the minimum tire vertical load and the corresponding maximum steering angle is set as a hard constraint in the OCP formulation. This constraint is active at most of the time during the maneuver as shown in Figure 5.6b.

The navigation at 30 m/s fails because the LIDAR detection range is not long enough to accommodate that speed and hence the prediction horizon is too short to prepare the vehicle to avoid the obstacles sufficiently early. Figure 5.7 shows the results of simulations with different LIDAR detection ranges. When a longer detection range of 140 m and a longer prediction horizon are used, the vehicle travels through

the field safely.

5.3.2 Test Case 2: Dense Obstacle Field

This simulation is to test the capability of the algorithm within a dense obstacle field. In this simulation, the vehicle speed is maintained at 20 m/s and there is no constraint on the final heading angle. Hence, Equation (5.12) is used as the cost function and the algorithm parameters are the same as the ones corresponding to 20 m/s in Table 5.2. Figure 5.8 shows the simulation results. The vehicle clears the obstacle field and reaches the target successfully using the algorithm.

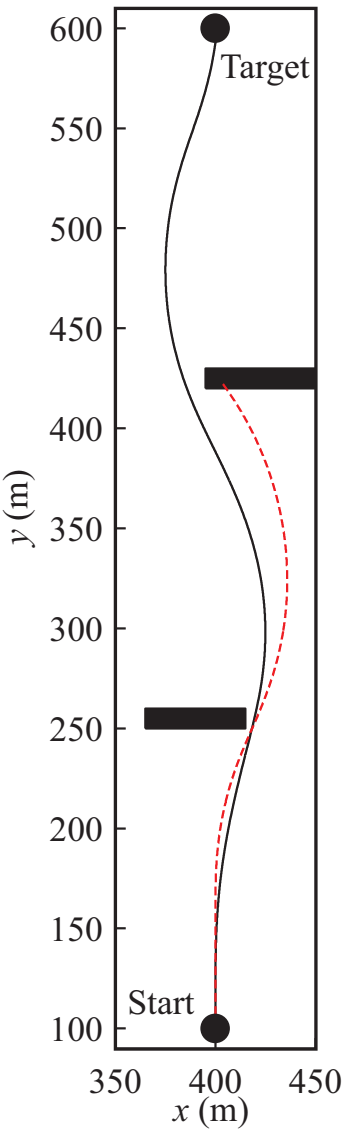
In this test case, at most of the steps, there are multiple feasible openings as exemplified by Figure 5.9. For each of the feasible openings, an OCP is formulated and solved. After all of them are solved, their objective function values are compared and the one with the smallest value is considered the best solution. In this example, the objective values of the calculated trajectories from right to left are 0.76, 0.74, 0.92, respectively. The smallest one is 0.74 and the control commands corresponding to the trajectory in the middle is sent to the plant.

5.3.3 Test Case 3: Double Lane Change

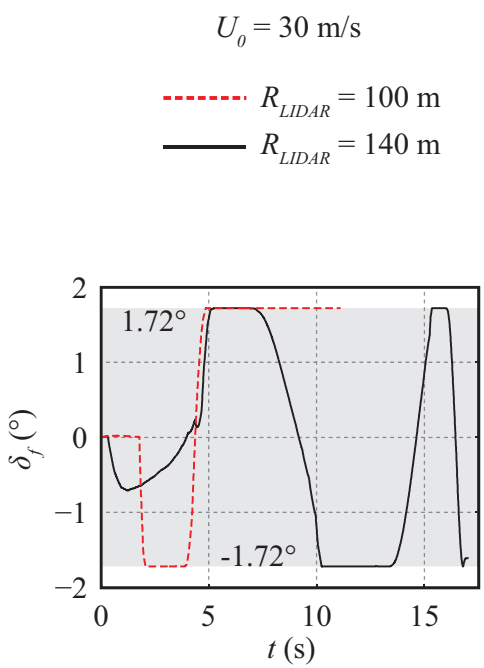
Although this thesis is concerned with unstructured environments, the proposed algorithm can also work for structured environments. The last simulation is to demonstrate the capability of the algorithm in a structured environment. The vehicle performs a double lane change maneuver at 15 m/s using the nonlinear MPC algorithm. Table 5.3 summarizes the parameters used.

Figure 5.10 shows the generated trajectory and the corresponding steering angle. Figure 5.10a shows the trajectory of the CoG of the plant and the corresponding trajectories of the four corners of the vehicle. It can be seen that all the trajectories are within the white space, which means that the vehicle is free from collision. Figure 5.10b is the corresponding steering sequence.

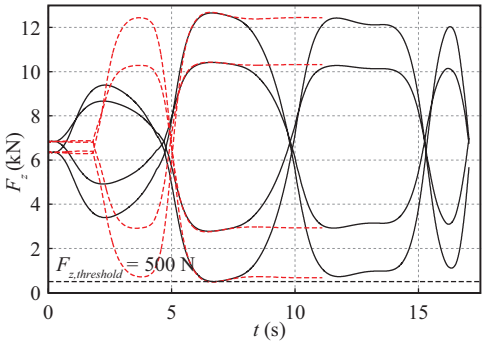
In this test case, in most of the steps, there are no 'openings' as defined in Figure 4.2. However, there are 'hypothetical openings' as defined in Section 4.3. The TRs are then defined as all regions with a feasible hypothetical opening. Figure 5.11 shows the use of a hypothetical opening.



(a) Trajectory.



(b) Steering angle. Shaded area represents the allowable values.

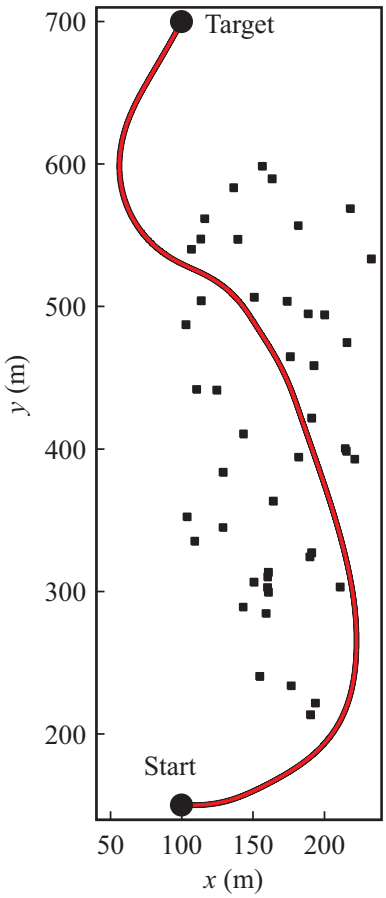


(c) Tire vertical loads.

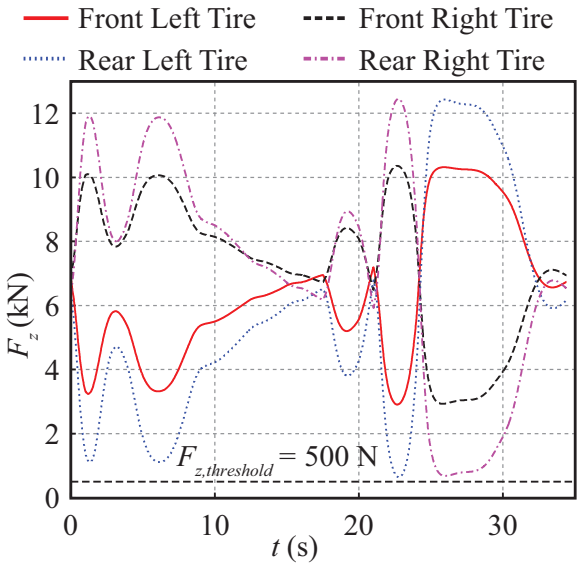
Figure 5.7: Results of simulations with different LIDAR detection ranges at 30 m/s.

Table 5.3: Algorithm Parameters for the Double Lane Change Test

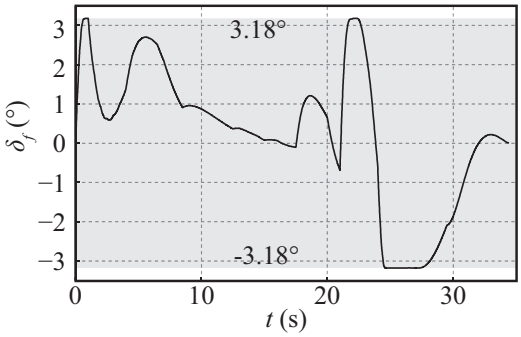
Parameter	Value
l_{SM} (m)	1.6
R_{LIDAR} (m)	50
$T_{p,\max}$ (s)	3.0
T_e (s)	0.3
$\varsigma_{f,\max}$ (°/s)	10
$\delta_{f,\max}$ (°)	5.14



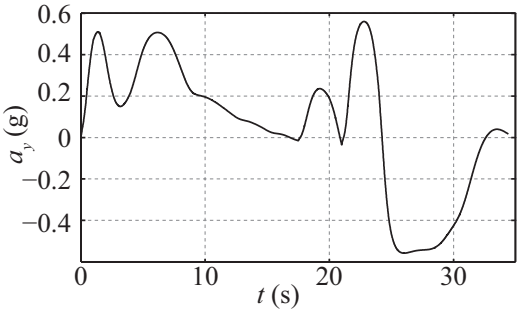
(a) Trajectory.



(b) Tire vertical loads.



(c) Steering angle. Shaded area represents the allowable values.



(d) Lateral acceleration.

Figure 5.8: Simulation results of navigation within a dense obstacle field.

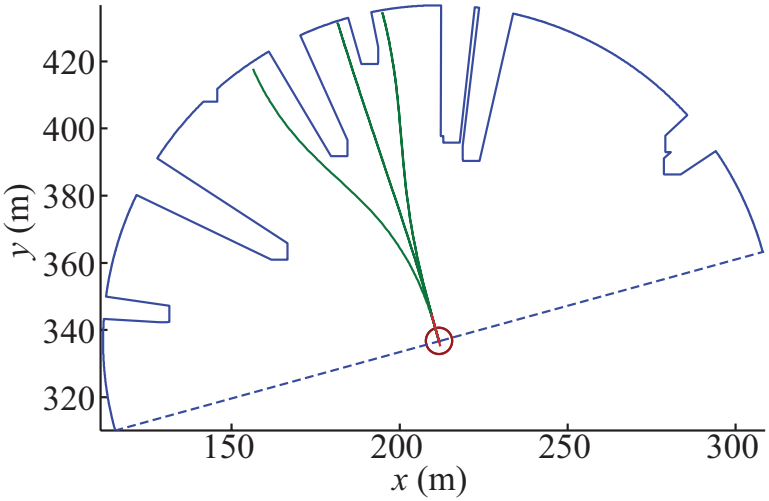
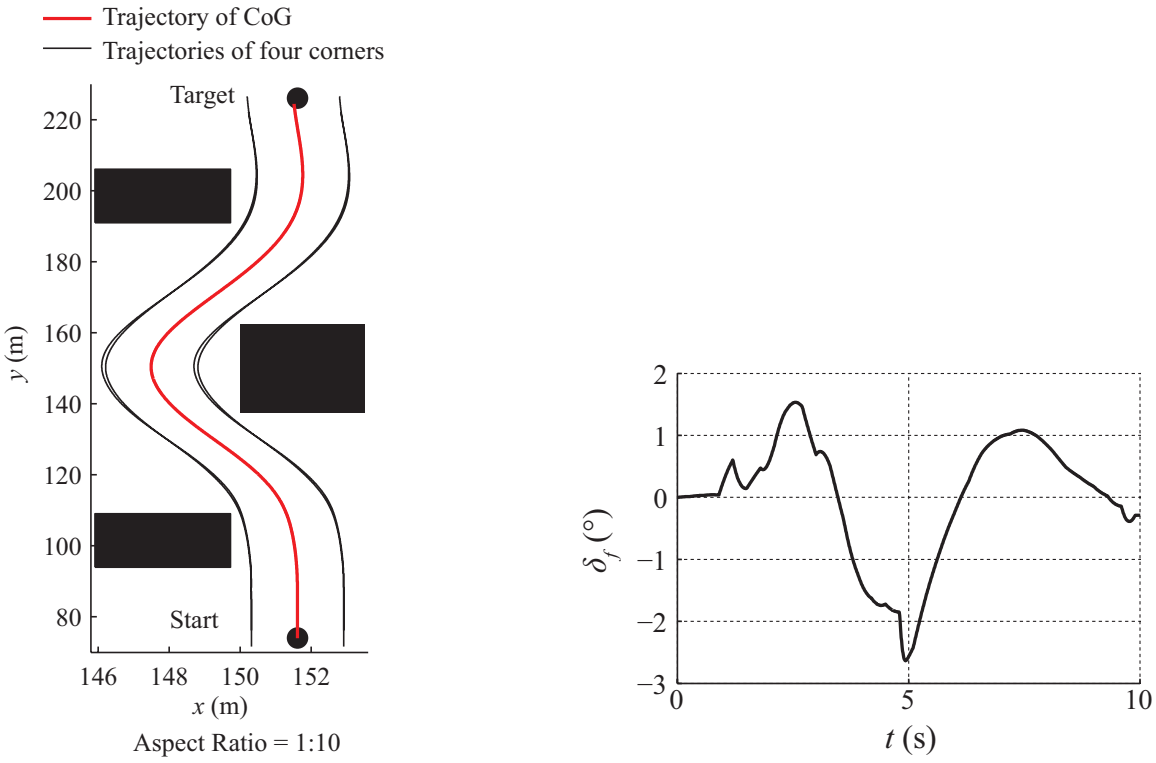


Figure 5.9: An example of the safe region with multiple feasible openings in test case 2.



(a) Trajectory.

(b) Steering angle.

Figure 5.10: Simulation results of the double lane change maneuver.

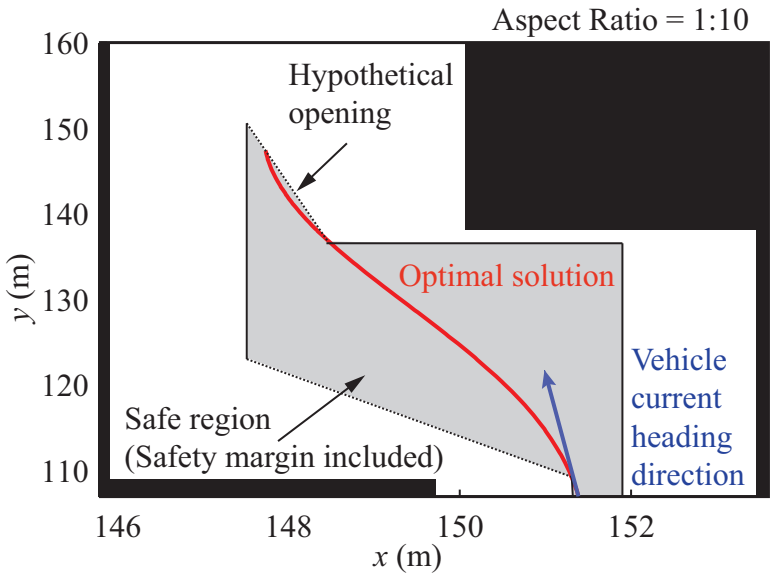


Figure 5.11: An example of the usage of a hypothetical opening in test case 3.

5.4 Discussion

It is assumed that the obstacles are static. This assumption is suitable for the context of this work considering the fact that in an unstructured environment AGVs typically encounter static obstacles more often. Nevertheless, extension to moving obstacles would certainly broaden the utility of the algorithm and is a recommended direction for further research.

It is also that the vehicle speed is provided to the algorithm as an input. The obstacle avoidance algorithm as presented in this chapter is not capable of determining the maximum speed that can be used to safely navigate through an obstacle field, because the on-board sensors provide information about the environment within only the close proximity of the vehicle. However, a conservative lower bound on the prediction horizon or a conservative lower bound on the sensor detection range can be imposed to ensure that the obstacle avoidance maneuver is performed early enough. These limits could be obtained from the trajectory for making a 90° turn when the initial steering angle is at the minimum bound, which is considered as the most extreme maneuver. For example, the trajectories from speeds ranging from 10 m/s to 30 m/s are shown in Figure 5.12. The time of completing this maneuver is considered as the minimum prediction time, which is summarized in Figure 5.13a. According to Equation (5.27), the minimum detection range is given by Figure 5.13b. As shown in the figure, the speed should be limited below 17 m/s when the LIDAR detection range is 100 m if this conservative bound is used. The third option is to let the

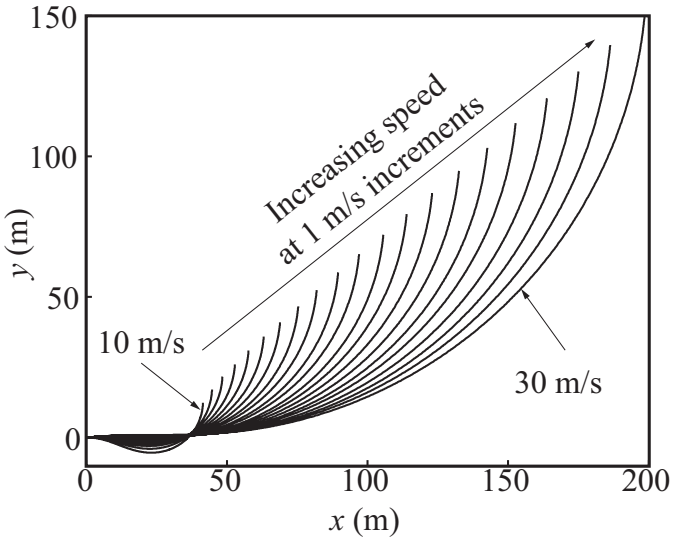
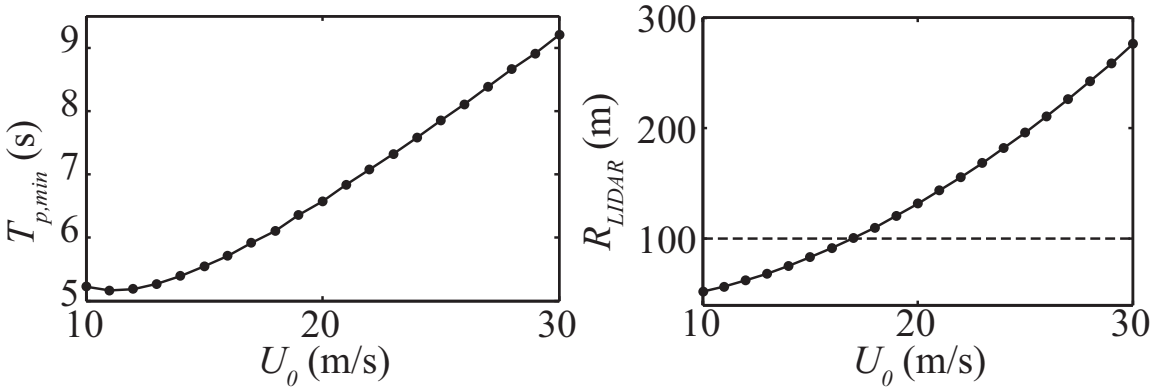


Figure 5.12: The trajectories of vehicle making a 90° turn at various speeds.



(a) Minimum prediction time.

(b) Minimum detection range

Figure 5.13: Limits on parameters to ensure early maneuver.

MPC algorithm control both the steering and vehicle speed. It is worth noting that the OCP formulation presented can be extended to include the vehicle longitudinal speed as another control variable in addition to steering angle, which is presented in Chapter 6.

In real systems, there may be several locations where time delays are introduced. Such delays can be lumped into two groups: the sensing delay and the control delay. These delays can have significant impact on the performance of the algorithm and hence need to be captured in the control loop. In a separate work, we have proposed methods to explicitly take delays into account to increase the robustness of the algorithm [2]. Specifically, the sensing delay is compensated by taking into account

Table 5.4: Five Sets of Weighting Parameters Used for Evaluation

	w_ψ	$w_d w_{\text{cf}}$	w_δ	w_{dt}
Set 1	1	10	0.1	10^{-4}
Set 2	10	10	0.1	10^{-4}
Set 3	1	100	0.1	10^{-4}
Set 4	1	10	1.0	10^{-4}
Set 5	1	10	0.1	10^{-3}

the differences of vehicle position and heading between the time when the LIDAR sensor is obtained and the time when the sensor data is used. The control delay is compensated by simulating a delayed control sequence, predicting the vehicle states after the delay with a vehicle model, and using the predicted states as initial states for planning.

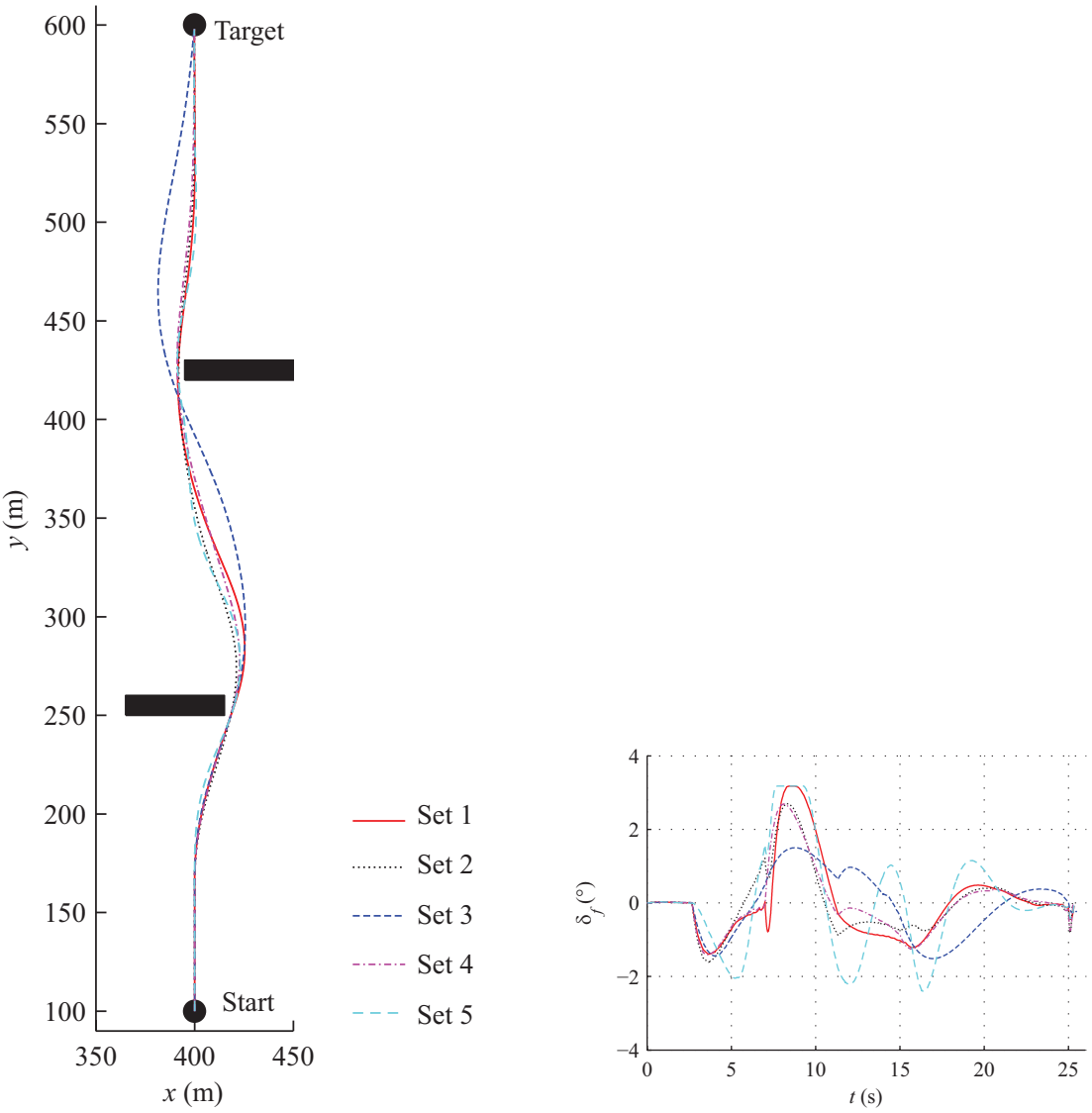
To generate a differentiable mathematical representation of the safe region from the LIDAR data, the region is first partitioned as described in Section 4.4. If the safe region could be used directly in the OCP formulation, the resulting optimal solution is expected to be the same as the optimal solution of the multi-phase OCP. Although a formal proof is not available, the fact that the same solution is obtained with different partitioning approaches could be a support for this argument. For example, the trajectories shown in Figure 4.6, Figure 4.7a, and Figure 4.7b are the same, although they are obtained using three different partitioning approaches.

There are four weighting parameters in the cost function. Note that all test cases presented in this chapter are run with the same set of weighting parameters, indicating that the algorithm can cover different test cases with the same set of weights. It is also interesting to point out how different choices of weights can affect the performance in a given test case. As an example, the test case 1 with vehicle speed at 20 m/s is ran with five sets of weighting parameters listed in Table 5.4.

As shown in Figure 5.14a and Figure 5.14b, all vehicle trajectories are collision free and all steering angle trajectories are within the maximum allowed bounds, which means that the vehicle is dynamically safe for all cases. Thus, the algorithm can be utilized with a wide range of weighting parameters.

Because all terms in the cost function have intuitive meanings, it is easy to adjust the value of the weighting parameters for different purposes. If the weight w_{cf} is increased, a smaller control effort is expected, which will result in turns with larger turning radius. If the weight w_{dt} is increased, the trajectory will tend to follow the line passing through the target location from the specified heading direction. Thus, sharper turns are expected. Simulation results in Figure 5.14 agree with these

expectations.



(a) Trajectory.

(b) Steering angle.

Figure 5.14: Simulation results to study the weighting parameters in the cost function.

5.5 Conclusion

This chapter presents a novel nonlinear MPC algorithm for obstacle avoidance in high-speed, large-size autonomous ground vehicles with high center of gravity that operate in unstructured environments, such as military vehicles. A multi-phase optimal control problem formulation is then adopted to accommodate the different position constraints introduced by each sub-region. The no-wheel-lift-off requirement,

which is the main dynamical safety concern for high-speed vehicles with a high CoG, is satisfied by limiting the steering angle within a range that is a function of vehicle speed. This range is obtained by simulating a fourteen DoF vehicle dynamics model offline. The cost function is formulated in a way to allow the vehicle reach the target position faster and, if needed, from a desired angle, and with minimal control effort. Simulation results show that the method can yield a satisfactory performance in a variety of test cases. The conclusion is that the method can enable a safe navigation of high-speed AGVs through an unstructured obstacle field, where safety is understood in terms of avoiding not only the obstacles, but also wheel lift-off.

CHAPTER 6

Variable Speed Formulation and Results

6.1 Introduction

The constant speed algorithm presented in Chapter 5 can achieve an optimal and smooth operation of AGVs at high speed through unstructured environments without collision while ensuring vehicle dynamical safety. However, the formulation assumes that the vehicle longitudinal speed is maintained constant, which can limit the mobility performance and the obstacle fields that can be cleared with this algorithm.

In this chapter, we extend the algorithm and develop a novel MPC formulation that simultaneously optimizes both the longitudinal speed and steering control commands for high speed obstacle avoidance taking into account dynamical safety. The novelty of the formulation includes

(1) A varying prediction horizon MPC is used to achieve a fixed distance prediction. This is prompted by two features of the proposed system. First, the terminal point of the planned trajectory is constrained at the LIDAR's maximum detection range in an effort to fully utilize as much information from the LIDAR as possible. Second, the variable vehicle speed necessarily leads to a variable prediction horizon with the previous constraint.

(2) The effects of powertrain and brake systems are taken into account through the relationship between acceleration and speed and the bounds on longitudinal jerk, acceleration, and speed. The vehicle's acceleration capability varies with the speed resulting from the powertrain and brake systems. To generate a speed

profile that can be tracked by the vehicle, the algorithm uses an offline generated look-up table to account for the acceleration and deceleration limitations.

(3) The no-wheel-lift-off requirement is considered through both hard and soft constraints using equations with empirical parameters that can predict tire vertical loads. A hard constraint bounds the vertical loads to be greater than a specified minimum threshold. A soft constraint is also used to provide a smooth approach to this threshold to prevent overshoot.

6.2 OCP Formulation

At each step of the MPC, one or more multi-phase OCPs are set up and solved to simultaneously optimize the steering angle and the reference speed profile. The formulation of the optimal control problem for obstacle avoidance in general form is given by:

$$
\begin{aligned}
&\underset{\boldsymbol{\xi},\,\boldsymbol{\zeta},\,T^1,\cdots,T^N}{\text{minimize}} && J = \mathcal{T}\left[\boldsymbol{\xi}^{(N)}\left(T^N\right), \boldsymbol{\xi}^{(1)}\left(T^0\right), T^N\right] \\
& && \quad + \sum_{i=1}^{N}\left\{\int_{T^{i-1}}^{T^i} \mathcal{I}^{(i)}\left[\boldsymbol{\xi}^{(i)}(t), \boldsymbol{\zeta}^{(i)}(t)\right] \mathrm{d}t\right\} && (6.1) \\
&\text{subject to} \quad \forall i=1,\cdots,N && \dot{\boldsymbol{\xi}}^{(i)}(t) = \boldsymbol{\mathcal{V}}\left[\boldsymbol{\xi}^{(i)}(t), \boldsymbol{\zeta}^{(i)}(t)\right] && (6.2) \\
& && \boldsymbol{\xi}^{(i)}_{\min}(t) \le \boldsymbol{\xi}^{(i)}(t) \le \boldsymbol{\xi}^{(i)}_{\max}(t) && (6.3) \\
& && \boldsymbol{\zeta}^{(i)}_{\min}(t) \le \boldsymbol{\zeta}^{(i)}(t) \le \boldsymbol{\zeta}^{(i)}_{\max}(t) && (6.4) \\
& && \boldsymbol{\mathcal{R}}^{(i)}\left[\boldsymbol{\xi}^{(i)}(t)\right] \le 0 && (6.5) \\
& && \boldsymbol{\mathcal{S}}^{(i)}\left[\boldsymbol{\xi}^{(i)}(t)\right] \le 0 && (6.6) \\
& && \boldsymbol{\xi}^{(i)}\left(T^{i-1}\right) = \boldsymbol{\xi}^{(i-1)}\left(T^{i-1}\right) && (6.7) \\
& && t \in \left[T^{i-1}, T^i\right], T^{i-1} < T^i && (6.8) \\
&\text{subject to} && \boldsymbol{\mathcal{F}}\left[\boldsymbol{\xi}^{(N)}\left(T^N\right), \boldsymbol{\xi}^{(1)}\left(T^0\right)\right] \le 0 && (6.9) \\
& && T^0 = 0,\ T_{p,\min} \le T^N = T_p \le T_{p,\max} && (6.10)
\end{aligned}
$$

By minimizing the cost function specified in Equation (6.1), subject to constraints defined in Equation (6.2) - Equation (6.8) for all phases, and constraints defined in Equation (6.9) - Equation (6.10) for the initial and end points of the prediction, the optimal state trajectories $\boldsymbol{\xi}^{*(i)}(t), t \in [T^{i-1}, T^i]$, the optimal control trajectories

$\boldsymbol{\zeta}^{*(i)}(t), t \in [T^{i-1}, T^i]$, and the instants of time T^i, $i = 1, \ldots, N$, for transitioning from one sub-region to the next are obtained, where N is the total number of phases; i.e., the number of sub-regions in a given sequence. The following sub-sections define the variables and explain the problem formulation in detail. The constraints are discussed first, and the cost function formulation is explained next.

6.2.1 Equation (6.2): Vehicle Dynamics Model

In the MPC framework, a model of the AGV is included in the OCP formulation explicitly to predict its behavior over the prediction horizon. In particular, a three DoF single track vehicle model is used with the longitudinal load transfer and tire nonlinearities taken into account. The state space equations can be written in the following form:

$$\dot{\boldsymbol{\xi}} = \boldsymbol{f}_{\text{3DoF}}(\boldsymbol{\xi}) + \boldsymbol{B}_{\text{3DoF}}\boldsymbol{\zeta} \tag{6.11}$$

with

$$\boldsymbol{f}_{\text{3DoF}}(\boldsymbol{\xi}) = \begin{bmatrix} U\cos\psi - (V + L_f\omega_z)\sin\psi \\ U\sin\psi + (V + L_f\omega_z)\cos\psi \\ \omega_z \\ a_x \\ (F_{y,f} + F_{y,r})/M_t - U\omega_z \\ (F_{y,f}L_f - F_{y,r}L_r)/I_{zz} \\ 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{B}^T_{\text{3DoF}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where the state vector is $\boldsymbol{\xi}^{\boldsymbol{T}} = [\ x \quad y \quad \psi \quad U \quad V \quad \omega_z \quad \delta_f \quad a_x\ ]$ and the control vector is $\boldsymbol{\zeta}^{\boldsymbol{T}} = [\ \gamma_f \quad J_x\ ]$.

6.2.2 Equation (6.3) and (6.4): State and Control Bounds

The steering rate γ_f is used as the control command to be optimized and the steering angle δ_f is set as an additional state variable of the system. The reason for choosing the steering rate as the control input instead of the steering angle is to obtain a smooth steering angle sequence and to facilitate implementing a limit on the steering rate. The same argument applies to the longitudinal jerk and speed. Although the control commands to be optimized are steering rate and longitudinal jerk, the outputs

of the algorithm are still the steering angle and the longitudinal speed.

Due to the mechanical limits of steering and actuator limits, constant bounds are imposed on the two variables δ_f and γ_f as follows.

$$\delta_{f,\min} \leq \delta_f(t) \leq \delta_{f,\max} \tag{6.12}$$

$$\gamma_{f,\min} \leq \gamma_f(t) \leq \gamma_{f,\max} \tag{6.13}$$

In addition, the longitudinal vehicle speed, acceleration, jerk values are bounded in accordance with the powertrain and brake dynamics of the real system.

$$U_{\min} \leq U(t) \leq U_{\max} \tag{6.14}$$

$$a_{x,\min}[U(t)] \leq a_x(t) \leq a_{x,\max}[U(t)] \tag{6.15}$$

$$J_{x,\min} \leq J_x(t) \leq J_{x,\max} \tag{6.16}$$

Figure 6.1a and Figure 6.1b are the longitudinal speed and acceleration profiles, respectively. These are the simulation results obtained with the plant model that considers the powertrain and brake dynamics. First, a full throttle command is applied until the vehicle reaches and maintains the maximum speed. Then, a full braking command is applied until the vehicle stops. Thus, Figure 6.1b shows the vehicle acceleration capability.

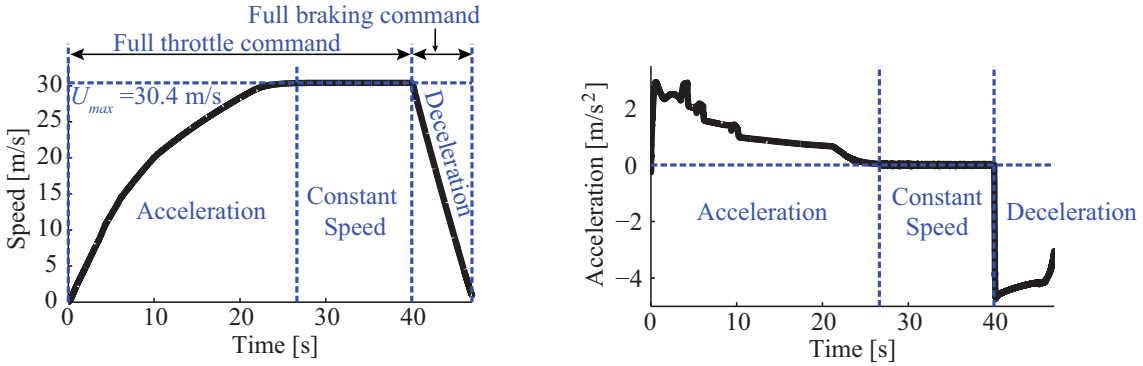
As shown in Figure 6.2, the acceleration capability depends on the instantaneous speed. The upper bound and lower bound that are used in the MPC are approximated using fourth order polynomials as follows:

$$a_{x,\max}(U) = c_1U^3 + c_2U^2 + c_3U + c_4 \tag{6.17}$$

$$a_{x,\min}(U) = c_5U^3 + c_6U^2 + c_7U + c_8 \tag{6.18}$$

where $c_1, \cdots, c_8$ are parameters obtained from polynomial fitting to the simulation data.

Constant upper and lower bounds are imposed on the longitudinal jerk to achieve a smooth acceleration profile. No bounds on lateral speed and yaw rate are included explicitly because the closed-form representations of the bounds as a function of the steering angle and the longitudinal speed could not be obtained. Thus, the implicit limits are directly included as constraints to ensure dynamical safety as discussed in the next Section.



(a) Longitudinal speed.

(b) Longitudinal acceleration.

Figure 6.1: Vehicle longitudinal speed and acceleration profiles of the plant model for a full throttle - full brake cycle.

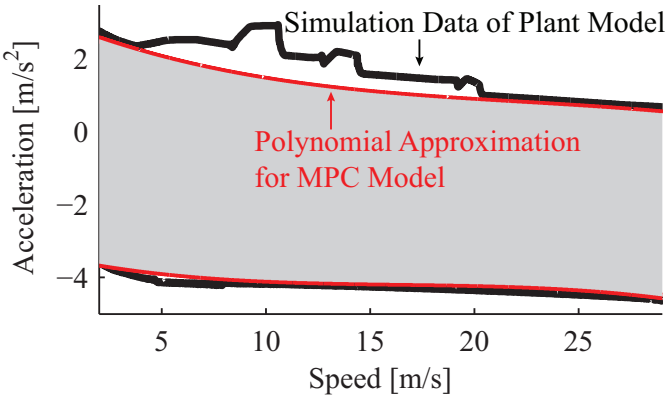


Figure 6.2: Vehicle acceleration limits as a function of vehicle longitudinal speed from the plant model simulation.

6.2.3 Equation (6.5): Safe Region Constraints

Avoiding collision with obstacles is enforced through position constraints; the vehicle trajectory must stay within the safe region that is obtained after the LIDAR data is processed. For each phase in the OCP, a set of position constraints is specified by Equation 4.2, which can be abbreviated into the following form.

$$\boldsymbol{\mathcal{R}}^{(i)}\left(x^{(i)}, y^{(i)}\right) \leq 0 \tag{6.19}$$

6.2.4 Equation (6.6): Dynamical Safety Constraints

In this study, ensuring the vehicle's dynamical safety is defined as avoiding single-wheel lift-off. This is a conservative criterion used to prevent rollover [27]. Prior work enforced this constraint through steering angle bounds [2, 52], or lateral acceleration bounds [113]. However, these approaches ignore the effect of longitudinal acceleration, which is an important factor to consider in the variable speed case. Hence, in this work, the no-wheel-lift-off requirement is taken into account directly by constraining the load on all four tires to be always positive.

The following equations are used to predict the tire vertical loads taking into account the vehicle's lateral load transfers [98].

$$F_{z,fl} = \tfrac{1}{2}\left(F_{z,f0} - \Delta F_{z,x}\right) - \Delta F_{z,yf} \tag{6.20}$$

$$F_{z,fr} = \tfrac{1}{2}\left(F_{z,f0} - \Delta F_{z,x}\right) + \Delta F_{z,yf} \tag{6.21}$$

$$F_{z,rl} = \tfrac{1}{2}\left(F_{z,r0} + \Delta F_{z,x}\right) - \Delta F_{z,yr} \tag{6.22}$$

$$F_{z,rr} = \tfrac{1}{2}\left(F_{z,r0} + \Delta F_{z,x}\right) + \Delta F_{z,yr} \tag{6.23}$$

where $\Delta F_{z,x}$ is the longitudinal load transfer; $\Delta F_{z,yf}$ is the front axle lateral load transfer; and $\Delta F_{z,yr}$ is the rear axle lateral load transfer.

These load transfers are approximated by the following relationships

$$\Delta F_{z,x} \approx K_{z,x}\left(\dot{U} - V\omega_z\right) \tag{6.24}$$

$$\Delta F_{z,yf} \approx K_{z,yf}\left(\dot{V} + U\omega_z\right) \tag{6.25}$$

$$\Delta F_{z,yr} \approx K_{z,yr}\left(\dot{V} + U\omega_z\right) \tag{6.26}$$

where $K_{z,x}$ is defined as the longitudinal load transfer coefficient; and $K_{z,yf}$ and $K_{z,yr}$ are defined as the front and rear lateral load transfer coefficients, respectively. These coefficients are obtained from several sets of simulations with the plant model.

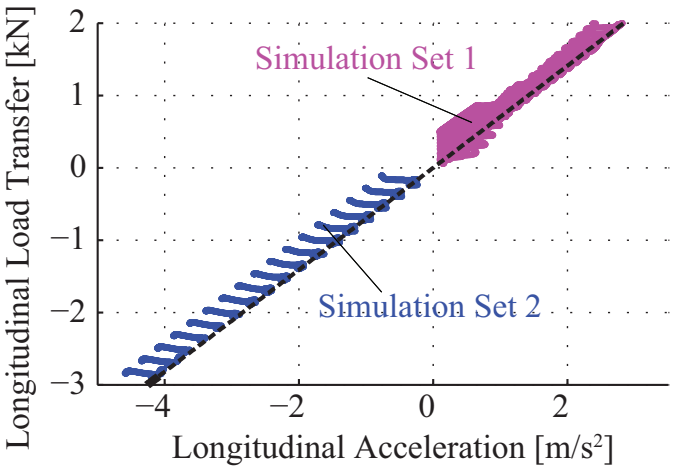


Figure 6.3: Longitudinal load transfer as a function of longitudinal acceleration of the plant model. A linear model is used to fit the obtained data.

Figure 6.3 and Figure 6.4 show the simulation results for the longitudinal and lateral load transfers with the plant model and the straight line fittings, whose slopes represent the constant load transfer coefficients.

To obtain the longitudinal load transfer coefficient, two sets of simulations are conducted. In the first set of simulations, the throttle command is maintained constant at different levels with zero braking command and zero steering angle. The results are used to study the longitudinal load transfer during vehicle acceleration. In the second set of simulations, the braking command is maintained constant at different levels with zero throttle command and zero steering angle. The results can then be used to study the longitudinal load transfer during vehicle deceleration. The black dashed line in Figure 6.3 is a line that passes through the origin with a slope estimated from these two sets of data. The deviation from this line to the upper left side is caused by aerodynamic drag.

The data for estimating the lateral load transfer coefficients is generated by following the maneuver specified by Figure 6.5a and Figure 6.5b. The vehicle is steered with a sinusoidal steering angle profile. In the meantime, the speed is changing at a slower frequency. Figure 6.4 shows the simulation data points and the fitted lines. Simulations with other maneuvers generated similar results, thus showing these results are independent of the maneuvers tested.

Even though the results from the simulations are not perfectly affine, the approximations in Equation (6.20) - (6.23) are considered sufficient for the purposes of this work, because the error introduced by these estimated coefficients is on the order of 5% as the comparative simulation results in Figure 6.5 illustrate.

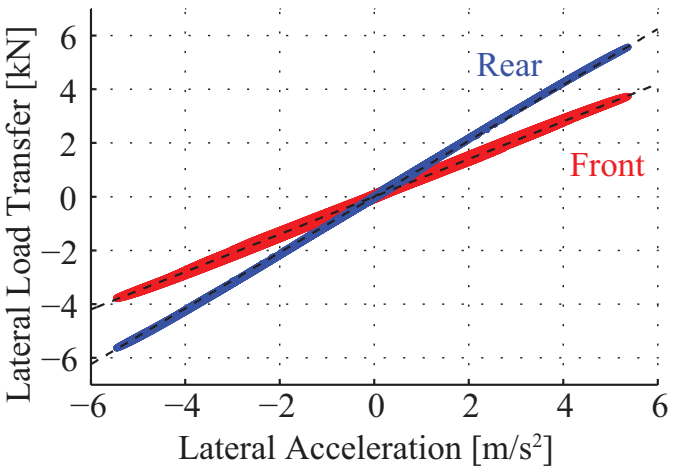
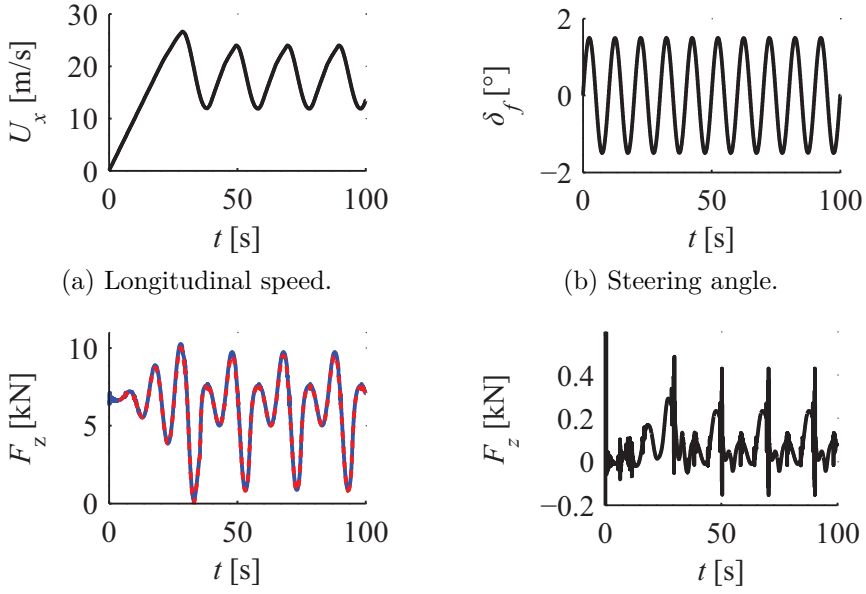


Figure 6.4: Front and rear lateral load transfers as functions of lateral acceleration of the plant model, respectively. Linear models are used to fit the obtained data.



(a) Longitudinal speed.

(b) Steering angle.

(c) Comparison of the actual and the predicted vertical loads.

(d) Difference of the actual and the predicted vertical loads.

Figure 6.5: A sample open-loop simulation used to validate the accuracy of vertical load prediction. The vehicle performs the maneuver specified by subplots (a) and (b). The blue solid line in subplot (c) is the vertical load of rear left tire of the plant model. The red dashed line is the prediction using Equation (6.22) with state values from the plant model simulation. The two lines overlap with each other and are indistinguishable, indicating that the vertical load prediction is sufficiently accurate.

With Equation (6.20) - (6.23), the no-wheel-lift-off requirement is taken into account directly by constraining the load on all four tires to be always positive. For the particular set of vehicle parameters considered, $F_{z,f0} > F_{z,r0}$, and $K_{z,yr} > K_{z,yf}$, vertical loads of the two tires on the back are constrained to be greater than a positive threshold value. In Figure 6.6, when the vehicle accelerations are within the entire shaded region, the vertical loads of the two front tires will be greater than $F_{z,\text{threshold}} = 1000$ N. However, to make sure that the vertical loads of the two rear tires are greater than this threshold, the vehicle accelerations need to stay within the light gray region. Thus, the range of vehicle accelerations for limiting rear tire vertical loads is smaller than the range for limiting front tire vertical loads. Thus, it is sufficient to include only the vertical load limits of the two rear tires.

$$\tfrac{1}{2}\left(F_{z,r0} + \Delta F_{z,x}\right) \pm \Delta F_{z,yr} \geq F_{z,\text{threshold}} \tag{6.27}$$

where $F_{z,\text{threshold}}$ is a positive minimum vertical load constant.

By substituting Equation (6.24) and Equation (6.26) into the above equation and substituting the dynamical equations for calculating $\dot{U}$ and $\dot{V}$ and rearranging the terms, the following inequalities as functions of the vehicle state variables are obtained.

$$\begin{aligned} &M_t\left(2F_{z,\text{threshold}} - F_{z,ro}\right) - K_{z,x}M_t\left(a_x - V\omega_z\right) \pm \\ &2K_{z,yr}F_{y,f}\left(U, V, \omega_z, \delta_f, a_x\right) + \\ &2K_{z,yr}F_{y,r}\left(U, V, \omega_z, \delta_f, a_x\right) \leq 0 \end{aligned} \tag{6.28}$$

They can be abbreviated into the following form.

$$\boldsymbol{\mathcal{S}}\left(U, V, \omega_z, \delta_f, a_x\right) \leq 0 \tag{6.29}$$

6.2.5 Equation (6.7) and (6.8): Continuity Constraints

The initial states of each phase are set to be the same as the final states of the previous phase as constrained by Equation (6.7).

Equation (6.8) specifies the requirement that time increases monotonically as the vehicle moves through the sub-regions. This enforces that the vehicle moves to the terminal region.

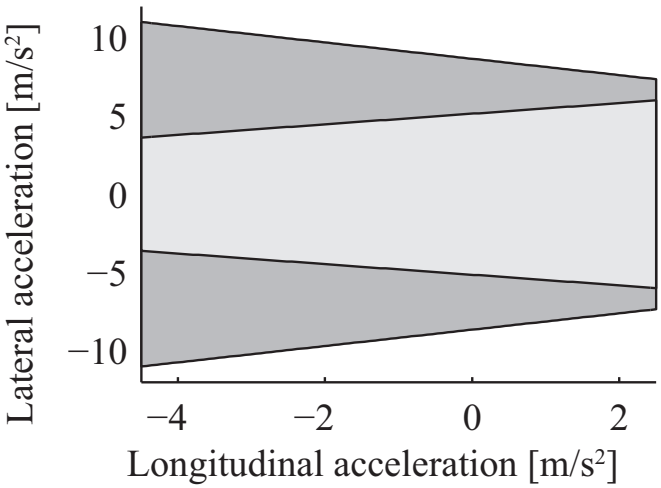


Figure 6.6: The range of vehicle accelerations for limiting rear tire vertical loads (light gray region) is smaller than the range for limiting front tire vertical loads (entire shaded region). As a result, it is sufficient to consider only the rear tire vertical loads in the formulation of the constraints.

6.2.6 Equation (6.9) and (6.10): Terminal Constraints

The terminal constraints are functions of only the initial states at T^0, and the final states at the end of the prediction horizon, T_p.

Three terminal constraints are included. The first terminal constraint specifies the initial states. For the first phase, the initial values of the states x, y, ψ, V, and ω_z are from the measurements. To maintain a smooth reference speed and a smooth steering sequence, the initial values of U, a_x, and δ_f are the values from the end of the execution horizon of the previous step.

$$\boldsymbol{\xi}^{(0)}\left(T^0\right) = \boldsymbol{\xi}^0 \tag{6.30}$$

The second terminal constraint limits the vehicle speed at the end of the prediction horizon. This limit is introduced to prepare for potential obstacles in the future. Specifically, the vehicle is allowed to accelerate at the beginning of the prediction horizon, but then is required to decelerate to a threshold speed at the end of the prediction horizon, since no obstacle information such as location, shape, and size are known *a priori*.

$$U^N\left(T^N\right) \leq U_{\text{threshold}} \tag{6.31}$$

The third terminal constraint ensures that the end of the vehicle trajectory is close to the LIDAR detection range. This constraint is active only when the target

is outside the LIDAR's detection range.

$$R_{\text{LIDAR}} - R_\sigma \leq \sqrt{\begin{array}{l}\left[x^N\left(T^N\right) - x^1\left(T^0\right)\right]^2 \\ + \left[y^N\left(T^N\right) - y^1\left(T^0\right)\right]^2\end{array}} \leq R_{\text{LIDAR}} \tag{6.32}$$

where R_σ is a relaxation constant.

These three constraints can be lumped in the form of Equation (6.9).

Finally, the initial time T_0 is specified as 0. Equation (6.10) specifies the limits on the prediction horizon. Because the longitudinal speed of the vehicle is to be optimized and can be varying and the third terminal constraint limits the distance between the initial position and the end position, the prediction horizon is not a constant parameter in terms of time. Instead, it is a design variable to be optimized. The upper bound, $T_{p,\max}$, and lower bound, $T_{p,\min}$ relate to the $U_{\max}$, and $U_{\min}$, respectively.

6.2.7 Equation (6.1): Cost Function

The cost function defines in what sense a trajectory is considered to be optimal. It consists of two parts: terminal cost and integral cost. In this work, the cost function formulation includes six terms, three terminal cost terms and three integral cost terms, that are linearly combined using relative weights as follows:

$$\begin{aligned} J = & \frac{s_f}{s_0} + w_\psi \psi_{\text{diff}}^2 + w_t T_p + \\ & w_{\text{dt}} \int_0^{T_p} \left[\sin\left(\psi_t\right)\left(x - x_t\right) - \cos\left(\psi_t\right)\left(y - y_t\right)\right]^2 \mathrm{d}t + \\ & w_{\text{fz}} \int_0^{T_p} \left[\tanh\left(-\frac{F_{z,rl} - F_{z,a}}{F_{z,b}}\right) + \tanh\left(-\frac{F_{z,rr} - F_{z,a}}{F_{z,b}}\right)\right] \mathrm{d}t + \\ & w_{\text{cf}} \int_0^{T_p} \left[w_\delta \delta_f^2 + w_\gamma \gamma_f^2 + w_J J_x^2\right] \mathrm{d}t \end{aligned} \tag{6.33}$$

where s_0 is given by Equation 5.13, s_f is given by Equation 5.14, ψ_{diff} is given by Equation 5.16, and T_p is the prediction horizon. If proper weighting factors are selected, the first three terms will result in a trajectory in which the end point of the predicted trajectory is close to the target, the final heading angle is pointing to the target, and the time used to cover the prediction distance is small.

Furthermore, three integral terms are included in the cost function.

The first term is used to minimize the integral over the prediction horizon of the square of distance to the line that is passing through the target $[x_t, y_t]$ along a desired

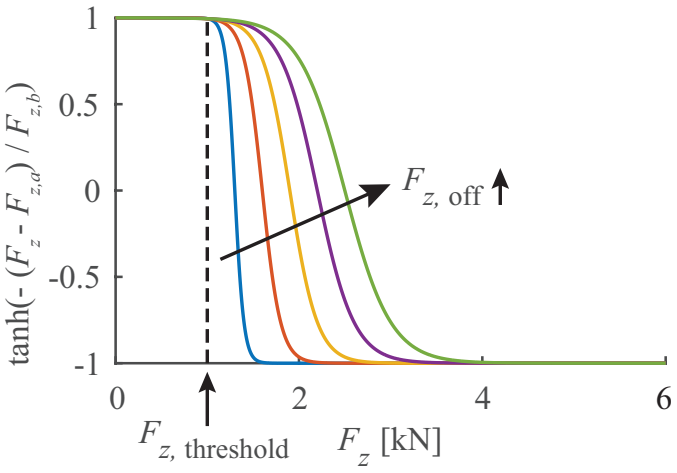


Figure 6.7: The effect of increasing $F_{z,\text{off}}$ in the soft constraint for no-wheel-lift-off requirement.

direction ϕ_t. This term is used to have the vehicle pass through the target from the desired direction. The term is included when there is a desired heading angle to pass the target.

The second term penalizes the cost when the tire vertical load is close to the specified threshold, $F_{z,\text{threshold}}$, and is used to provide a smooth approach to this threshold and prevent vehicle from operating at the limit unnecessarily. This is a soft constraint to ensure vehicle dynamical safety and is in addition to the hard constraints specified in Equation (6.6). The parameters $F_{z,a}$ and $F_{z,b}$ used in the definition are given by

$$\begin{aligned} F_{z,a} &= F_{z,\text{threshold}} + 3F_{z,\text{off}} \\ F_{z,b} &= F_{z,\text{off}} \end{aligned} \tag{6.34}$$

where $F_{z,\text{off}}$ relates to the transition of cost value as shown in Figure 6.7.

The third term is a regularization term minimizing the control effort that is defined as the integral of the weighted sum of δ_f^2, γ_f^2, and J_x^2.

The variables $w_\psi, w_t, w_{\text{dt}}, w_{\text{fz}}$, w_{cf}, w_δ, w_γ, and w_J are corresponding weighting parameters.

When the target position is within the sensor's detection range, the terms s_f/s_0 and $w_\psi \psi_{\text{diff}}^2$ are removed from the cost functions. Instead, Equation 5.21 is added to the OCP formulation.

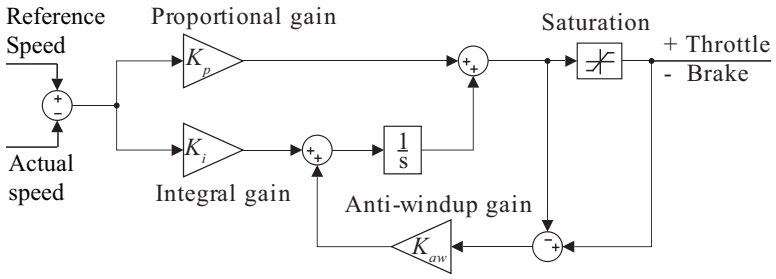


Figure 6.8: Vehicle longitudinal speed controller.

6.3 Simulation Results

This section presents numerical simulations of the developed nonlinear MPC obstacle avoidance algorithm with the fourteen DoF model as the plant. The generated reference speed profile from the MPC is tracked using a speed controller, which is a PI controller with saturation and anti-windup as shown in Figure 6.8. It takes the difference between the reference speed and actual vehicle speed as input and generates an output within the interval $[-1,\ 1]$, with positive values corresponding to throttle command and negative values corresponding to the brake command.

Three test cases are considered with three different obstacle fields. The first two test cases are simulated with both the constant speed formulation and variable speed formulation for comparison. These simulations are run with a vehicle initial speed of 20 m/s. These two sets of simulations are used to show the advantage of the combined steering and longitudinal speed optimization based MPC over the constant speed formulation. The third test case is used to show that the algorithm is capable of navigating the vehicle through a complex obstacle field, where complexity is understood in terms of obstacle density and variations in obstacle size. This simulation is run with a vehicle initial speed of 15 m/s.

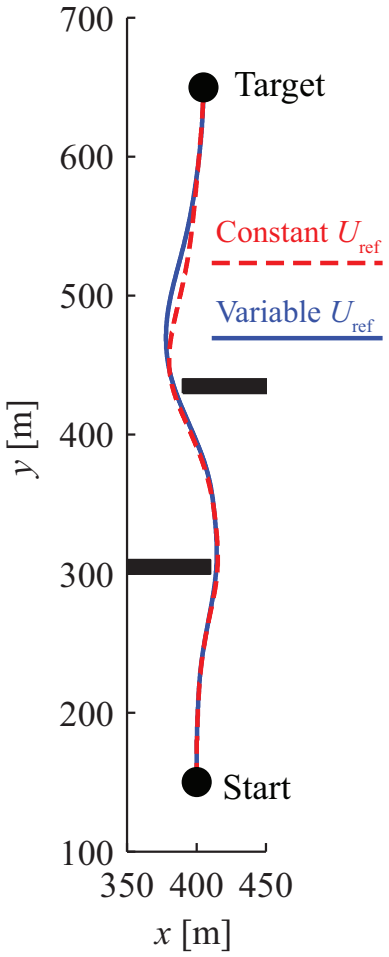
The LIDAR detection range is $R_{\text{LIDAR}} = 100$ m. The goal is to pass through the specified target from the 90 degrees direction in the global coordinate. At each step of the MPC, the optimization generates a trajectory within the 100 m LIDAR detection range. Only the first 0.5 s of the planned control command is executed by the vehicle. A new trajectory is planned every 0.5 s using the updated vehicle state information and obstacle information from the sensors. Note that the command within the 0.5 s is not constant; the sample rate for the control commands is 0.05 s. Table 6.1 summarizes the value of the parameters used in the simulation.

Table 6.1: Simulation Parameters

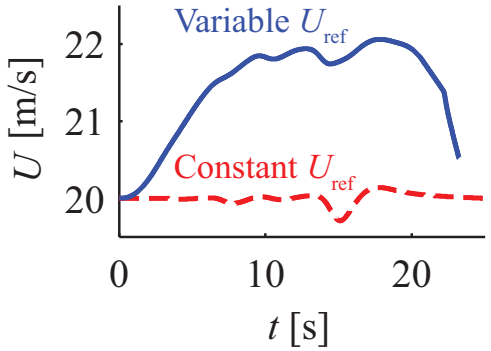
Symbol	Value	Unit
M_t	2689	kg
I_{zz}	4110	kg-m^2
L_f, L_r	1.58, 1.72	m
$K_{z,x}$	806	N/(m/s^2)
$K_{z,yf}$	675	N/(m/s^2)
$K_{z,yr}$	1076	N/(m/s^2)
$F_{z,\text{threshold}}$	1000	N
$[\delta_{f,\min}, \delta_{f,\max}]$	[-30, 30]	$^\circ$
$[\varsigma_{f,\min}, \varsigma_{f,\max}]$	[-5, 5]	$^\circ$/s
$[U_{\min}, U_{\max}]$	[5, 29]	m/s
$[J_{x,\min}, J_{x,\max}]$	[-5, 5]	m/s^3
c_1, c_2, c_3, c_4	-1.28e-4, 8.59e-3, -0.2257, 3.0828	-
c_5, c_6, c_7, c_8	-1.38e-4, 6.85e-3, -0.1204, -3.5589	-
R_{LIDAR}	100	m
R_σ	5	m
w_ψ, w_t	0.01, 0.05	-
$w_{\text{dt}}, w_{\text{fz}}, w_{\text{cf}}$	1e-5, 0.5, 1	-
w_δ, w_γ, w_J	0.1, 1, 0.01	-
$F_{z,a}, F_{z,b}$	1300, 100	N

6.3.1 Test Case 1: Less Challenging Case

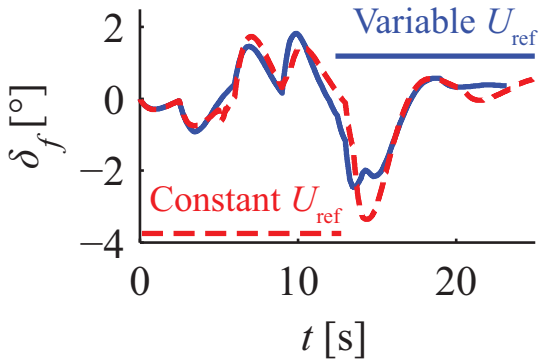
The results of the first test case are shown in Fig. 6.9. In this case, both the constant speed algorithm and variable speed algorithm navigate the vehicle through the obstacle field safely; i.e., collision-free as shown in Fig. 6.9a and without wheel lift-off as shown in Fig. 6.9d and Fig. 6.9e. In the constant speed case, because the speed controller used does not take the steering input into account, the speed has a maximum deviation of 0.3 m/s from the desired value of 20 m/s. In the variable speed case, the speed of the vehicle gradually increases from 20 m/s up to 22 m/s and decreases back towards 20 m/s within the final 3.5 s as desired. In this case, the vehicle arrives the target 1.7 s earlier out of the 25 s trajectory; i.e., about 7% faster. Hence, even though enforcing the speed to be constant can greatly simplify the control problem, the variable speed algorithm can better take advantage of the mobility capability of the AGV and avoid an unnecessarily conservative operation.



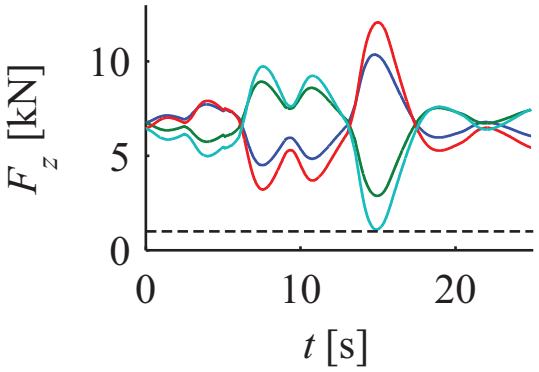
(a) Actual trajectory.



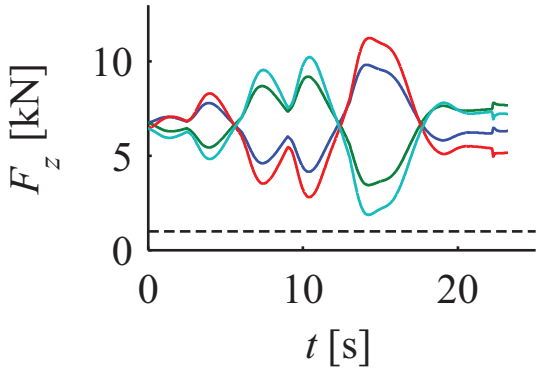
(b) Actual longitudinal speed.



(c) Steering angle.



(d) Tire vertical loads for the constant speed case.



(e) Tire vertical loads for the variable speed case.

Figure 6.9: Simulation results for test case 1. Both constant speed and variable speed navigations are successful; however, in the variable speed case, the vehicle arrives the target earlier. In the subplots (d) and (e), the black dashed lines indicate the minimum allowable vertical tire force.

6.3.2 Test Case 2: More Challenging Case

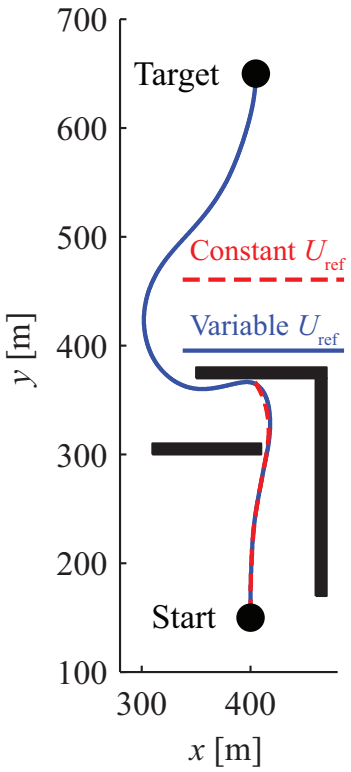
Not having a systematic way of determining the vehicle speed in the constant speed algorithm not only is a concern for causing an overly conservative vehicle operation, but, more importantly, can inadvertently cause safety problems as the second test case illustrates. The test case as shown in Fig. 6.10 is chosen to make it necessary to slow down to avoid the obstacle. Using the constant speed algorithm, the vehicle cannot avoid the obstacles safely as the simulation result confirms. With the constant speed algorithm, the vehicle collides with the obstacle after 11.5 s. However, with the variable speed algorithm, the vehicle is safely navigated through the obstacle field by decelerating to a speed around 10 m/s. The vehicle starts accelerating again after the obstacle is cleared. As shown in Fig. 6.10a and Fig. 6.10d, the trajectory is collision free. Fig. 6.10f shows that the tire vertical loads are all above the specified minimum threshold; hence, dynamical safety of the vehicle is ensured, as well. Fig. 6.11 shows the difference between the reference speed profile and the actual speed profile, which demonstrates the performance of the speed controller. The difference is relatively larger when the steering angle is large. Finally, Fig. 6.12 shows that the prediction time is varying at different steps.

6.3.3 Test Case 3: Dense Obstacle Field

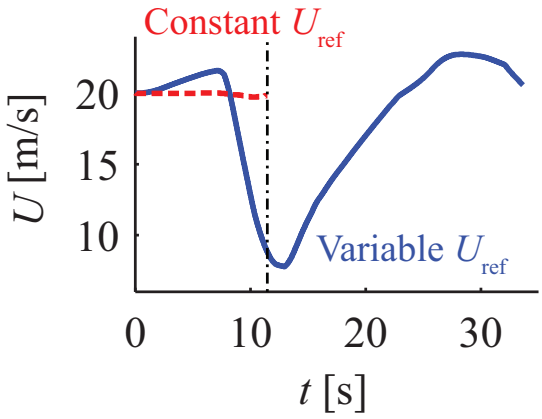
The third test case is the vehicle moving in an obstacle field with 40 obstacles of different sizes. Even though the vehicle speed is initialized at 15 m/s, the algorithm can recognize that the vehicle is actually capable of navigating through this complex obstacle field at a higher speed. Thus, it accelerates the vehicle to about 20 m/s while it navigates the vehicle through the field before it decelerates again to the desired final speed as it approaches the target position. The maximum speed is constrained by the acceleration capability and the sensor detection range.

6.4 Conclusion

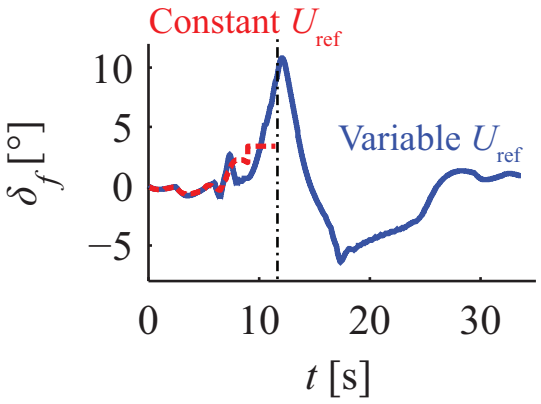
This chapter considers high-speed AGVs in unstructured environments without *a priori* information about the obstacles and presents a new MPC-based obstacle avoidance algorithm that optimizes the longitudinal speed and steering angle simultaneously to navigate the AGV safely and as quickly as possible to the target location. To this end, the algorithm is capable of exploiting the dynamic limits of the vehicle to maximize the vehicle's mobility performance. A multi-stage OCP formulation is used to incor-



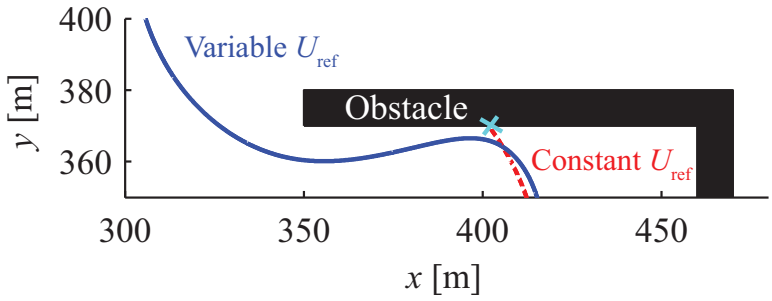
(a) Actual trajectory.



(b) Actual longitudinal speed.

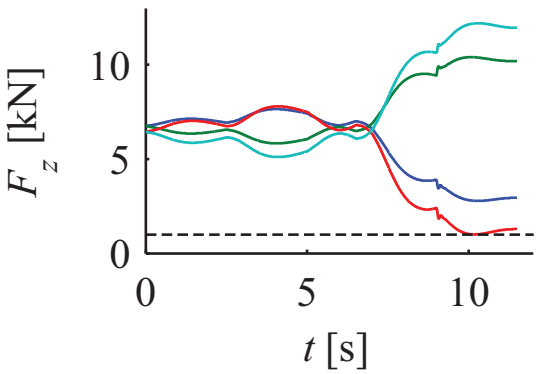


(c) Steering angle.

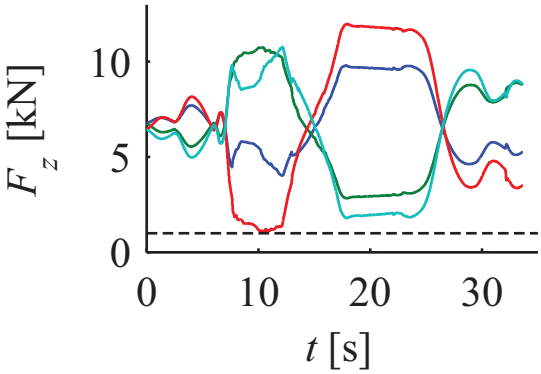


(d) Trajectory zoom in view.

Figure 6.10: Simulation results for test case 2. The constant speed navigation fails while the variable speed navigation is successful.

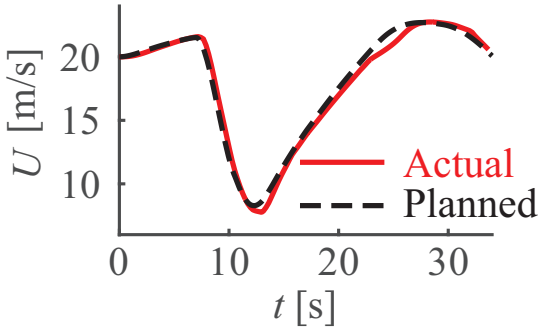


(e) Tire vertical loads for the constant speed case.

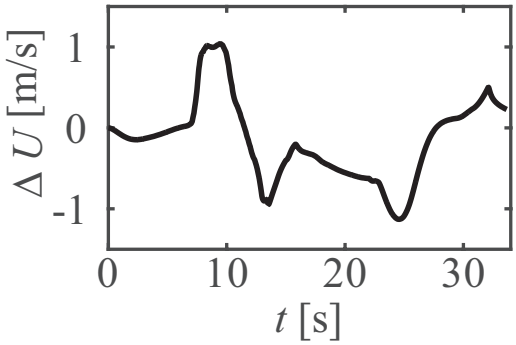


(f) Tire Vertical loads for the variable speed case.

Figure 6.10: (continued) Simulation results for test case 2. The constant speed navigation fails while the variable speed navigation is successful.



(a) Comparison of the reference and the actual speed profiles.



(b) Difference of the reference and the actual speed profiles.

Figure 6.11: Longitudinal speed in test case 2.

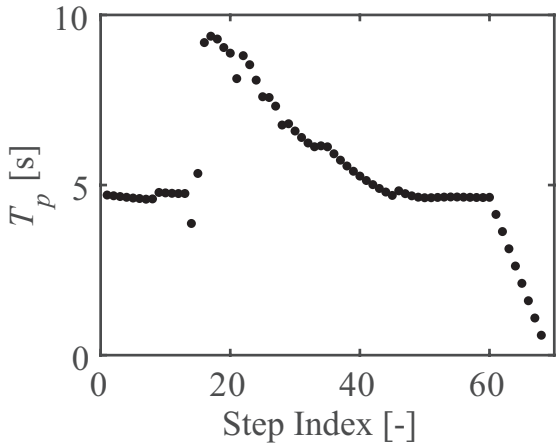
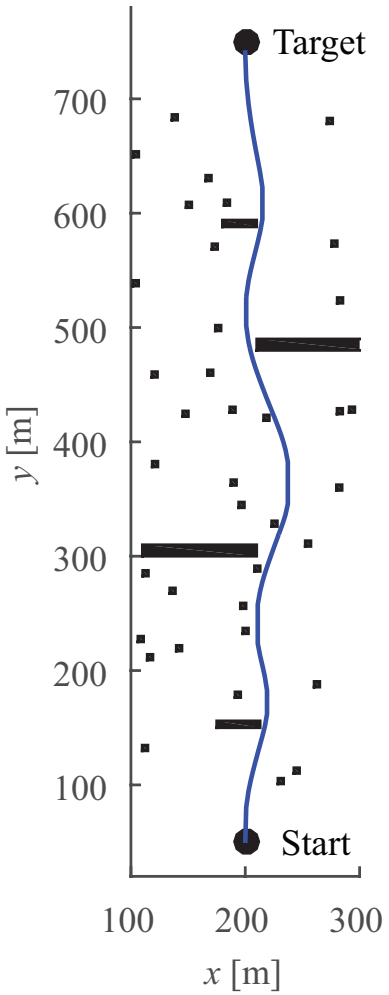
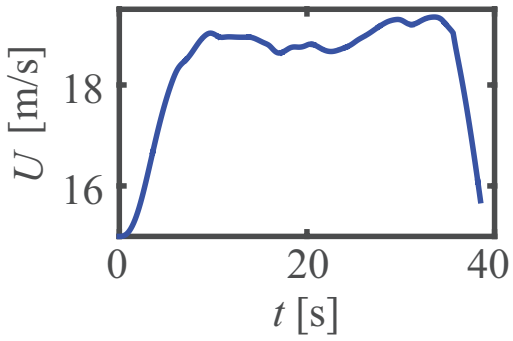


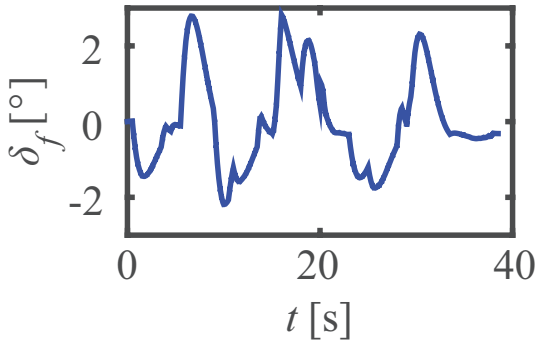
Figure 6.12: Varying prediction horizon in test case 2.



(a) Actual trajectory.



(b) Actual longitudinal speed.



(c) Steering angle.

Figure 6.13: Simulation results for test case 3. The algorithm is capable of navigating the vehicle through a complex obstacle field while increasing the speed to exploit the vehicle's mobility capability.

porate the obstacle data obtained from the on-board LIDAR sensor. The time length of the prediction horizon of the MPC is varying because a variable-speed trajectory is planned till the end of the sensor range. The powertrain and brake dynamics are taken into consideration through the bounds on the vehicle longitudinal speed, acceleration and jerk. The dynamical safety requirement is accounted for by enforcing a positive vertical load on all four tires as hard constraints. One term in the cost function also aims to provide a smooth approaching to the vertical load threshold. Three sets of numerical simulations are conducted to demonstrate the effectiveness of the algorithm. The conclusion is that the newly developed algorithm with variable speed and steering commands not only improves the performance of the vehicle by allowing it to operate closer to its dynamical limits, but also enables the safe clearance of obstacle fields that may not be cleared with steering control alone.

CHAPTER 7

Robust Formulation against Parametric Uncertainty and Results

7.1 Introduction

The goal of this chapter is to study the effect of parametric uncertainty on the obstacle avoidance algorithm and improve the robustness of the algorithm presented in Chapter 6. The parameter values used in previous simulations were assumed to be accurate, but in reality they are uncertain. This uncertainty can affect the effectiveness and task completion performance of the obstacle avoidance algorithm. To be more specific, parametric uncertainty denotes that the value of a parameter is constant but unknown. However, the probability distribution or bounds of the parameter value are assumed to be known *a priori* [114]. Robustness of this obstacle avoidance algorithm to parametric uncertainty and how it can be improved are research questions that have not yet been addressed in the literature and are thus the focus of this chapter.

The literature presents many methods to improve the robustness of an MPC algorithm by accounting for the given distributions of uncertainty within robust or stochastic MPC formulations [115, 116, 117], which have been applied in the context of automated driving and vehicle active safety [118]. There exist algorithms to handle general additive uncertainty, which can be applied to parametric uncertainty by approximating it as an additive uncertainty. Those algorithms are classified into the following three categories: i) open-loop min-max optimization based approach, where the satisfaction of the constraints for all possible uncertainty realizations are consid-

ered, e.g. [119]; ii) auxiliary-controller based approach, where a robustifying feedback control action is used to ensure that the actual state trajectories would not deviate from the feedforward trajectories beyond a specified bound, e.g. [120, 121]; and iii) scenario-based approach, where chance constraints are transformed into deterministic counterparts by evaluating them at a large number of uncertainty samples, e.g. [122]. In the latter approach, a scenario denotes a particular realization or sample of the uncertainty distributions. The same terminology is adopted throughout this chapter, as well.

Most of the algorithms from the first two categories, which are analytical robust or stochastic MPC algorithms, are restricted to linear systems and Gaussian distributions for the propagation of uncertain system states over the prediction horizon and/or the calculation of invariant tubes and auxiliary controller gains. However, Chapter 3 has concluded that model nonlinearity is essential for the application of interest. Although iterative online linearization of the model nonlinearity is possible [123], the linearized model is a good approximation only in a small region around the reference state and input vectors about which the linearization is performed. For obstacle avoidance in an unstructured environment, finding an optimal path is also part of the task besides avoiding obstacles, because no reference trajectory exists. This requires a long prediction horizon to achieve successful navigation in directions that may deviate significantly from the “current" direction. Thus, the assumption for the use of linearization may not hold.

In contrast to the first two categories, the scenario-based approach can be applied to any kind of model and uncertainty distribution, as long as a sufficient number of random samples are evaluated. However, the major drawback is that the problem size grows exponentially with the number of uncertain parameters and range of values that these parameters can take [124].

Besides the literature that considers the general additive uncertainty, some research efforts focus specifically on parametric uncertainty. For example, Walton [114] considers the uncertain parameter as an optimization variable and uses a cost functional that integrates the original cost multiplied by the probability density function over the parameter range. However, the application is limited to parametric uncertainty in both the cost function and dynamic model constraints of OCPs, excluding parametric uncertainty in path constraints. Xiong et. al. [125] transforms the original stochastic trajectory optimization problem into an equivalent deterministic one in the expanded higher-dimensional state space by the polynomial chaos expansion method. Path constraints can be handled in this approach. However, the number

of states in the expanded deterministic state equations is equal to the number of original states, n_s, multiplied by the total number of state-expansion coefficients, n_r. Furthermore, the total number of state-expansion coefficients is $n_r := {}^{(n_e+n_p)!}/_{n_e!n_p!}$, where n_p is the number of unknown parameters and, n_e is the order of the polynomial chaos expansion [126]. This number grows rapidly as the polynomial order n_e and/or the number of parameters n_p increases. Thus, this approach is not applicable to the problem space of this thesis due to a large number of parameters. Thus, the existing approaches to handle parametric uncertainty are not readily applicable to the problem of interest, which includes a nonlinear dynamics model, nonlinear path constraints, and a large number of parameters.

Therefore, in this chapter, a novel approach is developed to improve the robustness of the obstacle avoidance algorithm to parametric uncertainty. In particular, a double-worst-case formulation is developed for a robust satisfaction of the two safety requirements for high-speed obstacle avoidance in AGVs: collision-free and no-wheel-lift-off. Similar to the scenario-based approach, the proposed method also makes the constraints in the OCP formulation in the MPC algorithm robustly safe in all of the considered scenarios. However, instead of using hundreds of randomly generated scenarios, only two scenarios are considered, which are the two most likely worst-case scenarios corresponding to the two types of identified unsafe scenarios. In addition, instead of applying all the scenarios to check all the constraints, each scenario is only used to check the satisfaction of the relevant constraints, which helps to minimize the number of constraints in the augmented OCP formulation for robustness. The results show that the proposed formulation improves the robustness of the algorithm, albeit it cannot be guaranteed for all possible realizations of parametric uncertainty distributions because of the non-uniqueness and the approximation of the worst-case scenarios. The chapter also quantifies the trade-off between the robustness and task completion performance of the MPC-based obstacle avoidance algorithm.

The list of parameters used in the MPC model and the associated uncertainty levels are summarized in Table 7.1. It is assumed that the uncertainty distributions are independent and follow a uniform distribution with given upper and lower bounds. In particular, it is assumed that the mass and inertia terms have $\pm 10\%$ uncertainty around their nominal values; the CoG location measurements have $\pm 20\%$ uncertainty; and the tire parameters have $\pm 30\%$ uncertainty. The uncertainty levels are estimated based on the difficulty of measuring the parameter value. In general, the parameter values used in the tire model have large uncertainty, especially in the unstructured environment, which is often off-road. The vehicle CoG location measurements have

larger uncertainty compared to the mass and inertia measurements because the CoG location can easily shift. It is assumed that the uncertainty associated with the distance between the two axles, L_v, and the tire radius, R_t, are negligible because they can be easily and directly measured. The uncertainty range of the mass and inertia may include small changes due to fuel consumption but to account for the large changes is not intended, for example, due to additional payload. The results presented in the following sections are based on these assumptions of the uncertainty levels. When different assumptions are made, the particular results in terms of numerical values obtained, for example, the percentage of failure scenarios, may not hold, but the same methodology can be followed.

Table 7.1: List of parameters used in the MPC model.

Parameter	**Symbol**	**Nominal Value**	**Unit**	**Error Bounds[%]**
Sprung mass	M_s	2251.5	kg	±10
Unsprung mass	M_u	109.3	kg	±10
Yaw inertia	I_{zz}	4110.1	kg·m^2	±10
Front axle to rear axle distance	L_v	3.30	m	0
Front axle to CoG distance	L_f	1.58	m	±20
CoG height	h_{CG}	0.99	m	±20
Tire radius	R_t	0.55	m	0
Shape factor for lateral forces	P_c	1.5874	-	±30
Lateral friction	P_d	0.73957	-	±30
Variation of friction with load	$P_{d,z}$	-0.075004	-	±30
Lateral curvature	P_e	0.37562	-	±30
Variation of curvature with load	$P_{e,z}$	-0.069325	-	±30
Maximum value of stiffness	P_k	-10.289	-	±30
Load at which stiffness reaches maximum value	$P_{k,z}$	3.3343	-	±30

7.2 Evaluation of Robustness to Parametric Uncertainty

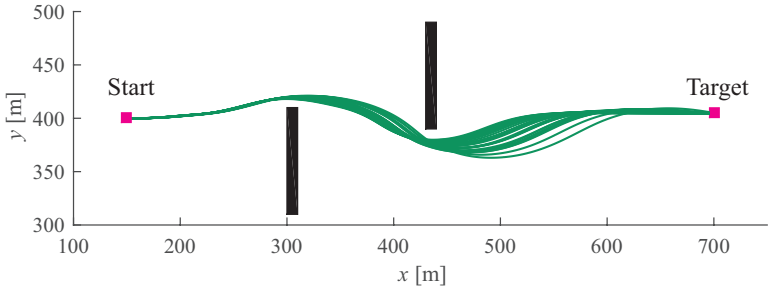
In previous simulations, it was assumed that there is no parametric uncertainty. Thus, the parameter values used in the MPC model were the nominal ones listed in Table 7.1. These nominal values were also used in the plant model, which was then used to demonstrate the effectiveness of the obstacle avoidance algorithm in closed-loop simulations. When there is parametric uncertainty, it is necessary to first

evaluate the robustness of the algorithm when nominal parameter values are used, because the MPC framework has inherent robustness due to feedback and re-planning over a moving horizon. Thus, it is possible that the MPC algorithm is already robust to parametric uncertainty and no additional strategies are needed. To perform this task, closed-loop simulations are conducted where each simulation uses a different set of parameter values in the plant model sampled from the parametric uncertainty distributions. The MPC model parameters, on the other hand, are kept at their nominal values in all simulations.

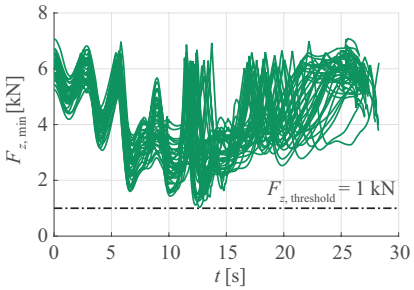
Because there are twelve parameters with uncertainty, proper sampling of the parameter values is necessary to cover most of the parametric uncertainty domain with a limited number of expensive simulations. In particular, Latin Hypercube Design (LHD) is used to generate a tractable collection of near-random samples of parameter values from the multidimensional distribution [127]. An LHD is performed in such a way that each of the n_d dimensions is divided into n_l equal levels and that there is only one sample at each level. A random procedure is used to determine the sample locations. Specifically, for this problem, there are 12 dimensions, and each of them is divided into 50 equal levels. The LHD used in this paper is optimized so that it has good space filling quality and low correlation between samples [128]. Thus, 50 sets of parameter values, in other words, 50 scenarios, are obtained.

Because there are two safety requirements for obstacle avoidance, namely, collision-free and no-wheel-lift-off, two obvious safety issues are collision and wheel lift-off, which are hard violations of the safety requirements. In addition, safety margins are included in the constraint formulations in the OCP. The violations of the plant to these constraints, i.e., moving into the safety margins without collision or wheel lift-off are soft violations. Both hard and soft violations are unacceptable in the simulation-based robustness study. In other words, when the distance to obstacle or tire vertical load is smaller than the corresponding safety margin, the AGV is considered as unsafe. The task of the AGV in these simulations is also to move from its start position to pass through a given target position from a specified direction. Thus, task completion is defined as moving through the final target position and heading in the specified direction. Thus, safety and task completion are both important. When the AGV passes through the target position from a different direction that has a greater than or equal to 10-degree difference compared to the specified one, it is considered as a failure even with no safety violations.

Fig. 7.1 shows the simulation results from the 50 scenarios with an obstacle map that consists of two obstacles. In this test, the final heading angle is required to be



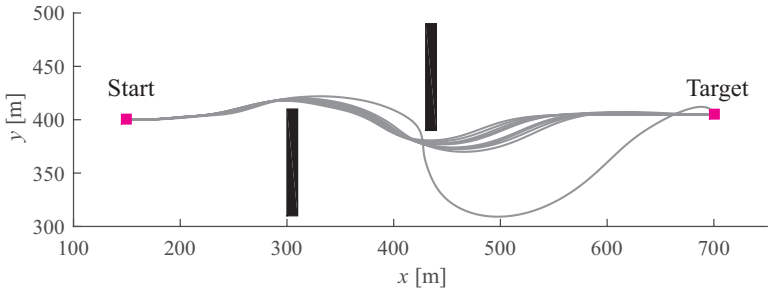
(a) Trajectory profiles of the 38 scenarios with no safety issues.



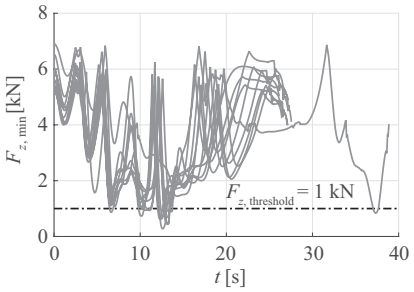
(b) Smallest vertical load profiles of the 38 scenarios with no safety issues.

Figure 7.1: Closed-loop simulation results with the LHD scenarios using the obstacle avoidance algorithm with nominal parameter values.

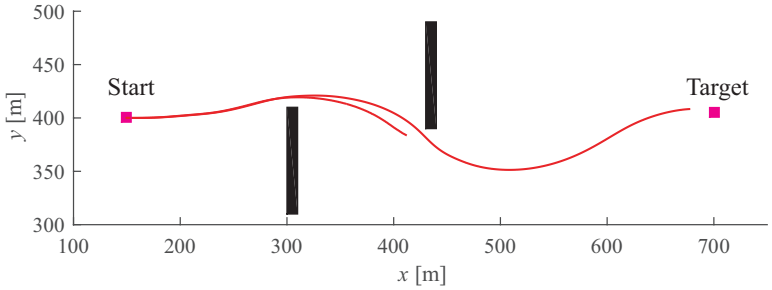
the same as the initial heading direction. In summary, the AGV in 38 out of the 50 scenarios successfully completes the task without any soft or hard violations of the safety requirements as shown in Fig. 7.1a and 7.1b. The remaining 12 scenarios lead to hard or soft violations. Soft violations are observed with 10 scenarios as shown in Fig. 7.1c and 7.1d. All the soft violations are the violation of the minimum vertical load threshold. No violation of the minimum distance to obstacle threshold is observed. One of the 10 scenarios results in a trajectory that moves very close to the obstacle before slowing down to avoid collision that leads to a significantly different and much longer route. Moreover, the task is not successfully completed with this scenario, because the final heading angle requirement is not met. Hard violations are observed with 2 scenarios. These simulations are terminated before the AGV reaches the target position, because wheel lift-off is observed as shown in Fig. 7.1e, and 7.1f. Therefore, it is concluded that the algorithm is not highly robust to the assumed parametric uncertainty distributions. Violations of the safety requirements are observed in 24% of the tested scenarios.



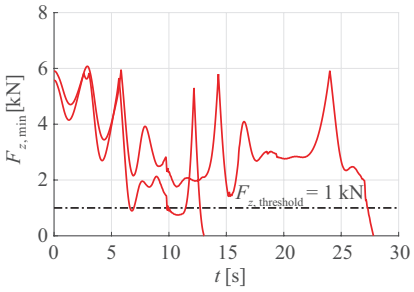
(c) Trajectory profiles of the 10 scenarios with violations of the vertical load threshold.



(d) Smallest vertical load profiles of the 10 scenarios with violations of the vertical load threshold.

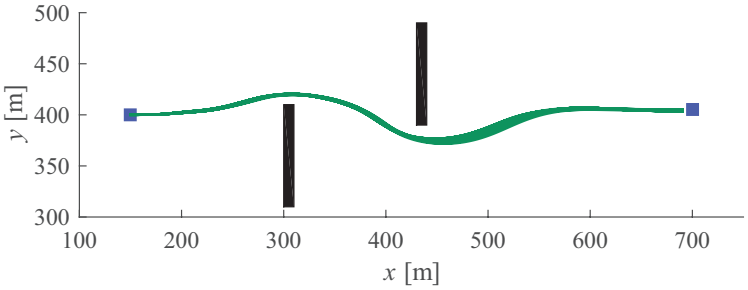


(e) Trajectory profiles of the 2 scenarios with single-wheel lift-off.

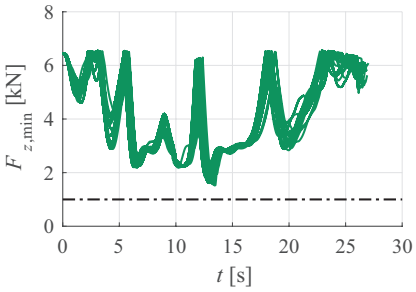


(f) Smallest vertical load profiles of the 2 scenarios with single-wheel lift-off.

Figure 7.1: (continued) Closed-loop simulation results with the LHD scenarios using the obstacle avoidance algorithm with nominal parameter values.



(a) Trajectory profiles.



(b) Smallest vertical load profiles.

Figure 7.2: Closed-loop simulation results with state measurement uncertainty.

7.3 Evaluation of Robustness to State Measurement Uncertainty

Besides parametric uncertainty, there are other sources of uncertainty. For example, one source of uncertainty is the errors in the knowledge of vehicle's locations, and current and future locations of the obstacles. Another source of uncertainty is the external disturbances due to wind and other factors. These sources might also cause problems and need to be considered [121].

In this section, the effect of the state measurement uncertainty on the robustness of the obstacle avoidance algorithm is evaluated. It is assumed that the estimation uncertainties of all states are independent and they follow uniform distributions. Thus, the upper and lower bounds are known. In particular, the vehicle position in global coordinates (x, y) has a ± 2.5 m error in each direction. The vehicle heading angle ψ has an error of $\pm 3°$.

In the simulation, at each step of the MPC update, the initial value of each state variable passed to the algorithm includes error sampled from the corresponding distribution. The parameter values used in the MPC model were the nominal ones listed in Table 7.1. The simulation is repeated 50 times. Figure 7.2 shows the simulation results with an obstacle map that consists of two obstacles.

Neither soft nor hard violations of the safety requirements are observed. It is concluded that the MPC-based obstacle avoidance algorithm is robust to the considered state measurement uncertainty distributions. Thus, when studying the effect of parametric uncertainty, the state measurement uncertainty is not considered so that the focus can be put specifically on the parametric uncertainty.

7.4 Improvement of Robustness to Parametric Uncertainty

A novel approach is proposed in this section to improve the robustness of the obstacle avoidance algorithm to parametric uncertainty. As mentioned in Section 7.1, the scenario-based approach can be applied to the considered problem with a nonlinear dynamics model. However, because the MPC model has a large number of parameters, hundreds of randomly generated scenarios are needed to provide a relatively high confidence level of the chance constraint, which is computationally intractable. To overcome this challenge, a double-worst-case formulation is developed. Systematic off-line simulations with various scenarios are performed to identify the scenarios that will lead to active constraints. Thus, the computational burden is shifted off-line. As a result, only two scenarios are considered in the proposed formulation, which are the two worst-case scenarios of different types corresponding to the two safety requirements for high-speed obstacle avoidance in AGVs: collision-free and no-wheel-lift-off. To further reduce the number of constraints in the augmented OCP formulation for robustness, each scenario is only used to check the satisfaction of the relevant constraints instead of applying all the scenarios to check all the constraints. The details of the proposed approach are discussed comprehensively in this section starting with a brief overview.

To give a definition of the worst-case scenario, the unsafe scenario is first defined. If the closed-loop results with a particular scenario reveal safety issues, that is, either soft or hard violations, the scenario is considered as unsafe. However, the justification of whether a scenario is unsafe or not depends on the obstacle field and the vehicle model and parameter values used in the MPC algorithm, because the control commands for the AGV with that scenario are generated by the algorithm in reaction to the given obstacle field. To reduce the number of inter-dependencies in the analysis and make it tractable, open-loop simulation results generated using constant-steering constant-acceleration maneuvers with the unsafe scenarios obtained

in Section 7.2 are used to analyze the characteristics of the unsafe scenarios in terms of vehicle responses. In Section 7.4.1, two types of unsafe scenarios are identified. Each of them is qualitatively defined based on a metric, which is a function of the AGV state variables, but no quantitative thresholds are set in the definition because of the inter-dependencies. For each type of unsafe scenario, the scenario that results in the smallest metric value is the worst-case scenario for a particular maneuver. It is essential to identify the worst-case scenarios, because if the worst-case scenarios can be made robustly safe, all the other scenarios will be safe, too. Open-loop simulations using a set of constant-steering constant-acceleration maneuvers that approximately covers the entire operation space of the AGV are used to obtain the worst-case scenario. However, the worst-case scenario of each type of unsafe scenario is not unique for all maneuvers because of model nonlinearities. Thus, the term most likely worst-case scenario is used to refer to the worst-case scenario that results in the smallest metric value for most of the considered maneuvers. If a scenario results in a smaller metric value than that from the most likely worst-case scenario with at least one maneuver, it is categorized as a less likely worst-case scenario. There may be multiple less likely worst-case scenarios for each type of unsafe scenarios. The procedure for obtaining the two most likely worst-case scenarios is presented in Section 7.4.2 in detail.

The procedure presented in Section 7.4.2 ignores the effects of interactions between the parameters because of the difficulty caused by the large number of parameters. To validate the obtained most likely worst-case scenarios and to identify the less likely worst-case scenarios that are noticeably different than the most likely worst-case scenarios, further open-loop simulations are performed with all the scenarios that consist of the upper or lower bound of all parameters using maneuvers picked according to the previous analysis. The results are presented in Section 7.4.3.

With the obtained most likely worst-case scenarios and less likely worst-case scenarios, two double-worst-case formulations are presented in Section 7.5. The first one considers only the two most likely worst-case scenarios and each of them is used to check only the relevant constraints. The second one also accounts for the less likely worst-case scenarios approximately in a way such that no additional scenarios are introduced in the formulation besides the two most likely worst-case scenarios.

7.4.1 Unsafe Scenarios

The scenarios corresponding to the closed-loop simulation results shown in Figure 7.1c, 7.1d, 7.1e, and 7.1f are unsafe scenarios for the considered obstacle field and the MPC algorithm using nominal parameter values. Open-loop simulation results using two representative constant-steering constant-acceleration maneuvers with these unsafe scenarios are shown in Figure 7.3 to assist the analysis of the characteristics of the unsafe scenarios in terms of vehicle responses. Figure 7.3a - 7.3c are results with constant acceleration; and Figure 7.3d - 7.3f are results with constant deceleration. These two maneuvers are selected because the longitudinal acceleration affects significantly the longitudinal load transfer and the sensitivity of a vehicle to steering.

As shown in Figure 7.3a, and 7.3d, in 11 out of the 12 open-loop results with the unsafe scenarios, the predicted tire vertical load values are almost always smaller than the values predicted with the nominal scenario along the prediction horizon. Thus, because of the differences between the vertical load value predictions, even when the control commands generated by the MPC algorithm with nominal parameter values satisfy the dynamical safety constraints in the OCP formulation, the AGV with these scenarios still encounters violations of the minimum-tire-vertical-load threshold or even wheel lift-off after the control commands are executed. Thus, the first type of unsafe scenario is characterized by the tire vertical load value. Because the vertical load profile is a timed sequence of data, the minimum tire vertical load value is used as the evaluation metric. A smaller value means a larger chance of being unsafe. As an example, the metric value of the nominal scenario with the given control commands for tire vertical load prediction is $\mathcal{M}_{\text{load}} = 1.72$ kN, which is the value indicated by the circle in Figure 7.4a.

For the one case that does not belong to the previous type, its trajectory is the "stiffest" of the 12 scenarios as shown in Figure 7.3b and 7.3e. In other words, the AGV with this scenario cannot turn as sharp as with the nominal scenario using the same control commands. Thus, the steering command generated by the MPC model with nominal parameter values is not large enough to steer the AGV with this scenario away from the obstacle early enough, and the AGV moves very close to obstacles due to lack of sufficient turning, which leads to the need for a large correction effort and a significant detour, and results in task incompletion. With a different obstacle field, violation of minimum-distance-to-obstacle threshold or even collision is possible. Thus, the second type of unsafe scenario is characterized by the trajectory "stiffness" that is defined as the area under the trajectory profile. The smaller the metric value is,

the "stiffer" the trajectory is. As an example, the metric value of the nominal scenario with the given control commands for trajectory prediction is $\mathcal{M}_{\text{traj}} = 128$ m^2, which is the area of the shaded region in Figure 7.4b.

In summary, the two types of unsafe scenarios identified are named as the small vertical load type and the "stiff" trajectory type. Note that these two types of unsafe scenarios are distinct; i.e., a small vertical load scenario is not a "stiff" trajectory scenario and vice versa as illustrated by the results in Figure 7.3.

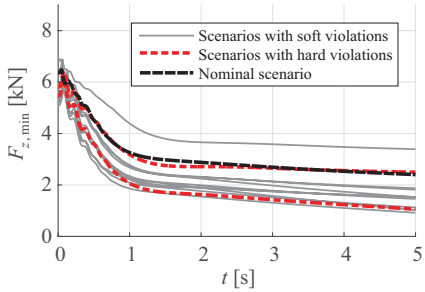
7.4.2 Most Likely Worst-Case Scenarios

The approach used to obtain the most likely worst-case scenario for each type of unsafe scenarios is presented in this section. Because of the model nonlinearities, it is difficult, if not impossible, to obtain the most likely worst-case scenarios analytically; hence, a numerical approach is used. Because there are twelve parameters, the interactions between the parameters are ignored because of the large number of combinations. The effect of each parameter value on the metrics is studied independently using off-line open-loop simulation results. A set of constant-steering constant-acceleration maneuvers is used for the evaluation, because this type of maneuver can be easily parameterized. Different combinations of steering angle, acceleration, and initial speed values are considered to approximately cover the entire operation space of the AGV. The sequence of values considered for each of the commands is listed in Table 7.2. The pairs of initial speed and acceleration shown in Figure 7.5 are used, where the pairs that generate speed less than 5 m/s or greater than 29 m/s along a 2 seconds simulation are eliminated. These two speed limits are used because a speed slower than 5 m/s does not pose any challenge and the maximum speed of the vehicle is 29 m/s. For each pair, the sequence of steering angle values specified in Table 7.2 is considered.

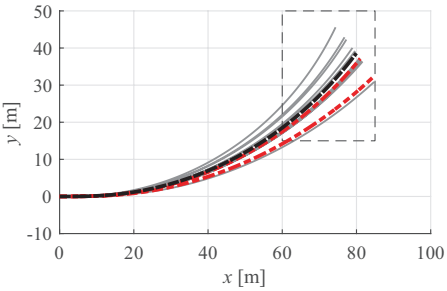
Table 7.2: Values of the commands used in the parameterized evaluation maneuvers.

Command	**Symbol**	**Unit**	**Minimum**	**Maximum**	**Interval**
Acceleration	a_x	m/s^2	-5	2	0.5
Steering angle	δ_f	$^\circ$	1	10	0.5
Initial speed	U_0	m/s	5	29	2

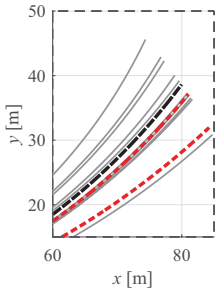
For each of the N_p parameters, N_v values within its uncertainty range, $[v_{min,i}, v_{max,i}]$, are considered, where i is the index of the parameter. In other words, N_v scenarios are evaluated, in which only one parameter value is varied and the other parameter values are maintained at their reference values. The initial reference values



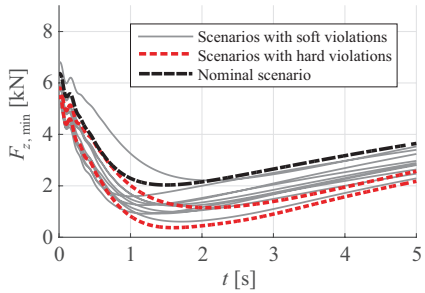
(a) Tire vertical load profiles: acceleration.



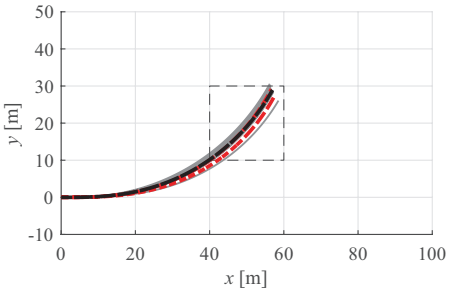
(b) Trajectory profiles: acceleration.



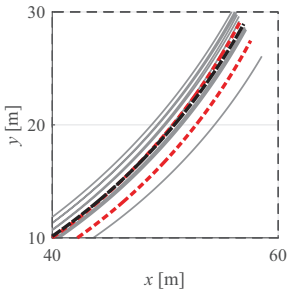
(c) Trajectory profiles (zoom-in view): acceleration.



(d) Tire vertical load profiles: deceleration.

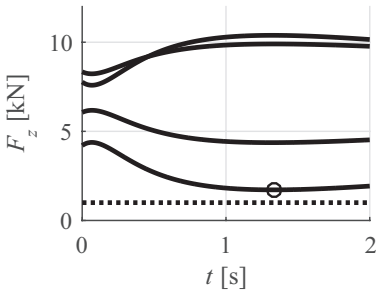


(e) Trajectory profiles: deceleration.

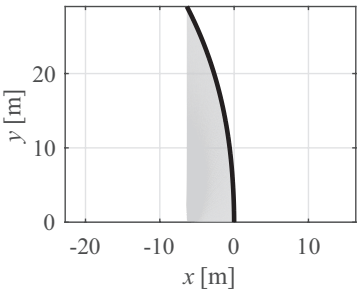


(f) Trajectory profiles (zoom-in view): deceleration.

Figure 7.3: The corresponding open-loop simulation results with unsafe scenarios from the closed-loop simulations compared with results with the nominal scenario.



(a) The minimum tire vertical load is the metric used for evaluating the tire vertical load prediction.



(b) The area under the trajectory is the metric used to characterize the trajectory "stiffness".

Figure 7.4: Definitions of evaluation metrics exemplified with the nominal scenario and control commands: $U_0 = 16$ m/s, $a_x = -1$ m/s^2, $\delta_f = 3°$.

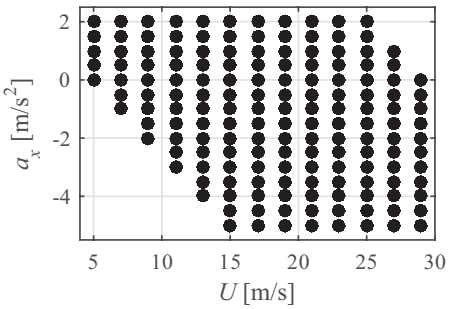


Figure 7.5: Pairs of longitudinal speed and acceleration commands used in the parameterized evaluation maneuvers for obtaining the worst-case scenarios.

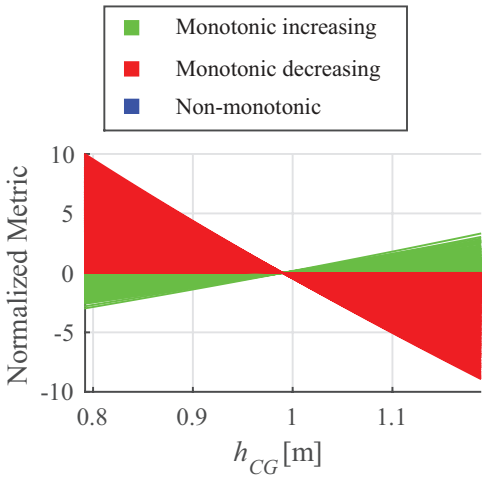
are the nominal parameter values. For each type of unsafe scenarios, after obtaining the most likely worst-case scenario using the initial reference values, Algorithm 1 is repeated to update the most likely worst-case scenario with the previously obtained most likely worst-case scenario as the new reference values. If the updated most likely worst-case scenario is the same as the previous one, the most likely worst-case scenario is finalized. Otherwise, Algorithm 1 is repeated again with the updated most likely worst-case scenario as the new reference values until it converges.

In Algorithm 1, for each of the considered parameters, the vehicle model is simulated for all the combinations of the N_v scenarios and N_m maneuvers. Two matrices are generated. The first one, I_{safe}, records whether the maneuver is safe or not with the considered scenario. The second one, $\mathcal{M}$, records the metric value calculated based on the vehicle response. Both of the matrices are of size N_m by N_v. Only when the maneuver is safe for all the considered scenarios, are the corresponding metric values used for the determination of most likely worst-case scenario. For example, Figure 7.6a illustrates the effects the parameter h_{CG} has on the trajectory stiffness metric value with 1127 safe maneuvers, where the relationships are all monotonic and approximately linear. Each line corresponds to one maneuver. The normalized metric is displayed, in which the median of each line is set to 0. Using these results, the value of the parameter that results in the smallest metric value is obtained. However, the value is not unique because it depends on the maneuver. In this example, the smallest metric value is achieved with the lower bound value of h_{CG} for 333 out of the 1127 maneuvers (about 30%). For the remaining 794 maneuvers, it is achieved with the upper bound value. Figure 7.6b shows corresponding maneuvers by the projected views of the operation space. In general, when the AGV accelerates, a larger h_{CG} value leads to a “stiffer" trajectory. In addition, it is observed that the variation range of the metric value also depends on the maneuver. When a “stiffer” trajectory is given by a larger h_{CG} value, the maximum variation range is 18.9, whereas when a “stiffer” trajectory is given by a smaller h_{CG} value, the maximum variation range is 6.3. Taking into account both the number of maneuvers and the value of the maximum deviation range, it is concluded that the upper bound value of h_{CG} is in the set of parameter values for most likely “stiffest” trajectory.

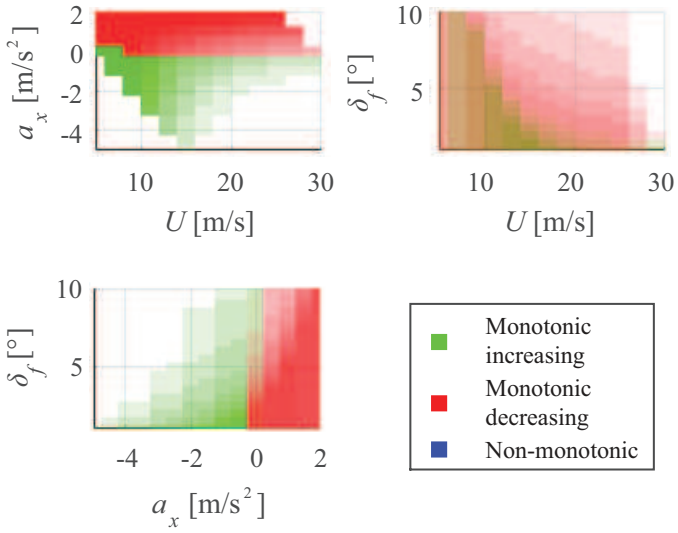
For the considered vehicle model and parametric uncertainty distributions, after repeating the Algorithm 1 two or three times, the most likely worst-case scenario of each type converges. Figure 7.7 summaries the effects of all parameters in the last run of the algorithm on the trajectory stiffness and vertical load metrics, respectively. In each figure, the top subfigure shows, the percentage of maneuvers that lead

Algorithm 1 Obtaining the most likely worst-case scenarios

procedure DATACOLLECTION(type, $v_{\max}$, $v_{\min}$, $\text{scenario}_{\text{ref}}$, maneuver)
  for $i \leftarrow 1$ **to** N_p **do** ▷ N_p parameters
    $v_{\text{seq},i} \leftarrow \text{linspace}\ (v_{\max,i}, v_{\min,i}, N_v)$
    $N_{in}(i) \leftarrow 0$, $N_{\text{de}}(i) \leftarrow 0$, $N_{ot}(i) \leftarrow 0$ ▷ Counters initialization
    $R_{in}(i) \leftarrow 0$, $R_{\text{de}}(i) \leftarrow 0$, $R_{ot}(i) \leftarrow 0$ ▷ Ranges initialization
    for $j \leftarrow 1$ **to** N_m **do** ▷ N_m maneuvers
      scenario $\leftarrow \text{scenario}_{ref}$ ▷ Reference scenario
      for $k \leftarrow 1$ **to** N_v **do** ▷ N_v scenarios
        $\text{scenario}(i) \leftarrow v_{\text{seq},i}(k)$ ▷ Current scenario
        $[F_z, x, y] \leftarrow \text{Model}(\text{maneuver}_j, \text{scenario})$ ▷ Model simulation
        if $F_z \geq F_{z,\text{threshold}}$ **then**
          $I_{\text{safe}}(j,k) \leftarrow 1$ ▷ Safe maneuver
        else
          $I_{\text{safe}}(j,k) \leftarrow 0$ ▷ Unsafe maneuver
        end if
        if type = "load" **then**
          $\mathcal{M}(j,k) \leftarrow f_{\text{load}}(F_z)$ ▷ Trajectory stiffness metric
        end if
        if type = "traj" **then**
          $\mathcal{M}(j,k) \leftarrow f_{\text{traj}}(x,y)$ ▷ Vertical load metric
        end if
      end for
      if $\text{sum}(I_{\text{safe}}(j,:)) = N_v$ **then** ▷ Array of all ones
        if $\text{length}(\text{find}(\text{diff}(\mathcal{M}(j,:)) < 0)) = 0$ **then**
          ▷ Monotonic increasing
          $N_{\text{in}}(i) \leftarrow N_{\text{in}}(i) + 1$
          $R_{\text{in}}(i) \leftarrow \max(R_{in}(i), \max(\mathcal{M}(j,:)) - \min(\mathcal{M}(j,:)))$
        else if $\text{length}(\text{find}(\text{diff}(\mathcal{M}(j,:)) > 0)) = 0$ **then**
          ▷ Monotonic decreasing
          $N_{\text{de}}(i) \leftarrow N_{\text{de}}(i) + 1$
          $R_{\text{de}}(i) \leftarrow \max(R_{\text{de}}(i), \max(\mathcal{M}(j,:)) - \min(\mathcal{M}(j,:)))$
        else ▷ Otherwise
          $N_{\text{ot}}(i) \leftarrow N_{\text{ot}}(i) + 1$
          $R_{\text{ot}}(i) \leftarrow \max(R_{ot}(i), \max(\mathcal{M}(j,:)) - \min(\mathcal{M}(j,:)))$
        end if
      end if
    end for
    if $\max(N_{\text{in}}(i) \cdot R_{in}(i), N_{\text{de}}(i) \cdot R_{\text{de}}(i)) \geq \cdot N_{ot}(i) \cdot R_{\text{ot}}(i)$ **then**
      if $N_{in}(i) \cdot R_{in}(i) \geq \cdot N_{\text{de}}(i) \cdot R_{\text{de}}(i)$ **then**
        $\text{scenario}_{\text{worst}}(i) \leftarrow v_{\min,i}$ ▷ Lower bound value
      else if $N_{\text{de}}(i) \cdot R_{\text{de}}(i) \geq \cdot N_{in}(i) \cdot R_{in}(i)$ **then**
        $\text{scenario}_{\text{worst}}(i) \leftarrow v_{\max,i}$ ▷ Upper bound value
      else
        $\text{scenario}_{\text{worst}}(i) \leftarrow \text{NaN}$ ▷ Detailed study needed
      end if
    else
      $\text{scenario}_{\text{worst}}(i) \leftarrow \text{NaN}$ ▷ Detailed study needed
    end if
  end for
end procedure



(a) The variation of the normalized trajectory stiffness metric value with the h_{CG} value for various evaluation maneuvers.



(b) The effect of control commands on the relationship between the trajectory stiffness metric value and the h_{CG} value.

Figure 7.6: Evaluation of the effects of the h_{CG} value and the maneuver on the trajectory stiffness metric value.

to a monotonically increasing relationship between the parameter value and metric value ($^{N_{in}}/_{N_{in}+N_{de}+N_{ot}}$), a monotonically decreasing relationship ($^{N_{de}}/_{N_{in}+N_{de}+N_{ot}}$), and a non-monotonic relationship ($^{N_{ot}}/_{N_{in}+N_{de}+N_{ot}}$). The maximum deviation range of the metric value for each relationship (R_{in}, R_{de}, and R_{ot}) is summarized in the second subfigure. The last subfigure combines the above information by introducing a voting metric, $N_\bullet \cdot R_\bullet$, and normalizing it so that the maximum voting metric for each of the parameter is 1. The voting metric values are then used to determine the most likely worst-case scenario of each type. For each parameter, if the voting metric of the monotonic increasing relationship is the largest, the parameter lower bound value is in the set of parameter values for most likely worst-case scenario, whereas if the voting metric of the monotonic decreasing relationship is the largest, the parameter higher bound value is in the set. However, if the non-monotonic relationship has the largest voting metric value, a more detailed discussion of the results is needed as exemplified below.

For the trajectory prediction, as shown in Figure 7.7a, besides the parameter h_{CG} that has been discussed in detail above, the effects of other parameters on the trajectory stiffness metric are almost independent of the maneuvers. Thus, it is straightforward to assemble the most likely worst-case scenario for "stiffest" trajectory based on the results, which is summarized in Table 7.3.

For the vertical load prediction, the effects of maneuvers are larger, but dominant trends still exist as shown in Figure 7.7b. Two parameters that need a more detailed discussion are L_f and P_d. For more than half of the maneuvers, the vertical load metric does not change monotonically with respect to the parameter value for both parameters. However, in 309 out of 356 maneuvers (about 87 %), the lower bound value of L_f results in the smallest vertical load metric. In 308 out of 355 maneuvers (about 87 %), the higher bound value of P_d results in the smallest vertical load metric. Thus, it is determined that the lower bound value of L_f and the higher bound value of P_d are in the set of parameter values for most likely smallest vertical load scenario. With the voting metric values and further analysis of the detailed results from the Algorithm 1, the most likely worst-case scenario for smallest vertical load is also determined and given in Table 7.3.

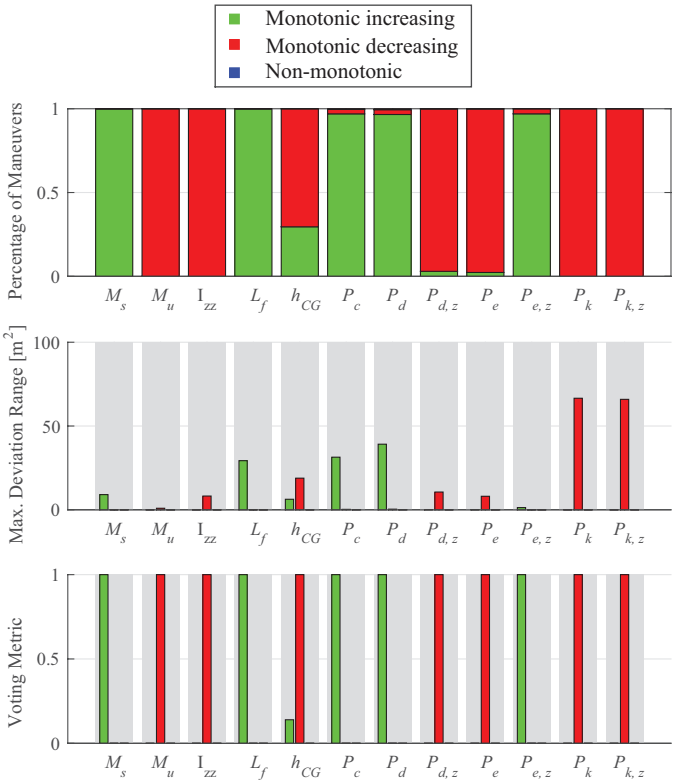
In addition, it is observed that the steering angle command does not have a significant effect on the metric values for both trajectory prediction and vertical load prediction. For the trajectory prediction, it is clear that the most likely worst-case scenario would be the actual worst-case scenario only with positive acceleration commands due to the fact that h_{CG} is monotonically decreasing only during acceleration.

Table 7.3: Parameter values of the nominal scenario and the most likely worst-case scenarios.

Parameter	**Nominal (Scenario 0)**	**"Stiffest" Trajectory (Scenario 1)**	**Smallest Vertical Load (Scenario 2)**	**Unit**
M_s	$\times 1.0$	$\times 0.9$	$\times 0.9$	kg
M_u	$\times 1.0$	$\times 1.1$	$\times 0.9$	kg
I_{zz}	$\times 1.0$	$\times 1.1$	$\times 0.9$	kg-m^2
L_f	$\times 1.0$	$\times 0.8$	$\times 0.8$	m
h_{CG}	$\times 1.0$	$\times 1.2$	$\times 1.2$	m
P_c	$\times 1.0$	$\times 0.7$	$\times 1.3$	-
P_d	$\times 1.0$	$\times 0.7$	$\times 1.3$	-
$P_{d,z}$	$\times 1.0$	$\times 0.7$	$\times 1.3$	-
P_e	$\times 1.0$	$\times 1.3$	$\times 0.7$	-
$P_{e,z}$	$\times 1.0$	$\times 1.3$	$\times 0.7$	-
P_k	$\times 1.0$	$\times 0.7$	$\times 1.3$	-
$P_{k,z}$	$\times 1.0$	$\times 1.3$	$\times 0.7$	-

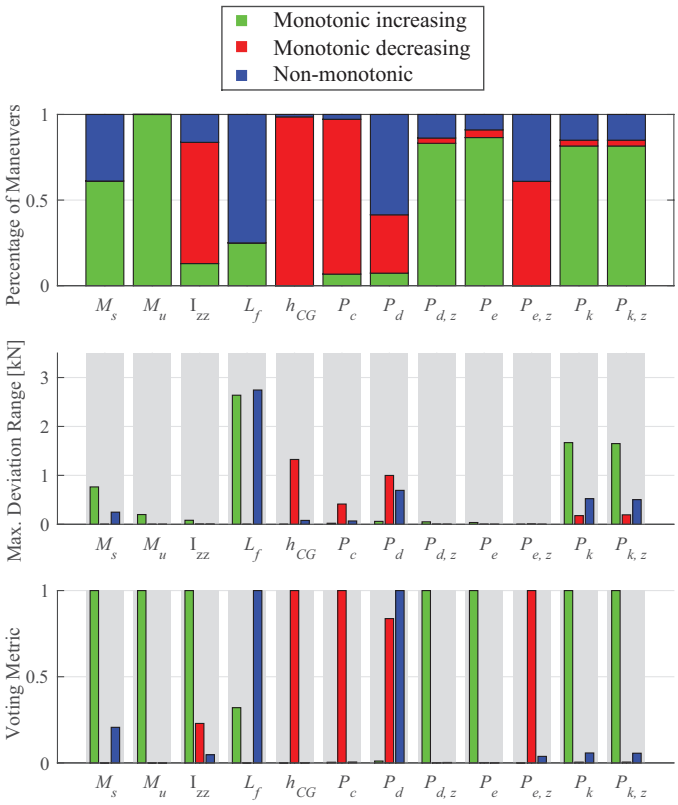
For the vertical load prediction, determining when the most likely worst-case scenario would be the actual worst-case scenario is more complicated. For the two most significant parameters in terms of deviation range besides L_f, which are P_k and $P_{k,z}$, the monotonicity of the vertical load metric value with respect to the parameter value as a function of the control commands are shown in Figure 7.8. It is observed that the most likely worst-case scenario would not be the actual worst-case scenario with high deceleration maneuvers starting from a high initial speed. Thus, the following three sets of control commands are used for the following open-loop simulations, which include acceleration, moderate deceleration, and hard deceleration maneuvers. The initial speeds of the three maneuvers are the same. The steering commands are selected to make the smallest vertical load approximately the same.

- Maneuver A: $U_0 = 16$ m/s, $a_x = +1$ m/s^2, $\delta_f = 3.0^\circ$
- Maneuver B: $U_0 = 16$ m/s, $a_x = -1$ m/s^2, $\delta_f = 2.5^\circ$
- Maneuver C: $U_0 = 16$ m/s, $a_x = -3$ m/s^2, $\delta_f = 1.5^\circ$



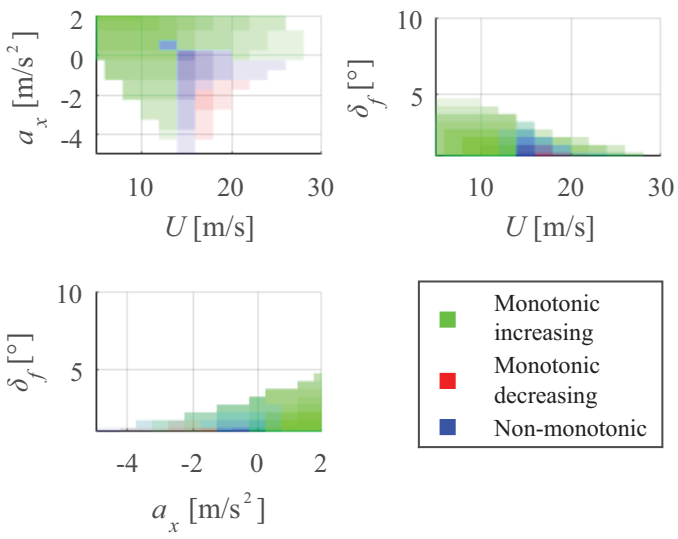
(a) Trajectory prediction.

Figure 7.7: Summary of the effects of all parameters on the prediction metrics using bar charts.

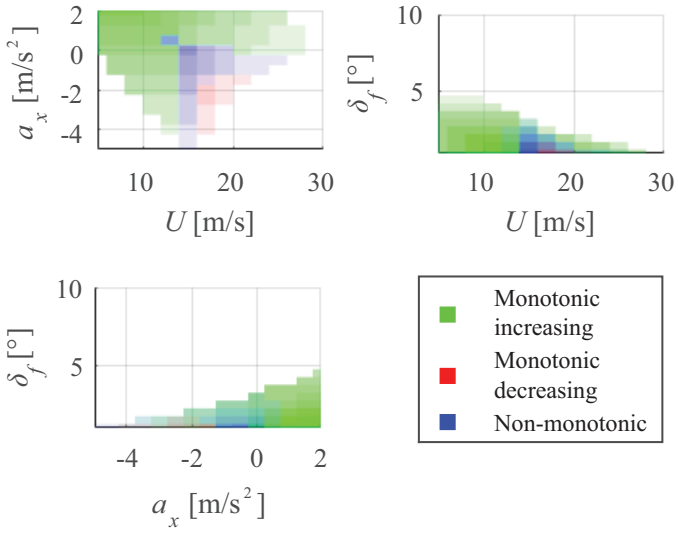


(b) Vertical load prediction.

Figure 7.7: (continued) Summary of the effects of all parameters on the prediction metrics using bar charts.



(a) P_k



(b) $P_{k,z}$

Figure 7.8: The monotonicity of the vertical load metric value with respect to the parameter value as a function of the control commands.

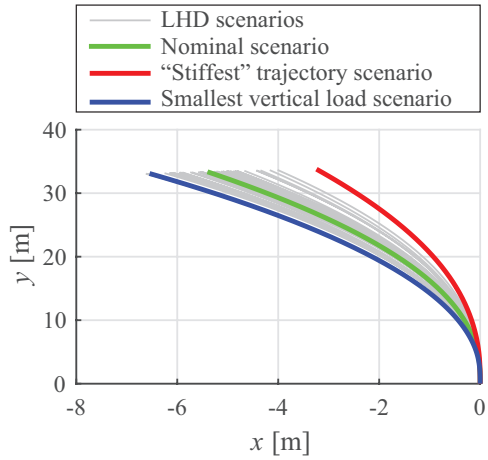
7.4.3 Open-Loop Simulation Results and Less Likely Worst-Case Scenarios

With Algorithm 1, the interactive effects between parameters are not considered. Although the algorithm is repeated several times to account for the interactive effects, it does so only in a limited sense. Therefore, to validate the obtained most likely worst-case scenarios, open-loop simulations are performed with the LHD scenarios used in Section 7.2 and with all the scenarios that consist of the upper or lower bound of all parameters. The latter set of results are also be used to identify the less likely worst-case scenarios that are noticeably different from the most likely worst-case scenarios.

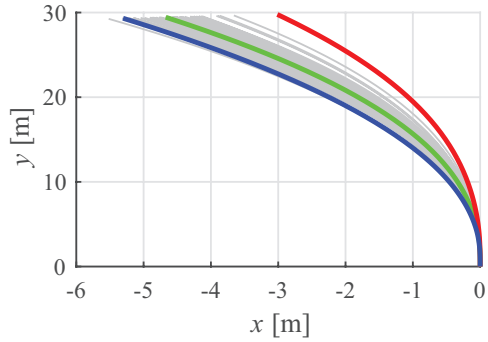
In Figure 7.9 and 7.10, open-loop simulation results with the scenarios listed in Table 7.3 are plotted on top of the open-loop simulation results with the LHD scenarios. All the trajectories from the LHD scenarios are more "flexible" than the trajectory from the most likely "stiffest" trajectory scenario. The smallest vertical loads from all of the LHD scenarios are also larger than the value from the most likely smallest vertical load scenario. However, it is possible that smallest vertical load scenario is not always the worst with hard deceleration as suggested by Figure 7.10c when a different set of initial state values is used.

As shown in Figure 7.7, most of the relationships between the parameter values and metric values are monotonic. Even if they are not, the smallest metric values are almost always achieved with either the upper bound or the lower bound of the parameter value. To exploit this observation, another set of open-loop simulations is then performed by assuming that each of the parameters can take only two values, which are the upper bound and the lower bound values. Thus, $2^{12} = 4096$ scenarios are considered. The results are shown in Figure 7.11 and Figure 7.12.

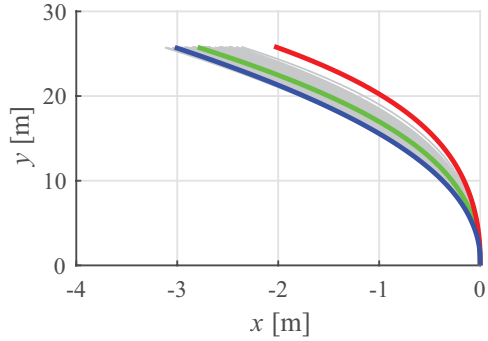
It is observed that, when the AGV accelerates, these two most likely worst-case scenarios are the actual worst-case scenarios among all the considered scenarios. However, when the AGV decelerates, they are not necessarily the actual worst-case scenarios and the differences become larger with harder deceleration. For the trajectory prediction, the smallest metric value is achieved with the lower bound value of h_{CG} for decelerating maneuvers. However, the upper bound value is used in the most likely worst-case scenario. Thus, there exists a worse scenario as shown in Figure 7.11c. Because the effect of h_{CG} on the trajectory prediction only ranks 6th as shown in the second subfigure of Figure 7.7a, the difference between the trajectories from the most likely worst-case scenario and the actual worst-case scenario is small and can be neglected. To confirm this inference, the less likely worst-case scenario for trajectory



(a) Maneuver A.

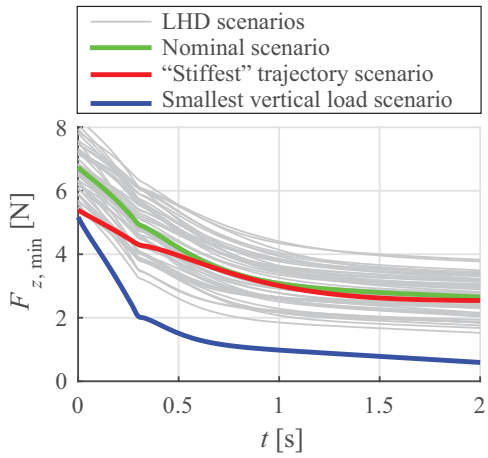


(b) Maneuver B.

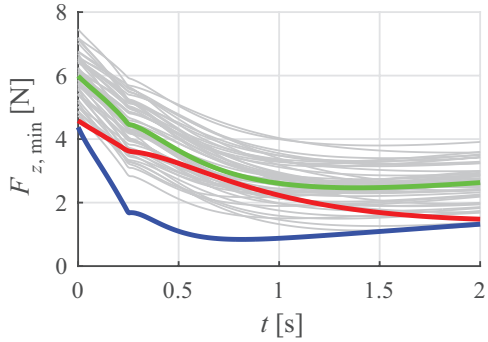


(c) Maneuver C.

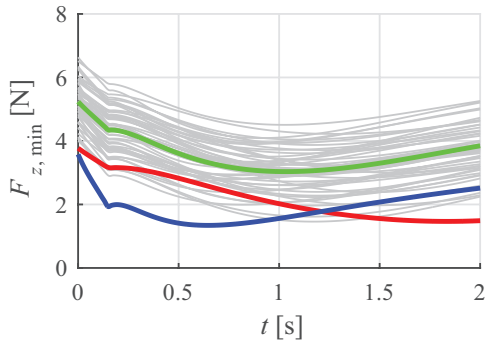
Figure 7.9: Trajectory profiles of the open-loop simulation results with the nominal scenario, the obtained most likely worst-case scenarios, and the LHD scenarios.



(a) Maneuver A.



(b) Maneuver B.



(c) Maneuver C.

Figure 7.10: Vertical load profiles of the open-loop simulation results with the nominal scenario, the obtained most likely worst-case scenarios, and the LHD scenarios.

prediction is listed in Table 7.4 and it is used in the closed-loop simulation in the next Section.

For the vertical load prediction, with acceleration and moderate deceleration, the most likely worst-case scenario is the actual worst-case scenario as shown in Figure 7.12a and 7.12b, respectively. However, it is not with hard deceleration as shown in Figure 7.12c. It would be beneficial if the vertical load profile generated by the scenario that results in the smallest metric value is the lower bound for vertical load along the entire simulation horizon. However, this is not the case; the lower bound consists of results from multiple scenarios as shown in Figure 7.13b and 7.13c. These scenarios are considered the less likely worst-case scenarios for vertical load prediction and are listed in Table 7.4.

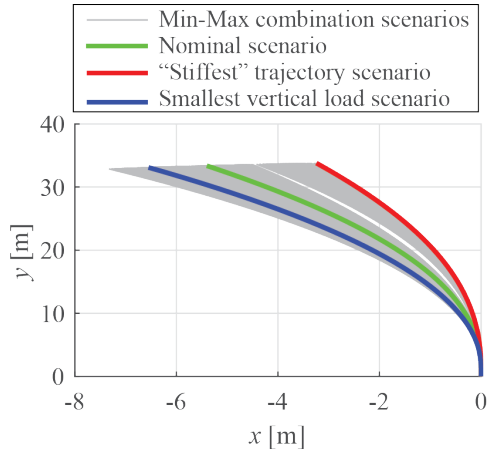
In summary, based on these two sets of open-loop results, it is concluded that the two most likely worst-case scenarios are at least very close to the actual worst-case scenarios. However, with hard deceleration, the differences between these scenarios are more significant. Thus, less likely worst-case scenarios are obtained with this type of maneuvers. These results provide an important part of the foundation for the proposed robust optimal control problem formulation.

7.5 OCP Formulation

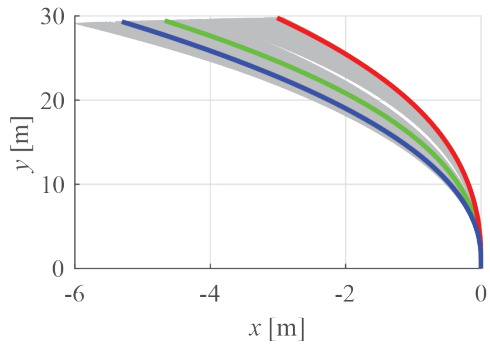
To improve the robustness of the obstacle avoidance algorithm, two approaches are considered. The first approach keeps the formulation as is and uses either of the two most likely worst-case scenarios in the MPC formulation. The second approach considers both types of the worst-case scenarios simultaneously, which is named as double-worst-case formulation. In this work, two double-worst-case formulations are proposed. The first one considers only the two most likely worst-case scenarios and each of them is used to check only the relevant constraints. The second one also accounts for the less likely worst-case scenarios approximately in a way such that no additional scenarios are introduced in the formulation besides the two most likely worst-case scenarios.

7.5.1 Original Formulation

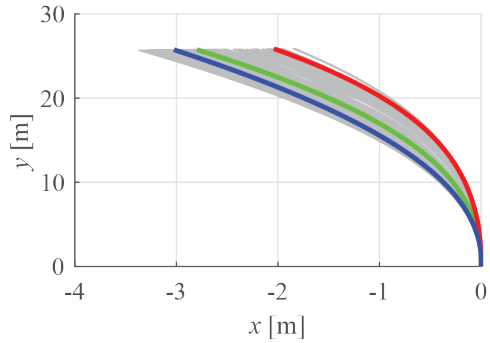
Because of the inherent robustness of the MPC framework, the original formulation with the nominal scenario, which is referred to as `MPC0`, can tolerate the parametric uncertainty to a certain degree. The results shown in Figure 7.1 are used as bench-



(a) Maneuver A.

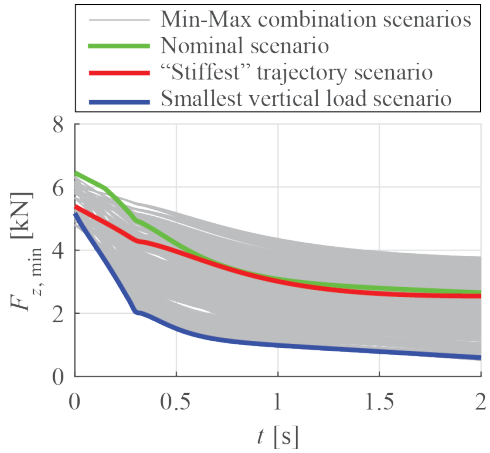


(b) Maneuver B.

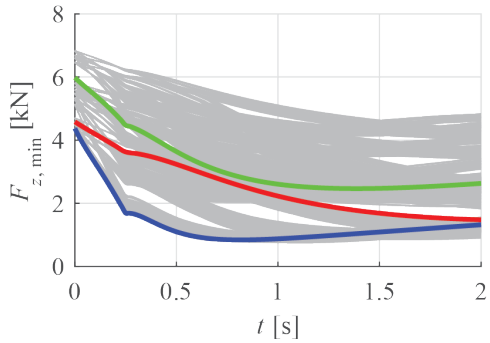


(c) Maneuver C.

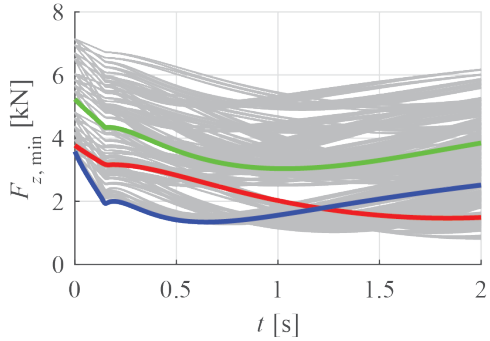
Figure 7.11: Trajectory profiles of the open-loop simulation results with the nominal scenario, the obtained most likely worst-case scenarios, and all combinations of parameter lower and upper bound values.



(a) Maneuver A.

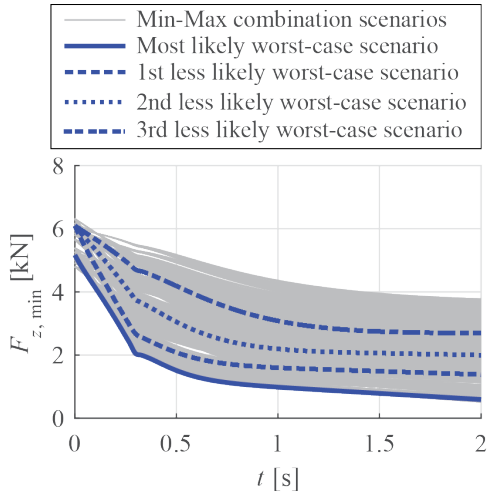


(b) Maneuver B.

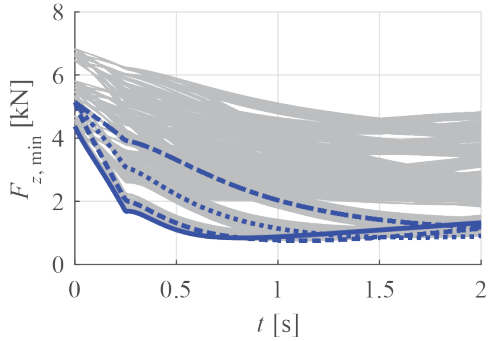


(c) Maneuver C.

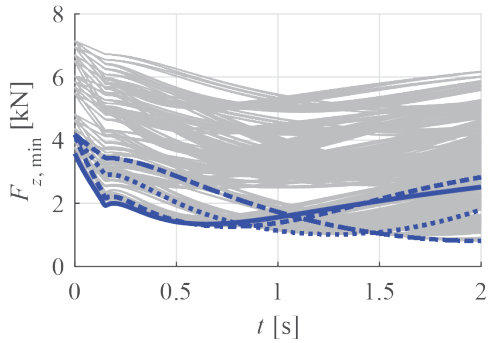
Figure 7.12: Vertical load profiles of the open-loop simulation results with the nominal scenario, the obtained most likely worst-case scenarios, and all combinations of parameter lower and upper bound values.



(a) Maneuver A.



(b) Maneuver B.



(c) Maneuver C.

Figure 7.13: Vertical load profiles of the open-loop simulation results with the most likely and less likely worst-case scenarios for vertical load prediction and all combinations of parameter lower and upper bound values.

Table 7.4: Parameter values of the less likely worst-case scenarios.

Parameter	**"Stiffest" Trajectory (Scenario 3)**	**Smallest Vertical Load (Scenario 4)**	**Unit**
M_s	$\times 0.9$	$\times$**1.1**	kg
M_u	$\times 1.1$	$\times 0.9$	kg
I_{zz}	$\times 1.1$	$\times 0.9$	kg-m^2
L_f	$\times 0.8$	$\times 0.8$	m
h_{CG}	$\times$**0.8**	$\times 1.2$	m
P_c	$\times 0.7$	$\times$**0.7**	-
P_d	$\times 0.7$	$\times$**0.7**	-
$P_{d,z}$	$\times 0.7$	$\times$**0.7**	-
P_e	$\times 1.3$	$\times$**1.3**	-
$P_{e,z}$	$\times 1.3$	$\times$**1.3**	-
P_k	$\times 0.7$	$\times 1.3$	-
$P_{k,z}$	$\times 1.3$	$\times 0.7$	-

Parameter	**Smallest Vertical Load (Scenario 5)**	**Smallest Vertical Load (Scenario 6)**	**Unit**
M_s	$\times$**1.1**	$\times$**1.1**	kg
M_u	$\times 0.9$	$\times 0.9$	kg
I_{zz}	$\times 0.9$	$\times 0.9$	kg-m^2
L_f	$\times 0.8$	$\times 0.8$	m
h_{CG}	$\times 1.2$	$\times 1.2$	m
P_c	$\times$**0.7**	$\times$**0.7**	-
P_d	$\times$**0.7**	$\times$**0.7**	-
$P_{d,z}$	$\times$**0.7**	$\times$**0.7**	-
P_e	$\times$**1.3**	$\times$**1.3**	-
$P_{e,z}$	$\times$**1.3**	$\times$**1.3**	-
P_k	$\times 1.3$	$\times$**0.7**	-
$P_{k,z}$	$\times$**1.3**	$\times$**1.3**	-

marks for evaluating the effectiveness and studying the trade-off in improving the robustness to parametric uncertainty.

Two other MPC implementations that are also considered are the original formulation with the most likely "stiffest" trajectory scenario (`MPC1`), or the original formulation with the most likely smallest vertical load scenario (`MPC2`). These two implementations don't increase the complexity of the OCP formulation but only change the parameter values used in the MPC model. The term most likely "stiffest" trajectory scenario is used to refer the most likely worst-case scenario based on the trajectory "stiffness". The term most likely smallest vertical load scenario is used similarly.

7.5.2 Double-Worst-Case Formulation

The first double-worst-case formulation (`MPC3`) considers the two most likely worst-case scenarios simultaneously and each of them is used to check only the relevant constraints, which requires the augmentation of the original OCP formulation. The augmented MPC model is specified by Equation (7.1) - Equation (7.12).

The control commands to be optimized are the longitudinal jerk, J_x, and steering acceleration, η_f, which are used to provide a smooth control of the vehicle.

$$\dot{U} = a_x \tag{7.1}$$

$$\dot{a}_x = J_x \tag{7.2}$$

$$\dot{\delta}_f = \gamma_f \tag{7.3}$$

$$\dot{\gamma}_f = \eta_f \tag{7.4}$$

where U is the longitudinal speed, a_x is the longitudinal acceleration, δ_f is the steering angle, and γ_f is the steering rate.

Two sets of lateral dynamics and yaw dynamics are calculated simultaneously with the parameter values from the two most likely worst-case scenarios, respectively. The first set of dynamic equations uses the parameter values, p_{traj}, of the most likely "stiffest" trajectory scenario, and the subscript traj is used to indicate the correspond-

ing parameters and variables.

$$\dot{V}_{\text{traj}} = \left(F_{y,f,\text{traj}} + F_{y,r,\text{traj}}\right)/M_{t,\text{traj}} - U\omega_{z,\text{traj}} \tag{7.5}$$

$$\dot{\omega}_{z,\text{traj}} = \left(F_{y,f,\text{traj}}L_{f,\text{traj}} - F_{y,r,\text{traj}}L_{r,\text{traj}}\right)/I_{zz,\text{traj}} \tag{7.6}$$

$$F_{y,f,\text{traj}} = \mathcal{P}_f\left(U, V_{\text{traj}}, \omega_{z,\text{traj}}, \delta_f, a_x;\ p_{\text{traj}}\right) \tag{7.7}$$

$$F_{y,r,\text{traj}} = \mathcal{P}_r\left(U, V_{\text{traj}}, \omega_{z,\text{traj}}, a_x;\ p_{\text{traj}}\right) \tag{7.8}$$

where V_{traj} is the lateral speed in the BFCF, $\omega_{z,\text{traj}}$ is the yaw rate, $F_{y,f,\text{traj}}$ and $F_{y,r,\text{traj}}$ are the tire lateral forces generated at the front axle and the rear axle, respectively, which are calculated using the pure-slip Pacejka Magic Formula tire model $\mathcal{P}$ as a function of vehicle state and control variables. The subscripts f, and r are used to represent front axle and rear axle, respectively.

The second set of dynamic equations uses the parameter values, p_{load}, of the most likely smallest vertical load scenario.

$$\dot{V}_{\text{load}} = \left(F_{y,f,\text{load}} + F_{y,r,\text{load}}\right)/M_{t,\text{load}} - U\omega_{z,\text{load}} \tag{7.9}$$

$$\dot{\omega}_{z,\text{load}} = \left(F_{y,f,\text{load}}L_{f,\text{load}} - F_{y,r,\text{load}}L_{r,\text{load}}\right)/I_{zz,\text{load}} \tag{7.10}$$

$$F_{y,f,\text{load}} = \mathcal{P}_f\left(U, V_{\text{load}}, \omega_{z,\text{load}}, \delta_f, a_x;\ p_{\text{load}}\right) \tag{7.11}$$

$$F_{y,r,\text{load}} = \mathcal{P}_r\left(U, V_{\text{load}}, \omega_{z,\text{load}}, a_x;\ p_{\text{load}}\right) \tag{7.12}$$

This set of equations, Equations (9) - (12), is the same as Equations (5) - (8) except that the subscript load is used to indicate that the parameter values are from the most likely smallest vertical load scenario and the variables are calculated using these parameter values.

Thus, two sets of vehicles state predictions from the vehicle dynamics model with two sets of parameter values are obtained. Each set is used to check the satisfaction of only the relevant constraints.

The vehicle heading angle and trajectory profile are calculated using the parameter values and the state predictions of the most likely “stiffest" trajectory scenario, which are then used in the cost function to drive the vehicle to the specified target position and are also used in the position constraints to keep the vehicle within the safe region

established using the LIDAR data.

$$\dot{\psi} = \omega_{z,\text{traj}} \tag{7.13}$$

$$\dot{x} = U\cos(\psi) - (V_{\text{traj}} + L_{f,\text{traj}}\omega_{z,\text{traj}})\sin(\psi) \tag{7.14}$$

$$\dot{y} = U\sin(\psi) + (V_{\text{traj}} + L_{f,\text{traj}}\omega_{z,\text{traj}})\cos(\psi) \tag{7.15}$$

where (x, y) is the vehicle's front center position in global coordinates and ψ is the yaw angle.

The constraints used to prevent wheel lift-off are given by:

$$F_{z,rl,\text{load}} \geq F_{z,\text{threshold}} \tag{7.16}$$

$$F_{z,rr,\text{load}} \geq F_{z,\text{threshold}} \tag{7.17}$$

where $F_{z,\text{threshold}}$ is the minimum vertical load threshold, $F_{z,rl,\text{load}}$ and $F_{z,rr,\text{load}}$ are the rear left and rear right tire vertical loads and are calculated as functions of the parameter values and state predictions of the most likely smallest vertical load scenario.

$$F_{z,rl,\text{load}} = \tfrac{1}{2}\left(F_{z,r0,\text{load}} + \Delta F_{z,x,\text{load}}\right) - \Delta F_{z,yr,\text{load}} \tag{7.18}$$

$$F_{z,rr,\text{load}} = \tfrac{1}{2}\left(F_{z,r0,\text{load}} + \Delta F_{z,x,\text{load}}\right) + \Delta F_{z,yr,\text{load}} \tag{7.19}$$

$$\Delta F_{z,x,\text{load}} \approx K_{z,x,\text{load}}\left(a_x - V_{\text{load}}\omega_{z,\text{load}}\right) \tag{7.20}$$

$$\Delta F_{z,yr,\text{load}} \approx K_{z,yr,\text{load}}\left(F_{y,f,\text{load}} + F_{y,r,\text{load}}\right)/M_{t,\text{load}} \tag{7.21}$$

where $F_{z,r0,\text{load}}$ is the static rear axle load; $\Delta F_{z,x,\text{load}}$ is the load transferred between the front and rear axles due to longitudinal acceleration; $\Delta F_{z,yr,\text{load}}$ is the load transferred between two rear wheels due to lateral acceleration; $K_{z,x,\text{load}}$ is the longitudinal load transfer coefficient; and $K_{z,yr,\text{load}}$ is the lateral load transfer coefficient of the rear axle.

The details of the two constraints introduced above and the rest of the OCP formulation are omitted here because they remain the same as and are discussed thoroughly in Section 6.2.

The second double-worst-case formulation (`MPC4`) also accounts for the less likely worst-case scenarios approximately in a way such that no additional scenarios are introduced in the formulation besides the two most likely worst-case scenarios. As discussed in Section 7.4.2 and demonstrated in Section 7.4.3, for vertical load pre-

diction, the discrepancy between the most likely worst-case scenario and the actual worst-case scenario becomes larger with higher deceleration. Three less likely worst-case scenarios are obtained, in which the sprung mass value is at its upper bound and the tire parameter values are at the opposite bounds besides the two stiffness related parameters compared to Scenario 2. It is possible to consider all three less likely worst-case scenarios in the formulation by further augmenting the state vector and increasing the number of constraints. However, this augmentation would increase the computational load for solving the OCP, which is not desired.

An alternative approach is developed as follows. As shown in Figure 7.13b and 7.13c, it is observed that the results of Scenario 2 and Scenario 4 are very close to each other, and the results of Scenario 5 lie in between the results of Scenario 4 and the results of Scenario 6. Thus, if only one less likely worst-case scenario is to be considered, it would be Scenario 6. Furthermore, Scenario 6 and Scenario 1 are the same besides three mass and inertia related terms, which have less impact on the metric value as shown in the second subfigure of Figure 7.7b. Therefore, Scenario 6 can be approximated by Scenario 1. Thus, to further improve the robustness of the algorithm, `MPC3` can be augmented by considering the following additional constraints for preventing wheel lift-off, which becomes `MPC4` that accounts for the less likely worst-case scenarios approximately.

$$F_{z,rl,\text{traj}} \geq F_{z,\text{threshold}} \tag{7.22}$$

$$F_{z,rr,\text{traj}} \geq F_{z,\text{threshold}} \tag{7.23}$$

where $F_{z,rl,\text{traj}}$ and $F_{z,rr,\text{traj}}$ are the rear left and rear right tire vertical loads and are calculated as functions of the parameter values and state predictions of the most likely "stiffest" trajectory scenario. They are calculated used the following set of equations, which is the same as Equations (18) - (21) except that a different set of parameter values is used.

$$F_{z,rl,\text{traj}} = \tfrac{1}{2}\left(F_{z,r0,\text{traj}} + \Delta F_{z,x,\text{traj}}\right) - \Delta F_{z,yr,\text{traj}} \tag{7.24}$$

$$F_{z,rr,\text{traj}} = \tfrac{1}{2}\left(F_{z,r0,\text{traj}} + \Delta F_{z,x,\text{traj}}\right) + \Delta F_{z,yr,\text{traj}} \tag{7.25}$$

$$\Delta F_{z,x,\text{traj}} \approx K_{z,x,\text{traj}}\left(a_x - V_{\text{traj}}\omega_{z,\text{traj}}\right) \tag{7.26}$$

$$\Delta F_{z,yr,\text{traj}} \approx K_{z,yr,\text{traj}}\left(F_{y,f,\text{traj}} + F_{y,r,\text{traj}}\right)/M_{t,\text{traj}} \tag{7.27}$$

Table 7.5 summarizes the number of variables, number of equality constraints, and

Table 7.5: Size of the OCP in the MPC formulations.

	MPC0/MPC1/MPC2
Number of variables	$N_{var} = N_{ph}\left[9\left(N_{no}+1\right)+2N_{no}+3\right]$
Number of equality constraints	$N_{con,eq} = N_{ph}\left[9N_{no}+1\right]$
Number of inequality constraints	$N_{con,neq} = \sum_{n=1}^{N_{ph}} N_{pt}^{n} N_{no} + N_{ev} + N_{ph}$
	MPC3
Number of variables	$N_{var} + 2N_{ph}\left(N_{no}+1\right)$
Number of equality constraints	$N_{con,eq} + 2N_{ph}N_{no}$
Number of inequality constraints	$N_{con,neq}$
	MPC4
Number of variables	$N_{var} + 2N_{ph}\left(N_{no}+1\right)$
Number of equality constraints	$N_{con,eq} + 2N_{ph}N_{no}$
Number of inequality constraints	$N_{con,neq} + \sum_{n=1}^{N_{ph}} 2N_{no}$

number of inequality constraints, in the transcribed NLPs of the original formulation with different scenarios (`MPC0`, `MPC1`, `MPC1`) and the two double-worst-case formulations (`MPC3` and `MPC4`) in terms of number of phases of the OCP, N_{ph}, number of nodes for discretization, N_{no}, number of path constraints, N_{pt}, and number of event constraints, N_{ev}. Thus, `MPC4` is the most computationally expensive formulation.

7.6 Simulation Results and Discussion

Two sets of simulations are conducted with all five MPC formulations. The first set uses the 50 LHD scenarios, whereas the second set uses the seven scenarios listed in Table 7.3 and Table 7.4, which are the nominal scenario and the six worst-case scenarios.

7.6.1 Latin Hypercube Design Scenarios

Fig. 7.14 shows the results with the LHD scenarios using the four MPC formulations that take into account the worst-case scenarios. The number of scenarios with safety issues of different types and the total number of failure scenarios are summarized in Table 7.6. Although a single simulation may have multiple safety issues, this only counts as one failure case. Therefore, the total number of failure scenarios is not the summation of the number of safety issues of all type in Table 7.6.

It is concluded that `MPC1` improves the robustness to parametric uncertainty

slightly compared with MPC0. However, MPC2 worsens the robustness significantly. Although none of the scenarios has wheel lift-off and the number of violations of the minimum-vertical-load threshold decreases, the number of violations of the minimum-distance-to-target threshold increases dramatically and consequently leads to task incompletion in terms of not satisfying the final heading angle requirement. Both MPC3 and MPC4 increase the robustness significantly, because all the results are dynamically safe and meet the task requirements. It is worth emphasizing that although the robustness is improved, it is not guaranteed for all possible scenarios with either MPC3 or MPC4, because only most likely worst-case scenarios are considered in the formulations.

It is expected that improving robustness to parametric uncertainty would sacrifice task completion performance. To evaluate this trade-off, four metrics are used to evaluate the task completion performance of the obstacle avoidance algorithm: time-to-target, average speed, acceleration effort, and steering effort. Besides the average speed, for which a larger metric value is preferred, smaller values are desired for the other three metrics. Only when the scenarios do not contain failure scenarios with all the considered controllers, are the comparisons of the metrics meaningful. Thus, only the 38 scenarios with no safety issues using MPC0 are considered, because besides MPC2, the other three controllers also do not lead to safety issues with these 38 scenarios. Table 7.7 summarizes the average metric values for each MPC formulation besides MPC2.

Comparing the results of using MPC1 to that of using MPC0, the first three performance metrics are slightly worse and the last one is slightly better. With MPC3 and MPC4, the first three performance metrics are moderately worse compared with those of both MPC0 and MPC1. Thus, it is concluded that there is a trade-off between robustness and task completion performance. Improving the robustness by considering the worst-case scenarios degrades the task completion performance. To be more specific, the average speed with MPC3 and MPC4 is about 11% slower than that with MPC0 because of the conservativeness of the formulation. The time-to-target with MPC3 and MPC4 is about 11% longer than that with MPC0 due to slower speed and/or longer route. Also, about 50% more acceleration effort is used with MPC3 and MPC4 than with MPC0. The steering effort required is almost the same with these three controllers. In summary, MPC3 and MPC4 both significantly improve the robustness of the obstacle avoidance algorithm to parametric uncertainty, however, at the cost of task completion performance degradation as discussed above.

The results generated using MPC2 are the worst among all the considered formu-

Table 7.6: Number of observed safety issues and failures scenarios with the LHD scenarios

	MPC0	**MPC1**	**MPC2**	**MPC3**	**MPC4**
Collision	0	0	0	0	0
Wheel lift-off	2	0	0	0	0
Soft collision [1]	0	0	31	0	0
Soft wheel lift-off [2]	10	9	4	0	0
Task incompletion	1	0	39	0	0
Total number of failure scenarios	12	9	40	0	0

[1] Soft collision indicates violation of minimum-distance-to-obstacle threshold.

[2] Soft wheel lift-off indicates violation of minimum-vertical-load threshold.

Table 7.7: Task completion performance evaluation metrics with the 38 LHD scenarios that are successful with `MPC0`.

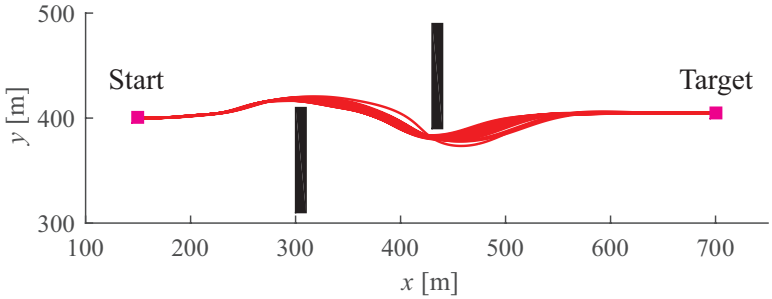
	MPC0	**MPC1**	**MPC3**	**MPC4**
Time-to-target [s]	27.1	27.7	30.2	30.2
Average speed [m/s]	20.8	20.1	18.6	18.6
Acceleration effort	11.3	12.8	16.7	16.8
Steering effort	29.4	27.1	29.0	29.3

lations in every aspect. The reason is that the trajectory generated using the most likely smallest vertical load scenario is significantly "flexible" as shown in Fig. 7.9 and 7.11. Thus, the steering command generated with `MPC2` is not large enough for almost all the scenarios to steer the AGV away from the obstacle. The AGV moves very close to obstacle before a sharp turn is made by hard deceleration, which leads to large control effort, significant detour, and task incompletion.

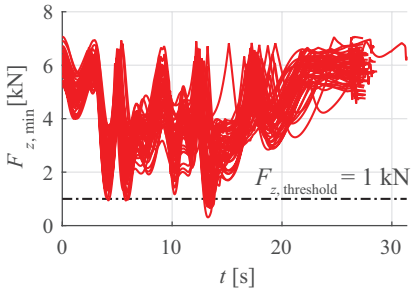
7.6.2 Worst-Case Scenarios

Fig. 7.15 shows the closed loop simulation results with the nominal scenario and the six worst-case scenarios using `MPC0`. Besides the simulation with the nominal scenario, the other six simulations lead to safety issues. Scenarios 2, 4, and 6 result in wheel lift-off. Results for scenarios 1 and 5 are with violations of the minimum-vertical-load threshold. Scenarios 1 and 3 lead to violations of minimum-distance-to-target threshold and fail to complete task. Thus, `MPC0`, which is the original formulation with nominal parameter values, is not robust to the worst-case scenarios.

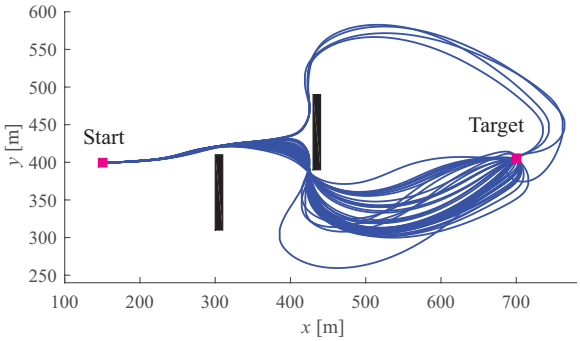
Fig. 7.16 shows the results with the other four MPC formulations. The number of scenarios with safety issues of different types and the total number of failure sce-



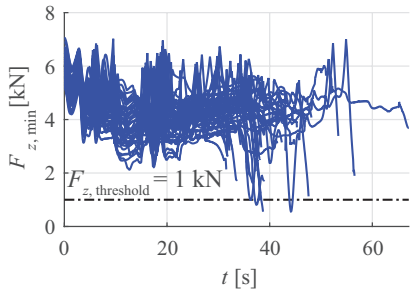
(a) Trajectory profiles with `MPC1`.



(b) Vertical load profiles with `MPC1`.

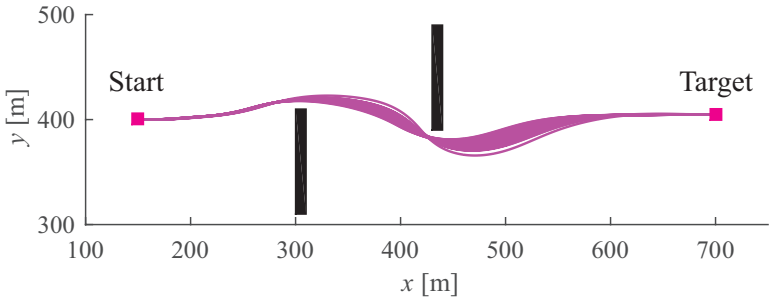


(c) Trajectory profiles with `MPC2`.

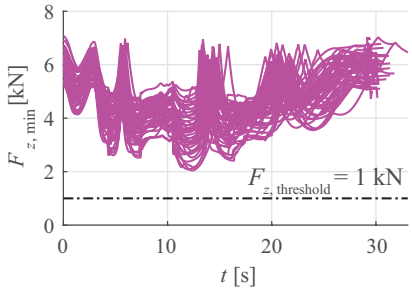


(d) Vertical load profiles with `MPC2`.

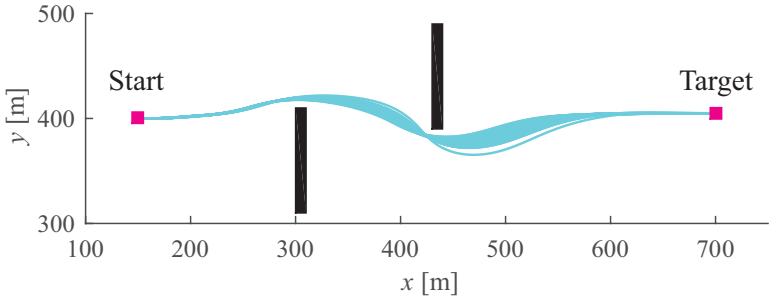
Figure 7.14: Close-loop simulation results with the 50 LHD scenarios using different controllers.



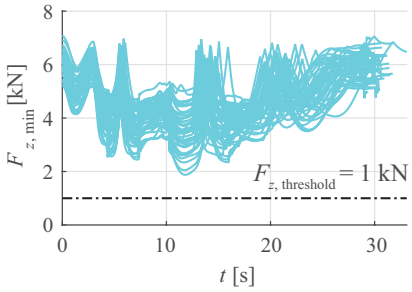
(e) Trajectory profiles with `MPC3`.



(f) Vertical load profiles with `MPC3`.



(g) Trajectory profiles with `MPC4`.



(h) Vertical load profiles with `MPC4`.

Figure 7.14: (continued) Close-loop simulation results with the 50 LHD scenarios using different controllers.

narios are summarized in Table 7.8. It is shown that these four controllers are more effective than the MPC0, although different levels of success are observed. Only when MPC4 is used, which is the most complex formulation out of the five, there are no safety issues or task completion issues in the simulation results with worst-case scenarios. With MPC1, scenario 2 leads to wheel lift-off, and results for scenarios 4 and 5 have violations of the minimum-vertical load threshold. With MPC2, only results for scenarios 2 and 4 are without safety issues or task completion issues. With MPC3, scenarios 1 and 6 result in violations of the minimum-vertical load threshold. Thus, MPC4 is the formulation that is robust to the parametric uncertainty in all considered scenarios among all the five considered formulations when the worst-case scenarios are considered. It is also concluded that the approach for approximately accounting for the less likely worst-case scenarios of the smallest vertical load type is effective.

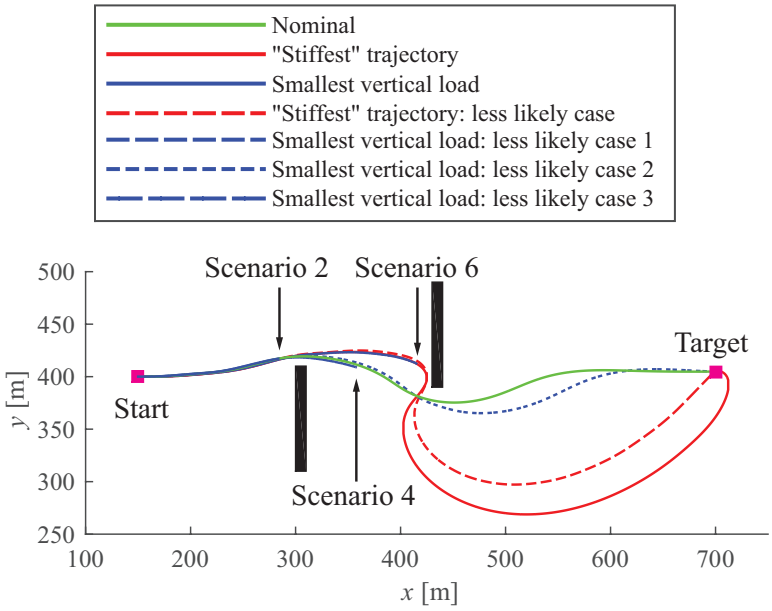
Finally, this set of results is not used to study the trade-off between the robustness and the task completion performance because the benchmark controller (MPC0) fails all the worst-case scenarios. In addition, if the task completion performance of MPC1, MPC3, and MPC4 are to be compared, only scenario 0 and scenario 3 can be used, because the three controllers do not lead to safety issues only with these two scenarios among the seven considered scenarios. However, these two scenarios are too few in number to generate statistically meaningful results.

In conclusion, results from simulations with stratified random scenarios and worst-case scenarios show that only the second double-worst-case formulation (MPC4) renders the algorithm robust to all uncertainty realizations tested. From the comparison of performance evaluation metrics of simulations with stratified random scenarios, it is also concluded that the improvement of robustness degrades performance, but arguably, to an acceptable level.

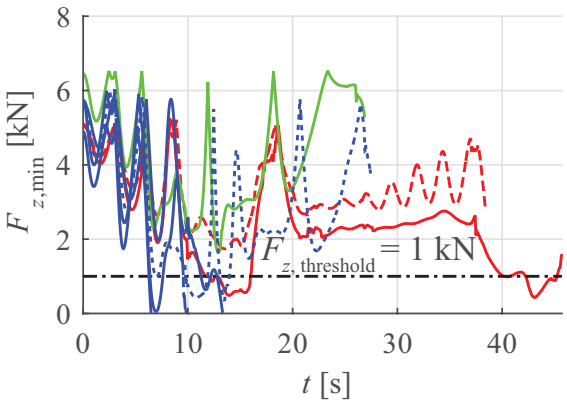
It is expected that the conclusions are generalizable to other obstacle fields, because the most likely worst-case scenarios are obtained independent of the obstacle configurations. The two sets of simulations discussed herein have also been repeated with another even more challenging obstacle field as discussed in Section 6.3.

7.7 Conclusion

Chapter 6 focused on large-size, high-speed autonomous ground vehicles within unknown and unstructured environments and developed a nonlinear model predictive control based obstacle avoidance algorithm that can simultaneously optimize both the reference longitudinal speed and the steering control command to navigate the vehicle

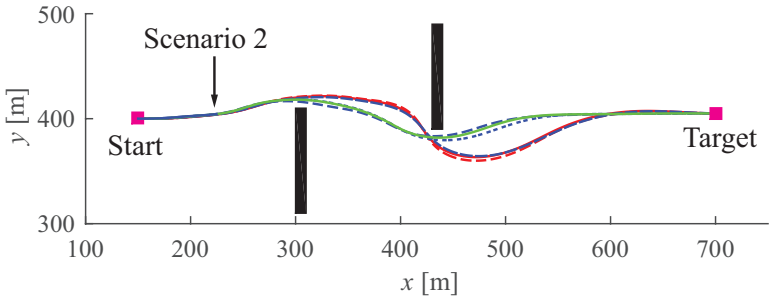


(a) Trajectory profiles with `MPC0`.

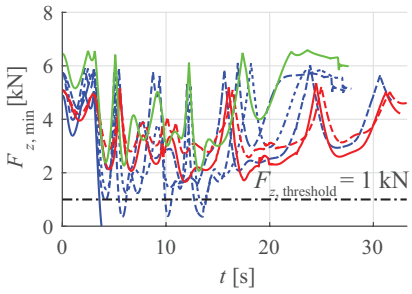


(b) Vertical load profiles with `MPC0`.

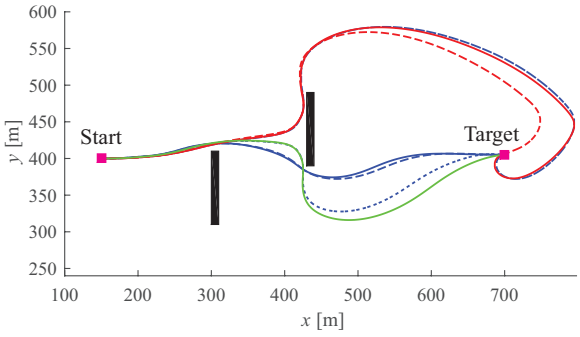
Figure 7.15: Closed-loop simulation results with the nominal scenario and the worst-case scenarios using `MPC0`.



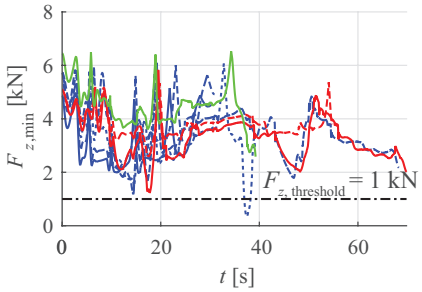
(a) Trajectory profiles with `MPC1`.



(b) Vertical load profiles with `MPC1`.

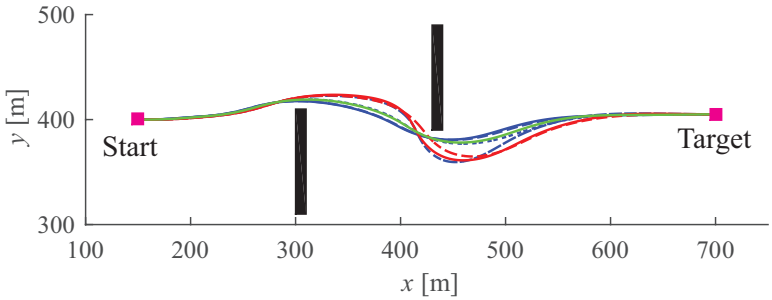


(c) Trajectory profiles with `MPC2`.

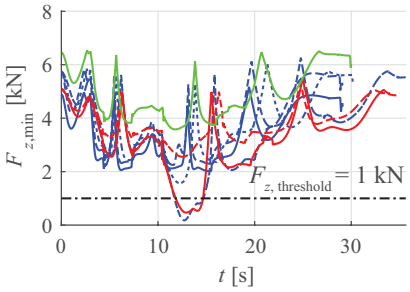


(d) Vertical load profiles with `MPC2`.

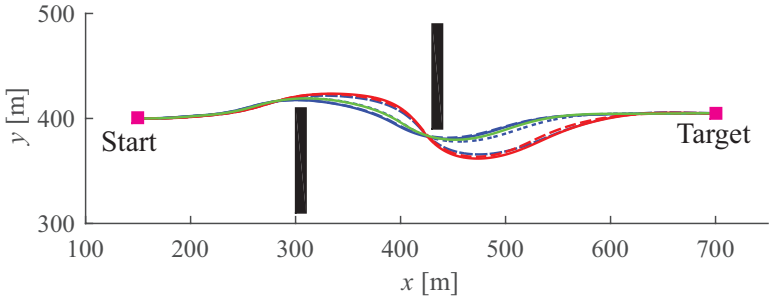
Figure 7.16: Closed-loop simulation results with the nominal scenario and the worst-case scenarios using different controllers.



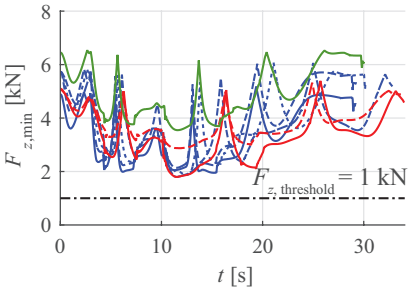
(e) Trajectory profiles with `MPC3`.



(f) Vertical load profiles with `MPC3`.



(g) Trajectory profiles with `MPC4`.



(h) Vertical load profiles with `MPC4`.

Figure 7.16: (continued) Closed-loop simulation results with the nominal scenario and the worst-case scenarios using different controllers.

Table 7.8: Number of observed safety issues and failures scenarios with the nominal scenario and the worst-case scenarios

	MPC0	**MPC1**	**MPC2**	**MPC3**	**MPC4**
Collision	0	0	0	0	0
Wheel lift-off	3	1	0	0	0
Soft collision [1]	2	0	5	0	0
Soft wheel lift-off [2]	2	2	1	2	0
Task incompletion	2	0	3	0	0
Total number of failure scenarios	6	3	5	2	0

[1] Soft collision indicates violation of minimum-distance-to-obstacle threshold.

[2] Soft wheel lift-off indicates violation of minimum-vertical-load threshold.

as quickly as possible to the target position while taking into account its dynamical safety.

In this chapter, the robustness of the algorithm with nominal parameter values to parametric uncertainty is first evaluated. Because there are twelve parameters with uncertainty, Latin Hypercube design is used to generate a tractable collection of parameter values from the multidimensional distribution. It is demonstrated that using the nominal parameter values in the algorithm leads to safety issues for 24% of the time with the considered parametric uncertainty distributions. Thus, improvement of robustness is necessary. The approach developed in this chapter uses a concept similar to the scenario-based approach, which makes the constraints in the MPC problem robustly safe in all of the considered scenarios. However, instead of using hundreds of randomly generated scenarios, only two scenarios are considered, which are the two most likely worst-case scenarios corresponding to the two types of identified unsafe scenarios (stiffest trajectory and minimum vertical load). In addition, instead of applying all the scenarios to check the satisfaction of all the constraints, each scenario is only used to check the relevant constraint. This helps to reduce the number of constraints in the extended formulation for robustness. In particular, two double-worst-case formulations are developed and both of them account for the robustness of the two safety requirements for obstacle avoidance simultaneously: collision-free and no-wheel-lift-off. The first one considers only the two most likely worst-case scenarios and each of them is used to check only the relevant constraints. In the second formulation the stiffest trajectory scenario is also used to check the vertical load constraints to account for the less likely worst-case scenarios in an approximate way without the need to introduce another scenario to the formulation. The task completion performance of the proposed double-worst-case formulations are compared to the original

formulation with different sets of scenarios. Results from simulations with stratified random samples and worst-case scenarios show that the second double-worst-case formulation renders the algorithm robust to all uncertainty realizations tested. The trade-off between the robustness and task completion performance is discussed based on the results with stratified random samples. It is concluded that the proposed double-worst-case formulations can improve the robustness of the algorithm at the cost of some acceptable reduction in task completion performance.

The approach presented in this chapter is a passive one. An active approach can be used in complement that reduces the uncertainty through an online nonlinear adaptive estimator to obtain more accurate estimations of the parameter values based on the difference between the model predictions and actual data collected from the vehicle. To this end, existing adaptive observer schemes such as the Extended Kalman Filter can be leveraged [129]. Future research could include adding such capability into the closed-loop system to reduce the task completion performance degradation due to conservativeness caused by large uncertainty ranges.

Besides parametric uncertainty and state measurement uncertainty, there are other sources of uncertainty. For example, one source of uncertainty is the external disturbances due to wind and other factors. These sources might also cause problems and need to be considered [121]. The robustness of the algorithm subject to all types of uncertainties is another topic requiring additional research.

CHAPTER 8

Preliminary Experimental Validation

Preliminary experimental tests are performed to validate the obstacle avoidance capability of the developed constant-speed algorithm in a limited way. These tests are conducted with the support of Quantum Signal, LLC.

8.1 Platform and Software

The vehicle platform is a four-wheeled, Kawasaki 4010 Mule utility vehicle [130] augmented with drive-by-wire capability. The maximum speed of the vehicle is about 10 m/s. The steering, throttle, and brake commands of the vehicle can be specified by a computer, which are then executed by the actuators. Fig. 8.1 is a picture of the frontal view of the vehicle. The vehicle parameter values are given in Table 8.1. A high-performance GPS-aided Inertial Navigation System (GPS/INS) [131] and an eight-layer scanning LIDAR sensor [132] are also installed on the platform. An onboard Linux PC processes the sensing data, runs the obstacle avoidance algorithm, and sends the motor commands to the vehicle.

Table 8.1: Parameter values of the vehicle platform

Parameter	Symbol	Value	Unit
Vehicle Mass	M	842	kg
Yaw moment-of-inertia	I_{zz}	628.7	kg-m^2
Front axle to CoG distance	L_f	1.01	m
Rear axle to CoG distance	L_r	0.86	m
Corning Stiffness	C_α	2542	N/deg

Figure 8.1: Kawasaki Mule

The constant-speed obstacle avoidance algorithm is used, in which only the steering angle sequence is optimized. The longitudinal speed of the vehicle is maintained at a specified constant value using a separate controller. In addition, in the obstacle avoidance algorithm, a linear tire model is used instead of the nonlinear Pacejka Magic Formula tire model because of the difficulty in obtaining parameter values. More discussions about the selection of tire model are included in Section 8.3.1.

Fig. 8.2 shows the schematic of the computer program running on a Linux PC. Five modules are evaluated in parallel, those are, INS interface, LIDAR interface, speed controller, obstacle avoidance algorithm, and DRIVER system. The INS interface connects to the GPS/INS sensor, parses the sensor outputs, and interprets the parsed data into desired vehicle state information, which includes the position and heading angle values in the global coordinates, the longitudinal and lateral speeds, and the yaw rate. Similarly, the LIDAR interface connects to the LIDAR sensor, parses the sensor outputs, and sends the obstacle information to the obstacle avoidance algorithm. The longitudinal speed of the vehicle is controlled to be a specified constant value using a PID controller. The speed controller generates the instantaneous throttle and brake commands. The obstacle avoidance algorithm takes all the sensor measurements as inputs and generates the desired steering angle sequence over the specified prediction horizon. Finally, the DRIVER system [133] translates the throttle, brake, and steering commands into corresponding motor commands, which control the maneuver of the vehicle. These modules run at different sampling frequencies, which are listed in

Table 8.2.

The obstacle avoidance algorithm module is implemented in MATLAB. The framework and the rest of the four modules of the computer program are implemented in C++. Thus, MATLAB engine API for C/C++ [134] is used to call the obstacle avoidance algorithm within the C++ framework. The computer program is run on a Linux PC with a 2.7 GHz i7 2620M processor and 8 GB RAM.

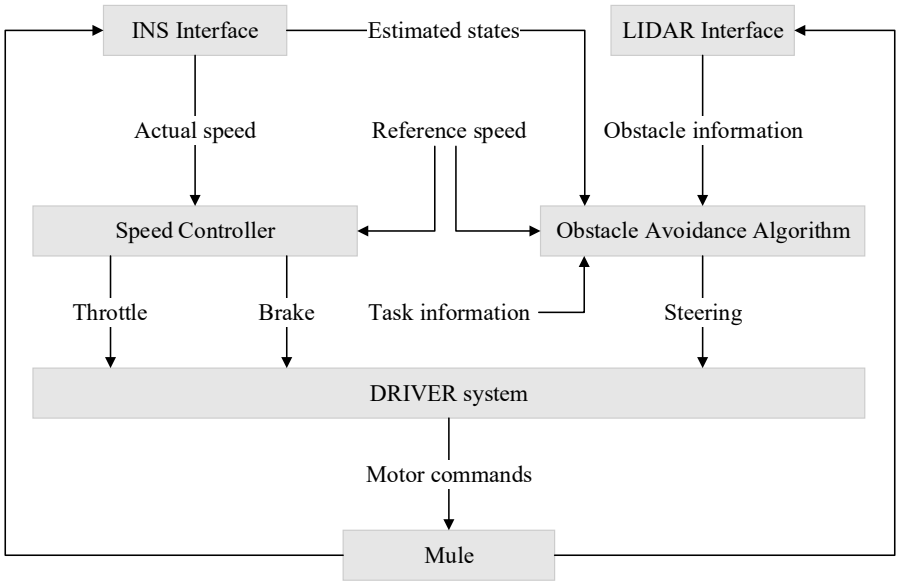


Figure 8.2: Schematic of the computer program running on-board the Kawasaki Mule

Table 8.2: Sampling frequency of different modules

Module	Sampling Frequency [Hz]
INS interface	8
LIDAR interface	25
Speed controller	100
Obstacle avoidance algorithm	1 or 2
DRIVER system	100

8.2 Results

The tests were conducted on a rough gravel ground with the size of about 60 × 50 m as shown in Fig. 8.3 [135]. In these tests, the speed of the vehicle was maintained at 3 m/s (6.7 mph) or 4 m/s (8.9 mph). Thus, the vehicle did not operate at its dynamical limit. The results are only used to demonstrate the obstacle avoidance capability of the algorithm in a limited way.

Figure 8.3: Satellite view of the test field.

8.2.1 Test Case: Steer-to-Target

The steer-to-target test is first performed. No obstacles are placed in the environment. The algorithm is used to drive the vehicle to move towards the specified target position.

Fig. 8.4 shows one set of experimental results. The vehicle initial position is at (0, 0) m with the initial heading angle in the positive-y direction. The target position is located at (50, 100) m. As shown in Fig. 8.4a, the algorithm is capable of steering the vehicle to change its heading direction and move towards the specified target position. The MPC algorithm updates the optimal trajectory and optimal steering angle sequence over a 5 s prediction horizon every 0.5 s. Thus, the execution horizon is 0.5 s. In Fig. 8.4b, the planned trajectories of all MPC update steps are plotted on top of each other. At each time step, starting from the estimated state information from the INS sensor, an optimal trajectory is predicted to move towards the target position. It is demonstrated that the vehicle follows the predicted path well in this test because the predictions are overlapped. Fig. 8.4c is the vehicle longitudinal speed profile and Fig. 8.4d shows the histories of the throttle and brake commands. The desired speed is 3 m/s. After a smooth transient period, the speed controller is capable of maintaining the speed at the desired value with a maximum error of 0.4 m/s in this test. No braking is used during the process. In Fig. 8.4e, the red-dashed line is the steering angle sequence generated by the MPC algorithm and the blue-solid line is the actual steering angle profile. The difference and significant oscillation are

likely due to the disturbances from the rough gravel terrain. More discussions about this problem can be found in Section 8.3.2. Finally, the lateral speed and yaw rate profiles are shown in Fig. 8.4f, which are the two state variables of the vehicle model used in the MPC algorithm.

The test was repeated and similar results were generated. In addition, a different target location was used and the tests were also successful completed.

In summary, the performance of the speed controller is at an acceptable level. With the state information from the INS sensor, the obstacle avoidance algorithm generates a smooth steering command. Albeit the fact that the actual steering sequence is rough and oscillatory because of the disturbances from the rough test ground, the vehicle can be successfully navigated to move towards the specified target position.

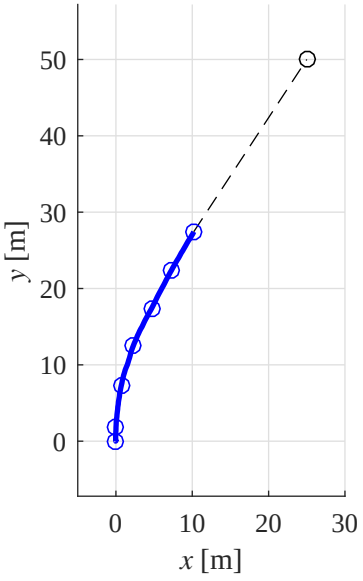
8.2.2 Test Case: Obstacle Avoidance

The obstacle avoidance tests are then performed. A plastic barrel is first used as the obstacle in the field as shown in Fig. 8.5.

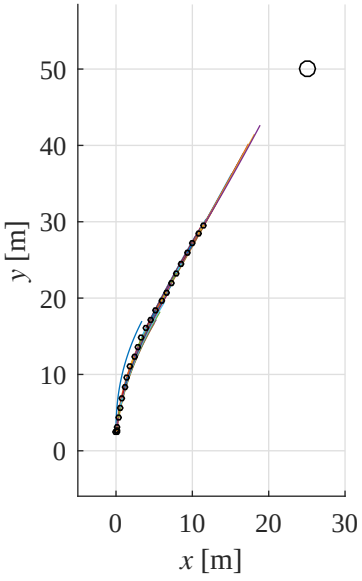
Fig. 8.6 shows one set of experimental results. The vehicle initial position is at (0, 0) m with the initial heading angle along the positive-x direction. The target position is located at (150, 0) m. The reference longitudinal speed is 3 m/s. The prediction horizon of the MPC algorithm is 5 s and the execution horizon is 1 s. The effective LIDAR detection range used is 10 m although the sensor is capable of detecting obstacles up to 100 m, which means that the algorithm only reacts to obstacles within 10 m range relative to the vehicle current position.

As shown in Fig. 8.6a, the algorithm is capable of navigating the vehicle to avoid the sensed obstacle and to move towards the target position. Fig. 8.6b shows the optimal trajectories of all MPC update steps. It is observed that although the task is completed successfully, the algorithm did not work perfectly as expected. Firstly, for the optimal control problem originating at around (11.5, 0) m, the computation time consumed was longer than the execution horizon. Steering command from the previous step was executed as a fail-safe approach. The discussion on the computational speed is presented in detail in Section 8.3.3. Secondly, it is also observed that the speed tracking performance is worse when the vehicle steers as shown in Fig. 8.6c. Section 8.3.5 includes more discussions on the issue of speed controller. More discussions about the use of LIDAR data can be found in Section 8.3.4.

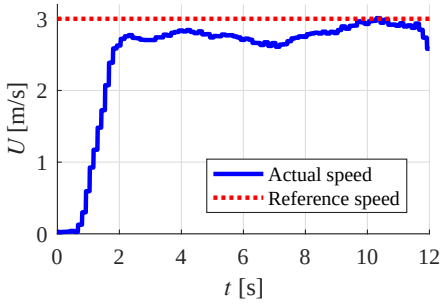
Fig. 8.7 is another set of experimental results, which has the same settings as the previous test. The only difference is that the relative initial position of the vehicle



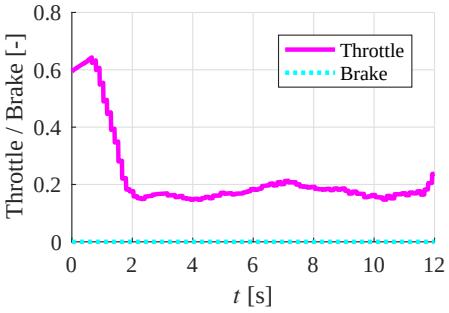
(a) Trajectory



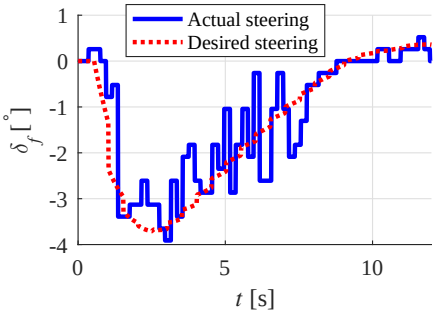
(b) Predictions



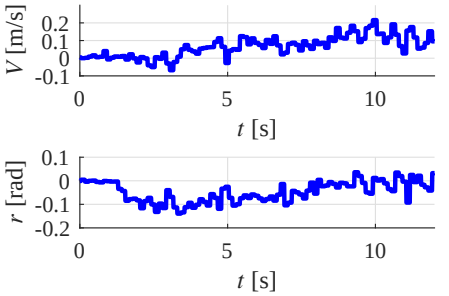
(c) Longitudinal speed



(d) Throttle / brake commands



(e) Steering angle



(f) Lateral speed & yaw rate

Figure 8.4: Steer-to-target test results.

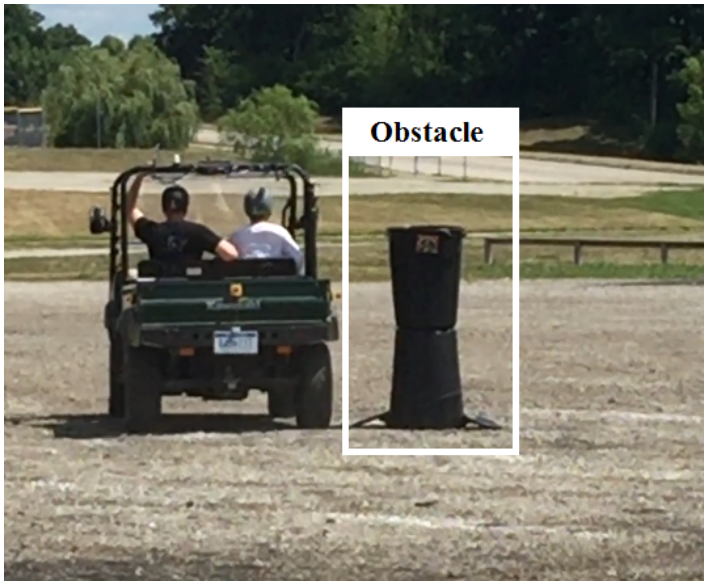
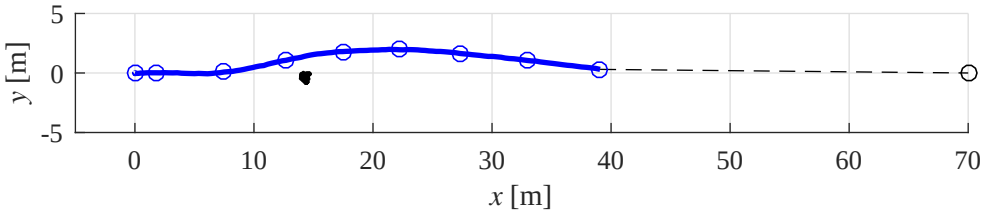


Figure 8.5: Illustration of the obstacle used in the test

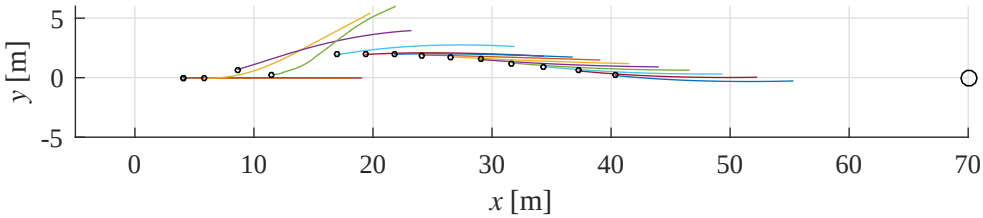
with respect to the obstacle was different. The vehicle also avoided the obstacle successfully. Fig. 8.8 shows a few snapshots of the video recorded during the test. The vehicle turned to the left to avoid the obstacle in front. Fig. 8.7a is the trajectory profile. It is observed that the vehicle did not move towards the target position after the avoidance maneuver. However, the planned trajectories from the MPC algorithm all tried to navigate the vehicle to the target position as shown in Fig. 8.7b. This discrepancy is likely due to the terrain, which is rough and non-flat. However, it is assumed that the terrain is uniform and flat in the algorithm. Hence, the vehicle would not follow the planned trajectory exactly. Moreover, the algorithm is updated only at frequency of 1 Hz, which seems to be not sufficient for this test case. The detailed discussions on the terrain effect and computation speed can be found in Section 8.3.2 and 8.3.3, respectively.

The third set of experimental results with the plastic barrel is shown in Fig. 8.9. The settings are still the same as the previous tests besides the target position and the execution horizon. The target position is changed from (0, 150) m to (50, 100) m. The execution horizon is reduced to 0.5 s from 1 s. In this test, the vehicle first changed its heading direction to move towards the target position. The obstacle was placed on its way to the target position. The vehicle avoided the obstacle by turning to the left. Thus, the task is successfully completed.

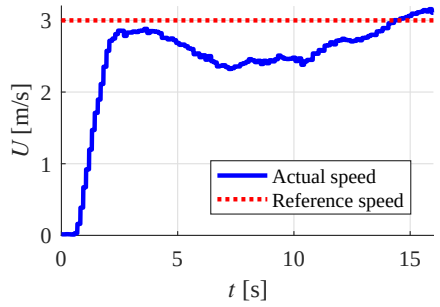
A larger obstacle as shown in Fig. 8.10 was also used. One set of experimental results with this obstacle is given in Fig. 8.11. In this test, the reference longitudinal



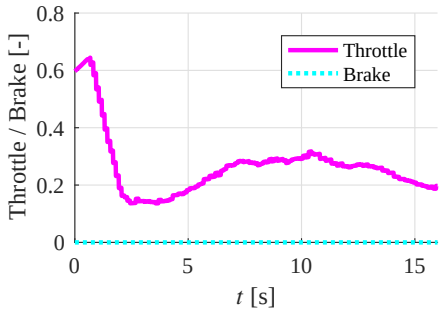
(a) Trajectory



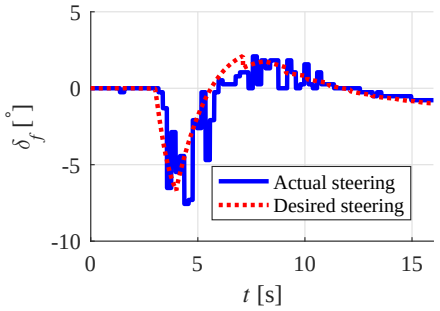
(b) Predictions



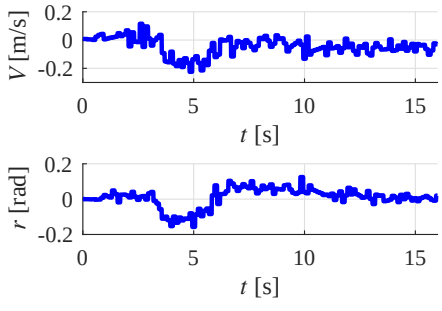
(c) Longitudinal speed



(d) Throttle / brake commands

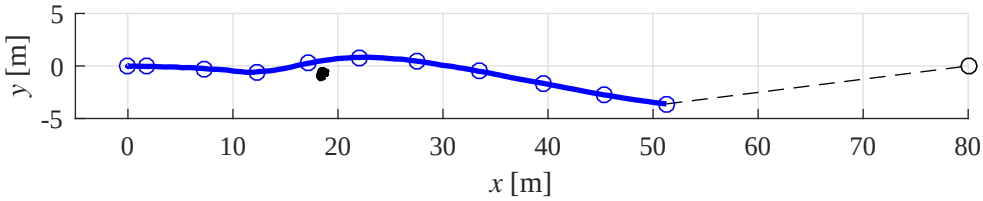


(e) Steering angle

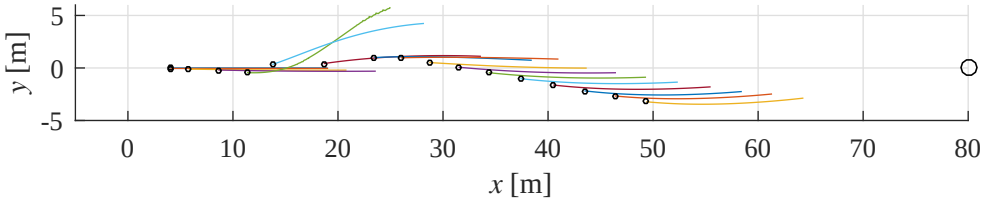


(f) Lateral speed & yaw rate

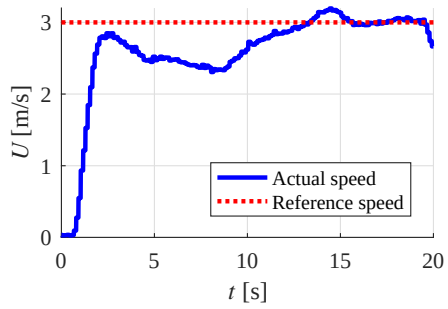
Figure 8.6: Obstacle avoidance test results.



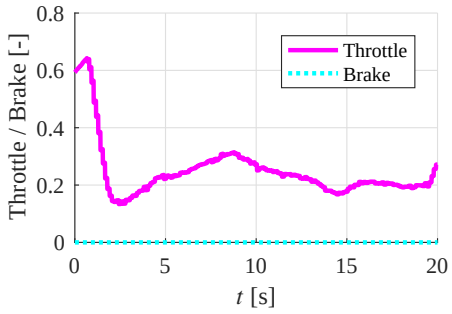
(a) Trajectory



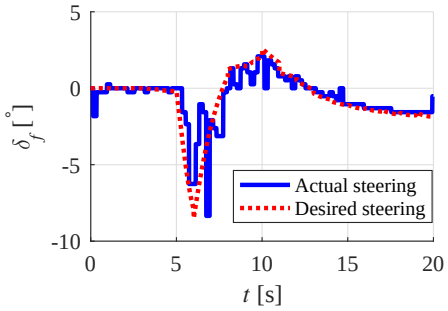
(b) Predictions



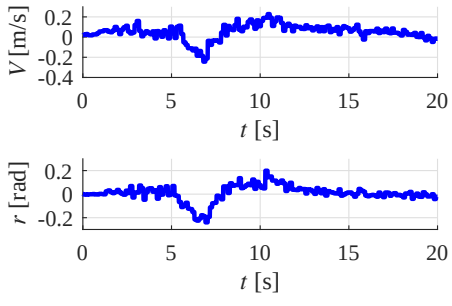
(c) Longitudinal speed



(d) Throttle / brake commands



(e) Steering angle



(f) Lateral speed & yaw rate

Figure 8.7: Obstacle avoidance test results.

(a) $t = 4$ s

(b) $t = 6$ s

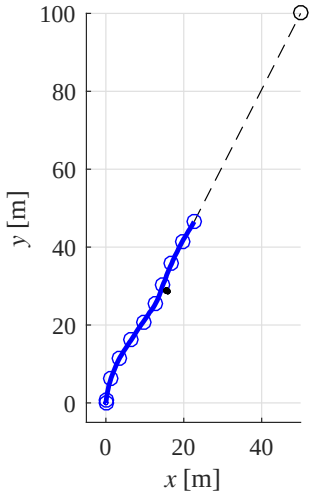
(c) $t = 8$ s

(d) $t = 10$ s

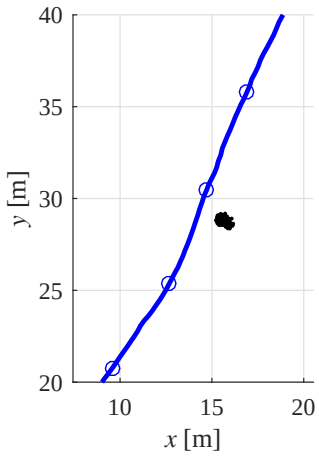
(e) $t = 12$ s

(f) $t = 14$ s

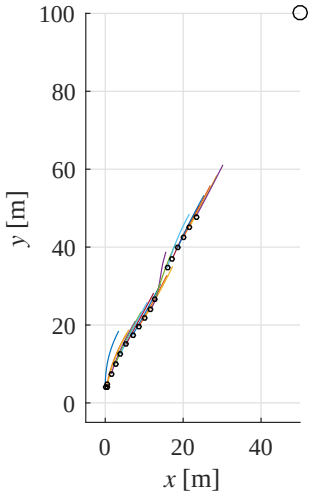
Figure 8.8: Snapshots of video.



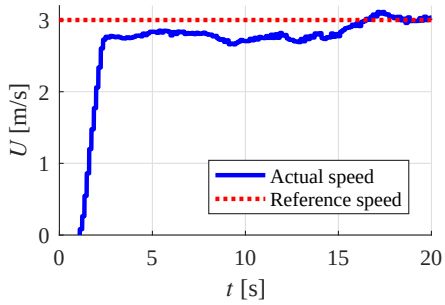
(a) Trajectory



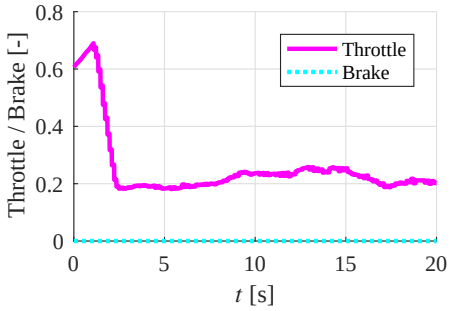
(b) Trajectory (zoom-in view)



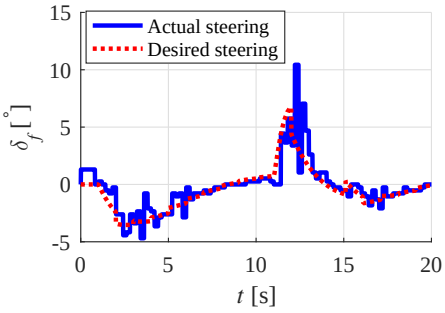
(c) Predictions



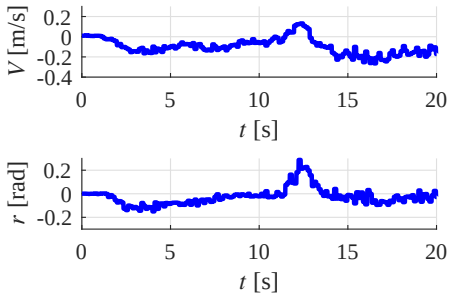
(d) Longitudinal speed



(e) Throttle / brake commands



(f) Steering angle



(g) Lateral speed & yaw rate

Figure 8.9: Steer-to-target test results.

Figure 8.10: Illustration of the obstacle used in the test

speed is set to be 4 m/s. The prediction horizon of the MPC algorithm is still 5 s, but the execution horizon is reduced to 0.5 s. The task is also successfully completed. Thus, it is shown that the algorithm is capable of handling a larger obstacle and a higher speed.

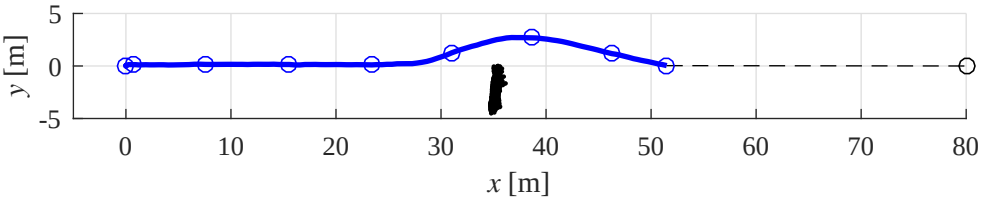
Repeated tests with the single plastic barrel and the larger obstacle were performed. Similar observations are obtained as discussed above. It is concluded that the constant-speed obstacle avoidance is capable of avoiding the detected obstacle when the AGV moves at a speed of 3 or 4 m/s.

8.3 Current Issues and Future Work

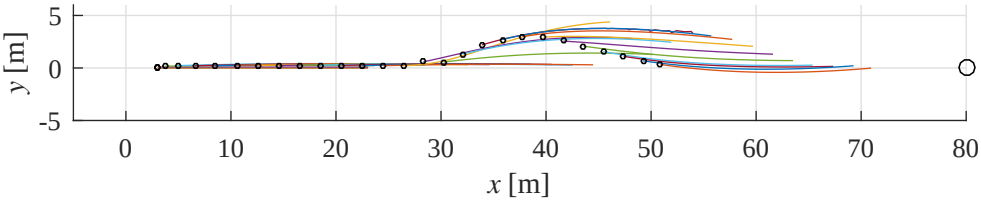
Through conducting the tests and analyzing the results, some issues of the obstacle avoidance algorithm itself and other issues related to the program used in the experimental validation are identified.

8.3.1 Tire Modeling

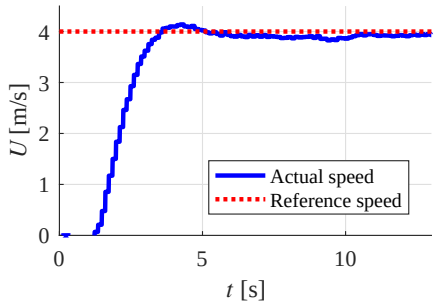
Due to time and resource limitations, this experimental validation excludes the characterization of the tire-terrain interaction relationships. Thus, parameter values of the nonlinear Pacejka Magic Formula tire model are not available. Instead, a simple linear tire model is used in the MPC model. Because the vehicle does not operate near its dynamical limit, a linear tire model could be sufficient. However, this statement



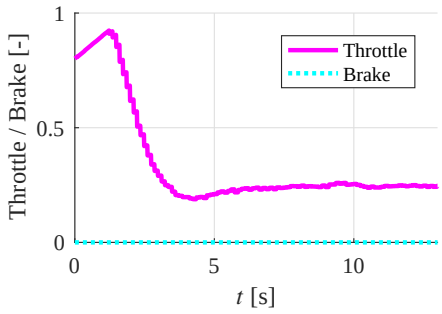
(a) Trajectory



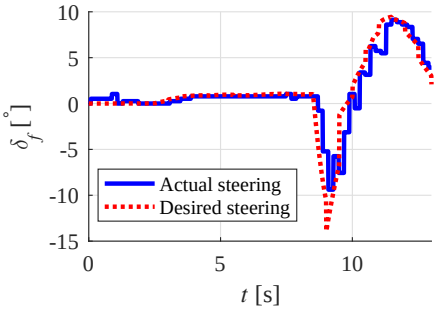
(b) Predictions



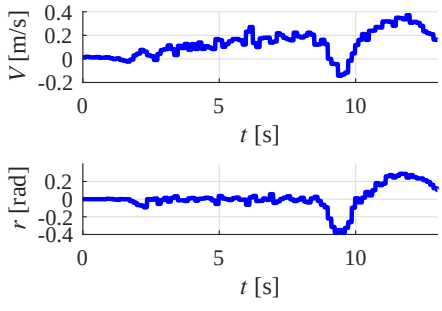
(c) Longitudinal speed



(d) Throttle / brake commands



(e) Steering angle



(f) Lateral speed & yaw rate

Figure 8.11: Obstacle avoidance test results.

is not experimentally validated. In addition, the corning stiffness used in the linear model is only a rough estimation.

For future work, characterization of the tire-terrain interaction in the particular experimental setup is needed in order to use the nonlinear tire model. More open-loop tests are also necessary to thoroughly validate the accuracy of the model prediction.

8.3.2 Terrain Effect

The algorithm is developed for an AGV on a uniform and flat terrain. However, the tests are conducted on a rough gravel ground. Three major issues are introduced by relaxing this assumption. Firstly, the MPC model uses a constant corning stiffness, whereas in reality, the parameter value changes as the vehicle moves. In other words, there exists parametric uncertainty. Secondly, the slope of the terrain has noticeable effects on the vehicle's responses. However, it is not captured in the MPC model. Thirdly, the steering command cannot be executed accurately on a gravel terrain. Large deviations and oscillatory behaviors are observed, which are likely to increase as the speed increases.

To address these issues, future work is needed. Regarding the first problem, the developed algorithm for handling parametric uncertainty can be incorporated. In addition, an online adaptive estimator can be used to obtain more accurate estimation of the parameter values. To address the second problem, assuming that the slope of the terrain can be estimated, the MPC model can be augmented to account for the slope information. For the third problem, the effect of the terrain roughness on the steering command execution needs to be thoroughly studied. In this way, the input uncertainty could be characterized. The effect of the input uncertainty on the performance of the algorithm also needs to be analyzed. If the effect is significant, the algorithm needs to be updated to incorporate this effect.

8.3.3 Computational Speed

The robustness of the MPC algorithm increases with a higher frequency. However, in this work, the execution horizon used is 0.5 s or 1.0 s due to the limitation of the computational speed, which might not be small enough. With longer execution horizon, the difference between the predicted and actual trajectories can become large before a correction effort is applied. In addition, if the optimal control problem is not solved within the desired interval, a fail-safe command need to be applied.

To improve the computational performance of the algorithm, three strategies can

be pursued. Firstly, the most straightforward approach is to use a more powerful computing device. Secondly, a MATLAB-free version of the algorithm would also help. Although mex functions are used in MATLAB to improve the computational performance, the use of mex functions leads additional time due to overhead. In addition, the use of MATLAB engine also introduces overhead. Thirdly, a more computationally efficient NLP solver that leverages techniques such as parallel implementation is the ultimate approach to pursue.

8.3.4 LIDAR Sensor

For future work, the detected data points from all 8 layers could be used, which would require object identification to exclude data points due to the ground. In addition, in this work, only instantaneous LIDAR data is used. Accounting for the LIDAR data from multiple consecutive steps would improve the accuracy and robustness of obstacle detection. This would require additional algorithm to keep track of the detected obstacles.

8.3.5 Speed Controller

As observed in some of the examples, the performance of the speed controller is not perfect. Large deviations from the reference speed are observed, especially when the vehicle steers.

In the future, a more advanced speed controller can be implemented to improve the speed tracking performance. The obstacle avoidance performance might also be improved because the collision-free and no-wheel-lift-off can only be guaranteed when the resulting control commands are executed precisely.

8.4 Conclusion

These preliminary results show that the developed algorithm is promising in terms of its capability of avoiding obstacles as demonstrated in Section 8.2. However, the experimental validation is not complete due to time and resource limitations. There are also many issues remain to be addressed as discussed in Section 8.3. Significant efforts are still required to thoroughly validate the algorithm.

CHAPTER 9

Conclusion

9.1 Summary

AGVs are attractive not only for reducing accidents and improving mobility in civilian applications, but also for increasing personnel safety and task completion performance in military applications. Realizing the full potential of AGVs relies on the development and successful deployment of many algorithms, including obstacle avoidance.

Obstacle avoidance refers to the task of generating control commands to navigate a vehicle safely around obstacles based on sensor measurements and/or an *a priori* map of the environment. Many obstacle avoidance algorithms have been developed in the literature for different AGV platforms and applications.

Different from most of the prior work, the context of interest for this paper is obstacle avoidance in large-size, high-speed AGVs within unknown and unstructured environments. This context is motivated by military applications, in which task completion performance is critical and hence high-speed maneuvering is desired. However, for large vehicles at high speeds, obstacle avoidance maneuvers can induce stability or handling issues, such as excessive sideslip, wheel lift-off, or even rollover. To this end, obstacle avoidance algorithms are needed that can utilize the knowledge of vehicle dynamics to maximize the task completion performance while guaranteeing the safety of the AGVs.

Specifically, a typical military truck on a flat terrain is considered in this work as a representative large-size AGV. Its mission is to move from an initial position to a given target position safely and as quickly as possible. Thus, the AGV is desired to be operated at high speed and close to its dynamic limits if necessary. The environment is unstructured, which means that there are no lanes or traffic rules to follow. In addition, between the initial and target positions, there exist obstacles,

whose locations, sizes, and shapes are unknown *a priori*, but are detected when they come into the range of a planar LIDAR sensor.

This work achieves its objective by developing a NLMPC-based obstacle avoidance algorithm. In the algorithm, a model of the AGV is used explicitly to predict and optimize future actions.

As a starting point, the vehicle longitudinal speed is first maintained to be a constant value and is provided to the algorithm as an input. The fidelity requirement of the vehicle model used for predicting vehicle trajectory in the MPC formulation is investigated.

First, open-loop simulations suggest that tire nonlinearity and longitudinal load transfer are both important factors to be included in the two DoF vehicle model in order to predict vehicle trajectory accurately. Closed-loop simulations with the MPC-based obstacle avoidance algorithm are also conducted using various obstacle fields. The performance is characterized by several metrics including the time to target, control effort, and integral of curvature. The results show that a two DoF vehicle model with linear tire model and constant axle loads can perform comparable to the fourteen DoF model when the vehicle travels at lower speed. However, a two DoF model with linear tire model or constant axle loads can fail to navigate the vehicle safely when the vehicle moves at higher speed within an obstacle field including large obstacles.

An OCP formulation is developed that optimizes the steering angle command to achieve a smooth operation of the vehicle through the obstacle field at high speed while ensuring vehicle safety. The novelty of the formulation is three-fold: 1) This work focuses on unstructured environments without a reference trajectory. A new cost function formulation is used that aims to find the shortest path to the target position in addition to approaching the target from a desired direction and minimizing the control effort. (2) To accommodate the complicated form of the obstacle-free region in the OCP formulation, the region is systematically partitioned, enabling a differentiable mathematical representation of the obstacle-free region and its inclusion in the OCP through a multi-phase approach. (3) The algorithm considers vehicles with relatively higher CoG and explicitly accounts for the vehicle dynamical safety in terms of avoiding single-wheel lift-off. This is achieved by limiting the steering angle within a range obtained offline using a fourteen DoF vehicle dynamics model. Simulations of an AGV in three different obstacle fields are given to demonstrate the effectiveness of the proposed algorithm.

The constant speed algorithm can achieve an optimal and smooth operation of

AGVs at high speed through unstructured environments without collision while ensuring vehicle dynamical safety. However, the formulation assumes that the vehicle longitudinal speed is maintained constant, which can limit the mobility performance and the obstacle fields that can be cleared with this algorithm. Thus, the formulation is later extended to simultaneously optimize both the longitudinal speed and steering control commands. The novelty of the formulation includes: (1) A varying prediction horizon MPC is used to achieve a fixed distance prediction. This is prompted by two features of the proposed system. First, the terminal point of the planned trajectory is constrained at the LIDAR's maximum detection range in an effort to fully utilize as much information from the LIDAR as possible. Second, the variable vehicle speed necessarily leads to a variable prediction horizon with the previous constraint. (2) The effects of powertrain and brake systems are taken into account through the relationship between acceleration and speed and the bounds on longitudinal jerk, acceleration, and speed. The vehicle's acceleration capability varies with the speed resulting from the powertrain and brake systems. To generate a speed profile that can be tracked by the vehicle, the algorithm uses an offline generated look-up table to account for the acceleration and deceleration limitations. (3) The no-wheel-lift-off requirement is considered through both hard and soft constraints using equations with empirical parameters that can predict tire vertical loads. A hard constraint bounds the vertical loads to be greater than a specified minimum threshold. A soft constraint is also used to provide a smooth approach to this threshold to prevent overshoot. Three sets of numerical simulations are conducted to demonstrate the effectiveness of the algorithm.

The parameter values in the previous simulation-based validations were assumed to be accurate, but in reality they are uncertain. This uncertainty can affect the effectiveness and task completion performance of the obstacle avoidance algorithm. Thus, the effect of parametric uncertainty on the obstacle avoidance is studied. It is demonstrated that using nominal parameter values in the algorithm leads to safety issues in about 25% of the evaluated scenarios with the considered parametric uncertainty distributions. Therefore, a novel approach is developed to improve the robustness of the obstacle avoidance algorithm to parametric uncertainty. In particular, a double-worst-case formulation is developed for a robust satisfaction of the two safety requirements for high-speed obstacle avoidance in AGVs: collision-free and no-wheel-lift-off. Similar to the scenario-based approach, the proposed method also makes the constraints in the OCP formulation in the MPC algorithm robustly safe in all of the considered scenarios. However, instead of using hundreds of randomly generated sce-

narios, only two scenarios are considered, which are the two most likely worst-case scenarios corresponding to the two types of identified unsafe scenarios. In addition, instead of applying all the scenarios to check all the constraints, each scenario is only used to check the satisfaction of the relevant constraints, which helps to minimize the number of constraints in the augmented OCP formulation for robustness. The results show that the proposed formulation improves the robustness of the algorithm, albeit it cannot be guaranteed for all possible realizations of parametric uncertainty distributions because of the non-uniqueness and the approximation of the worst-case scenarios. The trade-off between the robustness and task completion performance of the NLMPC-based obstacle avoidance algorithm is also quantified.

Finally, experimental validations are conducted with the constant speed algorithm. It is shown that the developed algorithm is promising in terms of its capability of avoiding obstacles. Some issues are identified to help improve the quality of the algorithm and future experimental validation.

9.2 Contributions

The original contributions of this thesis are listed as follows:

Evaluation of model fidelity requirement in the MPC formulation

A four-wheel truck driving at constant speed is considered as an example AGV platform. The AGV is navigated within an unstructured flat environment by the obstacle avoidance algorithm. To investigate the level of model fidelity needed in order for a MPC-based obstacle avoidance algorithm to be able to safely and quickly avoid obstacles even when the vehicle is close to its dynamic limits, five different representations of vehicle dynamics models are considered: four variations of the two DoF representation as lower fidelity models and a fourteen DoF representation with combined-slip Magic Formula tire model as a higher fidelity model. It is concluded that the two DoF representation that accounts for tire nonlinearities and longitudinal load transfer is necessary for the MPC-based obstacle avoidance algorithm in order to operate the vehicle at its limits within an environment that includes large obstacles. For less challenging environments, however, the two DoF representation with linear tire model and constant axle loads is sufficient. The details of the evaluation are presented in [1, 4].

Multi-phase OCP formulation taking into account sensory information

A planar LIDAR sensor is used to provide information about range and geometrical characteristics of the closest objects to the vehicle. The LIDAR returns the distance to the closest obstacle boundary in each radial direction. It is assumed that all obstacles of interest are at least the height of where the LIDAR sensor is mounted. The data from the sensor is used to form a safe region for the vehicle. However, in order to formulate an OCP, the mathematical representation of this safe region is required. In addition, all functions involved should be twice continuously differential so that the formulated OCP can be solved efficiently. To address this challenge, the safe region is partitioned into sub-regions and a multi-phase OCP formulation is used. After partitioning, each sub-region can be specified by a set of inequalities. The functions involved in these inequalities are not piecewise functions and are twice continuously differentiable. The formulation is first presented in [2] and is substantially extended in [5].

Optimization of steering and speed simultaneously taking into account vehicle dynamics

To develop an algorithm that simultaneously optimizes the steering and speed commands, the powertrain and brake dynamics are taken into consideration through the bounds on vehicle longitudinal speed, acceleration and jerk. In this study, ensuring the vehicle's dynamical safety is defined as avoiding single-wheel lift-off, which is a conservative criterion used to prevent rollover. Prior work enforced this constraint through steering angle bounds, or lateral acceleration bounds. However, these approaches ignore the effect of longitudinal acceleration, which is an important factor to consider in the variable speed case. Hence, in this work, the no-wheel-lift-off requirement is taken into account directly by enforcing a positive vertical load on all four tires at all times. The tire vertical loads are predicted using the three DoF vehicle model and estimated load transfer coefficients. In the cost function, a term is added to prevent the vehicle from operating at the limit unnecessarily. A detailed description is provided in [3] and [6].

Improvement of the algorithm robustness to parametric uncertainty.

The robustness of the MPC-based obstacle avoidance algorithm is improved.

Specifically, parametric uncertainties in the vehicle and tire models that are used in the MPC are considered and it is first demonstrated in simulation that using nominal parameter values does not yield a robust performance under the considered parametric uncertainty distributions; in about 25% of the cases the vehicle violates the no-collision or no-wheel-lift-off constraints. Then, a probabilistic analysis is performed on parameter value combinations that are most likely to cause collision or wheel lift-off, and a novel optimal control formulation is developed that uses two different sets of parameter values that represent the two most likely worst-case scenarios (collision and wheel lift-off). It is demonstrated that with the new formulation, MPC yields a robust performance in all the cases tested. Compared to other approaches in the literature that can be employed to increase robustness of MPC, the developed approach has the unique capability of handling the nonlinear optimal control problem with a high number of uncertain parameters in a more computationally efficient manner. The trade-off between the robustness and the task completion performance of the algorithm is also quantified. The formulation and results are summarized in [7] and [8].

Preliminary experimental validation of constant speed formulation.

Preliminary experimental tests are performed to validate the obstacle avoidance capability of the developed constant speed algorithm. The vehicle platform is a four-wheeled utility vehicle augmented with drive-by-wire capability. A high-performance GPS-aided Inertial Navigation System and an eight-layer scanning LIDAR sensor are also installed on the platform. An onboard Linux PC processes the sensing data, runs the obstacle avoidance algorithm, and sends the motor commands to the vehicle. The constant speed obstacle avoidance algorithm is used, in which only the steering angle sequence is optimized. The longitudinal speed of the vehicle is maintained at a specific constant value using a separate controller. Several sets of tests are conducted. First, the steer-to-target tests are performed. No obstacles are placed in the environment. The algorithm is used to drive the vehicle to move towards the specified target position. Second, the obstacle avoidance tests are performed. Obstacles of different sizes are used in the validation. It is shown that the developed algorithm is promising in terms of its capability of navigating to the target and avoiding obstacles. Some issues are identified to help improve the quality of the algorithm and future experimental validations.

9.3 Real-Time Implementation

The algorithm is currently implemented in MATLAB® for proof-of-concept. Thus, it does not yet run in real-time for all cases. Specifically, with the current implementation, it is estimated to be about 10 times slower than real-time for the constant speed algorithm and about 30 times slower for the variable speed algorithm in the worst case with a 3.5 GHz Intel® Xeon® processor. However, when fully implemented in a compiled language with an optimized code, the algorithm is expected to run in real time.

Because the real-time feasibility is an important consideration in the development of the obstacle avoidance algorithm, detailed discussions of the efforts made and the potential directions to be pursued in improving the computational efficiency are included below.

As presented in Chapter 1.7, the NLMPC obstacle avoidance algorithm consists of two parts: the LIDAR data processor and the control commands generator. Thus, the computational efficiencies of both parts are considered.

The high computational efficiency of the LIDAR data processor is achieved by adopting C/C++ implementation of several sub-functions.

The computation time of the control commands generator is mostly spent on solving the formulated OCPs. Regarding the solution technique, the *hp*-pseudospectral method is implemented partially in MATLAB® and partially in C++. The open source software library IPOPT [112] is used, which is implemented in C++.

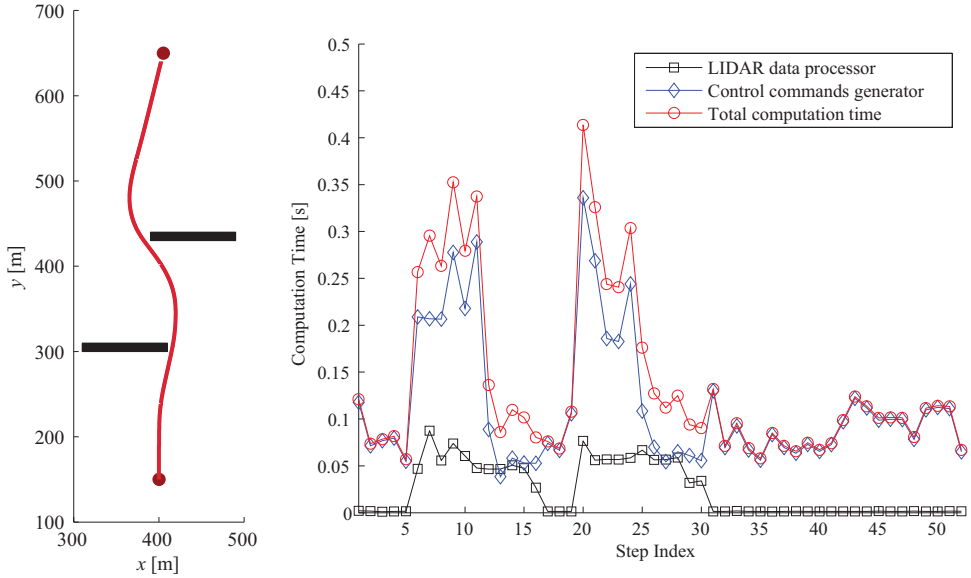
The *hp*-pseudospectral method is used because it utilizes unevenly spaced collocation points. High accuracy can be achieved with a much less number of collocation points compared to evenly spaced discretization, which results in much reduced computational load [136].

The selection of the IPOPT is based on a study of computation performance comparison of different NLP solvers [137]. In this study, the effect of the linear solvers used with the IPOPT on the computation performance is also considered. In particular, MUMPS [138] and MA27 from the HSL mathematical software library [139] are considered. According to the study, on average, IPOPT with MA27 as the linear solver is about 40% faster than IPOPT with MUMPS as the linear solver. Thus, MA57, which is an updated version of MA27, is used in this work.

Due to their inefficiency, numerical derivatives are not used with IPOPT. Instead, analytical representations of all first and second derivatives of all functions are manually provided to increase computational efficiency and accuracy. An order of mag-

nitude increase in computational speed was observed with the analytical derivatives compared to the numerical derivatives.

One example is presented to demonstrate the computation efficiency of the current partial MATLAB® and partial C++ implementation of the algorithm and to point out the bottleneck for further improvements. Figure 9.1b gives the computational time required at each step of the calculation for the given obstacle map shown in Figure 9.1a. The vehicle is navigated through this obstacle field at a constant speed of 20 m/s. The prediction horizon is 5 s and the execution horizon is 0.5 s. As shown in Figure 9.1b, the total computation time at all steps are below the execution horizon, which means that real-time performance is achieved for this test case.



(a) Trajectory (b) Computation time

Figure 9.1: Computation time of constant speed algorithm with a simple test case

In this example, the LIDAR data processor is not called when it is safe to ignore the obstacles and directly steer to the target. When it is called, the computation time is around 50 ms, which is about 10% of the execution horizon and thus is not significant. The variation of the computation time from step to step is small. The computation time of the control commands generator is about 10% to 70% of the execution horizon. The variation of the computation time from step to step is also significant.

Figure 9.2a shows the comparison of the computation times of two implementations of variable speed algorithm. One implementation is pure MATLAB® and another implementation is partial MATLAB® and partial C++. Neither of these

implementations could achieve real-time computation of all 95 OCP examples. With partial MATLAB® and partial C++ implementation, the percentage of examples that can be completed within real-time is improved from 43% to 64%. This is because of the 1.5 times computation performance improvement as shown in Figure 9.2b.

Thus, further increase of the computational efficiency of the LIDAR data processor might not be necessary. More efforts are needed to further improve the computation performance of the control command generator. In particular, the following strategies can be considered.

First of all, the most straightforward approach is to use a more powerful computing device. However, the improvement of computation performance might not be significant. Thus, other approaches are necessary.

Secondly, being an interpreted language, MATLAB® is not a computationally efficient platform. A pure C++ implementation of the algorithm needs to be pursued. In addition, in the current partial MATLAB® and partial C++ implementation, mex functions are used, which lead to additional time due to overhead. Thus, a pure C++ implementation is helpful in two aspects: improving the computation performance of the current MATLAB® functions and eliminating the overhead time.

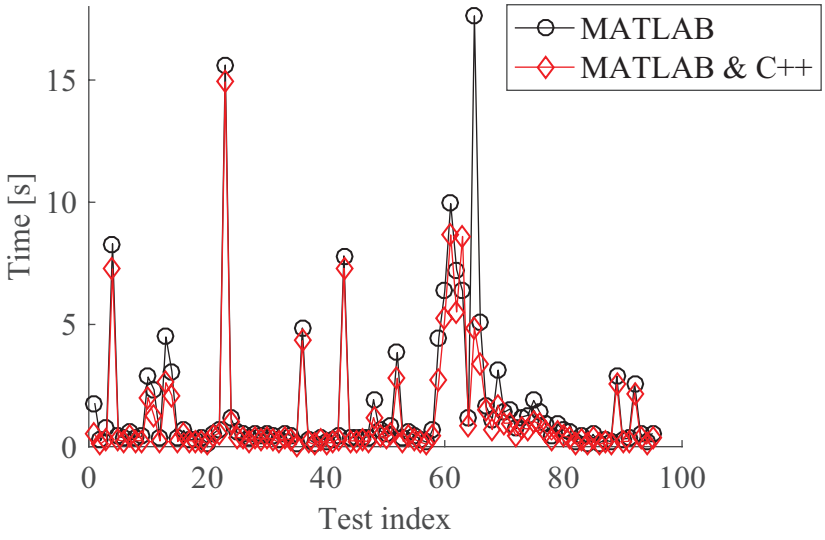
Thirdly, a better initial guess of the solution would be helpful to reduce the number of iterations needed to converge to the optimal solution. Because the computation time is highly proportional to the number of iterations, the use of proper initial guess could improve the computation performance. One promising approach to generate the initial guess is to use motion primitives.

Finally and most importantly, a more computationally efficient NLP solver that leverages techniques such as parallel implementation on GPUs is the ultimate approach to pursue. For example, parallel speedup can be achieved by exploiting problem structure and decomposing the internal linear algebra operations performed by the NLP algorithm [140]. The use of GPUs can significantly decrease the computation time.

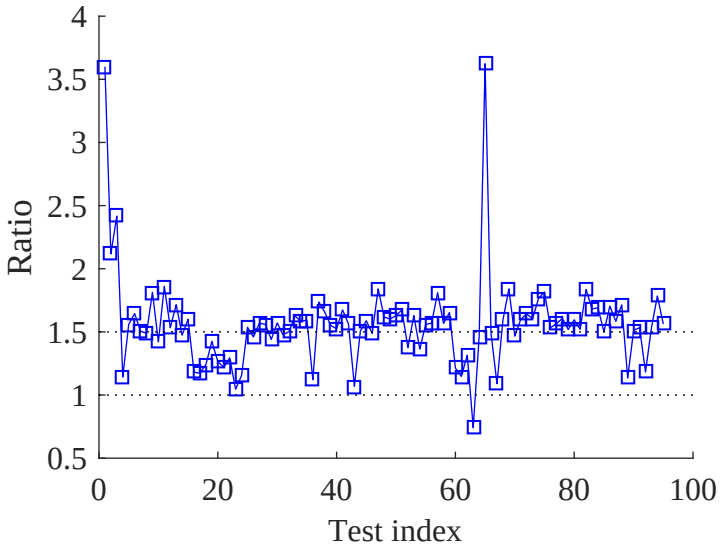
9.4 Future Work

As pointed out throughout this thesis, this work presents many possible directions for further work. They can be summarized as follows:

1. Regarding the study of the model fidelity requirements, future research opportunities are summarized as follows. First, the model fidelity requirement when



(a) Computation times of different implementations



(b) Ratio of the computation time of pure MATLAB implementation and the computation time of partial MATLAB partial C++ implementation.

Figure 9.2: Comparison of computation time with different implementations

the vehicle speed is added as a second control variable needs to be investigated. When the vehicle speed is varying, there are other important factors to consider, such as engine limits, brake limits, combined-slip tire model, etc. Second, this study pertains to the four-wheel truck. For a different vehicle, the procedures presented in this paper can be repeated. However, it would be beneficial if the model fidelity requirement can be determined easily for a series of platforms with different weight and CoG location. Last, the model fidelity requirement also needs to be re-evaluated when the vehicle parameter and state measurements uncertainties are accounted for or when the environment is non-flat.

2. The instantaneous LIDAR data are used in this work for simulation simplicity. Prior data are not stored. Even though this approach is sufficient to navigate the vehicle safely in the test cases presented in this thesis, tracking the LIDAR data in time could improve performance. In real world, it is beneficial to utilize LIDAR data from multiple steps and maintain a continuously updated map of the environment to provide more accurate and more complete obstacle information. This exploitation of prior measurements can be achieved by considering SLAM algorithms.

3. It is assumed that all obstacles can be detected. When only one planar LIDAR sensor is used, it is assumed that all obstacles are at least the height of where the LIDAR is mounted, which is in front of the vehicle. It is obvious that this assumption needs to be guaranteed because the algorithm cannot react to an obstacle that it is not aware of. In reality, a practical approach to ensure this assumption is to use multiple planar LIDAR sensors that face to different directions or a 3D LIDAR sensor. In this case, a data fusion algorithm that combines measurements from multiple number and/or types of sensors, is required to regenerate the obstacle-free region. As long as a safe region can be deduced from the sensor data, the algorithm is applicable. It is because that the algorithm presented in this thesis is not limited to be used with a planar LIDAR sensor.

4. Regarding the region partitioning approach, any partitioning technique can be used with the multi-phase optimal control problem formulation as long as it meets the following criteria: (1) computationally efficient; (2) easy to generate inequality representation of all sub-regions, where the functions involved in these inequalities are twice continuously differentiable. Two approaches are presented. It is demonstrated that the same solution is obtained with different

partitioning approaches. However, it is observed that the use of different partitioning approaches could affect the computational time. However, it is not clear yet what the fundamental reasons are. Thus, the following three questions need to be answered: (1) what the characteristics of a good partition are; (2) how to evaluate the quality of the partition; (3) how to generate a good partition systematically and efficiently.

5. A suitable cost function is essential for any optimization problem. It defines the performance measures of solutions. It should represent exhaustively and concisely the goals of the optimization task. The cost function of the variable speed algorithm includes eight terms and requires seven weighting factors. The weights selected based on engineering intuitions work well for the obstacle fields presented in the thesis and some others that are not included here. However, it is worth to investigate further on the formulation of cost function and selection of weighting factors.

6. The assumption of the obstacle being static is a limitation of the present work. Moving obstacles should be considered in the obstacle avoidance simulation that will represent a more generic and realistic application [141]. Time-varying position constraints could be used to handle moving obstacles. The difficulties are from two aspects: how to predict the motions of the moving obstacles along the prediction horizon and how to incorporate these motions as time-varying constraints in the OCP formulation efficiently.

7. The extension to applications on non-flat terrain is an important task. First, the term 'non-flat' needs to be defined. There are various types of non-flat terrains. It is very difficult, if not impossible, to handle all the situations with one algorithm. Thus, it is critical to prioritize the situations to be considered. One approach to classify the terrain is to use the relationship between the 'wavelength' of the terrain and the size of the vehicle. Second, the plant model that is used in to simulation-based study should be augmented so that it can be used as a benchmark for evaluating the performance of the vehicle in the defined non-flat terrain application. Third, an appropriate vehicle model to be included in the MPC formulation is to be selected. It should be able to take into account the terrain information, for example, the slope. Finally, it can be assumed that all required terrain information is available *a priori*. However, it is more realistic if the required information is estimated based on sensor measurements.

8. The approach presented in this thesis to handle parametric uncertainty is a passive one. An active approach can be used in complement that reduces the uncertainty through an online nonlinear adaptive estimator to obtain more accurate estimations of the parameter values based on the difference between the model predictions and actual data collected from the vehicle. To this end, existing adaptive observer schemes such as the Extended Kalman Filter can be leveraged. Future research could include adding such capability into the closed-loop system to reduce the task completion performance degradation due to conservativeness caused by large uncertainty ranges.

9. Besides parametric uncertainty, there are other sources of uncertainty. For example, one source of uncertainty is the errors in the knowledge of vehicle's locations, and current, and future locations of the obstacles. Another source of uncertainty is the external disturbances due to wind and other factors. These sources might also cause problems and need to be considered. However, when studying the effect of parametric uncertainty, the other sources of uncertainties are not considered so that the focus can be put specifically on the parametric uncertainty. Separately, the effect of the state measurement uncertainty on the robustness of the algorithm is also evaluated. However, the robustness of the algorithm subject to all types of uncertainties is another topic requiring additional research.

10. The experimental results show that the developed algorithm is promising in terms of its capability of avoiding obstacles. However, the experimental validation is not complete due to time and resource limitations. There are also many issues remain to be addressed as discussed in Section 8.3. Significant efforts are still required to thoroughly validate the algorithm.

APPENDIX A

Summary of Main Features of Five Representative MPC-based Obstacle Avoidance Algorithms

Table A.1: Summary of main features of five representative algorithms

Reference	[61]	[62]	[52]	[65]	[89]
Vehicle model (# of states)	3 DoF (6)	3 DoF (6)	3 DoF (5)	7 DoF (12)	2 DoF (5)
Control (# of inputs)	γ_f, U (2)	δ_f, $F_{d,f}$, $F_{d,r}$ (3)	δ_f, $F_{d,\text{left}}$, $F_{d,\text{right}}$ (3)	δ_f, T_{engine}, $T_{\text{brake},\star}$ (6)	δ_f (1)
Tire model	Linear	Simplified Magic formula	Pure-slip Magic formula	Combined-slip Magic formula	Linear
Levels of controller	Two-level	One-level	Two-level	One-level	One-level
Optimization goal	Minimize deviation from reference trajectory, x, y	Minimize deviation from reference trajectory, y	Minimize deviation from tracking states, $y, \psi, \dot{\psi}$	Minimize deviation from tracking states (all 12 states)	Minimize deviation from reference trajectory, x, y
Obstacle constraints	Soft constraints: potential function based on parallax value	Soft constraints: barrier function based on road bounds and unavoidable region	Soft constraints: potential function based on minimum distance	Hard constraints: lower and upper road bounds	Soft constraints: point-wise, repulsive potential function
Dynamic constraints	Fixed minimum turning radius	Maximum road force based on road friction	Maximum road force based on road friction	None	None
Max. speed [m/s]	5.5	20	20	10	5.5
Optimizer	Gradient search method	Generalized minimum residual method	NPSOL	qpOASES	Gradient search method
Control horizon/ prediction horizon	1/40	1/100	2/15	1/20	1/20
Sampling time [s]	0.05	0.002	0.05	0.1	0.05

APPENDIX B

The OCP Formulation and Dynamic Optimizer for Model Fidelity Evaluation

This Appendix presents the cost function and constraints formulation, as well as the dynamic optimizer components of the MPC-based obstacle avoidance algorithm used in Chapter 3 in detail.

B.1 Cost Function and Constraints

The cost function and constraints need to be specified properly to achieve the objective of avoiding the locally detected obstacles while guaranteeing vehicle dynamical safety and minimizing travel time. Thus, the OCP to be solved at each step of the MPC is formulated as

$$\underset{\boldsymbol{\xi},\boldsymbol{\zeta}}{\text{minimize}} \quad J = s_f + w_{\text{cf}} \cdot v_{\text{cf}} \tag{B.1}$$

$$\text{subject to} \quad \dot{\boldsymbol{\xi}}(t) = \boldsymbol{\mathcal{V}}\left[\boldsymbol{\xi}(t), \boldsymbol{\zeta}(t)\right] \tag{B.2}$$

$$\boldsymbol{\xi}(0) = \boldsymbol{\xi^0} \tag{B.3}$$

$$\tilde{\boldsymbol{\mathcal{R}}}\left[x(t), y(t)\right] \leq 0 \tag{B.4}$$

$$|\delta_f(t)| \leq \tilde{\delta}_{f,\max}(U_0) \tag{B.5}$$

$$|\gamma_f(t)| \leq \gamma_{f,\max} \tag{B.6}$$

$$t \in [0, T_p] \tag{B.7}$$

Equation (B.1) is the cost function. Equation (B.2) is the dynamic model of the vehicle represented as a set of first-order ODEs. Equation (B.3) specifies the vehicle initial condition. Equation (B.4) defines the position constraints due to the obstacles perceived by the LIDAR sensor. Equations (B.5) and (B.6) represent the bounds on the steering angle and steering rate, respectively. Equation (B.7) is the prediction horizon over which the optimal control problem is solved.

B.1.1 Cost Function

The cost function defines the soft requirement; that is, in what sense the trajectory is optimal. In this work, the cost function is formulated as

$$J = s_f + w_{\text{cf}} \cdot v_{\text{cf}} \tag{B.8}$$

where

$$s_f = \sqrt{[x_t - x(T_p)]^2 + [y_t - y(T_p)]^2} \tag{B.9}$$

$$v_{\text{cf}} = \int_0^{T_p} |\gamma_f(t)| \mathrm{d}t \tag{B.10}$$

Specifically, the cost function formulation includes two terms that are linearly combined using a relative weight w_{cf}. The first term is the distance s_f between the end point of the predicted trajectory $[x\left(T_p\right), y\left(T_p\right)]$ and the target $[x_t, y_t]$ as defined in Equation (B.9). A visual representation of this term is shown in Figure 5.3. Due to the constant speed assumption for the model fidelity evaluation study, this term also aims to minimize the remaining travel time. The second term is a regulation term minimizing the integral of magnitude of the steering rate as defined in Equation (B.10), where γ_f is the front wheel steering rate.

B.1.2 Constraints

The constraints represent the hard requirements for avoiding collision and ensuring vehicle dynamical safety. These requirements are hard in the sense that their violation is not allowed under any circumstances.

To avoid collision with obstacles, the vehicle should completely lie within the safe area detected by the LIDAR as exemplified in Figure 1.8b. The safe area can be considered as the inside of a simple polygon and can be defined using an inequality

compacted in the following form

$$\boldsymbol{\mathcal{R}}\left[x(t), y(t)\right] \leq 0, \forall t \in [0, T_p] \tag{B.11}$$

where the function $\boldsymbol{\mathcal{R}}(\cdot, \cdot)$ has no explicit form and is specified by a set of points from the LIDAR data because the safe area typically is non-convex. For a particular trajectory, point-in-polygon test is used to decide whether Equation (B.11) is satisfied or not.

To allow for some additional safety margin, a minimum acceptable distance between the center of gravity of the vehicle along the trajectory and any detected points by the LIDAR sensor (i.e., the boundaries of the obstacles) is included. Thus, the inequality describing the safe region, Equation (B.4), becomes Equation (B.12). If this requirement is satisfied, then the vehicle is considered as safe, that is, collision-free.

$$\tilde{\boldsymbol{\mathcal{R}}}\left[x(t), y(t)\right] \leq 0, \forall t \in [0, T_p] \tag{B.12}$$

where the function $\tilde{\boldsymbol{\mathcal{R}}}(\cdot, \cdot)$ indicates the safe region with safety margin included.

The second type of constraint is related to the dynamical safety of the vehicle. An upper bound on the steering angle magnitude as expressed by the following inequality constraint is used to prevent single-wheel lift-off.

$$|\delta_f(t)| \leq \tilde{\delta}_{f,\max}(U_0) \tag{B.13}$$

where the maximum steering angle $\tilde{\delta}_{f,\max}$ is a function of the vehicle speed U_0, which has no analytical expression and is represented using a look-up table that is described Section 5.2.3.

B.2 Dynamic Optimizer

Many high efficiency optimization algorithms have been considered in the literature to enable real-time operation with MPC. However, when the system dynamics are represented using the fourteen DoF formulation, this optimization problem will be very difficult to solve using standard optimization algorithms and is therefore subject to ongoing research. Instead, an exhaustive search approach is used here to solve this optimization problem approximately for the purpose of the model fidelity evaluation study. This approach is not a standard method for solving OCPs and is used solely for the purpose of evaluating model fidelity.

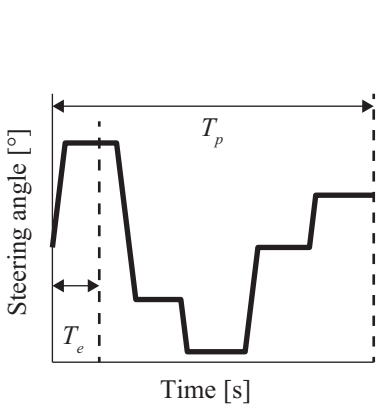
Towards this end, the search space is discretized; that is, instead of considering the entire continuous search space, only some discrete steering angles that include the maximum value, the minimum value, and zero are considered. The set of discrete steering angles is called the steering angle pool and is given by Equation (B.14). The entire prediction horizon is divided into intervals of the same length. At each interval, a steering angle is selected from the steering angle pool. The zero-order hold approach is used at each interval to form the steering angle sequence. Ramp transition is used between two different steering angles, where the slope is the maximum steering rate. Figure B.1a shows an example possible steering sequence. If the prediction horizon is divided into n_p intervals and the number of elements in the steering angle pool is n_s, the size of the search space will be $n_p^{n_s}$. Note that the execution horizon, T_e, can be shorter than the interval length, T_p/n_p. The control commands can be changed at the end of the execution horizon.

$$\bigcirc_{pool} = \left[\begin{array}{ccccccc} -\tilde{\delta}_{f,\max}(U_0) & \cdots & 0 & \cdots & \tilde{\delta}_{f,\max}(U_0) \end{array} \right] \tag{B.14}$$

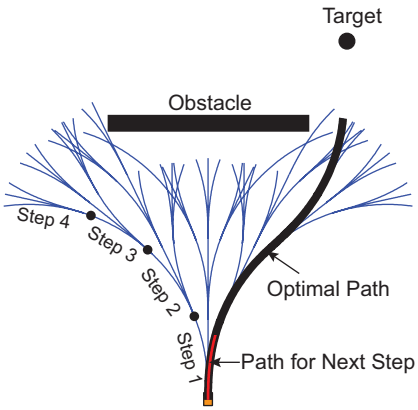
Note that Equation (B.10) in the cost function is then approximated by the following equation.

$$d = \sum_{i=1}^{n_p} |\delta_i - \delta_{i-1}| \tag{B.15}$$

The optimal control sequence that can minimize the cost function and satisfy the constraints is selected from the discrete steering command pool defined as Equation (B.14). Starting from the initial state, the control commands from the pool are applied to the vehicle and the resulting trajectories are checked for constraint violation. If constraints are not satisfied, the search branch is terminated. Otherwise, next step prediction is performed by using the end state of last simulation as the initial state and applying all the steering commands from the pool. This process is repeated until a predetermined number of steps in the prediction horizon is reached. Once all the feasible steps are thus determined, their costs are evaluated and the control sequence with minimum cost is considered as the optimal control sequence. The control command over the execution horizon is applied to the vehicle, and the process is iterated in the next simulation step. This idea is illustrated in Figure B.1b with $n_s = 3$ and $n_p = 4$ for clarity.



(a) Example of steering sequence.



(b) Illustration of the exhaustive search method using a steering command pool of three elements.

Figure B.1: Exhaustive search.

B.2.1 Algorithm Parameters

There are six parameters included in the algorithm as listed in Table B.1. Vehicle speed is maintained constant during the whole maneuvering process.

Number of intervals over the prediction horizon and number of elements in the steering angle pool determine the size of search space. Three different combinations of n_p and n_s are tested under the same condition: $n_p = 4, n_s = 5$; $n_p = 4, n_s = 7$; and $n_p = 5, n_s = 5$. The results are very close and thus the combination of $n_p = 4$ and $n_s = 5$ is used in the study presented in Chapter 3.

Table B.1: Parameters in algorithm for model fidelity evaluation

Parameter	**Symbol**	**Unit**
Vehicle speed	U_0	m/s
Number of intervals over the prediction horizon	n_p	-
Number of elements in the steering angle pool	n_s	-
Prediction horizon	T_p	s
Execution horizon	T_e	s
LIDAR detection range	R_{LIDAR}	m

The LIDAR detection range should be larger than the maximum travel distance along the vehicle's initial heading angle. This maximum distance is achieved when the vehicle moves straight in its initial heading angle. If the inequality

$$U_0 \cdot T_p \leq R_{\text{LIDAR}} \tag{B.16}$$

is satisfied, then all possible trajectories are covered by the LIDAR detection. This is required because if the trajectories are not totally covered by the LIDAR detection, it is unclear whether the trajectories are safe in terms of obstacle avoidance or not. This inequality can help with the selection of the appropriate sensor based on the specification of maximum detection range, or the vehicle velocity and prediction horizon combination for a given LIDAR range.

There is a lower bound on the prediction horizon to make sure that the vehicle detects and takes action to avoid the obstacle early enough.

$$T_p \geq T_{p,\min}\left(U_0, L_{\max}\right) + T_e \tag{B.17}$$

The lower bound consists of two parts. The first part corresponds to the minimum time for the vehicle to avoid hitting an obstacle of a fixed length. It is a function of vehicle speed and maximum obstacle size in the field as depicted in Figure B.2, which is created by simulating the fourteen DoF vehicle model. Note that even though the detailed obstacle information is detected using a planar LIDAR sensor, the maximum obstacle size can be assumed known *a priori*. The second part is the execution time, which is the duration of the computed command that is implemented.

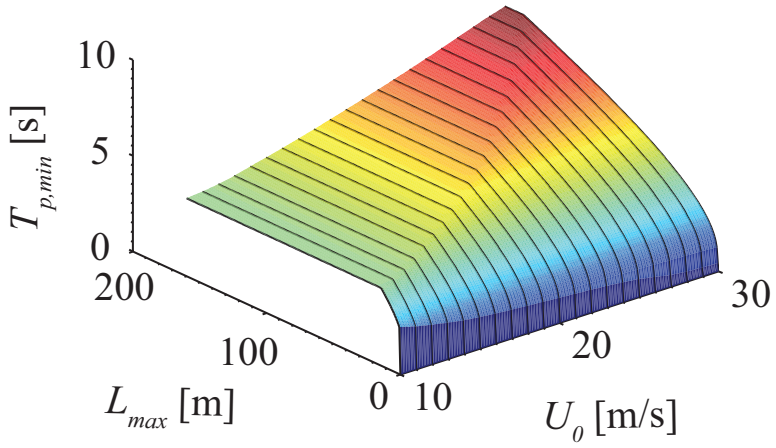


Figure B.2: Minimum Prediction Time.

BIBLIOGRAPHY

[1] Jiechao Liu, Paramsothy Jayakumar, James L Overholt, Jeffrey L Stein, and Tulga Ersal. The role of model fidelity in model predictive control based hazard avoidance in unmanned ground vehicles using LIDAR sensors. In *Dynamic Systems and Control Conference*. ASME, 2013.

[2] Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. A multi-stage optimization formulation for MPC-based obstacle avoidance in autonomous vehicles using a LIDAR sensor. In *Dynamic Systems and Control Conference*. ASME, 2014.

[3] Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. An MPC algorithm with combined speed and steering control for obstacle avoidance in autonomous ground vehicles. In *Dynamic Systems and Control Conference*. ASME, 2015.

[4] Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. A study on model fidelity for model predictive control-based obstacle avoidance in high-speed autonomous ground vehicles. *Vehicle System Dynamics*, pages 1–22, 2016.

[5] Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. A nonlinear model predictive control formulation for obstacle avoidance in high-speed autonomous ground vehicles in unstructured environments. *Vehicle System Dynamics*, in review.

[6] Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. Combined speed and steering control in high speed autonomous ground vehicles for obstacle avoidance using model predictive control. *IEEE Transactions on Vehicular Technology*, in review.

[7] Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. Improving the robustness of an MPC-based obstacle avoidance algorithm to parametric uncertainty using worst-case scenarios. *IFAC Journal Mechatronics: Special Issue on Automated and Intelligent Vehicles*, in review.

[8] Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. A double-worst-case formulation for improving the robustness of an MPC-based obstacle avoidance algorithm to parametric uncertainty. In *American Control Conference*. in review, 2017.

[9] Robin Murphy and James Shields. The role of autonomy in DoD systems. Technical report, Defense Science Board Task Force Report, 2012.

[10] Andrew Kerbrat. Autonomous platform demonstrator. Technical report, DTIC Document, 2010.

[11] Sebastian Thrun. Toward robotic cars. *Communications of the ACM*, 53(4): 99–106, 2010.

[12] National Research Council. *Application of lightweighting technology to military aircraft, vessels, and vehicles*. The National Academies Press, 2012.

[13] Patrick Norman Currier. *A method for modeling and prediction of ground vehicle dynamics and stability in autonomous systems*. PhD thesis, Virginia Polytechnic Institute and State University, 2011.

[14] Ryan W Wolcott and Ryan M Eustice. Visual localization within LIDAR maps for automated urban driving. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 176–183. IEEE, 2014.

[15] Steven M LaValle. *Planning algorithms*. Cambridge University Press, 2006.

[16] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986.

[17] Elon Rimon and Daniel E Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, 1992.

[18] Johann Borenstein and Yoram Koren. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, 1991.

[19] Jianwei Gong, Yulin Duan, Kai Liu, Yongdan Chen, Guangming Xiong, and Huiyan Chen. A robust multistrategy unmanned ground vehicle navigation method using laser radar. In *Intelligent Vehicles Symposium*, pages 417–424. IEEE, 2009.

[20] Dieter Fox, Wolframe Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Transactions on Robotics and Automation*, 4:1, 1997.

[21] Oliver Brock and Oussama Khatib. High-speed navigation using the global dynamic window approach. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 341–346. IEEE, 1999.

[22] Reid Simmons. The curvature-velocity method for local obstacle avoidance. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3375–3382. IEEE, 1996.

[23] Allan Manz, Ramiro Liscano, and David Green. *A comparison of realtime obstacle avoidance methods for mobile robots*. Springer, 1993.

[24] Jared L Giesbrecht. Local navigation for unmanned group vehicles. Technical Report DRDC Suffield TM 2005-038, DTIC Document, 2005.

[25] Voemir Kunchev, Lakhmi Jain, Vladimir Ivancevic, and Anthony Finn. Path planning and obstacle avoidance for autonomous mobile robots: A review. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pages 537–544. Springer, 2006.

[26] Shingo Shimoda, Yoji Kuroda, and Karl Iagnemma. High-speed navigation of unmanned ground vehicles on uneven terrain using potential fields. *Robotica*, 25(04):409–424, 2007.

[27] Bo-Chiuan Chen and Huei Peng. Rollover warning for articulated heavy vehicles based on a time-to-rollover metric. *Journal of Dynamic Systems, Measurement, and Control*, 127(3):406–414, 2005.

[28] Jared L Giesbrecht. Global path planning for unmanned ground vehicles. Technical Report DRDC Suffield TM 2004-272, DTIC Document, 2004.

[29] Yang Chen, Xingang Zhao, and Jianda Han. Review of 3D path planning methods for mobile robot. *Jiqiren(Robot)*, 32(4):568–576, 2010.

[30] Brandon D Luders, Sertac Karaman, and Jonathan P How. Robust sampling-based motion planning with asymptotic optimality guarantees. In *AIAA Guidance, Navigation, and Control Conference*, 2013.

[31] Frank Lingelbach. Path planning using probabilistic cell decomposition. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 467–472. IEEE, 2004.

[32] Han-Pang Huang and Shu-Yun Chung. Dynamic visibility graph for path planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2813–2818. IEEE, 2004.

[33] Priyadarshi Bhattacharya and Marina L Gavrilova. Roadmap-based path planning using the voronoi diagram for a clearance-based shortest path. *IEEE Robotics & Automation Magazine*, 15(2):58–66, 2008.

[34] Thierry Siméon, J-P Laumond, and Carole Nissoux. Visibility-based probabilistic roadmaps for motion planning. *Advanced Robotics*, 14(6):477–493, 2000.

[35] Min Gyu Park, Jae Hyun Jeon, and Min Cheol Lee. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. In *IEEE International Symposium on Industrial Electronics*, volume 3, pages 1530–1535. IEEE, 2001.

[36] Jin-Oh Kim and Pradeep K Khosla. Real-time obstacle avoidance using harmonic potential functions. *IEEE Transactions on Robotics and Automation*, 8 (3):338–349, 1992.

[37] Randal C Nelson and Jhon Aloimonos. Obstacle avoidance using flow field divergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(10):1102–1106, 1989.

[38] Ahmed Hussein, Heba Mostafa, Mohamed Badrel-din, Osama Sultan, and Alaa Khamis. Metaheuristic optimization approach to mobile robot path planning. In *International Conference on Engineering and Technology*, pages 1–6. IEEE, 2012.

[39] Ibrahim H Osman and Gilbert Laporte. Metaheuristics: A bibliography. *Annals of Operations Research*, 63(5):511–623, 1996.

[40] Amin Zargar Nasrollahy and Hamid Haj Seyyed Javadi. Using particle swarm optimization for robot path planning in dynamic environments with moving obstacles and target. In *Third UKSim European Symposium on Computer Modeling and Simulation*, pages 60–65. IEEE, 2009.

[41] Michael Brand, Michael Masuda, Nicole Wehner, and Xiao-Hua Yu. Ant colony optimization algorithm for robot path planning. In *International Conference on Computer Design and Applications*, volume 3. IEEE, 2010.

[42] Chuangling Liu, Huaiwang Liu, and Jingyu Yang. A path planning method based on adaptive genetic algorithm for mobile robot. *Journal of Information & Computational Science*, 8(5):808–814, 2011.

[43] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[44] Yang Wang and Stephen Boyd. Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology*, 18(2):267–278, 2010.

[45] Frank Allgower and Alex Zheng. *Nonlinear model predictive control*. Birkhauser, 2000.

[46] Francesco Borrelli, Paolo Falcone, Tamas Keviczky, Jahan Asgari, and Davor Hrovat. MPC-based approach to active steering for autonomous vehicle systems. *International Journal of Vehicle Autonomous Systems*, 3(2-4):265–291, 2005.

[47] 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.

[48] Paolo Falcone, Francesco Borrelli, Jahan Asgari, H Eric Tseng, and Davor Hrovat. A model predictive control approach for combined braking and steering in autonomous vehicles. In *Mediterranean Conference on Control & Automation*. IEEE, 2007.

[49] Roland Lenain, Benoit Thuilot, Christophe Cariou, and Philippe Martinet. Model predictive control for vehicle guidance in presence of sliding: application to farm vehicles path tracking. In *IEEE International Conference on Robotics and Automation*, pages 885–890. IEEE, 2005.

[50] Guilherme V Raffo, Guilherme K Gomes, Julio E Normey-Rico, Christian R Kelber, and Leandro B Becker. A predictive controller for autonomous vehicle path tracking. *IEEE Transactions on Intelligent Transportation Systems*, 10 (1):92–102, 2009.

[51] Paolo Falcone, Francesco Borrelli, J Asgari, HE Tseng, and Davor Hrovat. Low complexity MPC schemes for integrated vehicle dynamics control problems. In *9th International Symposium on Advanced Vehicle Control*, 2008.

[52] Yiqi Gao, Theresa Lin, Francesco Borrelli, Eric Tseng, and Davor Hrovat. Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads. In *Dynamic Systems and Control Conference*, pages 265–272. ASME, 2010.

[53] Efstathios Velenis, Panagiotis Tsiotras, and Jianbo Lu. Modeling aggressive maneuvers on loose surfaces: The cases of trail-braking and pendulum-turn. In *European Control Conference*, pages 1233–1240. IEEE, 2007.

[54] Krisada Kritayakirana and J Christian Gerdes. Using the centre of percussion to design a steering controller for an autonomous race car. *Vehicle System Dynamics*, 50(sup1):33–51, 2012.

[55] Petter Ogren and Naomi Ehrich Leonard. A convergent dynamic window approach to obstacle avoidance. *IEEE Transactions on Robotics*, 21(2):188–195, 2005.

[56] Adnan Tahirovic and Gianantonio Magnani. Passivity-based model predictive control for mobile robot navigation planning in rough terrains. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 307–312. IEEE, 2010.

[57] Adnan Tahirovic and Gianantonio Magnani. General framework for mobile robot navigation using passivity-based MPC. *IEEE Transactions on Automatic Control*, 56(1):184–190, 2011.

[58] James A Primbs, Vesna Nevistić, and John C Doyle. Nonlinear optimal control: A control Lyapunov function and receding horizon perspective. *Asian Journal of Control*, 1(1):14–24, 1999.

[59] Tobias Raff, Christian Ebenbauer, and Prank Allgöwer. Nonlinear model predictive control: A passivity-based approach. In *Assessment and Future Directions of Nonlinear Model Predictive Control*, pages 151–162. Springer, 2007.

[60] Yongsoon Yoon, Jongho Shin, H Jin Kim, Yongwoon Park, and Shankar Sastry. Model-predictive active steering and obstacle avoidance for autonomous ground vehicles. *Control Engineering Practice*, 17(7):741–750, 2009.

[61] J-M Park, D-W Kim, Y-S Yoon, H-J Kim, and K-S Yi. Obstacle avoidance of autonomous vehicles based on model predictive control. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 223(12):1499–1516, 2009.

[62] Masashi Nanao and Toshiyuki Ohtsuka. Nonlinear model predictive control for vehicle collision avoidance using c/gmres algorithm. In *International Conference on Control Applications*, pages 1630–1635. IEEE, 2010.

[63] Andrew Gray, Yiqi Gao, Theresa Lin, J Karl Hedrick, H Eric Tseng, and Francesco Borrelli. Predictive control for agile semi-autonomous ground vehicles using motion primitives. In *American Control Conference*, pages 4239–4244. IEEE, 2012.

[64] Yiqi Gao, Andrew Gray, Janick V Frasch, Theresa Lin, Eric Tseng, J Karl Hedrick, and Francesco Borrelli. Spatial predictive control for agile semi-autonomous ground vehicles. In *11th International Symposium on Advanced Vehicle Control*, 2012.

[65] Janick V Frasch, Andrew Gray, Mario Zanon, Hans Joachim Ferreau, Sebastian Sager, Francesco Borrelli, and Moritz Diehl. An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles. In *European Control Conference*, pages 4136–4141. IEEE, 2013.

[66] Alexander Katriniok and Dirk Abel. LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics. In *50th IEEE Conference on Decision and Control and European Control Conference*, pages 6828–6833. IEEE, 2011.

[67] Mohammad Ali, Andrew Gray, Yiqi Gao, J Karl Hedrick, and Francesco Borrelli. Multi-objective collision avoidance. In *Dynamic Systems and Control Conference*. ASME, 2013.

[68] Jeong hwan Jeon, Raghvendra V Cowlagi, Steven C Peters, Sertac Karaman, Emilio Frazzoli, Panagiotis Tsiotras, and Karl Iagnemma. Optimal motion planning with the half-car dynamical model for autonomous high-speed driving. In *American Control Conference*, pages 188–193. IEEE, 2013.

[69] 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.

[70] Alexander Liniger, Alexander Domahidi, and Manfred Morari. Optimization-based autonomous racing of 1:43 scale RC cars. *Optimal Control Applications and Methods*, 36(5):628–647, 2015.

[71] Rachid Attia, Jérémie Daniel, Jean-Philippe Lauffenburger, Rodolfo Orjuela, and Michel Basset. Reference generation and control strategy for automated vehicle guidance. In *Intelligent Vehicles Symposium (IV)*, pages 389–394. IEEE, 2012.

[72] Julia Nilsson, Paolo Falcone, Mohammad Ali, and Jonas Sjöberg. Receding horizon maneuver generation for automated highway driving. *Control Engineering Practice*, 41:124–133, 2015.

[73] GP Bevan, H Gollee, and J O'Reilly. Trajectory generation for road vehicle obstacle avoidance using convex optimization. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 224(4): 455–473, 2010.

[74] Stephen M Erlien, Susumu Fujita, and Joseph Christian Gerdes. Shared steering control using safe envelopes for obstacle avoidance and vehicle stability. *IEEE Transactions on Intelligent Transportation Systems*, 17(2):441–451, 2016.

[75] Christoph Voser, Rami Y Hindiyeh, and J Christian Gerdes. Analysis and control of high sideslip manoeuvres. *Vehicle System Dynamics*, 48(S1):317–336, 2010.

[76] Tulga Ersal, Burit Kittirungsi, Hosam K Fathy, and Jeffrey L Stein. Model reduction in vehicle dynamics using importance analysis. *Vehicle System Dynamics*, 47(7):851–865, 2009.

[77] Uwe Kiencke and Lars Nielsen. *Automotive control systems: for engine, driveline, and vehicle*. Springer Science & Business Media, 2005.

[78] Rajesh Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[79] M Azman, PD King, and Homer Rahnejat. Combined bounce, pitch, and roll dynamics of vehicles negotiating single speed bump events. *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics*, 221(1):33–40, 2007.

[80] Nicholas Cooper, David Crolla, Martin Levesley, and Warren Manning. Integration of active suspension and active driveline to ensure stability while improving vehicle dynamics. Technical report, SAE Technical Paper, 2005.

[81] C Ghike, T Shim, and J Asgari. Integrated control of wheel drive-brake torque for vehicle-handling enhancement. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 223(4):439–457, 2009.

[82] Terry D Day, Sydney G Roberts, and Allen R York. Simon: A new vehicle simulation model for vehicle design and safety research. Technical report, SAE Technical Paper, 2001.

[83] Taehyun Shim and Daniel Toomey. Investigation of active steering/wheel torque control at the rollover limit maneuver. Technical report, SAE Technical Paper, 2004.

[84] Taehyun Shim and Donald Margolis. Dynamic normal force control for vehicle stability enhancement. *International Journal of Vehicle Autonomous Systems*, 3(1):1–14, 2005.

[85] Taehyun Shim and Chinar Ghike. Understanding the limitations of different vehicle models for roll dynamics studies. *Vehicle System Dynamics*, 45(3):191–216, 2007.

[86] Jian Jun Zhu, Amir Khajepour, and Ebrahim Esmailzadeh. Dynamic response of a vehicle with planar suspension system (pss) under differential braking. *Vehicle System Dynamics*, 50(1):19–41, 2012.

[87] Justin Madsen. A stochastic framework for ground vehicle simulation. Master's thesis, University of Wisconsin, Madison, 2009.

[88] Yiqi Gao. *Model Predictive Control for Autonomous and Semiautonomous Vehicles*. PhD thesis, University of California, Berkeley, 2014.

[89] Muhammad Awais Abbas, Ruth Milman, and J Mikael Eklund. Obstacle avoidance in real time with nonlinear model predictive control of autonomous vehicles. In *IEEE 27th Canadian Conference on Electrical and Computer Engineering*. IEEE, 2014.

[90] HB Pacejka and IJM Besselink. Magic formula tyre model with transient properties. *Vehicle System Dynamics*, 27(S1):234–249, 1997.

[91] Dennis N Assanis, Walter Bryzik, Nabil Chalhoub, Zoran S Filipi, Naeim Henein, Dohoy Jung, Xiaoliu Liu, Loucas Louca, John Moskwa, Scott Munns, et al. Integration and use of diesel engine, driveline and vehicle dynamics models for heavy duty truck simulation. Technical report, SAE Technical Paper, 1999.

[92] Dennis N Assanis, Zoran Filipi, Steve Gravante, Dan Grohnke, Xinqun Gui, Loucas Louca, Geoff Rideout, Jeffrey Stein, and Yongsheng Wang. Validation and use of simulink integrated, high fidelity, engine-in-vehicle simulation of the international class VI truck. Technical report, SAE Technical Paper, 2000.

[93] Loucas S Louca, Jeffrey L Stein, and D Geoff Rideout. Generating proper integrated dynamic models for vehicle mobility using a bond graph formulation. *Ann Arbor*, 1001:48109–2121, 2001.

[94] Hyung Min Kim, Michael Kokkolaras, Loucas S Louca, George J Delagrammatikas, Nestor F Michelena, Zoran S Filipi, Panos Y Papalambros, Jeffrey L Stein, and Dennis N Assanis. Target cascading in vehicle redesign: a class VI truck study. *International Journal of Vehicle Design*, 29(3):199–225, 2002.

[95] Tulga Ersal, Mark Brudnak, Ashwin Salvi, Jeffrey L Stein, Zoran Filipi, and Hosam K Fathy. Development and model-based transparency analysis of an internet-distributed hardware-in-the-loop simulation platform. *Mechatronics*, 21(1):22–29, 2011.

[96] Karl Hedrick et al. Brake system modeling, control and integrated brake/throttle switching phase I. *California Partners for Advanced Transit and Highways*, 1997.

[97] Hans Pacejka. *Tire and vehicle dynamics*. Elsevier, 2005.

[98] Alessandro Rucco, Giuseppe Notarstefano, and John Hauser. Optimal control based dynamics exploration of a rigid car with longitudinal load transfer. *IEEE Transactions on Control Systems Technology*, 22(3):1070–1077, 2014.

[99] Tulga Ersal, Hosam K Fathy, D Geoff Rideout, Loucas S Louca, and Jeffrey L Stein. A review of proper modeling techniques. *Journal of Dynamic Systems, Measurement, and Control*, 130(6):061008, 2008.

[100] Edwin Brock Olson. *Robust and efficient robotic mapping*. PhD thesis, Massachusetts Institute of Technology, 2008.

[101] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, 1973.

[102] Bala R Vatti. A generic solution to polygon clipping. *Communications of the ACM*, 35(7):56–63, 1992.

[103] Angus Johnson. Clipper - an open source freeware library for clipping and offsetting lines and polygons., 2014. URL http://www.angusj.com/delphi/clipper.php.

[104] Mark Keil and Jack Snoeyink. On the time bound for convex decomposition of simple polygons. *International Journal of Computational Geometry & Applications*, 12(03):181–192, 2002.

[105] Yoshiaki Kuwata and Jonathan How. Receding horizon implementation of MILP for vehicle guidance (I). In *American Control Conference*, volume 4, page 2684, 2005.

[106] Tobias Achterberg. Scip: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[107] Robin Deits and Russ Tedrake. Efficient mixed-integer planning for uavs in cluttered environments. In *International Conference on Robotics and Automation*, pages 42–49. IEEE, 2015.

[108] Anil V Rao. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135(1):497–528, 2009.

[109] Christopher L Darby, William W Hager, and Anil V Rao. An hp-adaptive pseudospectral method for solving optimal control problems. *Optimal Control Applications and Methods*, 32(4):476–502, 2011.

[110] Christopher L Darby. *hp-Pseudospectral Method for Solving Continuous-Time Nonlinear Optimal Control Problems*. PhD thesis, University of Florida, 2011.

[111] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[112] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[113] Subhash Rakheja and Alain Piche. Development of directional stability criteria for an early warning safety device. Technical report, SAE Technical Paper, 1990.

[114] Claire Lydia Walton. *The Design and Implementation of Motion Planning Problems Given Parameter Uncertainty*. PhD thesis, University of California, Santa Cruz, 2015.

[115] Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In *Robustness in identification and control*, pages 207–226. Springer, 1999.

[116] Ali A Jalali and Vahid Nadimi. A survey on robust model predictive control from 1999-2006. In *CIMCA/IAWTIC*, page 207, 2006.

[117] Veronica Adetola and Martin Guay. Robust adaptive MPC for constrained uncertain nonlinear systems. *International Journal of Adaptive Control and Signal Processing*, 25(2):155–167, 2011.

[118] Ashwin Carvalho, Stéphanie Lefévre, Georg Schildbach, Jason Kong, and Francesco Borrelli. Automated driving: The role of forecasts and uncertainty - a control perspective. *European Journal of Control*, 24:14–32, 2015.

[119] DR Ramirez, T Alamo, EF Camacho, and D Munoz de la Pena. Min-max MPC based on a computationally efficient upper bound of the worst case cost. *Journal of Process Control*, 16(5):511–519, 2006.

[120] John M Carson, Behçet Açıkmeşe, Richard M Murray, and Douglas G MacMartin. A robust model predictive control algorithm augmented with a reactive safety mode. *Automatica*, 49(5):1251–1260, 2013.

[121] Yiqi Gao, Andrew Gray, H Eric Tseng, and Francesco Borrelli. A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles. *Vehicle System Dynamics*, 52(6):802–823, 2014.

[122] Georg Schildbach, Lorenzo Fagiano, Christoph Frei, and Manfred Morari. The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations. *Automatica*, 50(12):3009–3018, 2014.

[123] Ashwin Carvalho, Yiqi Gao, Andrew Gray, H Eric Tseng, and Francesco Borrelli. Predictive control of an autonomous ground vehicle using an iterative linearization approach. In *16th International IEEE Conference on Intelligent Transportation Systems*, pages 2335–2340. IEEE, 2013.

[124] Michael Maiworm, Tobias Bätlige, and Rolf Findeisen. Scenario-based model predictive control: Recursive feasibility and stability. *IFAC-PapersOnLine*, 48 (8):50–56, 2015.

[125] Fenfen Xiong, Ying Xiong, and Bin Xue. Trajectory optimization under uncertainty based on polynomial chaos expansion. In *AIAA Guidance, Navigation, and Control Conference*, page 1761, 2015.

[126] Dongbin Xiu and George Em Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24 (2):619–644, 2002.

[127] Felipe AC Viana. Things you wanted to know about the latin hypercube design and were afraid to ask. In *10th World Congress on Structural and Multidisciplinary Optimization*, pages 1–9, 2013.

[128] Felipe AC Viana, Gerhard Venter, and Vladimir Balabanov. An algorithm for fast optimal latin hypercube design of experiments. *International Journal for Numerical Methods in Engineering*, 82(2):135–156, 2010.

[129] Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013.

[130] Kawasaki mule. https://www.kawasaki.com/Products/2017-Mule-4010-4x4. Accessed: 2016-12-13.

[131] VectorNav VN-3000 rugged dual antenna GPS/INS. http://www.vectornav.com/products/vn300-rugged. Accessed: 2016-12-13.

[132] ibeo LUX 8L laserscanner. http://www.autonomoustuff.com/product/ibeo-lux-8l/. Accessed: 2016-12-13.

[133] DRIVER system. http://www.quantumsignal.com/robotics/teleoperation/driver/. Accessed: 2016-12-13.

[134] Introducing MATLAB engine api for C/C++ and Fortran. http://www.mathworks.com/help/matlab/matlab_external/introducing-matlab-engine.html. Accessed: 2016-12-13.

[135] Google map satellite view. https://www.google.com/maps/. Accessed: 2016-12-13.

[136] Shengbo Eben Li, Shaobing Xu, and Dongsuk Kum. Efficient and accurate computation of model predictive control using pseudospectral discretization. *Neurocomputing*, 177:363–372, 2016.

[137] Performance comparison of NLP solvers with GAMS globallib. https://www.coin-or.org/GAMSlinks/benchmarks/NLP/allSolver_110304/index.html. Accessed: 2016-12-13.

[138] MUMPS: a multifrontal massively parallel sparse direct solver. http://mumps.enseeiht.fr/. Accessed: 2016-12-13.

[139] The HSL mathematical software library. http://www.hsl.rl.ac.uk/. Accessed: 2016-12-13.

[140] Jia Kang. *An efficient interior-point decomposition algorithm for parallel solution of large-scale nonlinear problems with significant variable coupling*. PhD thesis, Texas A&M University, 2015.

[141] Huckleberry Febbo, Jiechao Liu, Paramsothy Jayakumar, Jeffrey L Stein, and Tulga Ersal. Moving obstacle avoidance for large, high-speed autonomous ground vehicles. In *American Control Conference*. in review, 2017.