Decomposition and Hamilton-Jacobi-Bellman Methods for Nonconvex Generalized Nash Equilibrium Problems

Andreas Britzelmeier

Vollständiger Abdruck der von der Fakultät für Luft- und Raumfahrttechnik der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Gutachter: 1. Prof. Dr. rer. nat. Matthias Gerdts
2. Prof. Dr. rer. nat. Karl Worthmann

Die Dissertation wurde am 07.06.2021 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 29.10.2021 angenommen. Die mündliche Prüfung fand am 04.11.2021 statt.

Andreas Britzelmeier
Decomposition and Hamilton-Jacobi-Bellman Methods for Nonconvex Generalized Nash Equilibrium Problems
MMXXI

First printing, June 2021

DOCUMENT BUILD
December 21, 2021 12:21pm | 48° 8′ 6.4464″ N 11° 34′ 55.1316″ E
This document was typeset in LaTeX using LuaTeX 1.0.4,
TeXLive 2017/Debian with TeXstudio 3.1.1.
The graphics are generated using `pgfplots` and `tikz`.
The bibliography is typeset using `bibtex`.

SUPERVISOR
Prof. Dr. rer. nat. Matthias Gerdts

INSTITUTION
Bundeswehr University Munich
Werner-Heisenberg-Weg 39
85577 Neubiberg

Decomposition and Hamilton-Jacobi-Bellman Methods for Nonconvex Generalized Nash Equilibrium Problems

Andreas Britzelmeier

Abstract

The subject of this dissertation is concerned with the theoretical investigation and the development of numerical methods for generalized Nash equilibrium problems subject to individual differential equations, state and control constraints as well as shared nonconvex constraints, complemented by an experimental validation of the methodological concepts.

To cope with the nonconvex shared constraints a partial penalty reformulation is proposed. A decomposition method with penalty selection, capable of solving the ensuing semi-discrete (constrained) Nash equilibrium problem is devised, which prevents the a priori imposition of hierarchies. Leveraging the underlying potential game's inherent structure, convergence of a sequence of solutions of the semi-discretized Nash equilibrium problem to a solution of the generalized Nash equilibrium problem is proven.

Then, a Hamilton-Jacobi-Bellman approach for solving standard Nash equilibrium problems, which constitute the subproblems of the decomposition method is pursued. Accordingly, a backward Semi-Lagrangian method for the numerical solution of the Hamilton-Jacobi-Bellman equation is devised. In conjunction with a suitable state space discretization, an approximate value function, for Nash equilibrium problems, satisfying the principle of dynamic programming is constructed. We then introduce an iterative algorithm for the recursive computation of optimal controls. Convergence to a global minimizer in finite time is shown based on an error estimate for the approximated value function. Furthermore, we investigate the convergence properties of higher-order Semi-Lagrange schemes in the context of time and control discrete value functions, as well as the influence of the order of approximation of the control on the resulting convergence rate. Eventually, exploiting the recursive problem structure of dynamic programming and a batching strategy, a GPU parallel dynamic programming algorithm is developed. Scalability and efficiency of the algorithm is demonstrated in a benchmark with CPU implementations.

Eventually, the numerical and experimental validation of the theoretical and algorithmic framework is conducted with respect to two tangible exemplary applications. The algorithmic framework is embedded in a model predictive control structure. To validate the feasibility of online path planning using the GPU parallel dynamic programming algorithm, a higher dimensional truck path planning problem is considered. In addition, an autonomous vehicle coordination problem modelled as a generalized Nash equilibrium problem with non-convex shared collision avoidance constraints is investigated. Numerical results instigate the capability of the algorithmic conceptualization to efficiently provide collision-free coordination by means of Nash equilibria. Based on the theoretical results, convergence properties are established. For the experimental evaluation, a test bench comprising a positioning system, a cloud for the online coordination and vehicle communication, as well as nonholonomic robots equipped with path-following controllers are developed. Numerical and experimental studies are conducted to validate the robustness and efficiency of the presented algorithms in the presence of measurement inaccuracies and transmission losses.

Zusammenfassung

Inhalt dieser Dissertation ist die theoretische Untersuchung und die Entwicklung numerischer Methoden für verallgemeinerte Nash Gleichgewichtsproblem mit individuellen Differentialgleichungen, Zustands- und Steuerungsbeschränkungen und gemeinsamen nicht konvexen Restriktionen, sowie eine experimentelle Validierung der methodologischen Konzepte.

Es wird ein Konzept zur Umformulierung der nicht-konvexen Nebenbedingung durch die Einführung geeigneter Strafterme konzipiert. Für das daraus resultierende und teildiskretisierte (restringierte) Nash Gleichgewichtsproblem wird ein Diagonalisierungsalgorithmus entwickelt, der eine a priori Implikation von Hierarchien durch eine Regulierung der Spielerwahl mittels der Strafterme unterbindet. Ausgehend von der zugrundeliegenden Struktur des Potenzialspiels wird für eine Folge von diskreten Lösungen des teildiskretisierten Nash Gleichgewichtsproblems Konvergenz gegen eine Lösung des verallgemeinerten Nash Gleichgewichtsproblems nachgewiesen. Anschließend verfolgen wir einen Hamilton-Jacobi-Bellman Ansatz zur Lösung von Standard Nash Gleichgewichtsproblem, welche die Teilprobleme der Diagonalisierungsmethode konstituieren. Entsprechend leiten wir ein rückwärtiges Semi-Lagrange Verfahren zur numerischen Lösung der Hamilton-Jacobi-Bellman Gleichung her. In Kombination mit einer geeigneten Zustandsdiskretisierung konstruieren wir eine approximierte Wertefunktion für Nash Gleichgewichtsproblem, die dem Prinzip der dynamischen Programmierung genügt. Mittels einer Fehlerabschätzung für die approximierte Wertefunktion wird die Konvergenz des Algorithmus zu einem globalen Optimum in endlicher Zeit bewiesen. Weiterhin untersuchen wir die Konvergenzeigenschaften von Semi-Lagrange Verfahren höherer Ordnung im Kontext von zeit- und steuerungsdiskreten Wertfunktionen, sowie den Einfluss der Approximationsordnung der Steuerung auf die resultierende Konvergenzrate. Schließlich wird unter Ausnutzung der rekursiven Problemstruktur der dynamischen Programmierung und einer Batching-Strategie ein GPU paralleler dynamischer Programmierungsalgorithmus entwickelt. In einem Benchmark mit CPU-Implementierungen wird die Skalierbarkeit und Effizienz des Algorithmus demonstriert. Letztendlich wenden wir uns der numerischen und experimentellen Validierung der theoretischen und algorithmischen Konzeptionierung anhand zweier anwendungsorientierter Beispiele zu. Die Algorithmen werden in eine Modell-Prädiktive Regelungsstruktur eingebettet. Die Realisierbarkeit einer Online-Bahnplanung mittels des GPU parallelen dynamischen Programmierungsalgorithmus wird anhand eines höher dimensionalen Bahnplanungsproblems eines Lastwagens validiert. Weiterhin betrachten wir ein Koordinierungsproblem autonomer Fahrzeuge, welches als ein verallgemeinertes Nash Gleichgewichtsproblem mit nicht konvexen, gemeinsamen Restriktionen zur Kollisionsvermeidung modelliert wird. Numerische Ergebnisse zeigen, dass die algorithmische Konzeptionierung eine effiziente und kollisionsfreie Koordinierung der Fahrzeuge durch Nash Gleichgewichte gewährleistet. Die Konvergenzeigenschaften werden auf Basis der theoretischen Ergebnisse nachgewiesen. Für die experimentelle Auswertung wird ein Prüfstand mit einem Positionierungssystem, einer Cloud für die online Koordinierung und Fahrzeugkommunikation, sowie nichtholonome Roboter, ausgerüstet mit Bahnfolgereglern, entwickelt. Numerische und experimentelle Studien bestätigen letztlich die Robustheit und Effizienz der vorgestellten Algorithmen unter Messungenauigkeiten und Übertragungsverlusten.

Acknowledgments

Whilst this thesis marks the end of a chapter of my everlasting adventure, it is the people with whom one has shared the often arduous road that matter. There are many inspiring and amazing people I am grateful to have met along the way who have helped me reach my destination.

First and foremost, I would like to express my deep gratitude to my thesis advisor, *Matthias Gerdts*, who welcomed me into his group, for his continued guidance, patience, inspiration and an endless supply of fascinating projects, as well as his counsel on teaching and research. His unassuming approach to research and science is a source of inspiration. He always had an open ear for my questions and at all times his door was open. Moreover, he gave me the freedom to explore the vastness of mathematics to an extent that I do not take for granted. I have benefited greatly from your invaluable wealth of knowledge and enthusiasm which constantly motivated me to reach further. Vielen Dank, Matthias.

Also I want to extend my appreciation and sense of respect to *Kurt Chudej*, who introduced me to numerical optimal control and encouraged me to pursue a doctorate in Munich. He was supervisor of my Bachelor's thesis and co-author of my first scientific publication. Thank you for taking a chance on me and believing in me.

A heartfelt gratitude goes to my colleague *Axel Dreves*, who introduced me to Nash equilibria, taught me the game of mathematics and offered advice and encouragement with a perfect blend of insight and humor. Thank you for your patience, valuable suggestions, ever encouraging and motivating guidance.

I wish to extend my sincere gratitude to *Karl Worthmann* for accepting to review my work and for taking the time to carefully read this document. Thank you for your valuable suggestions.

Moreover, I am grateful to my incredible colleagues in the Engineering Mathematics group who accompanied me along my path and made the time at the Bundeswehr University invaluable to me. Particularly, I would like to thank *Thomas Rottmann* and *Alberto De Marchi* for your wholehearted support, both professionally and morally.

Last, but not least, my warm and heartfelt thanks goes to my family, with my parents *Heidemarie* and *Hubert Britzelmeier*, and my brother *Thomas Britzelmeier*, for your endless support. Throughout my life you have always stood by my side, and this was no exception. Thank you for all of your love and for always encouraging me to pursue my goals. Thank you to my grandparents, *Johanna Urban*, *Maria Hölzle* with my late grandpa *Konrad Hölzle*, for always being there for me. I love all of you!

Publications

The following publications are included in parts or in an extended version in this thesis:

- Britzelmeier, A., Dreves, A., and Gerdts, M. (2019). Numerical solution of potential games arising in the control of cooperative automatic vehicles. In *2019 Proceedings of the Conference on Control and Its Applications*, Proceedings, pages 38–45. Society for Industrial and Applied Mathematics.

- Britzelmeier, A., Gerdts, M., and Rottmann, T. (2020a). Control of interacting vehicles using model-predictive control, generalized Nash equilibrium problems, and dynamic inversion. *IFAC-PapersOnLine*, 53(2):15146–15153.

- Britzelmeier, A. and Dreves, A. (2020). A decomposition algorithm for Nash equilibria in intersection management. *Optimization*, pages 1–38.

Furthermore, the following publications were part of my doctoral research, are however not explicitly covered in this thesis. The topics of these publications are outside of the scope of the material covered here:

- Britzelmeier, A. and Gerdts, M. (2018). Non-linear Model Predictive Control of Connected, Automatic Cars in a Road Network Using Optimal Control Methods. *IFAC-PapersOnLine*, 51(2):168–173.

- Britzelmeier, A., De Marchi, A., and Gerdts, M. (2018). An Iterative Solution Approach for a Bi-level Optimization Problem for Congestion Avoidance on Road Networks. In Falcone, M., Ferretti, R., Grüne, L., and McEneaney, W. M., editors, *Numerical Methods for Optimal Control Problems*, Springer INdAM Series, pages 23–38. Springer International Publishing, Cham.

- Britzelmeier, A. and Gerdts, M. (2020). A Nonsmooth Newton Method for Linear Model-Predictive Control in Tracking Tasks for a Mobile Robot With Obstacle Avoidance. *IEEE Control Systems Letters*, 4(4):886–891.

Contents

1 | INTRODUCTION

> *And this I believe: that the free, exploring mind of the individual human is the most valuable thing in the world. And this I would fight for: the freedom of the mind to take any direction it wishes, undirected.*
>
> — JOHN STEINBECK, EAST OF EDEN

Conflict is the apex of the selfishness of mankind. In all aspects of our lives, conflicts arise due to diverging or competing interests, such as in daily traffic, businesses, or simply when it comes to the economic interests of countries. If not resolved the resulting tensions culminate in dispute or as history has taught us even in war. Either leading to an unilateral improvement of the status of an individual while simultaneously impairing the opposition, or even worse, to a deterioration of the state of affairs of all involved. In order to avoid or resolve these tensions a consensus among the respective sides must be sought. Game theory provides a framework to characterize and analyse such conflict situations mathematically. In particular, the solution concept developed by *John F. Nash Jr.* (1928-2015) in [Nash, 1950, Nash, 1951] deals with this very issue by characterizing an equilibrium solution in which the individual cannot improve by unilaterally changing his actions while the others remain with their chosen course of action. While the equilibrium concept goes back as far as 1838 to [Cournot, 1838], where it was formally introduced in the context of oligopolistic economics, the foundation of game theory is indubitably rooted in the works of [von Neumann, 1928] and [von Neumann et al., 1944] concerning zero-sum games. A generalized form of Nash's equilibrium, which considers the influence of the adversaries' strategies on the set of strategies available to an individual player, was introduced in [Debreu, 1952] and [Arrow and Debreu, 1954]. While *K.J. Arrow* and *G. Debreu* introduced this generalization by the term social equilibrium, today it is most commonly known as *generalized Nash equilibrium*. Existence and uniqueness properties of generalized Nash equilibria became the center of scientific attention in the ensuing years. In 1965, in [Rosen, 1965] existence and uniqueness results together with the introduction of normalized solutions were presented by *J.B. Rosen*. Moreover, in [Rosenthal, 1973] some existence results in terms of pure strategy Nash equilibria were proven.

Whenever a shared interest in limited resources arises, generalized Nash equilibrium problems (GNEPs) naturally emanate from Nash equilibrium problems (NEPs). Consequently the GNEP soon established itself in many different fields of application as a versatile and powerful modeling tool, for instance in cloud computing [Ardagna et al., 2011, Cardellini et al., 2016], electricity generation [Nasiri and Zaccour, 2010, Contreras et al., 2013] or wireless communication [Han et al., 2012, Mochaourab et al., 2012]. However, as the interest in engineering applications increased, so did the necessity for computational methods to effectively solve the GNEP. Inherently, due to their complex structure GNEPs are challenging to solve and especially demanding in their analysis. The general approach therefore is to reformulate the problems into well known problem structures, such as optimization problems, Variational and Quasi-Variational Inequality (QVI) problems or fixed-point problems, leveraging the established methods and theory therein.

A special and widely studied subclass of GNEPs, are the jointly convex problems, where the strategy set of each player is convex. For these problems, many algorithms based on an optimization reformulation using the Nikaido-Isoda function [Nikaidô and Isoda, 1955] emanated in the recent years [Uryas'ev and Rubinstein, 1994][Dreves and Kanzow, 2011][von Heusinger et al., 2012]. Moreover, [Facchinei et al., 2007, Harker, 1991] and [Facchinei and Pang, 2009] suggest the reformulation of the jointly convex problem in terms of a Variational Inequality (VI) problem. However, these methods are only applicable to jointly convex problems. Unfortunately, most problems which arise in the context of engineering applications do not possess these convenient convexity properties and are usually more complex. Nevertheless, some suitable approaches for the solution of the general GNEP emerged.

In [Bensoussan, 1974] and [Harker, 1991] the coherence between QVI and GNEP was established. While QVI, indeed, are a generalization of VIs and the latter are theoretically and numerically well studied, compare [Facchinei and Pang, 2003], theoretical and algorithmic results for QVIs are still scarce. First existence results as well as a globally convergent algorithm based on a projection method are presented in [Chan and Pang, 1982]. Algorithmic approaches using gap functions with respect to QVI have been introduced in [Kubota and Fukushima, 2010, Fukushima, 2007, Taji, 2008]. In [Fukushima, 2007] a class of gap functions reducing the solution of QVI to the global minimization of a non differentiable gap function is presented. However, no algorithms are reported. A regularized gap function algorithm, using barrier techniques is proposed by [Kubota and Fukushima, 2010], which sequentially solves a GNEP with shared constraints. An extension to the regularized gap function approach then was introduced by [Taji, 2008]. Projection like methods, for solving the GNEP, which under certain assumptions converge globally are given in [Zhang et al., 2010]. [Pang and Fukushima, 2009] proposed a sequential penalty approach for general QVI, which reduces the solution of the QVI to a sequence of solutions of VIs under strong convergence assumptions. Considering variational inequalities, global uniqueness and existence results, as well as iterative algorithms and parametrised approaches can be found in [Facchinei and Pang, 2009, Nabetani et al., 2011]. While using the concatenated Karush-Kuhn-Tucker (KKT) system of all players is usually associated with an optimization reformulation, in [Facchinei et al., 2014] the KKT conditions are applied with respect to a QVI reformulation. Nevertheless, once a KKT system is acquired, suitable Newton type methods can be employed, see [Facchinei

et al., 2009, Dreves et al., 2014, Dreves, 2016]. In [Dreves et al., 2014, Dreves, 2016] error bounds are given for a hybrid algorithmic approach combining potential reduction and Newton methods, which provides global and fast local convergence. Various semismooth Newton approaches and their applications are investigated by [Facchinei et al., 2009]. A merit function approach as well as an interior point algorithm with global convergence results have been suggested in [Dreves et al., 2011, Dreves, 2011]. A more intuitive approach using an exact partial penalization is proposed in [Facchinei and Lampariello, 2011], where convergence is established using the properties of the KKT conditions.

Penalty methods provide a convenient way to rid the (usually nonconvex) GNEP of the troublesome shared constraints, effectively yielding a simpler NEP which in turn is solved using known methods from optimization or VI, see [Facchinei and Pang, 2006, Fukushima, 2011]. In [Facchinei and Pang, 2006] an exact penalty approach is suggested and conditions when a penalty reformulation is suitable are described in [Fukushima, 2011]. [Pang and Fukushima, 2009] were the first to inspire the application of a penalty reformulation with respect to GNEPs. However, with the exception of [Ba and Pang, 2020], which provide a study on exact penalization of GNEP and [Facchinei and Lampariello, 2011] and [Facchinei and Kanzow, 2010b], where a globally convergent penalty approach based on a gradient method along with numerical results is provided, convergence results as well as the theoretical investigation of penalty reformulations for GNEP are still in its infancy. If the GNEP omits a special structure it can be formulated in terms of a generalized potential game (GPG), see [Facchinei et al., 2011]. Moreover, in [Facchinei et al., 2011] the penalty approach is combined with a regularized Gauss-Seidel type decomposition method. Due to the nature of a game, where each player means to optimize his gain, decomposition methods in practice provide an intuitive algorithmic solution approach, see [Facchinei and Kanzow, 2010a] for a survey on this topic. Most of the approaches available for the solution and theoretical investigation of general GNEPs are either based on the reformulation to an optimization problem or of a VI, which subsequently are solved by suitable gradient methods or methods devised to solve the KKT system. Global convergence results are scarce or subject to strict conditions.

In the course of this work, an alternative solution approach for general nonconvex GNEPs based on the principle of dynamic programming, also known as Bellman's principle, see [Bellman, 1957], is introduced. In order to provide convergence results an algorithmic framework is developed, which fruitfully combines ideas, from nonconvex GNEPs, penalty methods, decomposition methods and dynamic programming. Since the subproblems of the individual players are usually coupled through nonconvex shared constraints, a partial penalty reformulation is employed to shift the troublesome constraints to the objective function. Moreover, a generalized decomposition method of Gauss-Seidel type, which takes into account the iterative alteration of the penalty terms is proposed. It can be shown that the reformulated GNEPs, without loss of generality, indeed are generalized potential games, making the decomposition method a suitable choice. Compared to the regularized Gauss-Seidel approach in [Facchinei et al., 2011], the penalty approach in this thesis guarantees that the subproblems of the respective GNEP are always solvable. A global solution of the discrete problem, then is computed employing approximate

dynamic programming, [Bertsekas, 2005]. Due to the curse of dimensions, the computational effort of dynamic programming increases exponentially with the dimension of the problem. Therefore, exploiting the structure and large dimensions of the spatial discretisation, a graphics processing unit (GPU) parallel dynamic programming algorithm is developed.

For the numerical and experimental validation of the proposed theoretical and algorithmical framework, a coordination problem of autonomous vehicles is considered. The objective of the problem is to navigate the vehicles to their desired destination by computation of optimal and collision free velocity profiles. In the context of the progressing automation of traffic, this problem is currently a crucial topic of research, since the safety of all traffic participants plays an essential role in autonomous driving. With respect to the experimental validation, the coordination and computation of velocity profiles needs to be feasible in real time. Therefore the proposed algorithmic framework is embedded in a model predictive control (MPC) setting, see [Grüne and Pannek, 2011]. Allowing us to define a model predictive dynamic programming algorithm, which on top of the GPU parallel algorithm yields a further amelioration of the computation time by reducing the dimensional load imposed on dynamic programming through the *curse of dimensions*.

The task of conflict avoidance employing multiple autonomous vehicles gives rise to numerous applications including intersection management, platooning or general path planning, which motivates the separation of the problem into a coordination problem, by means of collision free velocity profiles and a path following problem, under the assumption that each vehicle is assigned a precomputed trajectory, which may be obtained from a navigational system. On the premise of autonomous vehicles, we assume for our problem that the vehicles are able to communicate and have perfect information. Various approaches have emerged in recent years to address these problem classes, which can be distinguished into two groups. Either there is a central controller unit, which derives a hierarchy according to a prescribed set of rules, or the problem is addressed on the vehicle level in a decentralised manner. Considering the centralised approach, an intersection management problem with a central manager creating a schedule for the approaching vehicles is presented in [Dresner and Stone, 2008]. A bi-level model predictive controller scheme is proposed in [Hult et al., 2015] and [Hult et al., 2018, Hult et al., 2019], where time slots for each vehicle are allocated using a mixed-integer based heuristic on the upper level. On the lower level control policies for each vehicle are computed. In [Katriniok et al., 2017], semidefinite programming is used to cope with the nonconvex collision constraints, then a hierarchy based on fixed rules is computed. For a quadratic programming algorithm to solve a convex model predictive control coordination problem, see [Riegger et al., 2016]. The reduction of a hybrid problem to a linearized mixed integer problem for the coordination of a multi vehicle system is proposed in [Reinl and von Stryk, 2007]. In [Wang et al., 2018], the cooperative coordination of connected vehicles using an alternating direction method of multipliers is suggested. Model predictive control approaches with prescribed traffic rules are given in [Luo et al., 2016, Britzelmeier and Gerdts, 2018]. In [Dreves and Gerdts, 2018] a nonlinear MPC setting modelling the problem in each step of the MPC as a GNEP is considered. The players are coupled through the nonconvex collision avoidance constraints. While in [Dreves and Gerdts,

2018] a centralised controller is modelled, in this thesis a decentralised formulation is considered, see [Britzelmeier and Dreves, 2020], leading us to the second class of decentralised coordination approaches. An efficient distributed sequential quadratic programming algorithm with sensitivity updates is suggested in [Zanon et al., 2017]. [Makarem and Gillet, 2013] introduced a model predictive linear quadratic controller for an intersection crossing problem employing linear constraints to ensure collision avoidance and to reduce the computational effort. However, a different vehicle model and conflict constraints are used compared to the setting in this thesis. Another approach, where vehicles sequentially solve a local optimization problem with a velocity based negotiation for intersection crossing is provided in [de Campos et al., 2013]. [Alessandretti and Aguiar, 2020] suggest an a priori defined consensus path following setting, where vehicles share their coordination vector rather than their trajectories. Although, a variety of different approaches exists, the theoretical investigation, in particular convergence results are extremely scarce or nonexistent for these applications.

1.1 Outline and Contributions

This thesis aims to investigate the theory and develop numerical methods for nonlinear, nonconvex differential generalized Nash equilibrium problems with shared constraints, which can be conceived as finite-dimensional coupled optimal control problems. Moreover, the devised theory and methods are to be validated numerically as well as experimentally.

In Chapter 2, some fundamental results and definitions with respect to strategic games are briefly summarized. In particular, we provide an overview on game theoretical concepts associated with multi-agent coordination and establish a correlation linking cooperative and noncooperative games. Moreover, the solution concepts of Nash and Pareto are introduced. In the context of GNEPs we discuss existing approaches and their shortcomings, which later facilitates the subsumption of the problem class analysed within this thesis.

The cornerstone of this thesis is set in Chapter 3, where we state the class of differential GNEPs subject to nonconvex shared constraints and characterize its solution properties. Due to its coupling structure, standard approaches of continuous and discrete optimization are usually not suitable for these problems. A tailored partial penalty reformulation of the considered problem class is therefore proposed. Leveraging the potential game structure jointly with the penalty reformulation a decomposition algorithm capable of numerically solving the system of approximate Nash equilibrium problems is composed. Unlike existing decomposition methods, compare [Facchinei et al., 2011], the design of the proposed algorithm precludes the imposition of an a priori hierarchy and offers desirable convergence properties. We then expand the research on the convergence analysis of GNEPs and establish an approach bridging the discrepancy between the continuous and the penalized semi-discrete NEP. Albeit the principle ideas have already been published together with Axel Dreves in [Britzelmeier et al., 2019, Britzelmeier and Dreves, 2020], this chapter generalizes and extents these findings.

In Chapter 4, a value function reformulation for penalized NEPs is developed. It is based on the Hamilton-Jacobi-Bellman (HJB) equation, a solution of which is characterized by the value function. Then, a backward Semi-Lagrangian scheme for the numerical approximation of the solution of the Hamilton-Jacobi-Bellman (HJB) equation is proposed, extending and complementing the method in [Falcone and Ferretti, 2002]. Following a state space discretization, this yields an approximate value function. Together with an approximate dynamic programming (ADP) algorithm, harnessing the recursive properties of the control synthesis we derive an error estimate for the approximate value function and convergence to a global minimizer in finite time is shown. Moreover, we consider a control and time discrete higher-order backward Semi-Lagrangian scheme and procure the theoretical convergence properties of the approximation scheme to the continuous value function, extending the findings in [Falcone and Ferretti, 1998]. The methodology devised in this chapter can suitably be applied to solve the subproblems emanating in the decomposition method of Chapter 3. This optimization approach does not match any of the conventional approaches discussed in the previous section and its global convergence properties sets this method apart. Eventually, exploiting the dynamic programming structure we develop a versatile GPU parallel dynamic programming method which significantly improves the performance for large dimensional problems, and therefore relaxes the curse of dimensions to some extent. The algorithm is implemented in a Compute Unified Device Architecture (CUDA) and C++ routine. A benchmark together with a central processing unit (CPU) implementation of the ADP algorithm is conducted by means of an representative example and a summary of the numerical results is reported. For a particular application, the findings of this section except for the backward Semi-Lagrangian approximation, the convergence investigations of the higher order scheme and the parallel dynamic programming method, have been published in [Britzelmeier and Dreves, 2020].

Chapter 5 is dedicated to the numerical and experimental validation of the developed theoretical and algorithmic framework. Therefore, a traffic coordination problem and a higher dimensional path planning maneuver of a truck are considered. First, the algorithmic framework developed in the previous chapters is embedded in a nonlinear model predictive control (NMPC) setting, in order to achieve online computation capability, for the experimental validation. On that note, a brief overview on the concept of model predictive control is provided. Then, we derive the equations of motion, where for the coordination problem a longitudinal point mass model, see [Frego et al., 2016], and for the truck problem a single track model is developed. Accordingly, we investigate collision constraints as well as physical constraints and objectives. In particular, equivalency of objective functions with respect to their time minimal character are investigated. Then, we show that the coordination problem can be formulated as a decentralized differential game, by modeling the problem in each step of the NMPC as a GNEP. Moreover, within the provided theoretical framework convergence of the coordination problem is shown. The particular application was published in [Britzelmeier and Dreves, 2020]. The truck problem serves as high dimensional benchmark problem for the GPU parallel dynamic programming algorithm. Considering the experimental evaluation, a test bench for nonholonomic robots is constructed, where a HTC Vive System is utilized for positioning and a complex communication

system is developed. The nonholonomic robots are equipped with either a dynamic inversion controller or a linear model predictive controller for path tracking, [Britzelmeier and Gerdts, 2020, Britzelmeier et al., 2020b]. We report the experimental results and discuss the efficiency and robustness of the proposed algorithmic framework.
Chapter 6 summarizes and reflects on the main results. Furthermore, open questions and preceding research topics are discussed.

The Appendix, provides a collection of auxiliary results and definitions which meaningfully complement the theory and arguments provided in the preceding chapters.

1.2 NOTATIONS

Throughout this thesis the principle of a comprehensive, intuitive yet precise notation is pursued. To preserve conformity among the literature, wherever possible, we adhere to the standard notations in the respective fields, [Geiger and Kanzow, 2002, Rockafellar and Wets, 1998, Clarke, 1983]. In the interest of sententiousness, this section introduces essential conventions and established definitions. In view of the relevance to the ensuing discussions, we introduce some equivalence operators; a definition is henceforth denoted by $:=$ and indicates the equality of the expression on the left and on the right side of the symbol. The symbol $\equiv$ respectively states equivalency of the expressions.

SETS AND VECTORS

The set of real numbers is indicated by $\mathbb{R} := (-\infty, \infty)$. The set of the positive and strictly positive real numbers are conveniently defined by $\mathbb{R}_+ := [0, \infty)$ and $\mathbb{R}_{++} := (0, \infty)$. The set of integer numbers is denoted by $\mathbb{Z}$ and we adopt the convention

$$Z_0^+ := \{x \in \mathbb{Z} \mid x \geq 0\}, \qquad \mathbb{Z}^+ := \{x \in \mathbb{Z} \mid x > 0\}.$$

The set of natural numbers is denoted by $\mathbb{N} := \mathbb{Z}^+ \subset \mathbb{Z}$ and respectively $\mathbb{N}_0 = \mathbb{Z}_0^+$. An open interval is denoted by (a, b) or $]a, b[$, for given $a, b \in \mathbb{R}$ and a closed interval by $[a, b]$. The n-dimensional real valued vector space is constituted by $\mathbb{R}^n$ and the positive and strictly positive vector space are respectively denoted by $\mathbb{R}_+^n$ and $\mathbb{R}_{++}^n$. It will be distinguishable from the context if (a, b) indicates a pair or a vector in $\mathbb{R}^2$ such that no confusions may arise. Given a centre x and a radius r a closed ball, characterized by these metrics is indicated by $\mathbb{B}(x; r)$. For the closed unit ball of dimension n the notation $\mathbb{B}^n(x)$ is inaugurated. The closure of a subset $\mathcal{D} \subseteq \mathbb{R}^n$ is indicated by $\mathrm{cl}(\mathcal{D})$ and $\mathrm{int}(\mathcal{D})$ denotes the interior respectively. Then, the boundary of the subset $\mathcal{D}$ is defined as $\delta\mathcal{D} := \mathrm{cl}(\mathcal{D}) \backslash \mathrm{int}(\mathcal{D})$.

Given a vector $v \in \mathbb{R}^n$ and a matrix $M \in \mathbb{R}^{m \times n}$ with n, m indicating its dimensions, we write

$$v := \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} \qquad \text{and} \qquad M := \begin{pmatrix} v_{1,1} & \dots & v_{1,n} \\ \vdots & \ddots & \vdots \\ v_{m,1} & \dots & v_{m,n} \end{pmatrix}.$$

The i-th component of this vector is characterized by v_i. The elements of the matrix M are designated by subscripts i, j; namely $(m_{ij}) \in \mathbb{R}^{m \times n}$. In order to preempt any confusions that may arise, in the game theoretical setting the convention v_i^ν for the i-th component of the ν-th players' vector v is adopted. Throughout, the upper index designates the respective player. The transpose of v is denoted by $v^\top$. The n-dimensional unit vector with all elements equal to one is denoted by $\mathbb{1}$, the dimension will indubitably be clear from the context.

Sequences

The set $\{x_k \mid k \in K \subset \mathbb{N}\}$ is aggregated in the notation $\{x_k\}_{k \in K}$ and denotes a sequence assigning to every index $k \in K$ an element x_k in a given set X. Whenever $K = \mathbb{N}$ the index set may be neglected and we simply write $\{x_k\} \subset X$. Given a sequence of real numbers $\{x_k\} \subset \mathbb{R}^n$ with $k_n \in \mathbb{N}$ for all $n \in \mathbb{N}$ the notation $\{x_{k_n}\}$ denotes a subsequence of $\{x_k\}$.

Definition 1.1 (Accumulation Point)**.** *Let $\{x_k\} \subset \mathbb{R}^n$ be a sequence. Then $\hat{x}$ is said to be an accumulation point of that sequence if there exists a subsequence $\{x_{k_n}\}$ such that $\lim_{n\to\infty} x_{k_n} = \hat{x}$, i.e, for all $\varepsilon > 0$ there exists a $K \in \mathbb{N}$ such that, if $n \geq K$, $|x_{k_n} - \hat{x}| < \varepsilon$.*

Let us stress, that a bounded sequence in the real numbers always omits at least one accumulation point.

Infimum and Supremum

The function f is denoted to be extended real-valued, if it attains values in $\overline{\mathbb{R}} := [-\infty, \infty]$. The domain of f is characterized by the set

$$\operatorname{dom} f := \{x \in \mathbb{R}^n \mid |f(x)| < \infty\}.$$

Given a set $X \subset \mathbb{R}^n$ where $f : X \to \overline{\mathbb{R}}$ then the supremum of f is denoted by

$$\sup_{x \in X} f = \sup\{f(x) \mid x \in X\}$$

and the infimum by

$$\inf_{x \in X} f = \inf\{f(x) \mid x \in X\},$$

which characterise the least upper and the greatest lower bound respectively.

Definition 1.2 (Global Extrema). *Let $f : X \to \overline{\mathbb{R}}$ be a function on the extended real-valued set. Then, a global maximum of f is defined by*

$$\hat{x} \in \underset{x \in X}{\operatorname{argmax}} f(x) := \{x \in X \mid \forall y \in X, f(y) \leq f(x)\}$$

and respectively a global minimum is defined by

$$\hat{x} \in \underset{x \in X}{\operatorname{argmin}} f(x) := \{x \in X \mid \forall y \in X, f(y) \geq f(x)\}\,.$$

Definition 1.3 (Semicontinuity). *Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ and let $\hat{x} \in X \subseteq \mathbb{R}^n$. Then we say f is lower semicontinuous at $\hat{x}$ if and only if for every sequence $\{x_k\} \subseteq X$ that converges to $\hat{x}$,*

$$\limsup_{k \to \infty} f(x_k) \geq f(\hat{x}).$$

Accordingly, f is said to be upper semicontinuous at $\hat{x}$ if and only if

$$\liminf_{k \to \infty} f(x_k) \leq f(\hat{x}).$$

If the assertions hold for every $\hat{x} \in \mathbb{R}^n$, then f is lower respectively upper semicontinuous on $\mathbb{R}^n$.

SET-VALUED MAPPINGS

Let a point-to-set function be indicated by $\mathcal{F} : \mathbb{R}^n \rightrightarrows R^m$, i.e., the assignment of a set $\mathcal{F}(x) \subset \mathbb{R}^m$ to every $x \in \mathbb{R}^n$. Its domain is defined by

$$\operatorname{dom} \mathcal{F} := \{x \in \mathbb{R}^n \mid \mathcal{F}(x) \neq \emptyset\}\,.$$

The graph of $\mathcal{F}$ is characterized as a subset of $\mathbb{R}^n \times \mathbb{R}^m$, respectively

$$\operatorname{graph} \mathcal{F} := \{(x, y) \in \mathbb{R}^n \mid y \in \mathcal{F}(x)\}\,.$$

We speak of $\mathcal{F}$ as single valued, whenever $\mathcal{F}(x)$ at x is a singleton, i.e., it merely contains a single element. The notation $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^m$ is adopted in the case where $\mathcal{F}$ is single valued everywhere on $\mathbb{R}^n$. The following semi continuity property is perpetuated:

Definition 1.4 ([Rockafellar and Wets, 1998, Theorem 5.9]). *Consider a mapping $\mathcal{F} : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ and a point $\hat{x} \in \mathbb{R}^n$.*

- *Suppose $\mathcal{F}$ is convex-valued and $\operatorname{int} \mathcal{F}(\hat{x}) \neq \emptyset$, then $\mathcal{F}$ is inner semicontinuous relative to $\operatorname{dom}(\mathcal{F})$ if and only if for all $y \in \operatorname{int} \mathcal{F}(\hat{x})$ there exists a neighbourhood W of the point $(\hat{x}, y)$ such that $W \cap (\operatorname{dom}(\mathcal{F}) \times \mathbb{R}^m) \subset \operatorname{graph}(\mathcal{F})$)*
- *If $\mathcal{F}$ is graph-convex, i.e., the graph of $\mathcal{F}$ characterizes a convex set and $\hat{x} \in \operatorname{int}(\operatorname{dom}(\mathcal{F}))$, then $\mathcal{F}$ is inner semicontinuous at $\hat{x}$.*

Besides, some auxiliary notation should be inaugurated:

- The big-$\mathcal{O}$ notation is adopted. Given the sequences $\{x_k\} \subset \mathbb{R}$ and $\{\varepsilon_k\} \subset \mathbb{R}_{++}$ to indicate $\limsup_{k\to\infty} \frac{|x_k|}{\varepsilon_k} < \infty$ we write $x_k \in \mathcal{O}(\varepsilon_k)$.
- The projection $\Xi_{\mathcal{D}}[x]$ of a vector $x \in \mathbb{R}^n$ onto a nonempty, closed set $\mathcal{D} \subseteq \mathbb{R}^n$ is invoked by an Euclidean metric, that is

$$\Xi_{\mathcal{D}}[x] \coloneqq \underset{p\in\mathcal{D}}{\operatorname{argmin}} \ \frac{1}{2}\|p - x\|^2.$$

  Wherever the set $\mathcal{D}$ is convex the projection is unique.

2 Fundamentals of Game Theory

> *If people do not believe that mathematics is simple, it is only because they do not realize how complicated life is.*
>
> — John von Neumann

The intention of this chapter is to provide a comprehensive introduction to game theoretical paradigms and solution concepts, as well as crucial definitions and established results. The conventions and classifications provided here constitute the baseline upon which the remainder of this thesis is build on. In Section 2.1 concepts and criteria for the classification of games are discussed and the notions of cooperative and noncooperative games are established. In Sections 2.2 to 2.6 the solution concepts of Nash and Pareto are introduced. Then, the generalized Nash equilibrium problem is inaugurated. Subsequently, we state some fundamental definitions and provide a survey on reformulation approaches for solving GNEPs. For a more detailed introduction to game theory, especially with respect to Nash equilibrium problems we refer the interested reader to [Facchinei et al., 2009, Facchinei and Kanzow, 2010a] and [Fischer et al., 2014].

2.1 Coordination of Interacting Systems

The coordination of a system consisting of multiple agents, which usually pursue different targets and interact in a shared space, evoking conflicts, results in a complex interdependent set of problems. On a higher level, these problems can be addressed from two distinct perspectives. Either by employing a centralized or a decentralized setting.

A centralized approach is defined through a central controller unit which acts as a manager, prescribing rules to enforce a hierarchy. Furthermore, this includes centralization of information and the control of each individual agent, which is enforced through the central unit. A decentralized approach on the other hand refers to an implementation on the level of the individual

agent. Hence, conflicts are resolved by the agents themselves. Usually, in a decentralized setting, due to the absence of information aggregation, the state of information of each agent is limited or imperfect.

Naturally, the game theoretical approach arises from a decentralized setting, which will be the subject of this thesis, in order to investigate conflicts arising from multi agent systems. Therefore, in this section we briefly introduce some fundamental classifications and discuss approaches of game theoretical settings towards the coordination of multi agent systems. On a lower level, especially in game theory, the relation among the agents can be further distinguished by means of cooperation, the number of players or the state of information of each player. The concepts presented in this section are discussed in more detail in [Kanzow and Schwartz, 2018], [Osborne and Rubinstein, 1994] and [Krabs, 2005].

2.1.1 Classification of Games

A variety of classifications are proposed in the literature as a way to distinguish types of games and give a structure to the theory. In this thesis we mainly follow the definitions of [Kanzow and Schwartz, 2018] and [Osborne and Rubinstein, 1994]. With this in mind, let us first give a comprehensive characterization of a game. To this end, assume Z_ν is a vector space and $Z := \bigtimes_{\nu=1}^{N} Z_\nu$ denotes the respective product space. Along these lines, we refer to agents as players and throughout this thesis these terms are used equivalently.

Definition 2.1 (Strategic Game). *Let N indicate the number of players and $X_\nu \subset Z_\nu$ the set of strategies x^ν for each player $\nu = 1, \dots, N$. Moreover, let $\theta_\nu : X \to \mathbb{R}$ denote the utility or payoff function for every $\nu = 1, \dots, N$, where $X := \bigtimes_{\nu=1}^{N} X_\nu$ is a subset of the product space of all strategy sets. Then we call $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^{N}$ an N-player strategic game where every player $\nu = 1, \dots, N$ solves the optimization problem*

$$\min_{x^\nu} \quad \theta_\nu(x) \qquad \textit{subject to (s.t.)} \qquad x \in X \tag{2.1}$$

with the strategy vector $x := \left(x^1, \dots, x^N\right)$.

Remark 2.2. *An aspect which distinguishes game theory from the theory of optimization is related to the payoff function, which is not only dependent on the individual state variables, but might be subject to the variables of other players. Emphasizing the distinction in the terminology of game theory the respective objective function, as in (2.1), forthwith is indicated by $\theta_\nu(\cdot)$.*

Games can be categorized subject to three essential traits. The first property to consider is the *objective function* or *payoff function* for that matter. More precisely, which values it can assume. Essentially, a distinction can be drawn among three contingencies. First, if the payoff function becomes zero. Second, if it attains a constant value and lastly if it results in an arbitrary value distinct from the first two cases. As the first and the second case are unambiguously the same,

a mutual conversion is possible. Without sacrificing generality the following classes can be defined.

Definition 2.3 (Constant Sum Game)**.** *A game is called a constant sum game if there exists a constant* $c \in \mathbb{R}$ *such that* $\sum_{\nu=1}^{N} \theta_\nu(\varkappa) = c,\ \forall \varkappa \in X \coloneqq X_1 \times \ldots \times X_N$*. If the constant* $c = 0$*, then the game is called a zero sum game .*

Definition 2.4 (Non Constant Sum Game)**.** *A game is a non constant sum game if it is neither a constant nor a zero sum game.*

Another distinction can be made regarding the *state of information* of each player. Either a players' knowledge is restricted to his strategies and the current state of the game or he additionally has extensive information about the strategies and objectives of all the other players. Hence, a classification into games with perfect and with imperfect information can be made.

Definition 2.5. *A game is of perfect information if all players have knowledge about all strategies, payoff functions and possible outcomes of the game. If one or more players lack information about the game, i.e., strategies, payoff functions or outcomes, the game is of imperfect information.*

Further, a distinction concerning the *degree of cooperation* among the players may be drawn. In particular, we differentiate between cooperative and noncooperative games, which are discussed in more detail in the subsequent sections.

2.1.2 Cooperative Games

Contrary to noncooperative games, where the correlation between individual players is considered, cooperative games are concerned with the relation among groups of players. Further, the later also constitutes a subgame of noncooperative games. The outcome of such games are not individual strategies for each player but coalitions. Hence, often cooperative games are referred to as *coalitional games*. Therein, the inherent entanglements of players within a group are not of interest, rather how the coalitions are formed and more precisely how their payoff is distributed among the players to enforce a coalition. Following the definitions of [Osborne and Rubinstein, 1994] and [Krabs, 2005].

Definition 2.6. *A game of cooperative nature with transferable payoff is defined through a finite set of* N *players and a function* $C : 2^N \to \mathbb{R}$*, assigning to every nonempty subset* $S \subseteq \mathcal{N}$ *a value* $C(S)$*.*

The function $C(S)$ is the characteristic function of the coalition S, with $S \subseteq \mathcal{N}$ a subset of all players, assigning a worth to a certain coalitional configuration. A coalition then has the incentive to maximize (or minimize) a common or agreed upon goal which leads to optimal strategies $\hat{\varkappa}^\nu$ with $\nu \in S$, such that $\hat{\varkappa}$ maximizes the sum of their payoffs. In particular, for a

coalition S, with $S \neq \emptyset$, individual strategy sets X_ν, payoff functions θ_ν and with a feasible set $X = X_1 \times \ldots \times X_N$ the common goal is characterized by

$$\Theta_S(\varkappa) := \sum_{\nu \in S} \theta_\nu(\varkappa) \qquad \forall\ \varkappa \in X. \tag{2.2}$$

The corresponding characteristic function of (2.2) then is defined by

$$C(S) = \sup_{\varkappa^\nu, \nu \in S} \inf_{\varkappa^\nu, \nu \in \mathcal{N} \setminus S} \Theta_S(\varkappa).$$

Suppose, for all players $\theta_\nu(\varkappa) \geq 0$, and $\varkappa \in X$, the characteristic function $C(\cdot)$ is monotone; namely,

$$C(S) \leq C(\mathcal{N}) \qquad , S \subseteq \mathcal{N}.$$

Then, within this setting players are enticed to form the largest coalition, if the payoff $C(\mathcal{N})$ allows for a distribution among all players, provided that the payoff of each player is at least as large as if the respective player would not play the coalition, i.e.,

$$C(\mathcal{N}) \geq \sum_{\nu \in \mathcal{N}} C(\nu).$$

However, considering the case where every players' share of the coalitions payoff equals their individual payoff, that is no gain is achieved, the players have no incentive of joining the coalition [Krabs, 2005]. Therefore, a crucial aspect of coalitional games is the payoff distribution among the members of a coalition. If the distribution is defined such that a players gain is lower compared to playing alone the coalitions generate no value. Hence, the distribution must be defined in a way that every player at most achieves a higher payoff compared to his individual performance. This is captured by the definition of imputations

$$\mathcal{I}(C) = \left\{ \zeta \in \mathbb{R}^N \mid \zeta_\nu \geq C(\nu)\ \forall \nu \in \mathcal{N}, \sum_{\nu \in \mathcal{N}} \zeta_\nu = C(\mathcal{N}) \right\}.$$

A solution concept for coalitional games proposed by [von Neumann et al., 1944] is based on the idea of stable sets. Then, an imputation $\zeta \in \mathcal{I}(C)$ is favored in comparison to an arbitrary imputation $\overline{\zeta} \in \mathcal{I}(C)$ if there exists a coalition $S \subseteq \mathcal{N}$ where,

$$\zeta_\nu \geq \overline{\zeta}_\nu \qquad \forall \nu \in S \qquad \text{and} \qquad \sum_{\nu \in S} \zeta_\nu \leq C(S)$$

By means of this solution concept the largest coalition and a singleton coalition can be avoided.

Remark 2.7. *There exist various other solution concepts such as the core, the shapely value and many more, which are not discussed here but can suitably be adopted, compare [Osborne and Rubinstein, 1994, Petrosyan and Zaccour, 2016].*

2.1.3 Noncooperative Games

The terminology *noncooperative* in the context of games is often tainted with the presumption of an absolutely adverse or dualistic nature of the involved players. However, a more general description of noncooperative behaviour would be, that players do not engage in binding commitments, i.e., cooperations. Based on this and Definition 2.1 a noncooperative game can be defined as follows.

Definition 2.8. *An N-player game* $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^{N}$ *is referred to as noncooperative if it is no cooperative game.*

Next, let us formalize a noncooperative strategic game. Herein, let $\mathcal{N} := \{1, \dots, N\}$ denote the index set of the player. Then in terms of an individual player ν the noncooperative game for all $\nu \in \mathcal{N}$ reads

$$\min_{x^\nu} \quad \theta_\nu(x) \quad \text{s.t.} \quad x^\nu \in X_\nu(x),$$

with the feasible set $X_\nu(x)$ as well as the objective function dependent on the opposing players strategies. Let us stress, if either the objective function or the feasible set is independent of the opposing players strategy this nonetheless constitutes a noncooperative game. A noncooperative behaviour does not necessarily preempt an influence of other players' strategies on the objective of an individual player. It solely implies that the priority of each player is to optimize its own objective. Although, the coupling of the individual players' problems permits the modelling of complex scenarios. In practice these problems are typically challenging, if not impossible, to solve, since the optimal solution generally associated with the intrinsic optimization problem cannot be obtained using the same conditions, known from the theory of optimization, see [Kanzow and Schwartz, 2018]. More precisely, it will not always be possible to find a strategy $\hat{x} \in X$ which concurrently minimizes (or maximizes) the objective of every player simultaneously.

Remark 2.9. *Some dependencies on the strategies of the opposing players must be present in a game, otherwise the problem deteriorates to an optimization or optimal control problem.*

2.1.4 Antagonistic Games

A subclass of noncooperative games are the antagonistic games. The term antagonistic, however, tends to be used in an inflationary manner. Often player relations are deemed to be antagonistic

whenever they are not cooperating actively. However, the term antagonistic is actually precisely defined. First mentioned under the term strictly competitive games the concept was introduced by *Robert John Aumann* as early as 1961 in [Aumann, 1961]. Later the term antagonistic game was coined by *Anderson* in terms of a two person game, see [Anderson, 1976].

Definition 2.10. *A two-person game with payoff functions θ_1 and θ_2 is called an antagonistic game if θ_1 and θ_2 inherit the property that*

$$\theta_1(x^1, x^2) > \theta_1(y^1, y^2) \quad \text{, iff} \quad \theta_2(x^1, x^2) < \theta_2(y^1, y^2)$$

for all strategies $x^1, y^1 \in X_1$ and $x^2, y^2 \in X_2$.

Thus, every antagonistic game is a natural subset of a two-person game where the premise, if one player fares better, simultaneously the other player necessarily must do worse, holds. Therefore, in some definitions antagonistic games are seen only as a subset of zero sum games, where the trade off of each players payoff must be equal.

To this end, suppose we have a two player game $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^2$, with the individual problems defined by

$$\begin{aligned} \min_{\varkappa^1} \quad & \theta_1(\varkappa) \quad \text{s.t.} \quad \varkappa \in X, \\ \min_{\varkappa^2} \quad & \theta_2(\varkappa) \quad \text{s.t.} \quad \varkappa \in X, \end{aligned}$$

where $\theta_1(\varkappa) = -\theta_2(\varkappa)$ and $\theta_1(\varkappa) + \theta_2(\varkappa) = 0$ with $(\varkappa^1, \varkappa^2) \in X := X_1 \times X_2$. Evidently, every two person constant or zero-sum game is strategically equivalent to a corresponding antagonistic game.

This, however, constitutes a restricting definition of antagonistic games, since no arbitrary number of players can be considered. Extending this definition takes into account the concept of coalitional games, discussed previously. To this end, suppose a game $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^N$ consisting of N players. Therein, let $S \subseteq N$ be a subset of players forming a coalition with $S \neq \mathcal{N}$. An opposing coalition is then formed by players $\mathcal{N} \setminus S$. That is, coalitions S and $\mathcal{N} \setminus S$ have contradictory objectives,

$$C(S) = \sum_{\nu \in S} \theta_\nu$$

and respectively,

$$C(\mathcal{N} \setminus S) = \sum_{\nu \in N \setminus S} \theta_\nu.$$

where $C(\mathcal{N} \setminus S) = -C(S)$ and $C(\mathcal{N} \setminus S) + C(S) = 0$. For that matter a cooperative game, can at the same time be antagonistic. A property usually associated with noncooperative games.

Therefore, we like to introduce a more general definition of antagonistic games for N player games.

Definition 2.11 (General Antagonistic Game)**.** *An N-player game with coalition S, where $|S| \geq 1$ and coalition $\mathcal{N} \setminus S$ with $|\mathcal{N} \setminus S| \geq 1$ and individual payoff functions θ_ν is called an antagonistic game if*

$$\sum_{\nu \in S} \theta_\nu(x) > \sum_{\nu \in S} \theta_\nu(y) \quad \textit{iff} \quad \sum_{\nu \in \mathcal{N} \setminus S} \theta_\nu(x) < \sum_{\nu \in \mathcal{N} \setminus S} \theta_\nu(y)$$

holds for all strategies $x, y \in X$.

Considering the particular setting with $|S| = 1$ and $|\mathcal{N} \setminus S| = 1$, we have the two player case.

Remark 2.12. *Often the impression arises that cooperative and noncooperative games are mutually exclusive. However, a cooperative game can always be part of a noncooperative game in terms of an underlying subgame. Let us illustrate this with an example. Consider a noncooperative game with a set of players $\mathcal{N}$. A subset $S \subset \mathcal{N}$ of these players strives towards the same objective, that is, some players have common objectives motivating a coalition $C(S)$. Hence, we have a cooperative subgame. In terms of the noncooperative game, the coalition can then be viewed as a single player $\nu \leftarrow C(S)$. The number of players in the noncooperative setting then reduces by $|S| - 1$, that is the number of players joining a coalition. Consequently, the noncooperative game is henceforth composed of $N - |S| + 1$ players. Throughout this thesis we will focus on a noncooperative game setting, where we keep in mind that one player can represent a coalition.*

For noncooperative games, various solution concepts exist, the probably most prominent solution concept being the Nash equilibrium, which will be the focus of our attention in the subsequent chapters.

2.2 Solution Concepts

The concept of Nash equilibria was introduced by *John Forbes Nash* primordial in his doctoral thesis, cf. [Nash, 1951, Nash, 1950]. Next to the Pareto optimum, the Nash equilibrium constitutes one of the most important solution concepts in game theory.

Definition 2.13 (Nash Equilibrium)**.** *Given an N player strategic game $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^N$, then a vector $\hat{x}$ with $\hat{x}^\nu \in X_\nu$ and $\theta_\nu(\hat{x}) \leq \theta_\nu(\hat{x}^1, \ldots, \hat{x}^{\nu-1}, x^\nu, \hat{x}^{\nu+1}, \ldots, \hat{x}^N)$ for all $x^\nu \in X_\nu$ and $\nu = 1, \ldots, N$ is called a Nash equilibrium.*

This definition can be paraphrased to a Nash equilibrium being a point where no player can improve its payoff by an unilateral change of strategy, while the other players adhere to their strategies. Next, we want to introduce some notations frequently used in game theoretical settings, especially in the context of Nash equilibria. To this end, let $x := \left(x^1, \ldots, x^N\right)^\top$ be the

vector of strategies with components $x^\nu \in X_\nu$ for all $\nu = 1, \dots, N$. Often it is customary to emphasize the focused player ν by writing the strategies as $x = (x^\nu, x^{-\nu})^\top$ where

$$x^{-\nu} := \left(x^1, \dots, x^{\nu-1}, x^{\nu+1}, \dots, x^N\right),$$

contains all strategies x^μ of the rival players $\mu \neq \nu$. Note, however, that the ordering of x persists and remains unchanged by this notation.

Consequently, we can express (2.1) with emphasis on a particular player yielding,

$$\min_{x^\nu} \quad \theta_\nu(x^\nu, x^{-\nu}) \qquad \text{s.t.} \quad x^\nu \in X_\nu, \tag{2.3}$$

for all $\nu = 1, \dots, N$. Herein, the objective function depends on the decision variables (strategies) of all players. The problem delineated by (2.3) is referred to as a *standard Nash equilibrium problem.*

Remark 2.14. *Apart from theoretical constructs, especially in the setting of differential games, we have games of imperfect information. Hence, x is at least partially unknown, cf. [Kanzow and Schwartz, 2018].*

For completeness, let us also briefly revisit the definition of a *Pareto optimal solution*, see for instance [Ehrgott, 2005], aptly named after its inventor *Vilfredo Pareto*. Other frequently used expressions include *Pareto efficiency* or *efficient solution.*

Definition 2.15 (Pareto Optimum)**.** *Consider an N player game $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^N$. A vector $\hat{x} \in X$ is deemed an efficient solution or Pareto optimal, if for all $\nu = 1, \dots, N$ there is no $x \in X$ such that $\theta_\nu(x^\nu, x^{-\nu}) \leq \theta_\nu(\hat{x}^\nu, \hat{x}^{-\nu})$ and there exists a subset $\mathcal{Y} \subset \{1, \dots, N\}$ with $|\mathcal{Y}| \geq 1$, such that $\theta_\nu(x^\nu, x^{-\nu}) < \theta_\nu(\hat{x}^\nu, \hat{x}^{-\nu})$ for all $\nu \in \mathcal{Y}$.*

A Pareto optimal solution, therefore, is an outcome from which any attempt to improve by deviating to some other outcome will result in an impairment to someone else. Hence both, the Nash equilibrium and the Pareto optimum, give answers to different questions. The Pareto optimum gives an answer to what is efficient with respect to the payoff of each player, without consideration of the actual strategies. Whereas the Nash equilibrium is concerned with strategic feasibility. In order to illustrate the differences, we consider the following example.

Example 2.16 (Hunter/Gatherer)**.** Suppose we have a two player game $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^2$, where the two players are hunters (or gatherers) which are required to provide food for their community. Each player gets utility from the availability of food, which we can express through the sum of their combined efforts x^1 and x^2. Disutility is created through their personal expenditures for providing the food. Further, let $x^1, x^2 \in [0, 1]$ and let the payoff functions be given by

$$\begin{aligned}
\max_{x^1} \ \theta_1(x^1, x^2) &= (x^1 + x^2) - 1.2x^1 = x^2 - 0.2x^1, \\
\max_{x^2} \ \theta_2(x^2, x^1) &= (x^2 + x^1) - 1.2x^2 = x^1 - 0.2x^2.
\end{aligned}$$

Indifferent of the choice of the second player, from the viewpoint of the first player his payoff tapers if his personal effort $\varkappa^1 \neq 0$. Accordingly, it is best to not invest any effort, that is $\varkappa^1 = 0$. Similar, the best choice for the second player is $\varkappa^2 = 0$. Resulting in payoffs $\theta_1 = \theta_2 = 0$. Thus, $\varkappa = (0, 0)$ being a Nash equilibrium.

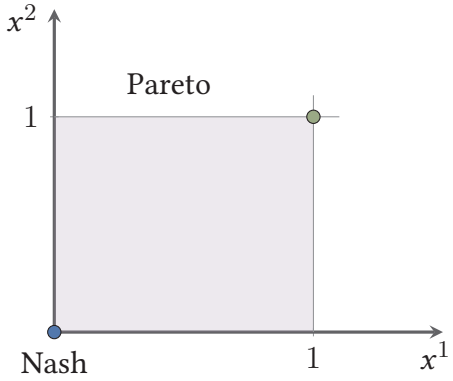


Figure 2.1: *Illustration of Nash (blue) and Pareto (green) optimal solutions of Example 2.16. The shaded area indicates the set of feasible strategies.*

Now, if the circumstances allow for the players to make an agreement to invest their maximum effort $\varkappa^1 = \varkappa^2 = 1$, both achieve higher payoffs $\theta_1 = \theta_2 = 0.8$. Where, $\varkappa = (1, 1)$ is a Pareto optimal solution, albeit no Nash equilibrium.

In Example 2.16 it can be observed that, in general, Nash equilibria and Pareto optimal solution not necessarily coincide. In the class of zero sum games discussed in the previous chapter, by constitution any combination of strategies results in a Pareto optimal point.

Let us stress that the subsequent discourse will be directed toward finite dimensional problems, i.e., $X_\nu \subseteq \mathbb{R}^{n_\nu}$.

2.3 Generalized Nash Equilibrium Problem

The generalized Nash equilibrium problem (GNEP) constitutes an extension of the Nash equilibrium problem (NEP) defined in (2.3) and is delineated by a finite set of players $\nu = 1, \ldots, N$, where each player holds an individual cost function and a strategy set X_ν. Compared to the NEP, both, the objective function $\theta_\nu : \mathbb{R}^n \to \mathbb{R}$, with $n = n_1 + \ldots + n_N$, may depend on the strategies of all player. Accordingly, as opposed to the NEP, also the strategy set $X_\nu(\varkappa^{-\nu}) \subseteq \mathbb{R}^{n_\nu}$ of the ν-th player might be contingent on the decision variables of the other players $\varkappa^{-\nu}$.

In a GNEP every player simultaneously strives to minimize (or maximize) its cost, or objective function, which is equivalent to solving the following problem.

Problem 2.3.1 (GNEP) . — *Consider a strategic game* $\Gamma = \{X_\nu(x^{-\nu}), \theta_\nu\}_{\nu=1}^N$. *Then, the optimization problem for each player* $\nu = 1, \ldots, N$ *reads,*

$$\min_{x^\nu} \quad \theta_\nu(x^\nu, x^{-\nu}) \quad \text{s.t.} \quad x^\nu \in X_\nu(x^{-\nu}).$$

Dependencies of the feasible sets X_ν on the strategies $x^{-\nu}$ usually emanate from coupling or shared constraints among the players. Most commonly, shared constraints appear in terms of inequality constraints, yielding

$$X_\nu(x^{-\nu}) = \{x^\nu \in \mathbb{R}^{n_\nu} \mid g^\nu(x^\nu, x^{-\nu}) \leq 0\}, \tag{2.4}$$

with a function $g^\nu : \mathbb{R}^n \to \mathbb{R}^{m_\nu}$ and the cumulative number of the coupling inequality constraints defined by $m = \sum_{\nu=1}^N m_\nu$. Furthermore, player individual equality constraints can be considered resulting in a more general expression for the feasible set, in particular

$$X_\nu(x^{-\nu}) := \{x^\nu \in \mathbb{R}^{n_\nu} \mid c^\nu(x^\nu, x^{-\nu}) = 0,\ g^\nu(x^\nu, x^{-\nu}) \leq 0\}\,, \tag{2.5}$$

where $c^\nu : \mathbb{R}^n \to \mathbb{R}^{\ell_\nu}$ and $\ell = \sum_{\nu=1}^N \ell_\nu$ aggregates the number of individual constraints. Consequently the joint strategy space of all players in the setting of a GNEP is defined by the Cartesian product of the individual strategy spaces

$$X(x) := \bigtimes_{\nu=1}^N X_\nu(x^{-\nu}). \tag{2.6}$$

Within this setting, a Nash equilibrium solution of Problem 2.3.1 is defined as follows, cf. [Kanzow and Schwartz, 2018].

Definition 2.17 (Generalized Nash Equilibrium)**.** *A vector* $\hat{x} \in X(\hat{x})$ *is called a generalized Nash equilibrium of a strategic game* $\Gamma = \{X_\nu(x^{-\nu}), \theta_\nu\}_{\nu=1}^N$, *if*

$$\theta_\nu(\hat{x}^\nu, \hat{x}^{-\nu}) \leq \theta_\nu(x^\nu, \hat{x}^{-\nu})$$

for all $x^\nu \in X_\nu(x^{-\nu})$ *and all player* $\nu = 1, \ldots, N$.

In reference to economics, i.e., the cradle of NEPs, the emphasis of research with respect to NEPs and GNEPs has been primarily directed at convex problems. Therefore, the theory outlined in this section relates to a convex problem setting. Hence, the subsequent assumptions are required.

Assumption. 1 . The objective function $\theta_\nu(\cdot, x^{-\nu})$ and the functions $g_i^\nu(\cdot, x^{-\nu})$ are continuous and convex in x^ν for fixed $x^{-\nu}$ and all player $\nu = 1, \ldots, N$ and $i = 1, \ldots, m_\nu$.

Assumption. 2 . The functions $\theta_\nu(\cdot)$ and $g_i^\nu(\cdot)$ for all $\nu = 1, \ldots, N$ and $i = 1, \ldots, m_\nu$ are at least C^1 functions.

Assumption. 3 . For all $\nu = 1, \ldots, N$ the set $X_\nu(x^{-\nu})$ is closed and convex for all $x^{-\nu}$.

Thus, together with boundedness of $X_\nu(\cdot)$ this ensures the solvability of Problem 2.3.1, cf. [Facchinei and Kanzow, 2010b]. Next, let us introduce the definition of an important and widely studied subclass, the *jointly convex* GNEPs, see for instance [Dreves, 2011].

Definition 2.18. *Consider a GNEP as in Problem 2.3.1 which satisfies Assumption 1 and Assumption 3. The GNEP is called jointly convex if there exists a common convex strategy set $X \subseteq \mathbb{R}^n$ such that for all $\nu = 1, \ldots, N$ the feasible set is given by*

$$X_\nu(x^{-\nu}) = \{x^\nu \in \mathbb{R}^{n_\nu} \mid (x^\nu, x^{-\nu}) \in X\} .$$

With respect to the feasible set defined in (2.4), Definition 2.18 is equivalent to $g^1 = \ldots = g^N = g$ with a common function $g(x)$, with convex components $g^\nu(\cdot)$ in all variables x, for all $\nu = 1, \ldots, N$. Then, the joint feasible set reads,

$$X = \{x \in \mathbb{R}^n \mid g(x) \leq 0\} .$$

Remark 2.19. *Assumptions on the convexity of the players' optimization problem appear to be imperative for investigating noncooperative games. One reason that immediately comes to mind is that in many cases without these assumptions a Nash equilibrium simply may not exist. Furthermore, it is not straight forward to administer abstract mathematical concepts to nonconvex or even convex games. Especially for games, originating from practical problems and applications, which are additionally governed by dynamic systems and constraints.*

2.4 Reformulations of the GNEP

Since GNEPs, due to their complex structure, are intrinsically difficult to solve, the general approach, therefore, is to suitably reformulate the problems into well known problem structures, such as optimization problems, Variational and Quasi-Variational Inequality Problems and leverage the established methods and theory therein. Hence, in the subsequent sections a survey on existing methods is given, compare [Facchinei and Kanzow, 2010a, Fischer et al., 2014].

2.4.1 KKT-Conditions of GNEP

A gateway for solving GNEPs is provided by utilizing the optimality conditions of the respective constrained optimization problem, more precisely the KKT conditions. Introduced in [Kuhn and Tucker, 1951, Karush, 1939], the first order necessary conditions are the foundation of various algorithms to derive optimal solutions, provided some regularity conditions are satisfied.
A summary of such algorithms can be found in [Nocedal and Wright, 2006, Geiger and Kanzow,

2002]. Subsequently, we consider Problem 2.3.1 to be additionally governed by shared equality constraints. Specifically, its feasible set is characterized by (2.5). Anterior to the definition of the KKT conditions for a GNEP, we define the player individual Lagrangian of Problem 2.3.1,

$$\mathcal{L}_\nu(x^\nu, x^{-\nu}, \lambda^\nu, \sigma^\nu) := \theta_\nu(x^\nu, x^{-\nu}) + \sum_{i=1}^{m_\nu} \lambda_i^\nu g_i^\nu(x) + \sum_{j=1}^{\ell_\nu} \sigma_j^\nu c_j^\nu(x), \tag{2.7}$$

where $\mathcal{L}_\nu : \mathbb{R}^n \times \mathbb{R}^{m_\nu} \times \mathbb{R}^{\ell_\nu} \to \mathbb{R}$ with the multipliers $\lambda^\nu \in \mathbb{R}^{m_\nu}$ and $\sigma^\nu \in \mathbb{R}^{\ell_\nu}$. Respectively, we have for the gradient of the Lagrangian in terms of the variables x^ν:

$$\nabla_{x^\nu} \mathcal{L}_\nu(x^\nu, \hat{x}^{-\nu}, \lambda^\nu, \sigma^\nu) := \nabla_{x^\nu} \theta_\nu(x^\nu, x^{-\nu}) + \sum_{i=1}^{m_\nu} \lambda_i^\nu \nabla_{x^\nu} g_i^\nu(x) + \sum_{j=1}^{\ell_\nu} \sigma_j^\nu \nabla_{x^\nu} c_j^\nu(x).$$

Using the Lagrangian of the individual players' problem, i.e., (2.7), and suppose $\hat{x}$ is a solution of the respective GNEP, then the KKT conditions for player ν can be defined.

$$\nabla_{x^\nu} \mathcal{L}_\nu(\hat{x}^\nu, \hat{x}^{-\nu}, \lambda^\nu, \sigma^\nu) = 0, \tag{2.8.1}$$
$$c_j^\nu(\hat{x}^\nu, \hat{x}^{-\nu}) = 0, \qquad j = 1, \dots, \ell_\nu, \tag{2.8.2}$$
$$\lambda^\nu \geq 0, \tag{2.8.3}$$
$$g_i^\nu(\hat{x}^\nu, \hat{x}^{-\nu}) \leq 0, \qquad i = 1, \dots, m_\nu, \tag{2.8.4}$$
$$\lambda_i^\nu g_i^\nu(\hat{x}^\nu, \hat{x}^{-\nu}) = 0, \qquad i = 1, \dots, m_\nu. \tag{2.8.5}$$

By combining the individual KKT conditions of every player $\nu = 1, \dots, N$ a system of KKT conditions for the GNEP is acquired:

$$\nabla_x \mathcal{L}(x, \lambda, \sigma) = 0, \tag{2.9.1}$$
$$c(x) = 0, \tag{2.9.2}$$
$$\lambda \geq 0, \tag{2.9.3}$$
$$g(x) \leq 0, \tag{2.9.4}$$
$$\lambda^\top g(x) = 0. \tag{2.9.5}$$

Herein, we have the concatenated vectors

$$\nabla_x \mathcal{L}(x, \lambda, \sigma) := \begin{pmatrix} \nabla_{x^1} \mathcal{L}_1(x, \lambda^1, \sigma^1) \\ \vdots \\ \nabla_{x^N} \mathcal{L}_N(x, \lambda^N, \sigma^N) \end{pmatrix}, \; g(x) := \begin{pmatrix} g^1(x) \\ \vdots \\ g^N(x) \end{pmatrix}, \; c(x) := \begin{pmatrix} c^1(x) \\ \vdots \\ c^N(x) \end{pmatrix}$$

and the combined vectors of the Lagrangian multipliers,

$$\lambda := \begin{pmatrix} \lambda^1 & \cdots & \lambda^N \end{pmatrix}^\top \in \mathbb{R}^m, \quad \sigma := \begin{pmatrix} \sigma^1 & \cdots & \sigma^N \end{pmatrix}^\top \in \mathbb{R}^\ell.$$

Suppose that for every player $\nu = 1, \dots, N$ a reasonable constraint qualification (e.g., linear-independence constraint qualification (LICQ), Mangasarian-Fromovitz constraint qualification (MFCQ), cf. [Geiger and Kanzow, 2002]) holds. Then, (2.9.1) to (2.9.5) are considered first order necessary conditions for the GNEP. Moreover, a local minimum can be characterized.

Theorem 2.20 ([Facchinei and Kanzow, 2010b, Theorem 4.6]). *Consider* Problem 2.3.1 *together with Assumptions 1 to 3.*

- *Let $\hat{x}$ be an equilibrium point of the GNEP at which all player's subproblems satisfy a constraint qualification. Then, there exists a triple $(\hat{x}, \hat{\lambda}, \hat{\sigma})$ that satisfies the system of (2.8.1) to (2.8.5) .*
- *Suppose $(\hat{x}, \hat{\lambda}, \hat{\sigma})$ solves (2.8.1) to (2.8.5) and Assumption 1 and Assumption 3 are satisfied by* Problem 2.3.1*, then $\hat{x}$ is an equilibrium of the GNEP.*

In the peculiar convex case, the KKT conditions are not only necessary but also sufficient conditions. A wide range of reformulations of the KKT conditions, which constitute the foundation of various algorithms, that efficiently solve the underlying optimization problem, have been introduced in the literature in recent years, see for instance [Facchinei et al., 2009, Dreves et al., 2014, Dreves, 2016].

Remark 2.21. *While first-order methods facilitate the identification of local optima, for the identification of global solutions they are insufficient. Indeed, the second order information would be required. However, the computational cost associated with obtaining the Hessian renders respective methods impractical in most cases.*

2.4.2 Quasi Variational Inequalities

A reformulation in terms of VIs has proven to be an essential accessory for the theoretical and algorithmic investigation of GNEPs, compare [Pang and Fukushima, 2009, Facchinei and Kanzow, 2010a] and [Pang and Scutari, 2011]. Suppose Assumptions 1 to 3 hold. Then, within the setting of Problem 2.3.1 let

$$F_\nu(x) := \nabla_{x^\nu} \theta_\nu(x^\nu, x^{-\nu})$$

with $F_\nu(x) \in \mathbb{R}^{n_\nu}$ and thus, for all players $\nu = 1, \dots, N$ the mapping $F : \mathbb{R}^n \to \mathbb{R}^n$ is constituted by

$$F(x) := (\nabla_{x^\nu} \theta_\nu(x^\nu, x^{-\nu}))_{\nu=1}^N . \tag{2.10}$$

The corresponding QVI problem is to find a vector $\hat{x} \in X(\hat{x})$ such that

$$(y - \hat{x})^\top F(\hat{x}) \geq 0, \tag{2.11}$$

for all $y \in X(\hat{x})$. Naturally the VI emerges from this setting if $X(x) = X$, that is if the feasible set is independent of the opposing players' strategies. [Bensoussan, 1974] was the first to associate the QVI as a reformulation of a GNEP. Paraphrasing his findings, a generalized Nash equilibrium can be characterized as a solution of the QVI.

Theorem 2.22. *Consider* Problem 2.3.1 *together with Assumptions 1 to 3. A vector $\hat{x}$ is a generalized Nash equilibrium if and only if it is a solution to* (2.11).

Note, that the assertion in the previous theorem holds in both ways. Thus, a solution of GNEP is also a solution to the respective QVI. Furthermore, it implies the same properties for the relation of NEP and VI. Unlike the vast applicability of results known from the study of VIs to NEPs, QVIs and their properties have so far only been scarcely investigated, cf. [Pang and Fukushima, 2009, Facchinei and Pang, 2009]. Nonetheless, in case of the jointly convex GNEP it is possible to fruitfully establish and nurture a relation between the solutions of GNEPs and VIs. Therefore, let us consider a jointly convex GNEP as in Definition 2.18. The associated KKT system of the VI resulting from (2.11) with the joint feasible set $X(x)$ defined through (2.5) and (2.6) is given by

$$F(x) + \lambda^\top \nabla g(x) + \sigma^\top \nabla c(x) = 0, \tag{2.12.1}$$
$$c(x) = 0, \tag{2.12.2}$$
$$\lambda \geq 0, \tag{2.12.3}$$
$$g(x) \leq 0, \quad \lambda^\top g(x) = 0. \tag{2.12.4}$$

Then, with (2.10) and for common convex constraints $g^1 = \ldots = g^N =: g$, and $c^1 = \ldots = c^N =: c$, we have the convex joint feasible set

$$X = \{x \in \mathbb{R}^n \mid g(x) \leq 0,\ c(x) = 0\}\,.$$

It is then discernible, that the KKT systems in (2.9.1) to (2.9.5) and (2.12.1) to (2.12.4) coincide. Eventually, this coherence leads to the following theorem, compare [Facchinei et al., 2007, Theorem 3.1].

Theorem 2.23. *Let* Problem 2.3.1 *be jointly convex together with Assumptions 1 to 3. Then, the following statements apply*

- *Let $\hat{x}$ be a solution to the VI governed by $F(x)$ and the feasible set X and $\hat{\lambda} \in \mathbb{R}^m$, $\hat{\sigma} \in \mathbb{R}^\ell$. Suppose $(\hat{x}, \hat{\lambda}, \hat{\sigma})$ satisfy* (2.12.1) *to* (2.12.4)*. Thus, $(\hat{x}, \hat{\lambda}, \hat{\sigma})$ also satisfy the KKT conditions in* (2.9.1) *to* (2.9.5) *with common multipliers $\hat{\lambda}^1 = \ldots = \hat{\lambda}^N$, $\hat{\sigma}^1 = \ldots = \hat{\sigma}^N$ and therefore $\hat{x}$ is a solution of the GNEP.*
- *Conversely, suppose that $\hat{x}$ is a solution of* Problem 2.3.1 *with the feasible set X such that the KKT conditions in* (2.9.1) *to* (2.9.5) *are satisfied with common multipliers $\hat{\lambda}^1 = \ldots = \hat{\lambda}^N$, $\hat{\sigma}^1 = \ldots = \hat{\sigma}^N$. Then $(\hat{x}, \hat{\lambda}, \hat{\sigma})$, with $\hat{\lambda} := \hat{\lambda}_1$ and $\hat{\sigma} := \hat{\sigma}_1$, is a KKT point of the VI given by* (2.11) *in conjunction with the feasible set X, and $\hat{x}$ itself is a solution of the VI.*

Remark 2.24. *It is important to emphasize that Theorem 2.23 requires the multipliers of the KKT conditions of the GNEP and of the respective VI to coincide. Then again, this does not imply that any solution of a GNEP simultaneously is a solution to the VI. On the contrary, the solution set described by the VI usually is much smaller than the one of the corresponding GNEP. Hence, not all solutions of the GNEP can be acquired if it is reduced to a VI.*

The solution of a jointly convex GNEP, which is also a solution of the VI is called a *variational equilibrium*. It is often also referred to as a *normalized equilibrium*. However, throughout this thesis the term variational equilibrium will be used, to not induce any confusion with the *normalized equilibrium* introduced by [Rosen, 1965].

2.4.3 Nikaidô-Isoda Function

A widely used asset for the analysis of GNEP introduced in [Nikaidô and Isoda, 1955], has been and still is the Nikaidô-Isoda function, also known as the Ky-Fan function. Consider Problem 2.3.1 and let x and y be two feasible points of a GNEP. Then, the function solely depending on the payoff functions of the players,

$$\Psi(x, y) := \sum_{\nu=1}^{N} \left[\theta_\nu(x^\nu, x^{-\nu}) - \theta_\nu(y^\nu, x^{-\nu})\right]. \tag{2.13}$$

is known as the Nikaidô-Isoda function. The intuition behind this equation is yet simple but elegant. For each player ν the sum over all players tracks the impact of a deviation in its decision variable from x to y while the other players remain with their choices $x^{-\nu}$. Consequently, leading to a rather self explanatory condition for a generalized Nash equilibrium. A solution to the GNEP is then delineated through an absence of possible improvement, cf. [Facchinei and Kanzow, 2010b]. Recalling the findings of [Nikaidô and Isoda, 1955]:

Theorem 2.25. *Suppose (2.13) is the Nikaidô-Isoda function of Problem 2.3.1, together with (2.6). Then, $\hat{x}$ is a solution to the GNEP if and only if,*

$$\sup_{y \in X(\hat{x})} \Psi(\hat{x}, y) = 0 \tag{2.14}$$

and $\hat{x} \in X(\hat{x})$.

The Nikaidô-Isoda function has become a useful tool in deriving further reformulations, as well as algorithms for solving the GNEP. However, the literature herein focuses mainly on jointly-convex GNEP, for instance [Uryas'ev and Rubinstein, 1994],[Dreves and Kanzow, 2011] and [von Heusinger and Kanzow, 2009]. Further results and methods using the Nikaidô-Isoda function can be found in [Facchinei and Kanzow, 2010a].

2.5 Potential Games

In this section, a concept inspired by vector analysis or more precisely by physics which likewise is applicable to game theory is discussed, the potential function. [Rosenthal, 1973] was the first making use of this reinterpretation of a game in terms of a potential function with respect to the rather small class of congestion games. The notion of a *potential game*, however, was primarily coined by the works of [Monderer and Shapley, 1996] in the context of Nash equilibrium problems. Over the years many variations of potential games have emerged. An overview and in depth discussion can be found in [Lã et al., 2016]. Essentially, four categories of potential games were suggested by [Monderer and Shapley, 1996], i.e., the *ordinal potential game*, the *weighted potential game*, the *exact potential game* and the *generalized potential game*.

Let $\Phi : \mathbb{R}^n \to \mathbb{R}$ indicate the potential function. Following [Monderer and Shapley, 1996], then a potential game is characterized as follows:

Definition 2.26. *Consider an N player game $\Gamma = \{X_\nu, \theta_\nu\}_{\nu=1}^N$.*

*(i) The game is called an **exact potential game**, if there exists a function $\Phi(x)$ such that for all player $\nu = 1, \dots, N$ and all $x^\nu, y^\nu \in X$,*

$$\theta_\nu(y^\nu, x^{-\nu}) - \theta_\nu(x^\nu, x^{-\nu}) = \Phi(y^\nu, x^{-\nu}) - \Phi(x^\nu, x^{-\nu}).$$

*(ii) The game is called a **weighted potential game**, if there exists a function $\Phi(x)$ with $w_\nu > 0$, such that for all player $\nu = 1, \dots, N$ and all $x^\nu, y^\nu \in X$,*

$$\theta_\nu(y^\nu, x^{-\nu}) - \theta_\nu(x^\nu, x^{-\nu}) = w_\nu \left(\Phi(y^\nu, x^{-\nu}) - \Phi(x^\nu, x^{-\nu})\right).$$

*(iii) The game is called an **ordinal potential game**, if there exists a function $\Phi(x)$ such that for all player $\nu = 1, \dots, N$ and all $x^\nu, y^\nu \in X$,*

$$\theta_\nu(y^\nu, x^{-\nu}) - \theta_\nu(x^\nu, x^{-\nu}) > 0 \Leftrightarrow \Phi(y^\nu, x^{-\nu}) - \Phi(x^\nu, x^{-\nu}) > 0.$$

*(iv) The game is called a **generalized ordinal potential game**, if there exists a function $\Phi(x)$ such that for all player $\nu = 1, \dots, N$ and all $x^\nu, y^\nu \in X$,*

$$\theta_\nu(y^\nu, x^{-\nu}) - \theta_\nu(x^\nu, x^{-\nu}) > 0 \Rightarrow \Phi(y^\nu, x^{-\nu}) - \Phi(x^\nu, x^{-\nu}) > 0.$$

Remark 2.27. *A potential function characterizes the change in a players objective function induced by the players deviation in his decision variable. Thus, the whole dynamic of a game can be expressed and reduced to a single optimization problem governed by the potential function.*

An exact potential game therefore, directly reflects the change in a players decision in the respective potential function. Furthermore, it is easy to see that the weighted potential game for $w_\nu = 1$ for all $\nu = 1, \dots, N$ becomes an exact potential game. On the contrary to the weighted and exact potential game, where the magnitude of deviation is reflected by the potential function,

in the ordinal potential game the deviation in the objective function and the potential function only must have the same tendency. Hence, the definition can be formulated as

$$\operatorname{sgn}\left(\theta_\nu(y^\nu, x^{-\nu}) - \theta_\nu(x^\nu, x^{-\nu})\right) = \operatorname{sgn}\left(\Phi(y^\nu, x^{-\nu}) - \Phi(x^\nu, x^{-\nu})\right).$$

Compared to the ordinal potential game the generalized ordinal potential game requires that a change in the objective function, may it be an increase or decrease, automatically implies an identical change of direction in the potential function. Existence results regarding these types of potential games can be found in [Monderer and Shapley, 1996].

Analogously, the concept of potential games can be extended towards GNEPs. [Facchinei et al., 2011] introduced the generalized potential game (GPG) for the jointly convex case.

Definition 2.28 (GPG). *Consider* Problem 2.3.1 *together with the feasible set in* (2.5)*. Additionally assume the joint feasible set X is nonempty and closed for all $\nu = 1, \dots, N$. Let $y^\nu, z^\nu \in X_\nu(x^{-\nu})$ be two feasible points. Then the corresponding game $\Gamma = \{X_\nu(x^{-\nu}), \theta_\nu\}_{\nu=1}^N$ is a generalized potential game if a continuous potential function $\Phi : \mathbb{R}^n \to \mathbb{R}$ exists where a positive deviation in strategies, i.e.,*

$$\theta_\nu(y^\nu, x^{-\nu}) - \theta_\nu(z^\nu, x^{-\nu}) > 0, \tag{2.15}$$

implicates similar behaviour of the potential function,

$$\Phi(y^\nu, x^{-\nu}) - \Phi(z^\nu, x^{-\nu}) \geq \xi\left(\theta_\nu(y^\nu, x^{-\nu}) - \theta_\nu(z^\nu, x^{-\nu})\right), \tag{2.16}$$

for all player $\nu = 1, \dots, N$ and $x^{-\nu}$, with forcing function $\xi : \mathbb{R}_+ \to \mathbb{R}_+$ with properties:

$$\lim_{k\to\infty} \xi(t_k) = 0 \Rightarrow \lim_{k\to\infty} t_k = 0.$$

Evidently, if the forcing function is the identity function and the feasible set degenerates to (2.6), then the GPG regresses to an exact potential game. Hence, by suitable adaption of the forcing function ξ and the structure and properties of the feasible set the GPG can be affiliated with any of the categories of potential games from Definition 2.26.

More interestingly, however, is the possibility to reformulate the GNEP into a single optimization problem solely governed by the potential function,

$$\min_x \quad \Phi(x) \quad \text{s.t.} \quad x \in X(x). \tag{2.17}$$

From [Facchinei et al., 2011] it is known that a global solution of (2.17) is also a Nash equilibrium. Despite the compact notation, these centralized problems usually are difficult to solve, since the potential function Φ is not necessarily convex and it is necessary to acquire a global solution, to find a Nash equilibrium [Facchinei et al., 2011]. The aforementioned difficulty together with the natural structure of generalized Nash equilibrium problems motivates the employment of decomposition methods. A decomposition approach allows to solve each individual players

subproblem in an iterative manner, given the other players decisions. Herein, the subproblems are always of smaller dimension and therefore easier to solve then the centralized single optimization problem. In the subsequent section, variations and properties of decomposition methods will be discussed in more detail.

Remark 2.29. *Constructing or identifying potential functions is generally a difficult task. Merely in some particular settings the construction is fairly straightforward. Suppose that the objective function solely depends on the players variables, i.e.,*

$$\theta_\nu(x) = \theta_\nu(x^\nu)$$

then it is evident that one possible potential function simply can be defined through the sum over all players' objective functions

$$\Phi(x) = \sum_{\nu=1}^{N} \theta_\nu(x^\nu).$$

Clearly, the potential function directly reflects a deviation of player ν-th decision in the sum. Similarly, any problem where a separation of the objective function into a joint contribution and an individual contribution is possible, can be considered a potential function, e.g.,

$$\theta(x) = a(x) + b_\nu(x^\nu) \quad \Rightarrow \quad \Phi(x) = a(x) + \sum_{\nu=1}^{N} b_\nu(x^\nu).$$

2.6 Decomposition Algorithms

As conversed in the previous section, solving (2.17) in a practical and easily comprehensive way is possible by means of decomposition methods. In particular, the nonlinear Jacobi-type method (Algorithm 1) and the Gauss-Seidel-type method (Algorithm 2). Both methods are intrinsically known from linear algebra, cf. [Stoer and Bulirsch, 2002, Chapter 8]. For an extensive discussion of this topic we refer to [Facchinei and Kanzow, 2010a, Fischer et al., 2014, Facchinei et al., 2011]. Since the centre of attention of this thesis is concerned with a generalization of the Gauss-Seidel approach, the Jacobi algorithm will only be briefly discussed for completeness.

Suppose $x_k \in X_\nu(x^{-\nu})$ is a given strategy for all $\nu = 1, \ldots, N$ at an instance $k \in \mathbb{N}$, then in a GNEP, naturally every player strives to find an optimal response x^ν_{k+1} to the given decisions of the other players $x^{-\nu}_k$. The fundamental concept of decomposition methods is to leverage this natural behaviour, by iteratively computing a best response for every player $\nu = 1, \ldots, N$. Let us first consider the Jacobi-type method.

Algorithm 1: Jacobi Method

(S.0) Choose $\varkappa_0 = \{\varkappa_0^1, \ldots, \varkappa_0^N\}$, with $\varkappa_0^\nu \in X_\nu(x^{-\nu})$ for all $\nu = 1, \ldots, N$. Set $k := 0$.

(S.1) If $\varkappa_k$ is a Nash equilibrium: TERMINATE.

(S.2) For all $\nu = 1, \ldots, N$, compute a solution $\varkappa_{k+1}^\nu$ of

$$\min_{x^\nu} \quad \theta_\nu\left(x^\nu, \varkappa_k^{-\nu}\right) \quad \text{s.t.} \quad x^\nu \in X_\nu(\varkappa_k^{-\nu}).$$

(S.3) Set $\varkappa_{k+1} := (\varkappa_{k+1}^1, \ldots, \varkappa_{k+1}^N)$, $k \leftarrow k+1$. Go to $(S.1)$.

Initially, a feasible starting point $\varkappa_0$ is selected. Subsequently, in (S.2) for every player $\nu = 1, \ldots, N$ a problem of type Problem 2.3.1 is to be solved, amounting to N optimization problems in every iteration k. Therefore, when the ν-th players problem is solved, the strategies $\varkappa_k^\mu$ of the opposing players $\mu \neq \nu$ are considered to be fixed. While this concept eases the application and facilitates parallelisation, conversely outdated information is used in the current players optimization. Let us elaborate on that. Suppose $\nu > 1$, then if it is the ν-th players turn to optimize, all players $\mu < \nu$ already finished their optimization. Thus, their updated variables,

$$\varkappa_{k+1}^1, \ldots, \varkappa_{k+1}^{\nu-1}, \varkappa_k^{\nu+1}, \ldots, \varkappa_k^N$$

are already at ν's disposal and could be utilized in the objective function as well as in the feasible set $X_\nu(x^{-\nu})$. The shortcomings of Algorithm 1 motivate the use of Algorithm 2.

Algorithm 2: Gauss-Seidel Method

(S.0) Choose $\varkappa_0 = \{\varkappa_0^1, \ldots, \varkappa_0^N\}$, with $\varkappa_0^\nu \in X_\nu(x^{-\nu})$ for all $\nu = 1, \ldots, N$. Set $k := 0$.

(S.1) If $\varkappa_k$ is a Nash equilibrium: TERMINATE.

(S.2) For all $\nu = 1, \ldots, N$, compute a solution $\varkappa_{k+1}^\nu$ of

$$\begin{aligned} \min_{x^\nu} \quad & \theta_\nu(\varkappa_{k+1}^1, \ldots, \varkappa_{k+1}^{\nu-1}, x^\nu, \varkappa_k^{\nu+1}, \ldots, \varkappa_k^N) \\ \text{s.t.} \quad & x^\nu \in X_\nu(\varkappa_{k+1}^1, \ldots, \varkappa_{k+1}^{\nu-1}, \varkappa_k^{\nu+1}, \ldots, \varkappa_k^N) \end{aligned}$$

(S.3) Set $\varkappa_{k+1} := (\varkappa_{k+1}^1, \ldots, \varkappa_{k+1}^N)$, $k \leftarrow k+1$. Go to $(S.1)$

Note that in (S.2) of the Gauss-Seidel type method for all player $\mu > \nu$ the current variables, i.e., $\varkappa_k^\mu$ are utilized, while for all $\mu < \nu$ the updated decision variables $\varkappa_{k+1}^\mu$ are incorporated.

Remark 2.30. *Even though, the decomposition methods are favorably used in practical applications, especially in engineering applications, convergence results are almost non existent. Even in case of a simple NEP potential game, where each players individual problem is convex, see [Powell, 1973].*

One now might suggest to solve (2.17) directly, using standard methods, on the supposition that the problem is convex. Certainly, standard methods would lead to a global solution and therefore to a Nash equilibrium. However, even though the problem is convex, the set of Nash equilibria usually is larger than the solution set of the optimization problem in (2.17). Consequently, this would lead to a restriction of the solution space. From the practical perspective, solving the centralized optimization problem would implicitly suggest extensive coordination and communication effort among selfish players. Often this assumption is not feasible. Proceeding to the nonconvex case, it is immediately discernible that applying standard methods to (2.17) for $\Phi(x)$ being a nonconvex function results in stationary points rather than the desired global solutions, compare [Facchinei et al., 2011].

Convex Case

[Facchinei et al., 2011] introduce an alteration of Algorithm 2, by adjoining a proximal term to the objective function. Furthermore, the regularized algorithm allows them to provide convergence results for a convex and nonconvex setting of (2.17). Subsequently these findings are recalled and discussed. To this end, the regularized individual subproblem reads as,

$$\min_{x^\nu} \quad \theta_\nu(x^\nu, x^{-\nu}) + \tau_\nu \|x^\nu - x_k^\nu\|^2 \tag{2.18}$$

for all player $\nu = 1, \dots, N$ and $x^\nu \in X(x^{-\nu})$, with $\tau_\nu \in \mathbb{R}_{++}$. Then, the altered Gauss-Seidel algorithm is defined in Algorithm 3.

Algorithm 3: Regularized Gauss-Seidel Method

(S.0) Choose $x_0 = \{x_0^1, \dots, x_0^N\}$, with $x_0^\nu \in X_\nu(x^{-\nu})$ for all $\nu = 1, \dots, N$ and a regularization parameter $\tau = (\tau_0^1, \dots, \tau_0^N) \in \mathbb{R}^n_{++}$. Set $k := 0$.

(S.1) If x_k is a Nash equilibrium: TERMINATE.

(S.2) For all $\nu = 1, \dots, N$, compute a solution x_{k+1}^ν of

$$\begin{aligned} \min_{x^\nu} \quad & \theta_\nu(x_{k+1}^1, \dots, x_{k+1}^{\nu-1}, x^\nu, x_k^{\nu+1}, \dots, x_k^N) + \tau_\nu \|x^\nu - x_k^\nu\|^2 \\ \text{s.t.} \quad & x^\nu \in X_\nu(x_{k+1}^1, \dots, x_{k+1}^{\nu-1}, x_k^{\nu+1}, \dots, x_k^N) \end{aligned}$$

(S.3) Set $x_{k+1} := (x_{k+1}^1, \dots, x_{k+1}^N)$, $k \leftarrow k+1$. Go to $(S.1)$

For the setting of the convex case, consider (2.17) and suppose that Assumptions 1 to 3 hold. Additionally, assume for all player $\nu = 1, \ldots, N$, that $X_\nu(x^{-\nu})$ is inner-semicontinuous relative to $\mathrm{dom}(X_\nu)$ (cf. Definition 1.4). Further, let the joint feasible set $X(x)$ be free of any structural assumptions. The course of Algorithm 3 is the same as Algorithm 2. Due to the regularization, together with convexity of $\theta_\nu(\cdot)$ and $X_\nu(\cdot)$ for all player $\nu = 1, \ldots, N$, the problem is well defined and always has a solution.

Theorem 2.31. *[Facchinei et al., 2011, Theorem 4.3] Suppose $\hat{x}$ is a cluster point of the sequence $\{x_k\}$ generated by Algorithm 3. Then, $\hat{x}$ is a Nash equilibrium of Problem 2.3.1.*

The convergence property of the above theorem states, that if there exists a cluster point, or more precisely if the sequence generated by Algorithm 3 omits a cluster point it is a solution of the regularized convex problem in (2.18), but not necessarily a global minimizer of the potential function Φ of the respective convex GPG.

Nonconvex Case

The far more challenging, yet also more interesting case arises if Assumption 1 and Assumption 3 are dropped. In other words, $\theta_\nu(\cdot)$ and the feasible set $X_\nu(\cdot)$ are no longer assumed to be convex, resulting in each players subproblem being nonconvex. [Facchinei et al., 2011] address these nonconvex problems by introducing a suitable update rule for the parameter τ_ν which drives it towards zero. Allowing to establish convergence of an adapted Gauss-Seidel algorithm for nonconvex problems.

Algorithm 4: Nonconvex: Regularized Gauss-Seidel Method

(S.0) Choose $x_0 = \{x_0^1, \ldots, x_0^N\}$, with $x_0^\nu \in X_\nu(x^{-\nu})$ for all $\nu = 1, \ldots, N$ and a regularization parameter $\tau_0 = \left(\tau_0^1, \ldots, \tau_0^N\right) \in \mathbb{R}^n_{++}$. Set $k := 0$.

(S.1) If x_k is a Nash equilibrium: TERMINATE.

(S.2) For all $\nu = 1, \ldots, N$, compute a solution x_{k+1}^ν of

$$\begin{aligned} \min_{x^\nu} \quad & \theta_\nu(x_{k+1}^1, \ldots, x_{k+1}^{\nu-1}, x^\nu, x_k^{\nu+1}, \ldots, x_k^N) + \tau_\nu \|x^\nu - x_k^\nu\|^2 \\ & x^\nu \in X_\nu(x_{k+1}^1, \ldots, x_{k+1}^{\nu-1}, x_k^{\nu+1}, \ldots, x_k^N) \end{aligned}$$

(S.3) Set

$$\tau_{k+1} = \max\left\{\min\left[\tau_k, \max_{\nu=1,\ldots,N}\left\{\|x_{k+1}^\nu - x_k^\nu\|\right\}\right], \frac{\tau_k}{10}\right\} \tag{2.19}$$

$x_{k+1} := (x_{k+1}^1, \ldots, x_{k+1}^N)$, $k \leftarrow k+1$. Go to $(S.1)$

However, the subproblems are no longer automatically well defined. Thus, it is questionable if a solution exists. Therefore, it is from here on assumed that for all player ν and in all iterations k a solution of the individual subproblems exist.

Remark 2.32. *In particular, the existence of a solution is guaranteed if θ_ν is a continuous function and $X_\nu(\cdot)$ is a compact set. However, in practical applications this is not always the case.*

Compared to Algorithm 3 the player individual regularization parameter τ is now updated in $(S.3)$. Where, by [Facchinei et al., 2011, Lemma 5.1] the update rule (2.19) ensures that τ_k strives to zero. Then, the convergence conditions for Algorithm 4 omitting a Nash equilibrium is summed up in the following theorem.

Theorem 2.33. *[Facchinei et al., 2011, Theorem 5.1] Assume for all player $\nu = 1, \ldots, N$ that the objective function $\theta_\nu(\cdot)$ is continuous, $X_\nu(x^{-\nu})$ is inner-semicontinuous relative to $\mathrm{dom}(X_\nu)$ and for all iterations k each subproblem (2.18) has a solution. Furthermore, assume that the sequence $\{x_k\}$ generated by Algorithm 4 has a cluster point and there exists an infinite subset of iterations K where τ_k is reduced. Then any cluster point of $\{x_k\}_K$ is a Nash equilibrium of Problem 2.3.1.*

Aside from problem specific applications, see [Pang et al., 2008] or where restrictive assumptions and conditions are required, [Facchinei and Pang, 2009], to the best knowledge of the author, [Facchinei et al., 2011] solely provides convergence results for a general nonconvex setting. Though, it is evident that Theorem 2.33 strongly relies on the assumption that a cluster point of the sequence generated by Algorithm 4 exists. Hence, naturally requires the subproblems to be solvable, which not always is guaranteed.

Remark 2.34. *An interesting, yet different approach for the distributed solution of optimization problems was presented in [Houska et al., 2016] and makes use of an augmented Lagrangian based algorithm in combination with an inexact Newton method.*

3 Generalized Nash equilibrium problems

> *As you will find in multivariable calculus, there is often a number of solutions for any given problem.*
>
> — John F. Nash

The purpose of this section is to propose an algorithmic framework for the computation of generalized Nash equilibria for a class of nonconvex differential GNEPs. More precisely, a penalty reformulation of the considered class of GNEPs subject to nonconvex shared constraints, as well as convergence results of a semi-discrete NEP to the solution of the continuous GNEP are derived. The method is based on the characterization of the reformulated GNEP as a generalized potential game, which provides a suitable structure for the use of decomposition methods, see [Facchinei et al., 2011]. In particular, the algorithm developed here is a generalized decomposition method of Gauss-Seidel type, which capitalizes on the iterative alteration of the penalty terms of the individual players. Contrary to the regularized Gauss-Seidel method given in [Facchinei et al., 2011], the penalty approach developed in this chapter guarantees that the subproblems of the respective GNEP are always solvable. Moreover, standard Gauss-Seidel methods, implicitly introduce hierarchies on the set of players through the fixed iterative sequence imposed on the players. As a consequence of the proposed penalty selection, a priori hierarchies can be avoided. In Section 3.1 first some essential definitions and concepts of optimal control theory are provided. Then, the coherence interfacing optimal control problems and generalized Nash equilibrium problems; nominally differential games is established. A partial penalty reformulation is proposed to cope with troublesome shared constraints. Convergence results of the semi-discrete NEP, which emanates from the penalty reformulation, to the solution of the continuous GNEP are provided in Section 3.2.4, before we state the decomposition algorithm with penalty selection in Section 3.2.5 and show that the proposed method omits a Nash equilibrium in a finite number of iterations. For a particular use case the conceptual ideas in this chapter have been partially published in [Britzelmeier and Dreves, 2020].

3.1 Optimal Control

In this section we provide the essential theoretical aspects of optimal control. From the philosophical perspective, the essence of optimal control is characterized by the pursuit of perfection in all things, that is, performing a task to the utmost efficiency under the given adversities in order to attain a certain objective. Projecting this concept onto a physical application, it translates to finding a control, which under given inhibitions, causes the process to be executed as efficient as possible. The purpose of optimal control is to mathematically represent such problems, on the basis of which a control can be found that ideally correlates with maximum efficiency. The notion of efficiency hereby is heavily influenced by the particular application and desired result. Since long, optimal control has established itself in various fields of application, ranging from the optimization of production processes and supply chains in the field of operations research, over the optimal control of the energy networks, to the control of autonomous vehicles or mobile robots. Whereby the type of problems and constraints in the individual applications can vary greatly. Within the framework of this thesis, only nonlinear constrained optimal control problems are considered, whose system dynamics are governed by ordinary differential equations. More generally, it is also possible to characterize the system dynamics by partial differential equations, cf. [Tröltzsch, 2009], or differential-algebraic equations, consult [Gerdts, 2011] for an extensive discussion of this topic. The fundamental theoretical framework presented in this section is in line with the explanations given in [Alt, 2012],[Gerdts, 2011],[Gerdts and Lempio, 2011] and [Stoer and Bulirsch, 2002].

3.1.1 Problem Formulation

Optimal control problems can be segregated into three main components. First, the objective function, also known as cost or utility function, which reflects intention or a desired outcome.

Definition 3.1 (Objective Function)**.** *Given a compact time interval* $[t_0, t_f]$*, with* $t_0 < t_f$*, the functions* $\varphi : \mathbb{R}^{n_x} \to \mathbb{R}$ *and* $f_0 : \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}$*, then the objective function is defined by*

$$J(x,u) := \varphi(x(t_f)) + \int_{t_0}^{t_f} f_0(u(t), x(t))\, dt. \tag{3.1}$$

The functional defined in (3.1), is composed of a boundary cost contribution $\varphi(x(t_f))$ and continuous costs, given by the integral part $f_0(u(t), x(t))$. A problem of this type is deemed a *Bolza-Problem*. Moreover, a problem with respect to (3.1) is referred to as a *Mayer-Problem* whenever $f_0(\cdot) \equiv 0$ and respectively as a *Lagrange-Problem* for $\varphi(\cdot) \equiv 0$. Second, the dynamical system, which describes the evolution of the process at hand and is ordained by the respective state and control variables. Hence, let $x(t) \in \mathbb{R}^{n_x}$ delineate the state of the system at time t and $u(t) \in \mathbb{R}^{n_u}$ the associated control, where $n_x \in \mathbb{N}$ and $n_u \in \mathbb{N}$ denote the dimensions of the

state and control vectors. Besides, let the system dynamics be governed by ordinary differential equations.

Definition 3.2 (System Dynamics). *Let $[t_0, t_f] \subset \mathbb{R}$, with $t_0 < t_f$ be a compact time interval. Moreover, let $x_0 \in \mathbb{R}^{n_x}$, as well as $x : [t_0, t_f] \to \mathbb{R}^{n_x}$, $f : \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ and $u : [t_0, t_f] \to \mathbb{R}^{n_u}$. Then, the differential equation*

$$x'(t) = f(u(t), x(t)), \tag{3.2}$$
$$x(t_0) = x_0 \tag{3.3}$$

is called the dynamics of the system.

The state and control constraints, impose limitations on the values the states and controls can assume and complement the optimal control problem. For instance, the constraints may result from physical or environmental limitations, which in practical problems are often formulated as element-wise box constraints

$$\mathbb{U} := \left\{ u \in \mathbb{R}^{n_u} \mid u_{j,min} \leq u_j \leq u_{j,\max}, j = 1, \ldots, n_u \right\}, \tag{3.4}$$

or inequality constraints.

The task within optimal control then is to find suitable state and control variables that comply with the constraints and simultaneously minimize a previously defined objective function. Thus, let us formulate the continuous nonlinear optimal control problem of Mayer-type.

Problem 3.1.1 . — *Given a compact time interval $[t_0, t_f] \subset \mathbb{R}$ with $t_0 < t_f$, find a control $u : [t_0, t_f] \to \mathbb{R}^{n_u}$ and the associated state $x : [t_0, t_f] \to \mathbb{R}^{n_x}$ such that*

$$\varphi(x(t_f)) \tag{3.5}$$

becomes minimal subject to the system of differential equations

$$x'(t) = f(u(t), x(t)), \tag{3.6}$$

the mixed control and state constraints,

$$g(u(t), x(t)) \leq 0, \tag{3.7}$$

with $g : \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_g}$, boundary conditions,

$$\chi(x(t_0), x(t_f)) = 0 \tag{3.8}$$

where $\chi : \mathbb{R}^{n_x} \to \mathbb{R}^{n_\chi}$ and the set constraints,

$$u(t) \in \mathbb{U}. \tag{3.9}$$

Herein, possible box constraints of the state x are considered to be enveloped by the mixed control and state constraints in (3.7). Let us mention, that the Mayer term is, by no means, exclusively contingent on $x(t_f)$, yet may additionally constitute a dependency on $x(t_0)$ as well. Moreover, if the functions of Problem 3.1.1 are not explicitly dependent on the time t, the problem is called *autonomous*, otherwise it is called *nonautonomous*.

Remark 3.3. *Without loss of generality only ordinary differential equations (ODEs) of order one are considered throughout this thesis, since any ODE of higher order can be transformed into a equivalent system of differential equations of order one, compare [Stoer and Bulirsch, 2002, p. 100].*

Transformation: Lagrange to Mayer-Problem

In Problem 3.1.1 we defined a problem of Mayer type, which, however, entails no limitation on the set of problems to be considered, since any Bolza or Lagrange problem can be transformed into a Mayer problem. To this end, an auxiliary state x_α is introduced, which reflects the Lagrange term, such that

$$x'_\alpha(t) = f_0(u(t), x(t)),$$

with initial value $x_\alpha(t_0) = 0$. Hence, let $\tilde{x} = (x, x_\alpha)^\top$ denote the extended state vector of dimension $n_x + 1$. Then, integration over the differential equation of the auxiliary state yields,

$$x_\alpha(t_f) = \int_{t_0}^{t_f} f_0(u(t), x(t)).$$

Consequently, for the adapted Mayer term we get

$$\tilde{\varphi}(\tilde{x}(t_f)) = \varphi(x(t_f)) + x_\alpha(t_f).$$

Remark 3.4. *In addition to the above transformation, any optimal control problem which is defined in a more general sense, for instance with free final time or as a nonautonomous problem, may always be regressed to a Mayer problem as in Problem 3.1.1 by means of suitable transformations, see for instance [Gerdts, 2011].*

3.1.2 Fundamentals of Constrained Nonlinear Optimization

For a more concise characterization of the smoothness and differentiability properties of Problem 3.1.1, some fundamental definitions and properties of function spaces are introduced in this section. In addition, a definition for a minimum of our problem can be formulated. The results presented here and beyond can be found in [Gerdts, 2011, Alt, 2012], as in this thesis we only give a short overview.

Subsequently, to define the optimal control problem appropriately, the required *Lebesgue* and *Sobolev spaces* are introduced. To this end, the definition of a *measurable function* and *Banach spaces* are considered to be known.

Definition 3.5 (L_p – *Spaces* [Gerdts, 2011, p.60]). *Let* $[t_0, t_f] \subset \mathbb{R}$, *with* $t_0 < t_f$, *delineate a compact interval.*

- *For* $1 \leq p < \infty$ *the Lebesgue space* $L_p\left([t_0, t_f]\right)$ *consists of all measurable functions* $f : [t_0, t_f] \to \mathbb{R}$ *with*

$$\int_{t_0}^{t_f} |f(t)|^p \, dt < \infty,$$

  where $|f(t)|^p$ *is Lebesgue integrable on* $[t_0, t_f]$.
- *For* $p = \infty$, *the Lebesgue space* $L_\infty\left([t_0, t_f]\right)$ *consists of all measurable functions* $f : [t_0, t_f] \to \mathbb{R}$ *for which,*

$$\operatorname*{ess\,sup}_{\tau \in [t_0, t_f]} |f(\tau)| := \inf_{N_0 \subset [t_0, t_f]} \sup_{\tau \in [t_0, t_f] \setminus N_0} |f(\tau)| < \infty,$$

  that is, f *is essentially bounded on* $[t_0, t_f]$. *Herein,* N_0 *is a set of Lebesgue measure zero.*
- *For* $1 \leq p \leq \infty$ *the Lebesgue space* $L_p^n\left([t_0, t_f]\right)$, *with* $n \in \mathbb{N}$ *denotes the product space*

$$L_p^n\left([t_0, t_f]\right) := L_p\left([t_0, t_f]\right) \times \ldots \times L_p\left([t_0, t_f]\right),$$

  where each element of $f \in L_p^n\left([t_0, t_f]\right)$ *is a mapping* $f : [t_0, t_f] \to \mathbb{R}$.

Remark 3.6. *Due to the integral being invariant towards deviations on a set of measure zero, the* $L_p([t_0, t_f])$ *spaces distinguish functions in terms of equivalence classes, i.e., two functions share the same equivalence class if they are equal almost everywhere on* $[t_0, t_f]$ *and only differ on a set of measure zero* N_0.

Before introducing the Sobolev spaces, we require the additional definition of absolute continuous functions. Properties of absolute continuous functions can be found in [Alt, 2012].

Definition 3.7 (Absolute Continuous Function). *Let* $[t_0, t_f] \subset \mathbb{R}$, *with* $t_0 < t_f$, *delineate a compact interval. A function* $f : [t_0, t_f] \to \mathbb{R}$ *is said to be absolutely continuous, if for every* $\varepsilon > 0$ *there exists a* $\delta(\varepsilon) > 0$ *such that for arbitrary* $m \in \mathbb{N}$

$$\sum_{i=1}^{m} |b_i - a_i| < \delta(\varepsilon) \qquad \Rightarrow \qquad \sum_{i=1}^{m} |f(b_i) - f(a_i)| < \varepsilon,$$

where $(a_i, b_i) \subset [t_0, t_f]$, $i = 1, \ldots, m$ *are disjoint intervals. Moreover, let* $AC\left([t_0, t_f]\right)$ *denote the space of absolutely continuous functions* $f : [t_0, t_f] \to \mathbb{R}$.

The Sobolev spaces are a natural solution space of differential equations. Moreover, with respect to partial differential equations, Sobolev spaces provide a framework for the analysis of the existence of weak solutions. In particular, the existence of viscosity solutions of HJB equations, which plays a significant role in the context of dynamic programming, cf. [Bardi and Capuzzo-Dolcetta, 2008].

Definition 3.8 ($W_{q,p}$ – *Spaces* [Gerdts, 2011, p.62])**.** *Given the compact interval* $[t_0, t_f] \subset \mathbb{R}$*, with* $t_0 < t_f$*.*

- *For* $1 \leq q, p \leq \infty$ *the Sobolev space* $W_{q,p}\left([t_0, t_f]\right)$ *consists of all absolutely continuous functions* $f : [t_0, t_f] \to \mathbb{R}$ *with absolutely continuous derivatives up to order* $q - 1$*,*

$$\|f\|_{q,p} < \infty \qquad \text{and} \qquad \frac{d^q}{dt^q} f \in L_p\left([t_0, t_f]\right)$$

where $\frac{d^q}{dt^q} f := f^{(q)}$ *is the derivative of order* q *and the norms are given by*

$$\|f\|_{q,p} := \left(\sum_{i=0}^{q} \|f^{(i)}\|^p\right)^{\frac{1}{p}}, \quad 1 \leq p < \infty,$$
$$\|f\|_{\infty} := \max_{0 \leq i \leq q} \|f^{(i)}\|_{\infty}.$$

- *For* $1 \leq p, q \leq \infty$ *the Sobolev space* $W^n_{q,p}\left([t_0, t_f]\right)$*, with* $n \in \mathbb{N}$ *denotes the product space*

$$W^n_{q,p}\left([t_0, t_f]\right) := W_{q,p}\left([t_0, t_f]\right) \times \ldots \times W_{q,p}\left([t_0, t_f]\right),$$

where each element of $f \in W^n_{q,p}\left([t_0, t_f]\right)$ *is a mapping* $f : [t_0, t_f] \to \mathbb{R}$*.*

Note, that the spaces $W_{q,p}\left([t_0, t_f]\right)$ for $1 \leq q, p \leq \infty$ and equipped with the norm $\|\cdot\|_{q,p}$ are Banach spaces. Before addressing the subsequent assertion, the definition of another smoothness property, i.e., Lipschitz continuity is required.

Definition 3.9 (Lipschitz Continuity)**.** *A function* $f : \mathbb{R} \to \mathbb{R}$ *is called Lipschitz continuous, if there exists a constant* $L \geq 0$*, such that*

$$|f(x_2) - f(x_1)| \leq L|x_2 - x_1|$$

for all $x_1, x_2 \in \mathbb{R}$*.*

Moreover, the following convention for the Lipschitz constants of an arbitrary function $f(\cdot)$ with n_α arguments is adopted. Let the Lipschitz continuity of f with respect to argument α be denoted by

$$L_{f,\alpha} \geq 0,$$

for all $\alpha = 1, \ldots, n_\alpha$. Now, for a function $f \in W_{1,1}\left([t_0, t_f]\right)$ the following continuity correlation can be established:

Lemma 3.10. *Given the compact interval* $[t_0, t_f] \subset \mathbb{R}$*, with* $t_0 < t_f$*, a function* $f \in W_{1,1}\left([t_0, t_f]\right)$ *and let the derivative exist and be bounded by a constant* $M > 0$*, i.e.,* $\|f'\|_\infty \leq M$*. Then, the function* f *is Lipschitz continuous on the interval* $[t_0, t_f]$*.*

Proof. With $f \in W_{1,1}\left([t_0, t_f]\right)$ is absolutely continuous, from the fundamental theorem of differential and integral calculus, cf. [Alt, 2012], we know that for arbitrary $t_1, t_2 \in [t_0, t_f]$

$$f(t_2) - f(t_1) = \int_{t_1}^{t_2} f'(\tau)\, d\tau$$

holds. Since $|f'(t)| \leq M$ we have

$$\begin{aligned} |f(t_2) - f(t_1)| = \left| \int_{t_1}^{t_2} f'(\tau)\, d\tau \right| &\leq \int_{t_1}^{t_2} |f'(\tau)|\, d\tau \\ &\leq |M| \int_{t_1}^{t_2} d\tau \\ &\leq |M||t_2 - t_1|. \end{aligned}$$

Thus, $f(t)$ is Lipschitz continuous on $[t_0, t_f]$, with a positive constant M. □

Furthermore, for functions $f \in W_{q,p}\left([t_0, t_f]\right)$, with q, p large enough a relation to continuous or continuous differentiable functions can be established.

Definition 3.11 (Space of k-times continuously differentiable functions)**.** *Given the compact interval* $[t_0, t_f] \subset \mathbb{R}$*, with* $t_0 < t_f$ *and* $n, k \in \mathbb{N}$*. Then, let* $C_k^n\left([t_0, t_f]\right)$ *delineate the space of k-times continuously differentiable functions.*

With respect to Problem 3.1.1 we consider the differential state $x \in W_{1,1}^{n_x}\left([t_0, t_f]\right)$ and the control $u \in L_1^{n_u}\left([t_0, t_f]\right)$. Moreover, throughout let $\|\cdot\|$ denote the Euclidean norm and let

$$\|\cdot\|_\infty := \operatorname*{ess\,sup}_{t \in [a,b]} \|\cdot\|$$

be the essential supremum norm. A global or local minimizer of Problem 3.1.1 is defined according to

Definition 3.12. *Suppose* $(\hat{u}, \hat{x}) \in \left(L_1^{n_u}\left([t_0, t_f]\right) \cap \mathbb{U}\right) \times W_{1,1}^{n_x}\left([t_0, t_f]\right)$ *is feasible for* Problem 3.1.1*. Then, we call* $(\hat{u}, \hat{x})$ *a*

(i) global (weak) minimizer, if

$$\varphi(\hat{x}(t_f)) \leq \varphi(x(t_f))$$

for all feasible (u, x)*.*

(ii) local (weak) minimizer, if

$$\varphi(\hat{x}(t_f)) \leq \varphi(x(t_f))$$

for all feasible (u, x)*, with*

$$\|x - \hat{x}\|_{1,1} \leq \varepsilon, \quad \|u - \hat{u}\|_1 \leq \varepsilon$$

for some $\varepsilon > 0$.

Remark 3.13. *In general, minimizers can further be distinguished into strict local and global minimizer if* $\leq$ *is substituted by* $<$*. Moreover, one can categorize further between strong and weak minimizer, see for instance [Gerdts, 2011, p.341].*

Furthermore, considering the constraints on the control $u(t) \in \mathbb{U}$, the function space of u needs to be adapted accordingly,

$$\mathcal{U} := \left\{ u \in L_1^{n_u}\left([t_0, t_f]\right) \mid u(t) \in \mathbb{U} \text{ almost everywhere (a.e.) in } [t_0, t_f] \right\}.$$

The state and control variables are defined on the set,

$$(u, x) \in \Sigma := \mathcal{U} \times W_{1,1}^{n_x}\left([t_0, t_f]\right).$$

Moreover, we can summarize the equality constraints imposed on Problem 3.1.1 yielding,

$$h(u, x) := \begin{pmatrix} f(u(t), x(t)) - x'(t) \\ \chi(x(t_0), x(t_f)) \end{pmatrix} = 0$$

with the function $h : \Sigma \to \mathbb{R}^{n_x} \times \mathbb{R}^{n_\chi}$. Then, the feasible set of Problem 3.1.1 is defined as

$$X := \{(u, x) \in \Sigma \mid h(u, x) = 0,\ g(u, x) \leq 0\}.$$

3.1.3 Discretization of Optimal Control Problems

Several approaches from the theory of optimization are available to solve the continuous constrained optimal control problem (OCP) in Problem 3.1.1 numerically. Essentially, one can distinguish three distinct approaches for the solution of the respective problem, i.e., indirect methods, direct methods and dynamic programming. Indirect methods are based on the formulation of a boundary value problem using the necessary conditions, which is then discretized and solved numerically. Therefore, this approach is often referred to as *"first optimize, then discretize"*. Direct methods, on the other hand, are based on the discretization of the infinite dimensional optimal control problem, which leads to a finite dimensional nonlinear program that can be solved by established optimization methods. Consequently, direct methods are referred to as *"first discretize, then optimize"* approaches. Suitable algorithms and results for direct and indirect methods are presented in [Gerdts, 2011],[Geiger and Kanzow, 2002] and [Nocedal and Wright, 2006]. The dynamic programming approach is based on the principle of optimality introduced by [Bellman, 1957], which together with its correlation to the Hamilton-Jacobi-Bellman (HJB)

equation is discussed in more detail in Chapter 4. Throughout this thesis a first discretize then optimize approach is pursued. Therefore, we subsequently present a discretization scheme which permits an approximation of Problem 3.1.1 by a finite dimensional optimization problem. To this end, a discretization of the state and control functions, respectively $x(\cdot)$ and $u(\cdot)$, with respect to time is adopted. Accordingly, we introduce a lattice

$$\mathbb{G}_h := \{t_\ell\}_{\ell=0,\dots,M} \tag{3.10}$$

with $M+1 \in \mathbb{N}$ the number of grid points and $h_\ell = t_{\ell+1} - t_\ell$ the step size for $\ell = 0, \dots, M-1$, where $t_0 < t_1 < \dots < t_M = t_f$. On the grid, the controls and states, as well as the constraints, are solely evaluated on the grid points t_ℓ, $\ell = 0, \dots, M$. Furthermore, let $x_h(t_\ell)$ for all $\ell = 0, \dots, M$ and $u_h(t_\ell)$ for all $\ell = 0, \dots, M-1$ constitute the approximations of the state and control functions on the grid $\mathbb{G}_h$.

Remark 3.14. *Often, considering simplicity, it is customary to choose an equidistant grid, where $h = h_\ell$ for all $\ell = 0, \dots, M-1$, with respect to the discretization. In the context of interpolation, this approach can beneficially be exploited, see Appendix A.2.*

Moreover, for the controls $u(t)$, e.g., a piecewise constant or a piecewise linear approximation may be considered. The system of differential equations

$$x'(t) = f(u(t), x(t)) \qquad \text{with} \qquad t \in [t_0, t_f]$$

is defined in terms of an initial value problem, with initial values $x(t_0) = x_0$. An extensive theory on the numerical solution of differential equations can be found in [Stoer and Bulirsch, 2002]. Without loss of generality, within the scope of this thesis *one step methods* will be employed.

Definition 3.15 (One-Step Method)**.** *Let $\Phi : \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R} \to \mathbb{R}^{n_x}$ be a continuous function and $\mathbb{G}_h$ a grid in compliance with (3.10). Then, the method*

$$x(t_{\ell+1}) = x(t_\ell) + h_\ell \Phi(u(t_\ell), x(t_\ell), h_\ell), \qquad \ell = 0, \dots, M-1$$

is called a one step method and the function Φ is called increment function.

One step methods are aptly named, since the function Φ is only dependent on the previous time step at t_ℓ. The simplest of the one step methods being the explicit Euler method, which for the dynamic of Problem 3.1.1 yields the approximation:

$$x(t_{\ell+1}) = x(t_\ell) + h_\ell f(u(t_\ell), x(t_\ell)),$$

for all $\ell = 0, \dots, M-1$.

Consequently, we can define the respective discrete optimal control problem.

Problem 3.1.2 . — *Given a compact time interval* $[t_0, t_f] \subset \mathbb{R}$ *with* $t_0 < t_f$*, find a control function* $u_h : \mathbb{G}_h \to \mathbb{R}^{n_u}$ *and the associated state function* $x_h : \mathbb{G}_h \to \mathbb{R}^{n_x}$ *such that*

$$\varphi(x_h(t_M)) \tag{3.11}$$

becomes minimal, subject to the discrete system of difference equations

$$x_h(t_{\ell+1}) = x_h(t_\ell) + h_\ell f(u_h(t_\ell), x_h(t_\ell)), \qquad \forall \ell = 0, \dots, M-1 \tag{3.12}$$

the pointwise mixed control and state constraints,

$$g(u_h(t_\ell), x_h(t_\ell)) \leq 0, \qquad \forall \ell = 0, \dots, M \tag{3.13}$$

with $g : \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_g}$*, boundary conditions,*

$$\chi(x_h(t_0), x_h(t_f)) = 0 \tag{3.14}$$

where $\chi : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_\chi}$ *and the set constraints,*

$$u_h(t_\ell) \in \mathbb{U}, \qquad \forall \ell = 0, \dots, M-1 \tag{3.15}$$

Remark 3.16. *Further details and approaches on the discretization of optimal control problems for direct and indirect methods are available in [Betts, 2020], [Stoer and Bulirsch, 2002] and [Gerdts and Lempio, 2011, Gerdts, 2011].*

3.2 Differential Generalized Nash Equilibrium Problems

With hindsight to the decomposition approach for solving nonconvex GNEPs in the previous section, the purpose of this section is to propose a general decomposition algorithm of Gauss-Seidel type without regularization. More specificly a decomposition algorithm with penalty selection. The algorithm developed here is of particular interest, since the penalty approach ensures that the indvidual subproblems are always solvable, which as mentioned before cannot be guaranteed in the regularized approach in [Facchinei et al., 2011]. Furthermore, due to the penalty selection scheme an a priori introduced hierarchy, which naturally emerges in the Gauss-Seidel scheme can be avoided. It can be shown that the considered GNEP also is a generalized potential game as introduced in [Facchinei et al., 2011]. We follow a first discretize then optimize approach, which is quite common in practical applications. The continuous constrained and nonconvex GNEP is reformulated into a penalized NEP, shifting the nonconvex constraints into the cost function. Then, the resulting standard NEP is semi-discretized. Prior to the statement of the algorithm, we provide a proof of convergence of a sequence of solutions of the semi-discretized

NEP to the solution of the continuous GNEP in the state variables, provided the latter omits a solution. Additionally, it is shown that a suitable optimal control can be found. Eventually, the decomposition algorithm with penalty selection is introduced to solve the discrete generalized potential game. Furthermore, a proof is provided that the algorithm converges in a finite number of iterations. A particular application of the results presented in this section have already been published in [Britzelmeier and Dreves, 2020].

3.2.1 Continuous Generalized Nash Equilibrium Problem

As established in the previous section, the task within an optimal control problem is to minimize a functional with respect to certain constraints. If the problem at hand is independent of system dynamics it is considered to be static, since no evolution over time is present. The imposition of the aspect of time accordingly implicates that the strategies themselves must be functions of time. Hence, the agents then can be considered to be substitutes of physical systems governed by kinematic or dynamic equations in the form of ordinary differential equations. Consequently, an individual agents problem is embedded into an optimal control problem. Peculiarly, the natural counterpart to optimal control problems within game theory, by adjoining opposing players, is called a differential game. First introduced and extensively studied by [Isaacs, 1999], differential games are strategic games where the states of each player are governed by differential equations. Simply speaking, a differential game is a system of coupled (parametric) optimal control problems. Conversely a differential game can be formulated as an optimal control problem by inhibiting other players by means of decomposition.

We consider a N player game $\Gamma = \{X_\nu(\varkappa^{-\nu}), \theta_\nu\}_{\nu=1}^N$ and subsequently give a more detailed definition of Problem 2.3.1. The game is played on a fixed time horizon $\left[t_0, t_f\right]$ and additionally governed by ordinary differential equations. In fact, the differential dependencies make the problem even harder to solve, compared to a common strategic game. Furthermore, considering the fixed time horizon as a preview horizon, permits a convenient transition to an MPC scheme later on. To begin with we define the controls

$$u = \left(u^1, \dots, u^N\right),$$

where each $u^\nu : [t_0, t_f] \to \mathbb{R}^{n_u}$ is a measurable function for all $\nu = 1, \dots, N$, with n_u the dimension of controls. Further, for each player $\nu = 1, \dots, N$ let us define the set of admissible controls

$$\mathbb{U}_\nu := \left\{u^\nu \in \mathbb{R}^{n_u} \mid u^\nu_{j,\min} \le u^\nu_j \le u^\nu_{j,\max}, j = 1, \dots, n_u\right\},$$

and let the function space of u^ν for all $\nu = 1, \dots, N$ be defined by

$$\mathcal{U}_\nu := \left\{u^\nu \in L_1^{n_u}\left([t_0, t_f]\right) \mid u^\nu(t) \in \mathbb{U}_\nu, \text{ for a.e. } t \in [t_0, t_f]\right\}.$$

Accordingly, let the states, for all $\nu = 1, \dots, N$ be absolutely continuous functions,

$$x = \left(x^1, \dots, x^N\right),$$

with $x^\nu : [t_0, t_f] \mapsto \mathbb{R}^{n_x}$ and $x^\nu \in W_{1,1}^{n_x}\left([t_0, t_f]\right)$. Herein, n_x indicates the state dimension. Unless otherwise indicated x^ν, u^ν are to be understood as vectors of their respective dimension. Then, the GNEP amounts to each player $\nu = 1, \dots, N$ solving the optimal control problem

$$\begin{aligned}
\min_{(u^\nu, x^\nu)} \quad & \varphi^\nu(x^\nu(t_f)) \\
\text{subject to} \quad & x^\nu(t_0) = x_0^\nu, \\
& (x^\nu)'(t) = f(u^\nu(t), x^\nu(t)) && , a.e.\ t \in [t_0, t_f], \\
& k^\nu(x^\nu(t)) \leq 0 && , \forall\ t \in [t_0, t_f], \\
& g^\nu(x(t)) \leq 0 && , \forall\ t \in [t_0, t_f], \\
& u^\nu(t) \in \mathbb{U}_\nu.
\end{aligned} \tag{GNEP}$$

Where x_0^ν is the initial value at time $t = t_0$. The ordinary differential equation describes the evolution of the system by its right hand side function $f : \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$. The control u^ν is bounded from above by $u_{\max}^\nu$ and from below through $u_{\min}^\nu$. Furthermore, two types of inequality constraints are considered. In particular, individual constraints $k^\nu(x^\nu(t)) \leq 0$, only depending on player ν's strategies and $g^\nu(x(t)) \leq 0$, which harbours shared and troublesome constraints, where the shared constraints depend on the decision variables of all player. Forthwith, we assume that for each player $\nu = 1, \dots, N$ the constraints $g^\nu : \mathbb{R}^{n_x} \to \mathbb{R}^{m_\nu}$, with $m_\nu \in \mathbb{N}$ the number of constraints, are not necessarily convex.

Moreover, let the individual constraints k^ν be defined such that,

$$k^\nu : \mathbb{R}^{n_x} \to \mathbb{R}^{q_\nu}, \tag{3.16}$$

incorporates boundary conditions on the state variables, where $q_\nu \in \mathbb{N}$ denotes the number of individual constraints for each player $\nu = 1, \dots, N$. Additional we require the following assumption on the set of admissible states.

Assumption. 4 (Compact Set). For all $\nu = 1, \dots, N$ assume that $\{x^\nu \mid k^\nu(x^\nu) \leq 0\}$ is compact.

3.2.2 Penalized Nash Equilibrium Problem

To cope with the troublesome nonconvex constraints, we choose to take the natural course of employing a penalty reformulation of the problem. Additionally, since the cost functions are of Mayer type and moreover, for all player in (GNEP) the objectives are independent of the rivals strategies, we have a generalized potential game. Enabling us to write (GNEP) in a more compact manner. Therefore, two sets are introduced. One holding the equality and box constraints as

well as the individual inequality constraints, and another holding the intricate and nonconvex constraints. To this end, for all player $\nu = 1, \dots, N$ let the feasible set be defined by

$$\begin{aligned} X_\nu := \big\{ (u^\nu, x^\nu) \mid\ & x^\nu(0) = x_0^\nu, \\ & (x^\nu)'(t) = f(u^\nu(t), x^\nu(t)), \quad \text{a.e. } t \in [t_0, t_f], \\ & k^\nu(x^\nu(t)) \leq 0, \qquad\qquad\quad \forall t \in [t_0, t_f], \\ & u^\nu(t) \in \mathbb{U}_\nu \qquad\qquad\qquad\qquad\quad \big\}. \end{aligned}$$

For the set holding the troublesome constraints we define

$$G(x) := \big\{ x \mid g^\nu(x(t)) \leq 0, \forall\ t \in [t_0, t_f], \forall \nu = 1, \dots, N \big\}.$$

With these definitions and the given potential game structure, the GNEP can be reformulated to a single optimization problem of the form

$$\min_{(u,x)} \quad \sum_{\nu=1}^{N} \varphi^\nu(x^\nu(t_f)) \quad \text{s.t.} \quad (u,x) \in \bigtimes_{\nu=1}^{N} X_\nu,\ x \in G(x), \tag{GPG}$$

namely a generalized potential game, cf.[Facchinei et al., 2011]. As we have established in Section 2.5, due to its complexity in conjunction with the nonconvex feasible set $G(x)$, solving (GPG) is a challenging if not impossible task.

Consequently, pursuing the penalization approach by adopting a function $P : \mathbb{R}^{Nn_x} \to \mathbb{R}$, which constitutes a penalty function if the function inherently satisfies the subsequent properties:

$$P(x) = 0, \qquad \text{if} \quad (u,x) \in \bigtimes_{\nu=1}^{N} X_\nu \quad \text{and} \quad P(x) > 0, \qquad \text{if} \quad (u,x) \notin \bigtimes_{\nu=1}^{N} X_\nu.$$

Due to its additive nature with respect to minimization, a penalty function daunts a potential violation of the constraints it encompasses. Moreover, the (GNEP) translates to a standard Nash equilibrium problem, which inherently reduces the complexity. To this end, we define the function

$$P(x) := \frac{1}{(t_f - t_0)} \sum_{\nu=1}^{N} \sum_{j=1}^{m_\nu} \int_{t_0}^{t_f} \max\{0, g_j^\nu(x(t))\}\, dt, \tag{3.17}$$

which characterizes the constraints imposed by the set $G(x)$, with the player individual contribution for all $\nu = 1, \dots, N$ to (3.17) denoted by

$$P^\nu(x) := \frac{1}{(t_f - t_0)} \sum_{j=1}^{m_\nu} \int_{t_0}^{t_f} \max\{0, g_j^\nu(x(t))\}\, dt, \tag{3.18}$$

characterizing the shared constraints $g^\nu(\cdot)$. Thus, disposing of the difficult nonconvexities and yielding

$$\min_{(u^\nu, x^\nu)} \quad \varphi(x^\nu(t_f)) + \rho P^\nu(x^\nu, x^{-\nu}) \qquad \text{s.t.} \quad (u^\nu, x^\nu) \in X_\nu \tag{NEP}$$

the penalized Nash equilibrium problem, with the positive constant ρ.

Remark 3.17. *Some remarks on the stated problems:*

- *By separating the constraint sets the underlying GNEP is still recognizable in (GPG), since as usual in the theory of GNEPs, the feasible set is dependant on the current strategies x through $x \in G(x)$.*
- *Further, the formulation of the objective function as sole Mayer term is not a restriction of the considered class of problems, since it is well known, that a Lagrange or Bolza type problem can be transformed into a Mayer problem and vice versa, compare Section 3.1.*
- *It is easily possible to introduce further shared constraints, by treating them as $g_{\nu\mu}$ functions as in Chapter 5. For notational simplicity, we restrict our presentation to the common constraints g^ν since they already possess by assumption the troublesome nonconvexity property.*

3.2.3 Discrete Penalized Nash Equilibrium Problem

The subsequent discretization scheme is adapted to solve (GNEP). Let

$$\begin{aligned} &\mathbb{G}_h := \{t_i \mid i = 0, \dots, M\} \\ &\text{with} \quad h = \frac{t_f - t_0}{M}, \end{aligned} \tag{3.19}$$

and $t_i = ih$ for $i = 0, \dots, M$ be an equidistant lattice on $[t_0, t_f]$. Accordingly, let the controls u_h^ν assume values on an equidistant grid, such that

$$\begin{aligned} &u_h^\nu \in \mathbb{U}_\nu^{h_u^\nu} := \left\{u \in \mathbb{R}^{n_u} \mid u = u_{\min} + jh_u^\nu,\ j = 0, \dots, M_u^\nu\right\} \\ &\text{with} \quad h_u^\nu = \frac{u_{\max}^\nu - u_{\min}^\nu}{M_u^\nu}. \end{aligned} \tag{3.20}$$

Moreover, we consider the finite dimensional subspaces of $L_1^{n_u}\left([t_0, t_f]\right)$ and $W_{1,1}^{n_x}\left([t_0, t_f]\right)$. In particular, the subspaces

$$\begin{aligned} L_{1,h}^{n_u}\left([t_0, t_f]\right) &:= \left\{u : [t_0, t_f] \to \mathbb{R}^{n_u} \mid \ u(t) = u(t_i),\ t \in [t_i, t_{i+1}),\ i = 1, \dots, M-1\right\} \\ W_{1,1,h}^{n_x}\left([t_0, t_f]\right) &:= \left\{x_h \in W_{1,1}^{n_x}\left([t_0, t_f]\right) \mid x_h(t) = x_h(t_i) + \frac{t - t_i}{h}\left(x_h(t_{i+1}) - x_h(t_i)\right),\right. \\ &\qquad \left. t \in [t_i, t_{i+1}],\ i = 0, \dots, M-1\right\} \end{aligned}$$

are defined and we introduce the set,

$$\mathcal{U}_\nu^{h_u^\nu} := \left\{ u_h \in L_{1,h}^{n_u} \mid u_h(t_i) \in \mathbb{U}_\nu^{h_u^\nu},\ i = 1, \dots, M \right\}$$

such that $u_h^\nu(t) \in \mathcal{U}_\nu^{h_u^\nu}$ denote piecewise constant approximations on the equidistant control grid. The discrete states $x_h^\nu \in W_{1,1,h}^{n_x}\left([t_0, t_f]\right)$ are defined by an explicit Euler scheme, which yields piecewise linear approximations,

$$x^\nu(t_{i+1}) = x^\nu(t_i) + hf(u^\nu(t_i), x^\nu(t_i)).$$

Additionally the following definitions are required to simplify the notation later on:

$$\begin{aligned} h_u &:= \max_{\nu=1,\dots,N} h_u^\nu \\ x_{\max} &:= \max_{\nu=1,\dots,N} x_{\max}^\nu \\ B &:= \max_{\nu=1,\dots,N} \max\left\{ |u_{\min}^\nu|, |u_{\max}^\nu| \right\}. \end{aligned}$$

Here, $x_{\max}^\nu$ characterises an upper bound on the admissible states.

With the discretization scheme and by collecting the remaining constraints, let the feasible set of the discrete penalized problem be defined by

$$\begin{aligned} X_\nu^h := \Big\{ \left(u_h^\nu, x_h^\nu\right) \in L_{1,h}^{n_u} \times W_{1,1,h}^{n_x} \mid\ & x_h^\nu(0) = x_0^\nu, \quad \text{and} \quad \forall i = 0, \dots, M-1 : \\ & x_h^\nu(t_{i+1}) = x_h^\nu(t_i) + h_i f(u_h^\nu(t_i), x_h^\nu(t_i)), \\ & k^\nu(x_h^\nu(t_i)) \le (h + h_u)^p, \\ & u_h^\nu(t_i) \in \mathbb{U}_\nu^{h_u^\nu} \Big\} \end{aligned}$$

for every $\nu = 1, \dots, N$ and a positive constant $p < 1$. It remains to introduce the discrete penalized problem. Therefore, we first define the joint penalty term of all player,

$$P(x_h) := \frac{1}{M} \sum_{\nu=1}^{N} \sum_{j=1}^{m_\nu} \sum_{i=0}^{M-1} \max\{0, g_j^\nu(x_h(t_{i+1}))\}, \tag{3.21}$$

thus, the individual contribution of each player is given by,

$$P^\nu(x_h) := \frac{1}{M} \sum_{j=1}^{m_\nu} \sum_{i=0}^{M-1} \max\{0, g_j^\nu(x_h(t_{i+1}))\}. \tag{3.22}$$

Eventually, using the penalty term (3.22) together with a penalty parameter $\rho := \frac{1}{(h+h_u)^p}$, we attain the discrete NEP,

$$\min_{(u_h^\nu, x_h^\nu)} \quad \varphi^\nu(x_h^\nu(t_f)) + \frac{1}{(h+h_u)^p} P(x_h^\nu, x_h^{-\nu}) \quad \text{s.t.} \quad (u_h^\nu, x_h^\nu) \in X_\nu^h \tag{NEP$_h$}$$

for all $\nu = 1, \dots, N$ and a constant $p < 1$. Furthermore, the following smoothness assumptions are required:

Assumption. 5 (Lipschitz Continuous Mayer-Term). φ^ν is Lipschitz continuous, with a Lipschitz constant L_φ on X_ν^h.

It immediately follows for all $\nu = 1, \dots, N$ with the states x^ν bounded and φ^ν continuous, that φ^ν is also bounded.

Assumption. 6 (Lipschitz Continuous Dynamics). For all $\nu = 1, \dots, N$ assume f is a Lipschitz continuous function with respect to all arguments on X_ν^h.

Consequently, for f a continuous function and x^ν, u^ν defined on a compact set we have,

$$(x^\nu)'(t) = f(u^\nu, x^\nu)$$

bounded. This in conjunction with x^ν absolutely continuous leads to x^ν being Lipschitz continuous.

Assumption. 7 (Control Monotonicity). For arbitrary controls $u \leq v$ let $f(u, x^\nu) \leq f(v, x^\nu)$ for all fixed $x^\nu \in X_\nu^h$.

Remark 3.18. *Let us comment on the assumptions:*

- *The continuity assumptions are common and impose no significant restrictions on the potential class of problems.*
- *Note, that the monotonicity assumption is a necessary assumption for the proof later on. Moreover, this assumption is quite restrictive and might not always be satisfied. For instance, if the dynamics depend linearly on the control and there exists a prefactor matrix, e.g.,*

  $$x'(t) = f(x(t)) + Mu,$$

  the admissibility of the assumption highly depends on the properties of the matrix M. Whenever, $M \geq 0$ componentwise the control monotonicity is satisfied. More general, control-affine systems, i.e.,

  $$x'(t) = f(x(t)) + \sum_{i=1}^{m} h_i(x(t))u_i$$

  constitute a sufficient condition for the control monotonicity to hold whenever $0 < h_i$ for all $i = 1, \dots, m$. An example where the assumption is satisfied is discussed in more detail in Chapter 5.

Next, given the monotonicity assumption of the controls, we can establish the following intermediate result.

Lemma 3.19. *Let Assumption 7 hold. Moreover, let $f_k : \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}$ be a measurable function. Then, for an arbitrary absolutely continuous function $x^\nu \in W_{1,1}^{n_x}([t_i, t_{i+1}])$, a measurable function $u^\nu : [t_i, t_{i+1}] \to [u^\nu_{\min}, u^\nu_{\max}]$, with $u^\nu \in L_1^{n_u}([t_i, t_{i+1}])$ and $t \in [t_i, t_{i+1}]$, as well as for each $k = 1, \dots, n_x$,*

$$\int_{t_i}^{t_{i+1}} f_k(u^\nu_{\min}, x^\nu(\tau))\, d\tau \leq \int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau))\, d\tau \leq \int_{t_i}^{t_{i+1}} f_k(u^\nu_{\max}, x^\nu(\tau))\, d\tau$$

holds.

Proof. It follows from the assumption that $f_k(u^\nu(\cdot), x^\nu(\cdot))$ is a measurable function. From the monotonicity property of Lebesgue integrable functions, compare [Alt, 2012, A1.12, p.86], for arbitrary measurable functions $f_{k,1} \leq f_{k,2}$ we have that

$$\int_\Omega f_{k,1}(\tau)\, d\tau \leq \int_\Omega f_{k,2}(\tau)\, d\tau.$$

By Assumption 7 we know for $u^\nu_{\min} \leq u^\nu(t)$, for all $t \in [t_i, t_{i+1}]$, $f_k(u^\nu_{\min}, x^\nu(t)) \leq f_k(u^\nu(t), x^\nu(t))$. Thus, setting

$$\begin{aligned} f_{k,1}(t) &= f_k(u^\nu_{\min}, x^\nu(t)), \\ f_{k,2}(t) &= f_k(u^\nu(t), x^\nu(t)). \end{aligned}$$

together with $\Omega = [t_i, t_{i+1}]$:

$$\int_{t_i}^{t_{i+1}} f_k(u^\nu_{\min}, x^\nu(\tau))\, d\tau \leq \int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau))\, d\tau,$$

and equivalently for $u^\nu(t) \leq u^\nu_{\max}$:

$$\int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau))\, d\tau \leq \int_{t_i}^{t_{i+1}} f_k(u^\nu_{\max}, x^\nu(\tau))\, d\tau,$$

concluding the proof. □

Remark 3.20. *Note that by our setting in (NEP_h) we only have a finite number of strategies, since the piecewise constant controls are defined on an equidistant grid, and therefore the states may assume only a finite number of piecewise linear functions. Hence, a solution of (NEP_h) exists. Moreover, the setting of the (NEP_h) can be characterised as a sampled-data system with zero-order hold.*

3.2.4 Convergence Theory

Before introducing the decomposition algorithm, we require some convergence properties of the stated class of problems. In particular, that by refining the discretization, an accumulation point of a sequence of solutions of the discretized (NEP$_h$) converges, at least in the state variables, to a solution of (GNEP). Additionally, it can be shown that a corresponding and also optimal control can be found. The theory developed in this section is applied to a coordination problem of autonomous vehicles in Chapter 5.

Initially we introduce the following technical estimate, which unburdens the proof of the subsequent results later on. Without loss of generality we subsequently assume that $t_0 = 0$ and the final time is a constant $t_f = T$, with $T > 0$.

Lemma 3.21. *Suppose two functions x, x_h satisfy for all $i = 0, \dots, M-1$*

$$|x(t_{i+1}) - x_h(t_{i+1})| \leq (1 + C_1 h)|x(t_i) - x_h(t_i)| + Lhh_u + C_2 h^2$$

with some constants $L, C_1, C_2 > 0$. Then we have for all $i = 0, \dots, M-1$,

$$|x(t_{i+1}) - x_h(t_{i+1})| \leq e^{C_1 T}|x(t_0) - x_h(t_0)| + (Lh_u + C_2 h)\frac{e^{C_1 T} - 1}{C_1}.$$

Proof. With the definition $h = \frac{T}{M}$ the known estimate $(1 + \frac{\alpha}{M})^M \leq e^{\alpha}$, for all $M \in \mathbb{N}$, $0 \leq \alpha$ and $0 \leq i \leq M-1$ we obtain

$$\begin{aligned}
|x(t_{i+1}) - x_h(t_{i+1})| &\leq (1 + C_1 h)|x(t_i) - x_h(t_i)| + Lhh_u + C_2 h^2 \\
&\leq (1 + C_1 h)^{i+1}|x(t_0) - x_h(t_0)| + (Lhh_u + C_2 h^2)\sum_{j=0}^{i}(1 + C_1 h)^j \\
&\leq \left(1 + \frac{C_1 T}{M}\right)^M |x(t_0) - x_h(t_0)| + (Lhh_u + C_2 h^2)\sum_{j=0}^{M-1}\left(1 + \frac{C_1 T}{M}\right)^j \\
&\leq e^{C_1 T}|x(t_0) - x_h(t_0)| + (Lhh_u + C_2 h^2)\frac{\left(1 + \frac{C_1 T}{M}\right)^M - 1}{\left(1 + \frac{C_1 T}{M}\right) - 1} \\
&\leq e^{C_1 T}|x(t_0) - x_h(t_0)| + (Lhh_u + C_2 h^2)\frac{e^{C_1 T} - 1}{\frac{C_1 T}{M}} \\
&= e^{C_1 T}|x(t_0) - x_h(t_0)| + (Lh_u + C_2 h)\frac{e^{C_1 T} - 1}{C_1}
\end{aligned}$$

□

The next lemma is an approximation result, showing that every feasible point of (GNEP) can be approximated through a sequence of feasible points for (NEP$_h$). The proof is very technical

and lengthy, but it is constructive in giving explicit formulas for the approximation and the error.

Lemma 3.22. *Let Assumptions 6 to 7 hold. Let (u, x) be a feasible point of (GNEP). Then for positive h, h_u sufficiently small, we can find a point (u_h, x_h) that is feasible for (NEP$_h$). Moreover, we have constants $C_3, C_4 > 0$ independent of h, h_u such that the following approximations holds:*

$$\begin{aligned} \|x - x_h\|_\infty &\leq C_3(h + h_u), \\ P(x_h) &\leq C_4(h + h_u). \end{aligned}$$

Proof. The proof is divided into four steps. Note that the set of admissible controls is compact.

1. First, a discrete approximation (u_h^ν, x_h^ν) of (u^ν, x^ν) for every $\nu = 1, \dots, N$ is defined recursively. By Lemma 3.19 we have for each $k = 1, \dots, n_x$,

$$\int_{t_i}^{t_{i+1}} f_k(u_{\min}^\nu, x^\nu(\tau))\, d\tau \leq \int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau))\, d\tau \leq \int_{t_i}^{t_{i+1}} f_k(u_{\max}^\nu, x^\nu(\tau))\, d\tau.$$

   Hence, using the intermediate value theorem there exists a constant control $\tilde{u}_h^\nu(t_i) \in \left[u_{\min}^\nu, u_{\max}^\nu\right]$ such that

$$\int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x(\tau))\, d\tau = \int_{t_i}^{t_{i+1}} f_k(\tilde{u}_h^\nu(t_i), x(\tau))\, d\tau \tag{3.23}$$

   holds for each $k = 1, \dots, n_x$. Next we define,

$$u_h^\nu(t_i) := \Xi_{\{u_{\min}^\nu + j h_u^\nu \mid j=0,\dots,M_u^\nu\}}(\tilde{u}_h^\nu(t_i))$$

   as the projection of $\tilde{u}_h^\nu(t_i)$ onto a grid point in $\left[u_{\min}^\nu, u_{\max}^\nu\right]$. Further, we define our approximation by piecewise constant controls,

$$u_h^\nu(t) = u_h^\nu(t_i), \qquad \forall t \in [t_i, t_{i+1}[\,.$$

   Moreover, setting $x_h^\nu(t_0) = x^\nu(t_0)$ and for the piecewise linear states, defined for all $t \in [t_i, t_{i+1}[$ and for all $i = 0, \dots, M$ and all $\nu = 1, \dots, N$

$$x_h^\nu(t) := x_h^\nu(t_i) + (t - t_i) f(u_h^\nu(t_i), x_h^\nu(t_i)). \tag{3.24}$$

2. Next, we estimate the error of x_h^ν to x^ν for all players $\nu = 1, \dots, N$. Therefore, consider an arbitrary player $\nu \in \{1, \dots, N\}$. Given the definitions we can write for each $k = 1, \dots, n_x$

$$\begin{aligned} &\left| x_k^\nu(t_{i+1}) - x_{h,k}^\nu(t_{i+1}) \right| \\ &= \left| x_k^\nu(t_i) + \int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau))\, d\tau - x_{h,k}^\nu(t_i) - h f_k(u_h^\nu(t_i), x_h^\nu(t_i)) \right| \end{aligned}$$

$$
\begin{aligned}
&\leq \left|x_k^\nu(t_i) - x_{h,k}^\nu(t_i)\right| + \left|\int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x_h^\nu(t_i))\, d\tau\right| \\
&\leq \left|x_k^\nu(t_i) - x_{h,k}^\nu(t_i)\right| + \left|\int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau)) - f_k(\tilde{u}_h^\nu(t_i), x^\nu(\tau))\, d\tau\right| \\
&+ \left|\int_{t_i}^{t_{i+1}} f_k(\tilde{u}_h^\nu(t_i), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x^\nu(t_i))\, d\tau\right| \\
&+ \left|\int_{t_i}^{t_{i+1}} f_k(u_h^\nu(t_i), x^\nu(t_i)) - f_k(u_h^\nu(t_i), x_h^\nu(t_i))\, d\tau\right| \\
&\leq \left|x_k^\nu(t_i) - x_{h,k}^\nu(t_i)\right| + \underbrace{\left|\int_{t_i}^{t_{i+1}} f_k(u^\nu(\tau), x^\nu(\tau)) - f_k(\tilde{u}_h^\nu(t_i), x^\nu(\tau))\, d\tau\right|}_{=0} \\
&+ \left|\int_{t_i}^{t_{i+1}} f_k(\tilde{u}_h^\nu(t_i), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x^\nu(t_i))\, d\tau\right| \\
&+ h\left|f_k(u_h^\nu(t_i), x^\nu(t_i)) - f_k(u_h^\nu(t_i), x_h^\nu(t_i))\right| \\
&= \left|x_k^\nu(t_i) - x_{h,k}^\nu(t_i)\right| + \left|\int_{t_i}^{t_{i+1}} f_k(\tilde{u}_h^\nu(t_i), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x^\nu(\tau))\right. \\
&\left. + f_k(u_h^\nu(t_i), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x^\nu(t_i))\, d\tau\right| \\
&+ h\left|f_k(u_h^\nu(t_i), x^\nu(t_i)) - f_k(u_h^\nu(t_i), x_h^\nu(t_i))\right| \\
&\leq \left(1 + \max_{k=1,\dots,n_x} \left\{L_{f,2}^k\right\} h\right) \left\|x^\nu(t_i) - x_h^\nu(t_i)\right\| \\
&+ \left|\int_{t_i}^{t_{i+1}} f_k(\tilde{u}_h^\nu(t_i), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x^\nu(\tau))\, d\tau\right| \\
&+ \left|\int_{t_i}^{t_{i+1}} f_k(u_h^\nu(t_i), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x^\nu(t_i))\, d\tau\right| \\
&\leq \left(1 + \max_{k=1,\dots,n_x} \left\{L_{f,2}^k\right\} h\right) \left\|x^\nu(t_i) - x_h^\nu(t_i)\right\| \\
&+ \int_{t_i}^{t_{i+1}} \max_{k=1,\dots,n_x} \left\{L_{f,1}^k\right\} \left\|\tilde{u}_h^\nu(t_i) - u_h^\nu(t_i)\right\| d\tau \\
&+ \left|\int_{t_i}^{t_{i+1}} f_k(u_h^\nu(t_i), x^\nu(\tau)) - f_k(u_h^\nu(t_i), x^\nu(t_i))\, d\tau\right| \\
&\leq \left(1 + \max_{k=1,\dots,n_x} \left\{L_{f,2}^k\right\} h\right) \left\|x^\nu(t_i) - x_h^\nu(t_i)\right\| + \max_{k=1,\dots,n_x} \left\{L_{f,1}^k\right\} hh_u \\
&+ \max_{k=1,\dots,n_x} \left\{L_{f,2}^k\right\} \max_{k=1,\dots,n_x} \left\{L_{x,1}^k\right\} \int_{t_i}^{t_{i+1}} \tau - t_i\, d\tau \\
&\leq \left(1 + \max_{k=1,\dots,n_x} \left\{L_{f,2}^k\right\} h\right) \left\|x^\nu(t_i) - x_h^\nu(t_i)\right\| + \max_{k=1,\dots,n_x} \left\{L_{f,1}^k\right\} hh_u
\end{aligned}
$$

$$
\begin{aligned}
&+ \frac{1}{2} \max_{k=1,\dots,n_x} \left\{ L^k_{f,2} \right\} \max_{k=1,\dots,n_x} \left\{ L^k_{x,1} \right\} h^2 \\
&= (1 + C_5 h) \left\| x^\nu(t_i) - x^\nu_h(t_i) \right\| + \max_{k=1,\dots,n_x} \left\{ L^k_{f,1} \right\} h h_u + C_6 h^2
\end{aligned}
$$

where we set

$$
\begin{aligned}
L_{f,1} &= \max_{k=1,\dots,n_x} \left\{ L^k_{f,1} \right\}, \\
L_{f,2} &= \max_{k=1,\dots,n_x} \left\{ L^k_{f,2} \right\}, \\
L &= \max_{k=1,\dots,n_x} \left\{ L^k_{x,1} \right\},
\end{aligned}
$$

and the constants,

$$
C_6 \coloneqq \frac{L}{2} L_{f,2} \qquad \text{and} \qquad C_5 \coloneqq L_{f,2}.
$$

Employing Lemma 3.21 together with the initial condition $x^\nu(t_0) = x^\nu_h(t_0)$ yields

$$
\| x^\nu(t_{i+1}) - x^\nu_h(t_{i+1}) \| \leq L_{f,1} h_u \frac{e^{C_5 T} - 1}{C_5} + C_6 h \frac{e^{C_5 T} - 1}{C_5}
$$

Given this estimate, we can define constants

$$
C_7 \coloneqq L_{f,1} \frac{e^{C_5 T} - 1}{C_5} \qquad \text{and} \qquad C_8 \coloneqq C_6 \frac{e^{C_5 T} - 1}{C_5}
$$

which are independent of h and h_u. Hence, the approximation error

$$
\| x^\nu(t_{i+1}) - x^\nu_h(t_{i+1}) \| \leq C_7 h_u + C_8 h
$$

holds at all time grid points $i = 0, \dots, M$. Then, by definition we obtain for arbitrary $i = 0, \dots, M-1$ and all $t \in [t_i, t_{i+1}]$,

$$
\begin{aligned}
\| x^\nu(t) - x^\nu_h(t) \| &= \| x^\nu(t) - x^\nu(t_i) + x^\nu(t_i) - x^\nu_h(t_i) - (t - t_i) f(u^\nu_h(t_i), x^\nu_h(t_i)) \| \\
&\leq \| x^\nu(t) - x^\nu(t_i) \| + \| x^\nu(t_i) - x^\nu_h(t_i) \| + h \| f(u^\nu_h(t_i), x^\nu_h(t_i)) \|.
\end{aligned}
$$

Using Lipschitz continuity of all x^ν for $\nu = 1, \dots, N$ with common constant $L > 0$ together with boundedness of $\| f(u^\nu_h(t_i), x^\nu_h(t_i)) \| \leq \tilde{C}$, by a constant $\tilde{C} > 0$ and the estimate $\| x^\nu(t_{i+1}) - x^\nu_h(t_{i+1}) \| \leq C_7 h_u + C_8 h$, we get

$$
\begin{aligned}
\| x^\nu(t) - x^\nu_h(t) \| &\leq L h + C_7 h + C_8 h_u + h \| f(u^\nu_h(t_i), x^\nu_h(t_i)) \| \\
&\leq (L + C_7 + \tilde{C}) h + C_8 h_u \\
&= C_9 h + C_8 h_u
\end{aligned}
$$

where the constant $C_9 := (L + C_7 + \tilde{C})h$ being independent of the step sizes h and h_u. Since all the estimates hold for arbitrary $\nu \in \{1, \dots, N\}$, this proves the approximation error

$$\|x - x_h\|_\infty \leq C_3(h + h_u),$$

with the constant $C_3 := \max\{C_8, C_9\} > 0$ independent of h, h_u.

3. Next, we show that $x_h^\nu(t)$ is feasible for NEP_h, for all $\nu = 1, \dots, N$ and all $t \in [t_i, t_{i+1}]$. Using Lipschitz property of the state variables we can estimate,

$$\begin{aligned} k^\nu(x_h^\nu(t_{i+1})) - k^\nu(x^\nu(t_{i+1})) &\leq \|k^\nu(x^\nu(t_{i+1})) - k^\nu(x_h^\nu(t_{i+1}))\| \\ &\leq L_{k,1} \left|x^\nu(t_{i+1}) - x_h^\nu(t_{i+1})\right| \\ &\leq L_{k,1} C_3 \, (h + h_u) \end{aligned}$$

From this it follows

$$\begin{aligned} k^\nu(x_h^\nu(t_{i+1})) &\leq \underbrace{k^\nu(x^\nu(t_{i+1}))}_{\leq 0} + L_{k,1} C_3 \, (h + h_u) \\ &\leq L_{k,1} C_3 \, (h + h_u) \\ &= C_{10} \, (h + h_u) \\ &\leq (h + h_u)^p \end{aligned}$$

Therefore, for $p \leq 1$ if h, h_u sufficiently small $x_h^\nu(t_{i+1})$ is feasible for NEP_h.

4. Given an arbitrary point $x_h(t)$ and a feasible point $x(t)$, where $t \in [t_i, t_{i+1}]$. Then we know,

$$\|\max\{0, g^\nu(x_h(t))\} - \max\{0, g^\nu(x(t))\}\| \leq \|g^\nu(x_h(t))\|$$

since $g^\nu(x(t)) \leq 0$. With this we have,

$$\begin{aligned} \max\{0, g^\nu(x_h(t))\} &\leq \max\{0, g^\nu(x_h(t)) - g^\nu(x(t))\} \\ &\leq \max\{0, |g^\nu(x(t)) - g^\nu(x_h(t))|\} \\ &= \|g^\nu(x(t)) - g^\nu(x_h(t))\| \\ &\leq L_g \|x_h(t) - x(t)\| \\ &\leq C_3 L_g (h + h_u) \end{aligned}$$

This together with (3.21) leads to

$$\frac{1}{M} \sum_{\nu=1}^{N} \sum_{i=0}^{M-1} \sum_{j=1}^{m_\nu} \max\{0, g_j^\nu(x_h(t_{i+1}))\} \leq C_3 L_g N \max_{\nu=1,\dots,N} \{m_\nu\} \, (h + h_u)$$

resulting in

$$P(x_h) \leq C_4 (h + h_u)$$

where the positive constant $C_4 := C_3 L_g N \max_{\nu=1,\dots,N} \{m_\nu\}$, thus completing the proof. □

With this, we have shown that under the given assumptions, every feasible point of (GNEP) can be approximated through a sequence of feasible points of (NEP_h) for sufficiently small h and h_u. Furthermore, we have shown that the penalty term is bounded from above by a constant term $C_4 (h + h_u)$, which solely depends on the discretization of the time and control variables. Thus, adequately for sufficiently small h and h_u vanishes, which is important to achieve a feasible solution of (GNEP). Before we can move on to the main result of this chapter, another technical result is required. This later on allows us to make use of *Filippov's Lemma* (see Lemma A.5).

Lemma 3.23. *Suppose Assumption 6 holds. The set-valued function*

$$\mathcal{F}(x(t)) := \begin{pmatrix} \tilde{f}_1(\mathcal{U}^1, x^1(t)) \\ \vdots \\ \tilde{f}_1(\mathcal{U}^N, x^N(t)) \\ \vdots \\ \tilde{f}_{n_x}(\mathcal{U}^1, x^1(t)) \\ \vdots \\ \tilde{f}_{n_x}(\mathcal{U}^N, x^N(t)) \end{pmatrix} \qquad \text{with} \qquad \tilde{f}(\mathcal{U}^\nu, x^\nu) := \left\{ f(u^\nu, x^\nu) \mid u^\nu \in \left[u^\nu_{\min}, u^\nu_{\max}\right] \right\}$$

is upper semicontinuous and there exists a constant $C > 0$ such that for all $t \notin \{t_0, \dots, t_M\}$

$$\begin{pmatrix} (x^1_{h,1})'(t) \\ \vdots \\ (x^N_{h,1})'(t) \\ \vdots \\ (x^1_{h,n_x})'(t) \\ \vdots \\ (x^N_{h,n_x})'(t) \end{pmatrix} \in \mathcal{F}(x_h(t)) + Ch\mathbb{B}^{n_x N}(0)$$

where the closed unit ball of dimension $n_x N$, centred at the origin is denoted by $\mathbb{B}^{n_x N}(0)$.

Proof. With $u^\nu \in \left[u^\nu_{\min}, u^\nu_{\max}\right]$ and f Lipschitz continuous by Assumption 6 then by [Aubin and Cellina, 1984, Proposition 1, p. 47] the set valued function $\mathcal{F}(\cdot)$ is continuous and therefore upper semicontinuous. By definition we have for $t \in]t_j, t_{j+1}[$,

$$\begin{pmatrix} (x_{h,1}^1)'(t) \\ \vdots \\ (x_{h,1}^N)'(t) \\ \vdots \\ (x_{h,n_x}^1)'(t) \\ \vdots \\ (x_{h,n_x}^N)'(t) \end{pmatrix} = \begin{pmatrix} f_1(u_h^1(t_j), x_h^1(t_j)) - f_1(u_h^1(t_j), x_h^1(t)) \\ \vdots \\ f_1(u_h^N(t_j), x_h^N(t_j)) - f_1(u_h^N(t_j), x_h^N(t)) \\ \vdots \\ f_{n_x}(u_h^1(t_j), x_h^1(t_j)) - f_{n_x}(u_h^1(t_j), x_h^1(t)) \\ \vdots \\ f_{n_x}(u_h^N(t_j), x_h^N(t_j)) - f_{n_x}(u_h^N(t_j), x_h^N(t)) \end{pmatrix} + \begin{pmatrix} f_1(u_h^1(t_j), x_h^1(t)) \\ \vdots \\ f_1(u_h^N(t_j), x_h^N(t)) \\ \vdots \\ f_{n_x}(u_h^1(t_j), x_h^1(t)) \\ \vdots \\ f_{n_x}(u_h^N(t_j), x_h^N(t)) \end{pmatrix}$$
$$\in \max_{\nu=1,\dots,N} L_{f,2} |x_h^\nu(t_j) - x_h^\nu(t)| \mathbb{B}^{n_x N}(0) + \mathcal{F}(x_h(t)).$$

Now we can use,

$$\begin{aligned} \max_{\nu=1,\dots,N} \|x_h^\nu(t_j) - x_h^\nu(t)\| &= \max_{\nu=1,\dots,N} \|(t - t_j) f(u_h^\nu(t_j), x_h^\nu(t_j))\| \\ &\leq h \max_{\left(u_h^\nu, x_h^\nu\right) \in X_{\nu,j}^h} \|f(u_h^\nu(t_j), x_h^\nu(t_j))\| \\ &\leq h\tilde{C} \end{aligned}$$

and with the previous result we get the constant

$$C := L_{f,2}\tilde{C},$$

and therefore have shown that,

$$\begin{pmatrix} (x_{h,1}^1)'(t) \\ \vdots \\ (x_{h,1}^N)'(t) \\ \vdots \\ (x_{h,n_x}^1)'(t) \\ \vdots \\ (x_{h,n_x}^N)'(t) \end{pmatrix} \in \mathcal{F}(x_h(t)) + Ch\mathbb{B}^{n_x N}(0).$$

□

Remark 3.24. *This is also true if the number of states n_x is not the same for all players $\nu = 1, \dots, N$. Let n_x^ν be the state dimension of the ν-th player. Then, with $n_x^\Sigma := \sum_{\nu=1}^N n_x^\nu$ the unit ball becomes* $\mathbb{B}^{n_x^\Sigma}(0)$.

It remains to show that an accumulation point of a sequence of solutions of (NEP$_h$) is a generalized Nash equilibrium of (GNEP).

Theorem 3.25. *Assume that (GNEP) has a feasible point and Assumption 4 holds. Then (NEP_h) has a solution $(\hat{u}_h, \hat{x}_h)$ and there exists a constant $C_{11} > 0$ such that*

$$P(\hat{x}_h) \leq C_{11}\,(h + h_u)^p$$

for all sufficiently small h and h_u. Further, for $\|(h, h_u)\| \longrightarrow 0$ the sequence $\{\hat{x}_h\}$ has an accumulation point $\hat{x}$ with respect to uniform convergence, and there exists a control $\hat{u}$, which together with $\hat{x}$ is a generalized Nash equilibrium of (GNEP).

Proof. By assumption (GNEP) has a feasible point and as a consequence of Lemma 3.22, (NEP_h) has at least one feasible point $(\tilde{u}_h, \tilde{x}_h)$ for sufficiently small h and h_u.
Due to the composition of (NEP_h), it is evident that we have a potential game, with an exact potential function

$$\Phi(x_h) := \sum_{\nu=1}^{N} \varphi^\nu\left(x_h^\nu(t_f)\right) + \frac{1}{(h + h_u)^p} P(x_h).$$

Emanating from the conception of the discretization, there is only a finite number of possible controls u_h. Moreover, the states x_h are uniquely defined by u_h. Then, this implies that also the number of feasible points is finite, which satisfy $\Phi(x_h) \leq \Phi(\tilde{x}_h)$. One of them, which corresponds to $(\hat{u}_h, \hat{x}_h)$, leads to the minimum of the potential function and hence, constitutes a Nash equilibrium of (NEP_h). Therefore, with

$$\begin{aligned}
\Phi(\hat{x}_h) &= \sum_{\nu=1}^{N} \varphi^\nu\left(\hat{x}_h^\nu(t_f)\right) + \frac{1}{(h + h_u)^p} P(\hat{x}_h) \\
&\leq \sum_{\nu=1}^{N} \varphi^\nu\left(\tilde{x}_h^\nu(t_f)\right) + \frac{1}{(h + h_u)^p} P(\tilde{x}_h) \\
&= \Phi(\tilde{x}_h)
\end{aligned}$$

and Lemma 3.22 we acquire,

$$\begin{aligned}
P(\hat{x}_h) &\leq (h + h_u)^p \sum_{\nu=1}^{N} \left(\varphi^\nu(\tilde{x}_h^\nu(t_f)) - \varphi^\nu(\hat{x}_h^\nu(t_f))\right) + P(\tilde{x}_h) \\
&\leq (h + h_u)^p \sum_{\nu=1}^{N} \left(\varphi^\nu(\tilde{x}_h^\nu(t_f)) - \varphi^\nu(\hat{x}_h^\nu(t_f))\right) + C_4(h + h_u).
\end{aligned}$$

By Assumption 5 we know that $\sum_{\nu=1}^{N}(\varphi^\nu(\tilde{x}_h^\nu(t_f)) - \varphi^\nu(\hat{x}_h^\nu(t_f)))$ is bounded. This in conjunction with $p \leq 1$ implies the existence of a constant $C_{11} > 0$ such that

$$P(\hat{x}_h) \leq C_{11}\,(h + h_u)^p \tag{3.25}$$

holds. Considering $(\hat{u}_h^\nu, \hat{x}_h^\nu)$ being a feasible point for (NEP$_h$), then for all $\nu = 1, \dots, N$ and all $t \notin \{t_0, \dots, t_M\}$

$$\begin{aligned}\|(\hat{x}_h^\nu)'(t)\| &= \|f(\hat{u}_h^\nu(t), \hat{x}_h^\nu(t))\| \\ &\leq \max_{(u_h^\nu, x_h^\nu) \in X_\nu^h} \|f(u_h^\nu(t_j), x_h^\nu(t_j))\| \leq \tilde{C}.\end{aligned}$$

With the estimate $k^\nu(\hat{x}_h^\nu(t_i)) \leq (h + h_u)^p$ together with (3.16) it follows that $\hat{x}_h^\nu(t_i)$ is bounded. Next we have,

$$k^\nu(\hat{x}^\nu(t_i)) = \lim_{\substack{h \to 0 \\ h_u \to 0}} k^\nu(\hat{x}_h^\nu(t_i)) \leq \lim_{\substack{h \to 0 \\ h_u \to 0}} (h + h_u)^p = 0.$$

Hence, together with $\hat{x}$ Lipschitz continuous we have $k^\nu(\hat{x}(t)) \leq 0$ and therefore $(\hat{u}_h^\nu(t), \hat{x}_h^\nu(t))$ being bounded for every $\nu = 1, \dots, N$ and $t \in [t_0, t_f]$. Then, boundedness together with Lemma 3.23 allows us to utilize Theorem A.6. Thus, we have a subsequence $\{x_h^\nu\}$ converging uniformly to some $\hat{x}$ for $\|(h, h_u)\| \to 0$. Comprising, for all player $\nu = 1, \dots, N$ we have the constraints $k^\nu(\hat{x}_h^\nu(t_i)) \leq (h + h_u)^p$, for all $t \in [t_0, t_f]$. Consequently, by the above estimate this additionally must hold for the accumulation point $\hat{x}^\nu$ for all $\nu = 1, \dots, N$.

Given the previous results, we can apply Filippov's Lemma, see for instance Lemma A.5. Herein, in conjunction with Assumption 4, we have the compact set

$$\Omega := \{(u, x) \mid u^\nu \in \left[u_{\min}^\nu, u_{\max}^\nu\right], k^\nu(x^\nu) \leq 0, \forall \nu = 1, \dots, N\},$$

the continuous function

$$\phi(t, x) := \phi(t, u, x) := \begin{pmatrix} f(x_1^1(t), u_1^1(t)) \\ \vdots \\ f(x_1^N(t), u_1^N(t)) \\ \vdots \\ f(x_{n_x}^1(t), u_{n_x}^1(t)) \\ \vdots \\ f(x_{n_x}^N(t), u_{n_x}^N(t)) \end{pmatrix},$$

and the bounded and measurable function

$$\Psi(t) := \begin{pmatrix} (x_1^1)'(t) \\ \vdots \\ (x_1^N)'(t) \\ \vdots \\ (x_{n_x}^1)'(t) \\ \vdots \\ (x_{n_x}^N)'(t) \end{pmatrix} \quad \text{with} \quad \|\Psi\|_\infty \leq \max\{x_{\max}^\nu, \tilde{C}\}.$$

Then, we acquire a control $\hat{u} \in \bigtimes_{\nu=1}^{N} \left[u_{\min}^{\nu}, u_{\max}^{\nu}\right]$ such that $(\hat{u}, \hat{x})$ satisfies the differential equations in the feasible set of (GNEP). Thus, it follows that $(\hat{u}, \hat{x}) \in \bigtimes_{\nu=1}^{N} X_{\nu}$.

Now using $\|(h, h_u)\| \to 0$ together with (3.25) and continuity of all functions involved implies for all $t \in [t_0, t_f]$ that

$$0 = \lim_{\|(h,h_u)\| \to 0} P(\hat{x}_h) = P(\hat{x}) \tag{3.26}$$

thus, $\hat{x} \in G(\hat{x})$. Therefore, $(\hat{u}, \hat{x})$ is feasible for (GNEP). It remains to prove that this is also a generalized Nash equilibrium. To this end, let an arbitrary $\nu \in \{1, \dots, N\}$ and an arbitrary feasible point of (GNEP) of the form $((u^{\nu}, \hat{u}^{-\nu}), (x^{\nu}, \hat{x}^{-\nu}))$ be given. By Lemma 3.22, we can find for $((u^{\nu}, \hat{u}^{-\nu}), (x^{\nu}, \hat{x}^{-\nu}))$ an approximating sequence (u_h, x_h) that is feasible for (NEP_h) and satisfies an error bound of order $(h + h_u)$. By chain rule P is Lipschitz continuous with constant $L_P > 0$. This, and the fact that $(\hat{u}_h, \hat{x}_h)$ is a solution of (NEP_h), leads to

$$\begin{aligned}
\varphi(\hat{x}_h^{\nu}(t_f)) + \frac{P(\hat{x}_h)}{(h+h_u)^p} &\leq \varphi(x_h^{\nu}(t_f)) + \frac{P(x_h^{\nu}, \hat{x}_h^{-\nu})}{(h+h_u)^p} \\
&= \varphi(x_h^{\nu}(t_f)) + \frac{1}{(h+h_u)^p}\left(P(x_h^{\nu}, \hat{x}_h^{-\nu}) - P(\hat{x}_h)\right) + \frac{P(\hat{x}_h)}{(h+h_u)^p} \\
&\leq \varphi(x_h^{\nu}(t_f)) + \frac{L_P}{(h+h_u)^p}\left\|x_h^{\nu}(t) - \hat{x}_h^{\nu}(t)\right\|_{\infty} + \frac{P(\hat{x}_h)}{(h+h_u)^p} \\
&\leq \varphi(x_h^{\nu}(t_f)) + \frac{L_P C_3 (h+h_u)}{(h+h_u)^p} + \frac{P(\hat{x}_h)}{(h+h_u)^p}
\end{aligned}$$

Subtracting $\frac{P(\hat{x}_h)}{(h+h_u)^p}$ on both sides, taking the limit $\|(h, h_u)\| \to 0$ and using $p < 1$, we obtain

$$\varphi(\hat{x}^{\nu}(t_f)) \leq \varphi(x^{\nu}(t_f)).$$

for all feasible points $((u^{\nu}, \hat{u}^{-\nu}), (x^{\nu}, \hat{x}^{-\nu}))$ of (GNEP). Since $\nu \in \{1, \dots, N\}$ was arbitrary, this holds for all $\nu = 1, \dots, N$, and we have shown that $(\hat{u}, \hat{x})$ is a generalized Nash equilibrium of (GNEP). □

3.2.5 A Decomposition Algorithm

In this section, we eventually state a decomposition algorithm tailored for the penalty reformulation of (NEP_h), where the shared and nonconvex constraints are shifted to the cost function by means of penalty terms. We further set $\varkappa_h^{\nu} := (u_h^{\nu}, x_h^{\nu})$, and write $f_{\nu}(\varkappa_h^{\nu}) := \varphi(x_h^{\nu}(t_f))$. To abbreviate the strategies of all players except player ν, we accordingly use the notation $\varkappa_h^{-\nu}$. Then, in our standard Nash equilibrium problem (NEP_h), each player $\nu = 1, \dots, N$ solves

$$\min f_{\nu}(\varkappa_h^{\nu}) + \rho P(x_h^{\nu}, x_h^{-\nu}) \quad \text{s.t.} \quad \varkappa_h^{\nu} \in X_{\nu}^{h}.$$

Since P^ν contains all terms of P that depend on x_h^ν the problem has the same Nash equilibria as,

$$\min f_\nu(\varkappa_h^\nu) + \rho P^\nu(x_h^\nu, x_h^{-\nu}) \quad \text{s.t.} \quad \varkappa_h^\nu \in X_\nu^h$$

for all $\nu = 1, \dots, N$. Since this is a generalized potential game, compare (GPG), a Gauss-Seidel type algorithm, as suggested in [Facchinei et al., 2011], which is able to treat nonconvex GNEPs by exploiting a regularization term could be applied. However, we will develop a variation of the standard Gauss-Seidel decomposition method, where we do not have a prescribed order of players, and we will see, that due to the finite number of feasible strategies, and since we solve penalized standard Nash games, our algorithm works without any regularization. Furthermore, through the penalization reformulation, possible intricacies emanating from an empty feasible set that may occur if the nonconvex constraints remain in the strategy set, can be avoided. Hence, guaranteeing that the subproblems of the individual player $\nu = 1, \dots, N$ are always solvable. Let us first state the algorithm.

Algorithm 5: Decomposition Algorithm with Penalty Selection [DAPS]

(S.0) Choose a starting point $\varkappa_0 = (\varkappa_0^1, \dots, \varkappa_0^N) \in \bigtimes_{\nu=1}^N X_\nu^h$ and set $k := 0$,

$$(c(1), \dots, c(N)) = 0,$$
$$(q(1), \dots, q(N)) = 0.$$

(S.1) If $(q(1), \dots, q(N)) = \mathbb{1}$: STOP.

(S.2) Compute

$$A_k := \{\nu \in \{1, \dots, N\} \mid q(\nu) \neq 1, P^\nu(x_{h,k}) \geq P^\mu(x_{h,k}) \; \forall \mu = 1, \dots, N\}.$$

Select a player $\nu_k \in \operatorname{argmax}_{\nu \in A_k} c(\nu)$.

(S.3) For $\nu = \nu_k$ compute a global solution $\varkappa_{h,k+1}^\nu$ of

$$\min f_\nu(\varkappa_h^\nu) + \rho P^\nu(x_h^\nu, x_{h,k}^{-\nu}) \quad \text{s.t.} \quad \varkappa_h^\nu \in X_\nu. \tag{3.27}$$

(S.4) Set

$$\varkappa_{h,k+1} = \begin{cases} \varkappa_{h,k}, & \text{if } f_\nu(\varkappa_k^\nu) + \rho P^\nu(x_{h,k}^\nu, x_{h,k}^{-\nu}) = f_\nu(\varkappa_{k+1}^\nu) + \rho P^\nu(x_{h,k+1}^\nu, x_{h,k}^{-\nu}), \\ (\varkappa_{h,k+1}^\nu, \varkappa_{h,k}^{-\nu}), & \text{else.} \end{cases}$$

(S.5) If $\varkappa_{k+1} \neq \varkappa_k$, set $(q(1), \dots, q(N)) = (0, \dots, 0)$. Set $q(\nu_k) = 1$,

$$c(\nu) = \begin{cases} c(\nu) + 1, & \text{for all } \nu \neq \nu_k, \\ 0, & \text{for } \nu = \nu_k, \end{cases}$$

$k \leftarrow k + 1$, and go to (S.1).

In the first step (S.0) an initial yet feasible state x_0 is chosen and the index lists $c, q \in \mathbb{R}^N$ are initialized to zero. Here, $c(\cdot)$ constitutes a counter, tracking the turns a player was not allowed to optimize. The indicator set $q(\cdot)$ charts which players' turn it was in the previous iteration. In (S.2) a player is selected for whom the optimization then is conducted. Let us stress that we do not have an a priori defined order, but the order is iteratively adopted according to each players contribution to the penalty term of the generalized potential game. In this sense, the potential function is a measure or indicator of the overall state of the players' ecosystem. Therefore, a favourable state of the ecosystem is associated with a low penalty value. Hence, the player with the highest penalty contribution is chosen to optimize, in order to rectify his penalty contribution $P^\nu(x_{h,k})$. Once a player optimized his strategy the corresponding index in the queue list is set to one, i.e., $q(\nu) = 1$. Thus, this ensures that the player who had the last turn cannot immediately be chosen to optimize yet again. Intuitively, this would otherwise lead to the same outcome, since no other player had the chance to deviate from their strategy. If there is a tie, the player that has not been chosen for the longest time, i.e., the one with the highest $c(\nu)$, is selected. If there is still a tie, an arbitrary player is selected. The main effort of the algorithm is the solution of the optimization problem in (S.3). This will be achieved through a dynamic programming approach and is content of the next chapter. Here, we only mention, if no improvement compared to $\varkappa^\nu_{h,k}$ is made, we remain with the old iterate $\varkappa_{h,k+1} = \varkappa_{h,k}$ in (S.4) otherwise the new strategy $\varkappa_{h,k+1}$ is adopted.

If there are $N - 1$ iterations in a row, where no improvement is made, all variables $q(\nu)$ are set to 1, and the algorithm stops in (S.1), because no player can improve the situation. Let us stress, that for two player games the players are chosen alternately, and we have the standard Gauss-Seidel algorithm. For an arbitrary number of players the Gauss-Seidel algorithm is a special case of Algorithm 5. Hence, it is covered by the following convergence theory.

Through the penalty approach we employ here, we have, in contrast to the general Gauss-Seidel method considered in [Facchinei et al., 2011], a fixed feasible set X^h_ν for every player $\nu = 1, \dots, N$. For the regularized variant of the algorithm with a cyclic choice of the players, the convergence theory of [Facchinei et al., 2011] states that cluster points are generalized Nash equilibria. The convergence of the decomposition method with penalty selection is given by the following theorem.

Theorem 3.26. *Let X^h_ν be nonempty and contain only a finite number of strategies for all players $\nu = 1, \dots, N$. Then, Algorithm 5 terminates with a Nash equilibrium of (NEP$_h$) after a finite number of iterations.*

Proof. Since for all $\nu = 1, \dots, N$ the feasible set X^h_ν is nonempty and contains only a finite number of strategies, (3.27) has always a solution $\varkappa^\nu_{h,k+1}$, which satisfies

$$f_\nu(\varkappa^\nu_{h,k+1}) + \rho P^\nu(x^\nu_{h,k+1}, x^{-\nu}_{h,k}) \leq f_\nu(\varkappa^\nu_{h,k}) + \rho P^\nu(x^\nu_{h,k}, x^{-\nu}_{h,k})$$

Since P^ν contains all terms of P that depend on x^ν, this is equivalent to,

$$f_\nu(\varkappa^\nu_{h,k+1}) + \rho P(x^\nu_{h,k+1}, x^{-\nu}_{h,k}) \leq f_\nu(\varkappa^\nu_{h,k}) + \rho P(x^\nu_{h,k}, x^{-\nu}_{h,k})$$

By definition of the potential function

$$\Lambda(\varkappa_h) := \sum_{\nu=1}^{N} f_\nu(\varkappa^\nu_h) + \rho P(x_h),$$

we consequently get

$$\begin{aligned} 0 \geq & f_\nu(\varkappa^\nu_{k+1}) + \rho P(x_{h,k+1}) - f_\nu(\varkappa^\nu_k) - \rho P(x_{h,k}) \\ = & \Lambda(\varkappa_{h,k+1}) - \Lambda(\varkappa_{h,k}). \end{aligned}$$

Therefore, the sequence $\{\Lambda(\varkappa_{h,k})\}$ is monotonically non-increasing. Next, we assume that $\varkappa_{h,k+1} \neq \varkappa_{h,k}$. Then, by (S.4), there must be a strict decrease, i.e.,

$$\Lambda(\varkappa_{h,k+1}) < \Lambda(\varkappa_{h,k}).$$

Since the total number of strategies is finite and $\{\Lambda(\varkappa_{h,k})\}$ is monotonically non-increasing, this is only possible for a finite number of iterations. Thus, we finally must have $\varkappa_{h,k+1} = \varkappa_{h,k}$ for $N-1$ iterations in a row, and the algorithm terminates after a finite number of K iterations with a feasible point $\varkappa_{h,K} \in \bigtimes_{\nu=1}^{N} X^h_\nu$. By (S.3), this point satisfies

$$f_\nu(\varkappa^\nu_{h,K}) + \rho P(x^\nu_{h,K}, x^{-\nu}_{h,K}) \leq f_\nu(\varkappa^\nu_h) + \rho P(x^\nu_h, x^{-\nu}_{h,K})$$

for all $\varkappa^\nu_h \in X^h_\nu$ and for all $\nu = 1, \ldots, N$. Hence, x_K is a Nash equilibrium of (NEP$_h$). □

Remark 3.27. *Let us summarize the results of this chapter. First, a penalty reformulation of the nonconvex differential GNEP was provided, which led to a standard Nash game. In Lemma 3.22 we have shown that the discrete states of (NEP$_h$) for sufficiently small h and h_u converge to the states of the continuous (GNEP) and that the penalty is bounded from above. The existence of a solution of (NEP$_h$) and that an accumulation point of a sequence of solutions of these Nash equilibrium problems is a solution of (GNEP) was shown. For the solution of (GNEP) with the respective penalty reformulation a suitable decomposition method with penalty selection was introduced. Theorem 3.26 states, that the proposed algorithm terminates within a finite number of iterations, where in each iteration a semi-discrete optimal control problem needs to be solved globally. This peculiar problem is discussed in the context of dynamic programming in the subsequent chapter.*

4 | Dynamic Programming

> *Rationality of thought imposes a limit on a person's concept of his relation to the cosmos.*
>
> — *John F. Nash*

After an algorithmic framework and convergence results, by reformulating the nonconvex differential (GNEP) using a penalty approach in combination with a tailored decomposition method were established in the previous chapter, it remains to provide a solution concept which globally solves the semi-discrete (NEP_h) in every iteration of Algorithm 5.

Therefore, the purpose of this chapter is to establish a suitable method for the computation of a global solution of the subproblem in the previous chapter. Due to its advantageous global solution properties, dynamic programming, which is based on *Richard E. Bellmann's* optimality principle, see [Bellman, 1957], offers to be a suitable methodology for solving the subproblems. To adhere to the penalty approach, a penalized value function has to be derived. For this purpose, the penalty function is treated as a Lagrangian term, which allows to embed the difficult state constraints into the value function. Another approach to handle a constrained optimal control problem by means of an auxiliary value function for a continuous problem is shown in [Altarovici et al., 2013, Assellaou, 2015], where as compared to the partial penalization approach within this thesis, the epigraph of the value function is characterized by a value function of an auxiliary unconstrained problem. First, in Sections 4.2 to 4.5 the fundamental concepts of dynamic programming and the theory of viscosity weak solutions is revisited, which is discussed in further detail in [Bardi and Capuzzo-Dolcetta, 2008, Alt, 2012, Crandall and Lions, 1983] and [Bertsekas, 2005]. Moreover, a backward Semi-Lagrangian scheme is introduced, which is based on the forward Semi-Lagrangian scheme in [Falcone and Ferretti, 1998], allowing us to establish a link between the HJB equation, the dynamic programming principle and the discrete value function. Then, the penalized value function of (GNEP) is derived in Section 4.6 along with error estimates of the approximated value function to the continuous value function. Additionally, it is shown that the approximate dynamic programming algorithm, see for instance [Bertsekas, 2005], converges in finite time to a global minimizer of the subproblems. Eventually, in Section 4.7 we introduce a GPU parallel dynamic programming algorithm which significantly improves the performance for large dimensions, and therefore relaxes the course of dimensions to some extent.

4.1 Preliminaries

Some additional notation is required to properly introduce and discuss the theory and concepts of dynamic programming. To this end, given a function $f : \mathbb{U} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ which governs the evolution of the considered system through the nonlinear ordinary differential equation

$$x'(t) = f(u(t), x(t)). \tag{4.1}$$

for all $t \in [t_0, t_f]$, where $t_0 < t_f$ indicates the final time and t_0 the initial time, respectively. Furthermore, let $x(t) \in \mathbb{R}^{n_x}$ with $\mathbb{R}^{n_x}$ the set of feasible states and $u(t) \in \mathbb{U}$, with $\mathbb{U} \subset \mathbb{R}^{n_u}$ the nonempty and compact set of admissible control values.

In compliance with the settings of the previous chapters, let the control $u : [t_0, t_f] \to \mathbb{R}^{n_u}$ be a measurable function. Likewise, let the state $x : [t_0, t_f] \to \mathbb{R}^{n_x}$ be an absolutely continuous function in $W_{1,1}^{n_x}\left([t_0, t_f]\right)$. Herein, the state and control dimensions are denoted by n_x, $n_u \geq 1$ respectively. In addition, to ensure regularity we assume that Assumption 5 and Assumption 6 hold. In virtue of the Lipschitz continuity of $f(u, \cdot)$, by the above assumption, in conjunction with compactness of $\mathbb{U}$ together with some initial value condition $x(t_0) = z$, (4.1) admits a unique solution. Forthwith let $x_{z,t_0}^u(t)$ indicate this solution and satisfy

$$x_{z,t_0}^u(t) = z + \int_{t_0}^{t} f(u(\tau), x(\tau))\, d\tau, \quad \forall\, t \in [t_0, t_f]. \tag{4.2}$$

Given a performance measure, the task within optimal control then is to derive a control law which minimizes the cost. To this end, let the measure of taking an action u until time $t := t_f$ be defined by the cost function

$$J(u, z, t_0) := e^{-\alpha t_f} \varphi(x_{z,t_0}^u(t_f)) + \int_{t_0}^{t_f} e^{-\alpha\tau} f_0(u(\tau), x_{z,t_0}^u(\tau))\, d\tau. \tag{4.3}$$

We refer to $f_0 : \mathbb{U} \times \mathbb{R}^{n_x} \to \mathbb{R}$ as the running cost function (Lagrange Term). The function $\varphi(x_{z,t_0}^u(t_f))$ designates the terminal cost (Mayer Term). With this in mind, we forthwith take into account the following additional assumption:

Assumption. 8 (Lipschitz Continuity Lagrange). The function $f_0 : \mathbb{U} \times \mathbb{R}^{n_x} \to \mathbb{R}$ is Lipschitz continuous for fixed $u \in \mathbb{U}$ with a constant $L_{f_0} \geq 0$.

Remark 4.1. *The Bolza problem defined in (4.3), is a more general formulation of the objective defined in (3.1), where the exponential terms $e^{-\alpha t}$ reflect shadow pricing. Thus, for $\alpha = 0$, (3.1) and (4.3) are equivalent. These discount terms commonly appear within economical applications.*

Supplementary, in order to consider unconstrained optimal control problems let $\mathcal{D} \subset \mathbb{R}^{n_x}$ be nonempty and closed. Then the set of feasible trajectories of the initial value problem (4.1), with

initial state z on some interval $[t, t_f]$ is defined by,

$$\mathcal{X}_{\mathcal{D}}(z;[t,t_f]) := \left\{ x := x^u_{z,t} \mid x \in \mathbb{R}^{n_x},\ x^u_{z,t}(\tau) \in \mathcal{D},\ \forall \tau \in [t,t_f] \right\}. \tag{4.4}$$

It is evident that in the case where $\mathcal{D} \neq \mathbb{R}^{n_x}$ under Assumption 6, $\mathcal{X}_{\mathcal{D}}(z;[t,t_f])$ is not closed in $W^{n_x}_{1,1}([t,t_f])$, see for instance [Altarovici et al., 2013].

Remark 4.2. *Without loss of generality and to unburden the notation we further set the lower time bound $t_0 = 0$. Furthermore, let the initial state as of now be $z = x_0$.*

It remains to characterize under which conditions an optimal control of the above system (4.2) to (4.3) exists. Therefore, we first discuss the unconstrained minimization problem in terms of a finite horizon. Then, an approach for the constrained case is presented in Section 4.4.

4.2 Viscosity Solutions

A prominent approach in solving optimal control problems is based on dynamic programming. The fundamental idea of this method entails the initial introduction of a value function ϑ which reflects the intrinsic value of the problem relative to an initial value and a given control. Therefore, ϑ is required to satisfy a functional equation known as the dynamic programming principle (DPP) (see [Bellman, 2003]).

Moreover, if ϑ is suitably smooth, it characterizes a solution of the infinitesimal version of the DPP, i.e., the Hamilton-Jacobi-Bellman (HJB) equation, which in the case of deterministic control problems constitutes a first order partial differential equation of type,

$$F(x, \vartheta(x), \nabla\vartheta(x)) = 0, \tag{4.5}$$

with $x \in \mathcal{D} \subset \mathbb{R}^{n_x}$ and $\mathcal{D}$ an open set. The HJB equation and respectively the DPP can be perceived as surrogates of the OCP. Therefore provide a convenient and compact representation to derive an optimal feedback law through the synthesis procedure. The relations and the essential concept associated with dynamic programming are illustrated in Figure 4.1.

Aforementioned smoothness of the value function, however, is not guaranteed. Especially, if nonlinear optimal control problems are considered, a classical solution of HJB cannot be expected, since everywhere differentiability of the value function usually is not satisfied, e.g., due to jumps in the gradient. Generalized solutions, namely, locally Lipschitz continuous functions which satisfy the HJB equation almost everywhere, on the other hand omit infinitely many solutions and therefore are not unique [Fleming and Soner, 2006]. Moreover, if the OCP additionally is subject to state constraints, the value function might be discontinuous.

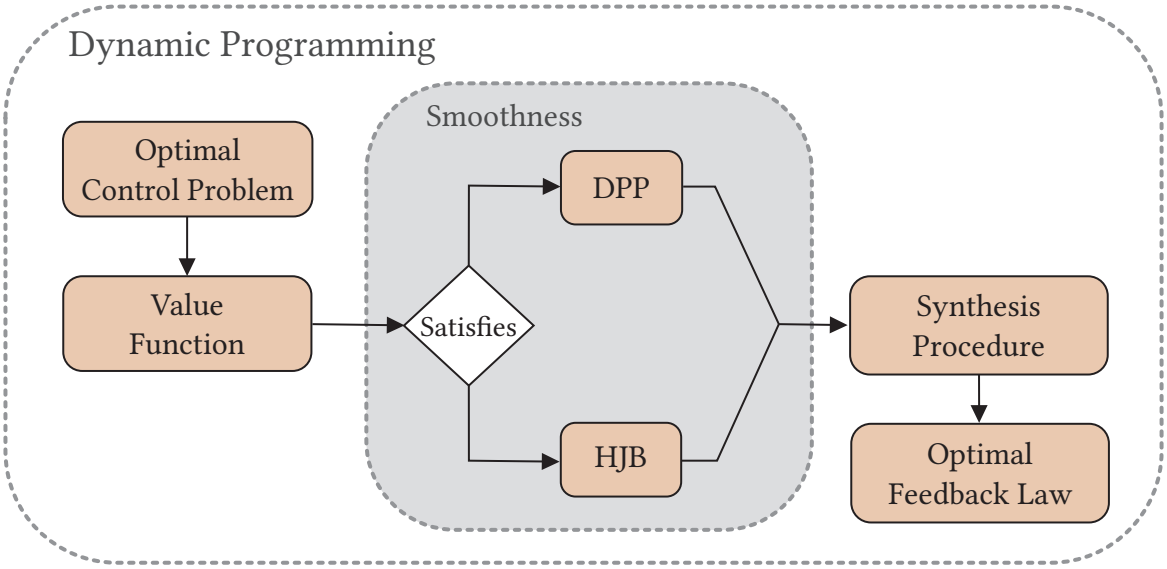


Figure 4.1: *Illustration of the general concept of the dynamic programming procedure and its proximity to the principle of dynamic programming and the Hamilton-Jacobi-Bellman equation, with respect to the smoothness of the value function.*

A weaker notion of solution is required. In [Crandall and Lions, 1983], [Crandall et al., 1984] and [Lions, 1982] the framework of viscosity solutions was introduced, which allows to characterize the value function, emanating from a nonlinear optimal control problem as a solution of (4.5):

Definition 4.3. *A function ϖ is a viscosity solution of (4.5) if and only if for any function $\psi \in C^1(\mathcal{D})$ the following conditions hold simultaneously:*

(a) ϖ is a viscosity supersolution if at any local minimum $\hat{x} \in \mathcal{D}$ for $\varpi - \psi$,

$$F(\hat{x}, \varpi(\hat{x}), \nabla\psi(\hat{x})) \geq 0$$

(b) ϖ is a viscosity subsolution if at any local maximum $\hat{x} \in \mathcal{D}$ for $\varpi - \psi$,

$$F(\hat{x}, \varpi(\hat{x}), \nabla\psi(\hat{x})) \leq 0.$$

There exist a vast amount of different definitions of a viscosity solution, see for instance [Bardi and Capuzzo-Dolcetta, 2008] for an extensive discussion on this topic. Let us elaborate on what Definition 4.3 suggests. The validity of the subsolution condition (respectively supersolution) is verified by introducing a smooth test function ψ and show that this function is touching the graph of ϖ from above (below) at any point $\hat{x}$. An illustration of this concept is presented in Figure 4.2.

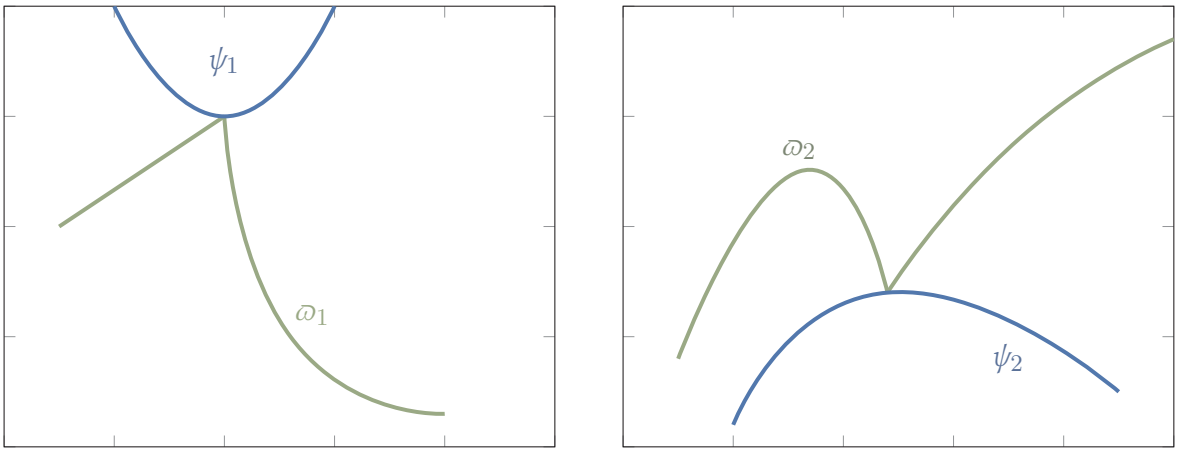


Figure 4.2: *Illustration of test functions used to validate viscosity in the context of Definition 4.3: (left) test function ψ_1, with respect to a viscosity subsolution touching the value function from above; (right) test function ψ_2, with respect to a viscosity supersolution touching the value function from below.*

Uniqueness of the solution then follows by means of the comparison principle [Bardi and Capuzzo-Dolcetta, 2008, Chapter I, p. 7]. In short, if ϖ_1 is a viscosity subsolution and ϖ_2 is a viscosity supersolution of the associated HJB equation, then it must hold that $\varpi_1 \leq \varpi_2$.

Remark 4.4. *Let us comment on this:*

- *It is well known, that the definition of sub- and supersolution, due to the relation to a continuous function, can be extended to upper and lower semicontinuouity as a requirement for ϖ to be a solution of (4.5) in the viscosity sense.*
- *In fact, in the constraint case, where ϑ is possibly discontinuous, discontinuous viscosity solutions can be defined, as suggested by [Ishii and Koike, 1996], in terms of upper and lower semicontinuous envelopes.*

4.3 Unconstrained Problem

The purpose of this section is to provide a reformulation of an unconstrained optimal control problem in terms of a value function, to derive an optimal control. Additionally, in the continuous case the value function then can be characterized by the HJB equation. Some classical results and a general discussion of the topic can be found in [Bardi and Capuzzo-Dolcetta, 2008], [Kirk, 2004] and [Lions, 1982].

Let $t_f < +\infty$. Consider the Bolza cost functional in (4.3) together with the first order differential equation (4.1). Hence, the problem without pure state constraints aims to find a suitable control $u(t)$ such that:

$$\begin{aligned}
\min_{u} \quad & J(u,z,t) = e^{\alpha(t-t_f)}\varphi(x^u_{z,t}(t_f)) + \int_t^{t_f} e^{\alpha(t-\tau)} f_0(u(\tau), x^u_{z,t}(\tau))\, d\tau \\
\text{s.t.} \quad & x'(\tau) = f(u(\tau), x(\tau)) && \tau \in [t, t_f] \\
& u(t) \in \mathbb{U} && \tau \in [t, t_f] \\
& x(t) = z
\end{aligned} \tag{uOCP}$$

Faced with a dynamical system, the value function serves as a substitute of the optimal value of the respective cost functional, defined over an interval $[t, t_f]$, initiated at time t and at state z. The corresponding value function of (uOCP) then is,

$$\vartheta(t,z) \coloneqq \inf_{u \in \mathcal{U}} J(u,z,t). \tag{4.6}$$

Suppose, the tuple $(\hat{x}, \hat{u})$ is optimal, then it follows from the above definition that $\vartheta(t,z) = J(\hat{u},z,t)$ at time t. For the finite horizon problem the following regularity result, indicating Lipschitz continuity of the value function ϑ holds.

Proposition 4.5. *[Bardi and Capuzzo-Dolcetta, 2008, Chapter III, Proposition 3.1] Assume Assumptions 5 to 6 and Assumption 8 hold. Then there exists a constant $L_\vartheta > 0$, depending on L_φ, t_f and L_{f_0}, for all $t, s \in [t_0, t_f]$ and $x, y \in \mathbb{R}^{n_x}$ implying that ϑ is Lipschitz continuous, such that*

$$|\vartheta(t,x) - \vartheta(s,y)| \leq L_\vartheta \left(|t-s| + |x-y|\right)$$

holds.

Bellman's principle of optimality, primordial introduced in [Bellman, 2003] in terms of a discrete setting, suggests that given an optimal control $\hat{u} : [t_0, t_f] \to \mathbb{R}^{n_u}$, at some time $t_0 \leq t \leq t_f$ produces a state $x(t)$. Then, minimizing the remaining subproblem starting at time instance t and using $x(t)$ as a new initial state, the truncated policy $\tilde{u} : [t, t_f] \to \mathbb{R}^{n_u}$ must be optimal (see [Bertsekas, 2005] for instance). Otherwise, there would be incentive to deviate towards a different policy. Let us roughly sketch the idea in continuous time:

$$\begin{aligned}
\vartheta(t,z) &= \inf_{u \in \mathcal{U}} \left\{ \int_t^{t_f} e^{\alpha(t-\tau)} f_0(u(\tau), x^u_{z,t}(\tau))\, d\tau + e^{\alpha(t-t_f)}\varphi(x^u_{z,t}(t_f)) \right\} \\
&= \inf_{u \in \mathcal{U}} \left\{ \int_t^{s} e^{\alpha(t-\tau)} f_0(u(\tau), x^u_{z,t}(\tau))\, d\tau + \int_s^{t_f} e^{\alpha(t-\tau)} f_0(u(\tau), x^u_{z,t}(\tau))\, d\tau \right. \\
&\qquad \left. + e^{\alpha(t-s)} e^{\alpha(s-t_f)} \varphi(x^u_{z,t}(t_f)) \right\} \\
&= \inf_{u \in \mathcal{U}} \left\{ \int_t^{s} e^{\alpha(t-\tau)} f_0(u(\tau), x^u_{z,t}(\tau))\, d\tau + e^{\alpha(t-s)} \vartheta(s, x^u_{z,t}(s)) \right\}
\end{aligned}$$

Inspecting the individual terms of the sum, the value function at time t and state $x(t) = z$ can be calculated in terms of the value function at time t_f. Eventually, with this, we can state the principle of dynamic programming, compare [Bardi and Capuzzo-Dolcetta, 2008].

Definition 4.6 (Dynamic Programming Principle)**.** *Let Assumptions 5 to 6 and Assumption 8 hold. Thus, for all $s \in [t, t_f]$, $z \in \mathbb{R}^{n_x}$, let*

$$\vartheta(t,z) = \inf_{u\in\mathcal{U}} \left\{ \int_t^s e^{\alpha(t-\tau)} f_0(u(\tau), x_{z,t}^u(\tau))\, d\tau + e^{\alpha(t-s)}\vartheta(s, x_{z,t}^u(s)) \right\}. \tag{4.7}$$

Hence, in dynamic programming the value function ϑ is required to satisfy the functional (4.7), which can be employed to derive an optimal policy. Moreover, Definition 4.6 implicitly states that for all controls $u \in \mathbb{U}$ the function,

$$s \mapsto \int_t^s e^{\alpha(t-\tau)} f_0(u(\tau), x_{z,t}^u(\tau))\, d\tau + e^{\alpha(t-s)}\vartheta(s, x_{z,t}^u(s)) \tag{4.8}$$

is nondecreasing. Further, suppose there exists an optimal control $\hat{u} \in \mathcal{U}$ minimizing the objective function, then the function defined in (4.8) is constant, cf. [Bardi and Capuzzo-Dolcetta, 2008]. Additionally, by derivation of the value function ϑ, provided ϑ is continuously differentiable, one obtains a partial differential equation, namely the HJB equation. Recalling the definition of the dynamic Hamiltonian,

$$\mathcal{H}(x,p) \coloneqq \sup_{u\in\mathbb{U}} \{-f(u,x)\cdot p - f_0(u,x)\},$$

and the natural terminal condition $\vartheta(t_f, z) = \varphi(x_{z,t}^u(t_f))$, the value function constitutes as a solution of

$$-\partial_t\vartheta(t,z) + \alpha\vartheta(t,z) + \mathcal{H}(z, \nabla\vartheta(t,z)) = 0$$

in $\mathbb{R}^{n_x}\times[t_0, t_f]$, with $\nabla\vartheta(t,z)$ the gradient in terms of the states and $\partial_t\vartheta(t,z)$ the partial temporal derivative of the value function. Commencing from or towards a state z, the HJB equation characterizes the evolution of the value function with respect to the state trajectories.

Remark 4.7. *Let us comment on the stated assertions:*

- *In Section 4.5 it will become apparent that the HJB has a discrete counterpart, i.e., the Bellman equation.*
- *Classic solutions on the HJB equation in general might neither be unique nor exist, even under suitable assumptions such as Lipschitz continuity, the existence of a solution can not be guaranteed (e.g. Eikonal equation). Moreover, the value function usually is not differentiable, rendering the above characterization impractical.*
- *The concept of viscosity solutions by [Crandall and Lions, 1983], provides a framework in terms of weak solutions as well as existence and uniqueness properties for ϑ.*

- *Usually there are two ways to go about solving the HJB equation. In the case where ϑ is differentiable, the HJB equation is a necessary and sufficient condition for an optimum. More common however is to compute a discrete approximation of the HJB equation and employ numerical methods, to solve the discretized problem. The later approach will be discussed in more detail in Section 4.5.*

4.4 Constrained Problem

So far, our discussion has been restricted to unconstrained optimal control problems. Naturally, in many applications the states and controls are, however, subject to physical limitations or boundaries, which appear as constraints within the control problems and need to be considered to appropriately model the problem.

Consider a finite horizon optimal control problem as in (uOCP), i.e., $t_f < +\infty$. Further, let $\mathcal{D} \subset \mathbb{R}^{n_x}$ be a closed nonempty subset, such that, without further specification, $\mathcal{D}$ harbors the constraints imposed on the states. The set of admissible controls is then defined by

$$\check{\mathcal{U}}(t,z) := \left\{ u : [t_0, t_f] \to \mathbb{R}^{\nu} \mid x^u_{z,t}(\tau) \in \mathcal{D},\ \forall \tau \in [t_0, t_f] \right\}.$$

Moreover, for the value function associated with the constrained optimal control problem we have

$$\vartheta(t,z) := \begin{cases} \inf_{u \in \check{\mathcal{U}}(t,z)} \int_t^{t_f} f_0(u(\tau), x^u_{z,t}(\tau))\, d\tau + \varphi(x^u_{z,t}(t_f)), & \text{if } \mathcal{X}_{\mathcal{D}}\left(z; [t, t_f]\right) \neq \emptyset \\ +\infty, & \text{if } \mathcal{X}_{\mathcal{D}}\left(z; [t, t_f]\right) = \emptyset \end{cases} \tag{4.9}$$

Herein, we employed the common convention to set the value function to infinity whenever it is not feasible.

Remark 4.8. *Infinity can be substituted by any constant value larger than the maximum, the value function attains, in order to achieve an equivalent definition. This approach often is even recommended, which becomes easier to understand later on, when we discuss the discrete problem formulation.*

The imposition of state constraints results in the value function to be possibly discontinuous, provided no other assumptions are made. Moreover, ϑ is not necessarily defined on the boundary of the constrained state space $\partial\mathcal{D}$. Hence, it is no longer immediately obvious that by standard arguments of DPP (Definition 4.6), the value function ϑ is a solution of the associated state constrained HJB equation [Bardi and Capuzzo-Dolcetta, 2008, Chapter III, p. 155]:

$$-\partial_t \vartheta(t,z) + \mathcal{H}(z, \nabla\vartheta(t,z)) = 0, \qquad \text{in } (t_0, t_f) \times \mathcal{D} \tag{4.10}$$

$$\vartheta(t_f, z) = \varphi, \qquad \text{in } \mathcal{D}. \tag{4.11}$$

To retain a feasible problem assume $\mathcal{X}_{\mathcal{D}}\left(z;[t_0,t_f]\right) \neq \emptyset$ for all $z \in \mathbb{R}^{n_x}$. Leading to an extended definition, adopting the notation of *constrained viscosity solutions* by [Soner, 1986a, Soner, 1986b].

Definition 4.9. *A function $\hat{\vartheta}$ is a constraint viscosity solution of (4.10) if it is a subsolution on $(t_0,t_f)\times\mathcal{D}\setminus\partial\mathcal{D}$ and a supersolution on $(t_0,t_f)\times\mathcal{D}$.*

However, Definition 4.9 does not imply any uniqueness properties on $\hat{\vartheta}$ as a solution of (4.10). Therefore, additional controllability assumptions are required, especially to ensure continuity of $\hat{\vartheta}$ close to the boundary of $\mathcal{D}$ [Capuzzo-Dolcetta and Lions, 1990, Ishii and Koike, 1996, Soner, 1986a]. Otherwise the state constrained HJB admits several solutions in the constrained viscosity sense (see [Bokanowski et al., 2011] for instance).

Considering suitable constraint qualifications allows to cope with these continuity deficiencies. [Soner, 1986a] established the inward pointing constraint qualification, stating, if at every $z\in\mathcal{D}$ there exists a control $u\in\mathbb{R}^{\nu}$ and a constant β for which

$$f(u,z)\eta(z) < -\beta < 0,$$

where $\eta(z)$ denotes the outward normal vector. Thus, the inward pointing constraint qualification ensures that from any initial point in $\mathcal{D}$ there exists a control and a corresponding feasible trajectory such that for all times $t\in[t_0,t_f]$, mentioned trajectory remains within $\mathcal{D}$. A variation of the inward pointing condition, requiring weaker assumptions is presented in [Capuzzo-Dolcetta and Lions, 1990, Bardi and Capuzzo-Dolcetta, 2008].

A different constraint qualification complementing the inward pointing condition, was introduced by [Frankowska and Plaskacz, 2000, Frankowska and Vinter, 2000] , requiring for any point $z\in\partial\mathcal{D}$, that there exists a control $u\in\mathbb{R}^{n_u}$ such that

$$f(u,z)\eta(z) > 0. \tag{4.12}$$

To put it another way, coming from the inside of $\mathcal{D}$ any point on the boundary can be reached. Moreover, contrary to the inward pointing condition, (4.12) is topological invariant. Under this condition ϑ is not necessarily continuous, yet characterizes as a unique lower semicontinuous solution of (4.10)

At the cue of lower semicontinuity, [Ishii and Koike, 1996] established an extension of Soners' inward pointing condition by definition of lower and upper semicontinuous envelopes $\underline{\vartheta}$ and $\overline{\vartheta}$ respectively

$$\begin{aligned}\overline{\vartheta}(t,z) &:= \limsup_{r\to 0}\left\{\vartheta(\cdot,y)\mid y\in\mathcal{D},|y-z|\leq r\right\},\\ \underline{\vartheta}(t,z) &:= \liminf_{r\to 0}\left\{\vartheta(\cdot,y)\mid y\in\mathcal{D},|y-z|\leq r\right\}.\end{aligned}$$

Definition 4.10. *A bounded function $\hat{\vartheta}$ is called a viscosity subsolution of (4.10) if*

$$-\partial_t \overline{\vartheta}(t,z) + \mathcal{H}(z, \nabla \overline{\vartheta}(t,z)) \leq 0, \qquad \text{in } \mathcal{D},$$

respectively a supersolution in the viscosity sense if

$$-\partial_t \underline{\vartheta}(t,z) + \mathcal{H}(z, \nabla \underline{\vartheta}(t,z)) \geq 0, \qquad \text{in } \mathcal{D}.$$

Another course to cope with unpleasant constraints, through reformulation of the original control problem by means of exact penalization, was adapted by [Altarovici et al., 2013]. More precisely, the value function of an auxiliary unconstrained problem, is utilized to characterize the epigraph of ϑ. The unconstrained problem then can be characterized by classical arguments as the unique viscosity solution of the associated Hamilton-Jacobi (HJ) equation, compare [Assellaou, 2015, Altarovici et al., 2013]. [Assellaou, 2015] applied the auxiliary reformulation approach to probabilistic control problems in finite and infinite horizon settings, and relaxed the deterministic theory to the local Lipschitz setting.

4.5 Numerical Approximation

Solving the HJB equation numerically is a challenging task. In general, with respect to optimal control problems, solutions to partial differential equation (PDE) emanate by application of the discrete dynamic programming principle. Therefore, in this section approximation methods for the solution of PDEs are discussed. In particular, the finite difference method used by [Crandall and Lions, 1984] with respect to HJB equations and the Semi-Lagrangian approach.

Finite difference schemes naturally approximate functions at uniform grid points and derivatives through finite differences or higher order polynomials on the given mesh. Higher order interpolation schemes, however give rise to oscillatory behaviour of the solution if the solution is fraught with discontinuities. An adaptive interpolation scheme, namely essentially non-oscillatory (ENO) proposed in [Harten et al., 1987] and [Harten et al., 1986], provides higher order accuracy even if discontinuities in the solution are present.

Proposed by [Courant et al., 1952] the Semi-Lagrangian method enjoys great popularity with applications emanating from meteorology and fluid dynamics, see [Staniforth and Côté, 1991] for instance. An initial adaptation to stationary HJB was presented by [Falcone, 1991]. As far as dynamic HJB equations are concerned, similar schemes have been applied with relation to optimal control problems (see [Falcone and Ferretti, 2002] and [Falcone and Giorgi, 1999]). Commonly, forward HJB approaches are considered ([Cacace et al., 2014] for instance). Here a backward Semi-Lagrangian scheme is proposed, providing a foundation for the development of an algorithm using backward propagation. This especially proves beneficial for problems where only the final state is known.

4.5.1 Finite Difference Scheme

To begin with, let us recall the (backward) HJB equation, compare for instance [Bardi and Capuzzo-Dolcetta, 2008, Chapter III, p. 155], of the finite horizon problem (4.10), where, for simplicity, we neglect the discount terms, yielding

$$-\partial_t \vartheta(t,z) + \mathcal{H}(z, \nabla\vartheta(t,z)) = 0, \qquad \text{in } (t_0,t_f)\times\mathcal{D}, \tag{4.13}$$

$$\vartheta(t_f,z) = \varphi(x^u_{z,t}(t_f)), \qquad \text{in } \mathcal{D}. \tag{4.14}$$

Acquiring a finite difference scheme, for solving (4.13) numerically, involves the approximation of the partial derivatives by finite differences. Initially a semi-discrete scheme is introduced. Therefore, let the spatial domain be governed by a uniform mesh $x_{k,i} = kh^i_x$ and let $e_i \in \mathbb{R}^{n_x}$ denote the unit vector of the respective dimension for $k \in \mathbb{N}_0$ and all $i = 1,\dots,n_x$. The approximation of the value function then is denoted by $\vartheta(t,z_k)$. The semi-discrete approximation of (4.13) then is defined by

$$\begin{aligned} -\partial_t\vartheta(t,z_k) &= -H(z_k,\vartheta^+(t,z_k),\vartheta^-(t,z_k)),\\ \vartheta(t_f,z_k) &= \varphi(x^u_{z,t}(t_f)), \end{aligned}$$

where $H(z_k,\vartheta^+(t,z_k),\ \vartheta^-(t,z_k))$ denotes a numerical Hamiltonian and $\vartheta^+(t,z_k),\vartheta^-(t,z_k)$ are approximations to the right and respectively left derivatives of ϑ at z_k. A first order monotone scheme of this type was introduced by [Crandall and Lions, 1984], such that for all $i = 1,\dots,n_x$,

$$\vartheta^+_i(t,z_k) = \frac{\vartheta(t,z_k+h^i_x e_i)-\vartheta(t,z_k)}{h^i_x} \quad \text{and} \quad \vartheta^-_i(t,z_k) = \frac{\vartheta(t,z_k)-\vartheta(t,z_k-h^i_x e_i)}{h^i_x}.$$

Here, the derivates are approximated by spatial forward and backward differences. Accordingly, the time derivative is approximated by a forward difference, complementing the discretization scheme. To this end, consider the lattice $\mathbb{G}_h$ and let $t_\ell = \ell h$ with $\ell \in \mathbb{N}_0$, $h = \frac{t_f-t_0}{M}$ and

$$-\frac{\vartheta(t_{\ell+1},z_k)-\vartheta(t_\ell,z_k)}{h} = -H(z_k,\vartheta^+(t_\ell,z_k),\vartheta^-(t_\ell,z_k))$$

yielding,

$$\begin{aligned} -\vartheta(t_{\ell+1},z_k)+\vartheta(t_\ell,z_k)+hH(z_k,\vartheta^+(t_\ell,z_k),\vartheta^-(t_\ell,z_k)) &= 0,\\ \vartheta(t_M,z_k) &= \varphi(x(t_M)). \end{aligned}$$

In particular, for a first order approximation of ϑ^+_k, a stencil $\{x_k,x_{k+1}\}$ containing x_k is chosen. Then, a suitable interpolating polynomial $p(x)$ containing this stencil must be defined. Since the stencil only contains two points, the polynomial is of degree one. Next, we set $\vartheta^+_k = p'(x_k)$ as an approximation to the right derivative of the value function. Thus, acquiring for all $i = 1,\dots,n_x$,

$$\vartheta_i(t_\ell,z_k) = \frac{\vartheta(t_\ell,z_k+h^i_x e_i)-\vartheta(t_\ell,z_k)}{h}.$$

Accordingly, this approach can be adapted for the left derivative ϑ_k^- with a left shifted stencil. Note, that the stability of finite difference methods essentially depends on the Courant number $C = v\frac{h}{h_x}$, where v denotes a (propagation) velocity. Let us stress, that the velocity is not necessarily constant but can variate in time and space, i.e. $v(x,t)$. Then, the approximation is considered numerically stable if the Courant-Friedrich-Lewy (CFL) condition, i.e., $|C| \leq 1$ is satisfied (see [Courant et al., 1928]). To illustrate the numerical stability, consider the following example.

Example 4.11. Let the advection equation with initial condition $u(0,x) = e^{-100(x-0.4)^2}$ be defined by

$$\partial_t u + c\partial_x u = 0,$$

where c is a positive constant. To solve the advection equation a first-order upwind scheme is used to approximate the partial derivatives. That is, a forward difference for the temporal derivative and a backward difference for the spatial derivative is employed, yielding

$$u(t_{\ell+1}, x_k) = u(t_\ell, x_k) - c\frac{h}{h_x}(u(t_\ell, x_k) - u(t_\ell, x_{k-1})).$$

Thus, we have an approximation with first order accuracy in space and time. Further, let $x_k \in [0,1]$, $t_\ell \in [0, 0.5]$ for all $k = 1, \ldots, N_x$ and $\ell = 1, \ldots, M$. Herein, N_x and M denote the number of grid points to approximate the spatial and respectively the temporal domain. Let us consider two sets of discretization parameters to illustrate the aspect of numerical stability and its relation to the Courant number. First, choose $N_x = 100$ and $M = 50$ and $c = 1$. Thus, the Courant number is $C = 1 \leq 1$. For the second setting, we increase the spatial discretization such that $N_x = 150$, yielding a Courant number of $C = 1.5$. In Figure 4.3, it becomes clear, that if the Courant number no longer satisfies the CFL condition, the solution is no longer stable and the graph explodes. Indicated by heavy oscillations in the numerical solution.

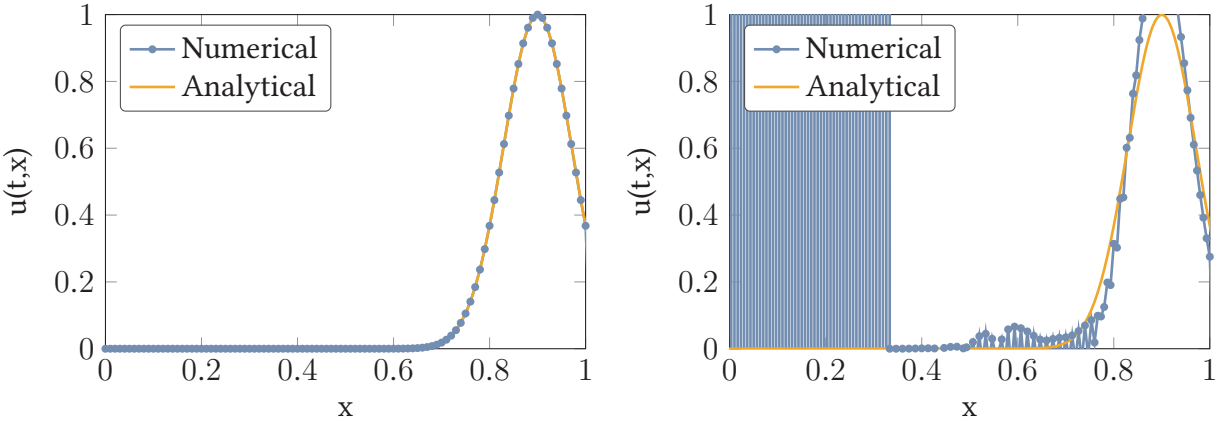


Figure 4.3: *Analytical (orange) and numerical (blue) solution of the advection equation. With $h_x = 0.01$ and $h = 0.01$, i.e., Courant number of $C = 1$ (left) and $h_x = 0.067$ and $h = 0.01$, i.e., Courant number of $C = 1.5$ (right) at $t = 0.5$.*

Remark 4.12. *Alternatively higher order finite difference schemes can be obtained similarly, tend however to oscillations if the solution is not regular. As an example, this would be the case if a fixed stencil is used, while traversing over a discontinuity of the solution.*

A popular interpolation scheme to cope with these problems is the ENO method, see for instance [Bokanowski et al., 2010],[Osher and Shu, 1991] and [Shu, 2007]. This procedure was later extended by [Jiang and Peng, 2000] resulting in the weighted essentially non-oscillatory (WENO) scheme. The idea of ENO is to dynamically adjust the stencil, commencing from the first order stencil as above, and iteratively adding grid points to the left and right. Then, by comparison of the Newton divided differences, adapting the stencil, exhibiting a lower deviation with respect to the first degree interpolation polynomial. For instance, if a third order approximation is chosen and suppose the resulting stencil is $\{z_{k-2}, z_{k-1}, z_k, z_{k+1}\}$ then the approximation of the left derivative becomes

$$\vartheta^-(t, z_k) = \frac{1}{h_x}\left(\frac{1}{6}\vartheta(t, z_{k-2}) - \vartheta(t, z_{k-1}) + \frac{1}{2}\vartheta(t, z_k) + \frac{1}{3}\vartheta(t, z_{k+1})\right).$$

Likewise, a higher order discretization scheme, e.g. a Runge-Kutta method of order three, can be introduced to approximate the temporal derivative.

4.5.2 Backward Semi-Lagrangian Scheme

Similar to the finite difference schemes, the Semi-Lagrangian method uses an Eulerian framework, respectively a regular grid, however, in conjunction with an estimation of the total derivative. With respect to HJB equations and optimal control problems, given a spatial grid, a Semi-Lagrangian scheme can be obtained by discretization of the dynamic programming principle in time. The combination of Eulerian and Lagrangian properties renders the disposition of larger time steps, compared to finite difference schemes, possible. More precisely, the numerical solution is unconditionally stable, hence not restrained by the limitations imposed by the CFL condition, compare [Bates and McDonald, 1982].

Equivalently to the (standard) forward HJB, cf. [Falcone and Ferretti, 2002], we want to derive a Semi-Lagrangian scheme with respect to the backward HJB equation,

$$-\partial_t \vartheta(t, z) + \mathcal{H}(z, \nabla\vartheta(t, z)) = 0, \tag{4.15}$$

$$\vartheta(t_f, z) = \varphi(x^u_{z,t}(t_f)). \tag{4.16}$$

Herein, the Hamiltonian is given by,

$$\mathcal{H}(z, \nabla\vartheta(t, z)) = \sup_{u\in\mathbb{U}} \{-f(u, x)\nabla\vartheta(t, z) - f_0(u, x)\}$$

Let the lattice be defined by $t_\ell = \ell h$, $\ell \in \mathbb{N}_0$ and $x_k = k h_x$, for $k \in \mathbb{N}_0$. To obtain the backward Semi-Lagrangian scheme, the directional derivative in the Hamiltonian is approximated by

$$-f(u, z_k)\nabla\vartheta(t_\ell, z_k) = -\frac{\vartheta(t_\ell, z_k) - \vartheta(t_\ell, z_{k-1})}{h_x} f(u, z_k) + O(h_x)$$

where z_k emerges from the evolved dynamical system. Replacing h_x by an explicit Euler approximation of the dynamics,

$$h_x = z_{k+1} - z_k = h f(u, z_k)$$

yields,

$$-f(u, z_k)\nabla\vartheta(t_\ell, z_k) = -\frac{\vartheta(t_\ell, z_k) - \vartheta(t_\ell, z_{k-1})}{h} + O(h_x). \tag{4.17}$$

Substitution of (4.17) into the Hamiltonian together with an approximation of the temporal derivative in (4.15) by backward finite differences gives,

$$-\frac{\vartheta(t_\ell, z_{k-1}) - \vartheta(t_{\ell-1}, z_{k-1})}{h} + \sup_{u\in\mathbb{U}}\left\{-\frac{\vartheta(t_\ell, z_k) - \vartheta(t_\ell, z_{k-1})}{h} - f_0(u, z_{k-1})\right\} = 0.$$

Multiplication by h leads to

$$-\vartheta(t_\ell, x_{k-1}) + \vartheta(t_{\ell-1}, x_{k-1}) + \sup_{u\in\mathbb{U}}\{-\vartheta(t_\ell, z_k) + \vartheta(t_\ell, z_{k-1}) - h f_0(u, z_{k-1})\} = 0.$$

Then solving for $\vartheta(t_{\ell-1}, x_{k-1})$

$$\begin{aligned}\vartheta(t_{\ell-1}, x_{k-1}) &= \vartheta(t_\ell, x_{k-1}) - \sup_{u\in\mathbb{U}}\{-\vartheta(t_\ell, z_k) + \vartheta(t_\ell, z_{k-1}) - h f_0(u, z_{k-1})\}\\ &= \vartheta(t_\ell, z_{k-1}) + \inf_{u\in\mathbb{U}}\{\vartheta(t_\ell, z_k) - \vartheta(t_\ell, z_{k-1}) + h f_0(u, z_{k-1})\}\\ &= \inf_{u\in\mathbb{U}}\{\vartheta(t_\ell, z_k) + h f_0(u, z_{k-1})\}\end{aligned}$$

Eventually, performing an index shift $k \longrightarrow k+1$ yields the time explicit Semi-Lagrangian scheme:

$$\vartheta(t_\ell, z_k) = \inf_{u\in\mathbb{U}}\{\vartheta(t_{\ell+1}, z_{k+1}) + h f_0(u, z_k)\}\,. \tag{4.18}$$

Heed that z_{k+1} in $\vartheta(t_{\ell+1}, z_{k+1})$ most certainly is not a grid point, and therefore the value of $\vartheta(t_{\ell+1}, z_{k+1})$ is not known. Though, by means of interpolation, taking into account the neighbouring grid points it can be approximated. The most common approach is to employ a linear interpolation. In Appendix A.2 we present a bilinear interpolation scheme. Moreover, an extension to higher dimensions is straightforward. However, as far as higher order polynomials are concerned for the restitution of $\vartheta(t_{\ell+1}, z_{k+1})$, their tendency to introduce oscillation in the

case of discontinuities also applies here. Making an ENO procedure a suitable alternative to linear interpolation. The introduced approximation error only depends on the interpolation step (compare [Falcone and Ferretti, 2002]). An error estimate for a more complex problem with respect to linear interpolation in higher dimension is provided in Section 4.6.3.

4.6 Application to Games

As established in Section 2.5 to characterize a solution of a GNEP as a Nash equilibrium a desirable property of the solution is to be a global optimum. By design, dynamic programming proves to be a proficient method to obtain global optima of the respective optimal control problem, motivating its adaptation to GNEP. In this section, a solution concept for the subproblems of the decomposition method, based on the principle of dynamic programming is derived. After introducing some additional notations, a relation to the Hamilton-Jacobi-Bellmann equation with the continuous penalty reformulation of (GNEP) is established. Subsequently, the continuous problem is approximated by the backward Semi-Lagrangian scheme, leading to the discrete value function of the subproblems of the individual player. To numerically solve the approximate value function the ADP algorithm otherwise known as value iteration is applied. We refer to [Bellman, 1957], [Bertsekas, 2005],[Bardi and Capuzzo-Dolcetta, 2008] and [Gerdts and Lempio, 2011]. Moreover, an error estimate of the resulting approximate value function is obtained on the basis of which the convergence of the subproblem to a global minimizer in finite time is proven. Additionally, convergence results for the semi-discrete value function to the continuous value function under suitable conditions are procured. Since it is well known that, despite its favourable solution properties, dynamic programming is burdened with high computational efforts, a parallel algorithm, suitable for GPU computation, is proposed, to cope with this burden.

4.6.1 Penalized Value Function

In Section 3.2.1 a generalized Nash equilibrium problem in terms of an individual player ν was introduced. Let us recur to this problem:

$$
\begin{aligned}
\min_{(u^\nu, x^\nu)} \quad & \varphi^\nu(x^\nu(t_f)) \\
\textit{subject to} \quad & x^\nu(t_0) = x_0^\nu, \\
& (x^\nu)'(t) = f(u^\nu(t), x^\nu(t)) && , a.e.\ t \in [t_0, t_f], \\
& k^\nu(x^\nu(t)) \leq 0 && , \forall\ t \in [t_0, t_f], \\
& g^\nu(x(t)) \leq 0 && , \forall\ t \in [t_0, t_f], \\
& u^\nu(t) \in \mathbb{U}_\nu.
\end{aligned}
$$

Consequently, as we recall, pursuing the penalization approach by instigating a penalty function, which characterizes the shared constraints $g^\nu(x(t))$ yields a Nash equilibrium problem in the form of (NEP). Moreover, to keep the notation concise, let $P^\nu_{t\to\tau}(x^\nu, x^{-\nu})$, denote the penalty function in (3.17), where the index $t \to \tau$ indicates the limits of the integral therein. Then, using the dynamic programming principle in Definition 4.6 a recursive definition of the continuous value function can be established.

Definition 4.13. *For fixed $x^{-\nu}$ at time $t \in [t_0, t_f)$, $t_0 < \tau \leq t$ and initial states $z^\nu \in X_\nu$, the value function ϑ of (NEP), is recursively defined by*

$$\begin{aligned}\vartheta(t_f, z^\nu) &:= \varphi(x^\nu(t_f)),\\ \vartheta(t, z^\nu) &:= \min_{x^\nu \in X_\nu(t,z^\nu)} \left[P^\nu_{t\to\tau}(x^\nu, x^{-\nu}) + \vartheta(\tau, x^\nu(\tau))\right].\end{aligned}$$

Before we state the approximate dynamic programming algorithm, see [Bertsekas, 2005], some additional notations are required. First, we instigate a discretization in the state variables. To this end, for each player $\nu = 1, \ldots, N$ and each state $i = 1, \ldots, n_x$ let

$$h^{i,\nu}_x := \frac{x^{i,\nu}_{\max} - x^{i,\nu}_{\min}}{M^{i,\nu}_x},$$

denote the step size of the respective state, where $h^\nu_x \in \mathbb{R}^{n_x}_{++}$, and $M^\nu_x \in \mathbb{N}^{n_x}$ is the number of grid points. On that note, we can state the definition of the respective spatial lattice

$$\mathbb{G}^\nu_x := \bigtimes_{i=0}^{n_x} \left\{x^{i,\nu}_{\min} + j h^{i,\nu}_x \mid j = 0, \ldots, M^{i,\nu}_x\right\}.$$

Moreover, let $t_\ell \in \mathbb{G}_h$ denote an initial date and $z^\nu_\ell = x^\nu_h(t_\ell)$, not necessarily in $\mathbb{G}^\nu_x$, indicate an initial point for each player $\nu = 1, \ldots, N$. Within the scope of these definitions, for one time step

$$\begin{aligned}X^h_\nu(t_\ell, z^\nu_\ell) := \big\{(u^\nu_h, x^\nu_h) \in L^{n_u}_{1,h} \times W^{n_x}_{1,1,h} \mid\ & x^\nu_h(t_\ell) = z^\nu_\ell,\\ & x^\nu_h(t_{\ell+1}) = x^\nu_h(t_\ell) + h_\ell f(u^\nu_h(t_\ell), x^\nu_h(t_\ell));\\ & k^\nu(x^\nu_h(t_\ell)) \leq (h + h_u)^p\\ & u^\nu_h(t_\ell) \in \mathbb{U}^{h^\nu_u}_\nu\big\},\end{aligned}$$

defines the discrete feasible set. Besides, to alleviate the notation the abbreviation $\varkappa^\nu_h = (u^\nu_h, x^\nu_h)$, is sustained and let

$$\xi_\nu(\varkappa^\nu_h) := x^\nu_h(t_\ell) + h f(\varkappa^\nu_h),$$

delineate the function of the evolution of the dynamics in one step. Following the previous sections, the use of explicit Euler as an approximation scheme is maintained. Moreover, let us

introduce the mapping,

$$\mathcal{P}^\nu(x^\nu, x^{-\nu}) := \sum_{j=1}^{m_\nu} \max\{0, g_j^\nu(x)\},$$

with the vector $(x^\nu, x^{-\nu}) \in \mathbb{R}^{Nn_x}$. Then, for $\ell = 0, \dots, M-1$ we define the contribution of the ν-th player in the ℓ-th step to the penalty term by

$$P_\ell^\nu(x_h^\nu, x_h^{-\nu}) := \mathcal{P}^\nu(x_h^\nu(t_{\ell+1}), x_h^{-\nu}(t_{\ell+1})). \tag{4.19}$$

Perpetuating the convention in (4.9), we can now recursively define the time discrete value function.

Definition 4.14. *For the discrete penalized Nash equilibrium problem* (NEP$_h$) *with fixed* $x_k^{-\nu}$ *at time* t_ℓ *and initial states* z_ℓ^ν*, the approximate value function is recursively defined by*

$$\vartheta_h(t_M, z_M^\nu) := \varphi^\nu(z_M^\nu), \tag{4.20}$$

$$\vartheta_h(t_\ell, z_\ell^\nu) := \min_{\varkappa_h^\nu \in X_\nu^h(t_\ell, z_\ell^\nu)} \left[P_\ell^\nu(x_h^\nu, x_h^{-\nu}) + \vartheta_h(t_{\ell+1}, \xi_\nu(\varkappa_h^\nu))\right] \qquad \forall \ell = M-1, \dots, 0. \tag{4.21}$$

with $\vartheta_h : \mathbb{G}_h \times \mathbb{R}^{n_x} \to \mathbb{R}$.

In the standard dynamic programming approaches, if the state is not feasible, that is $\bar{x}^\nu \notin X_\nu$, the convention that $\vartheta(\cdot, \bar{x}^\nu) = \infty$ is used. Indeed, this convention serves as a viable filter to handle infeasible points, by assigning a huge value to this state and control combination. Thus, rendering it unattractive as a possible solution. Combined with an interpolation, however, this implicates an essential disadvantage. By the nature of the recursive computation of the value function, the values thus assigned are propagated to neighbouring grid points. As a consequence, the feasible set is reduced, effecting possible solution trajectories.

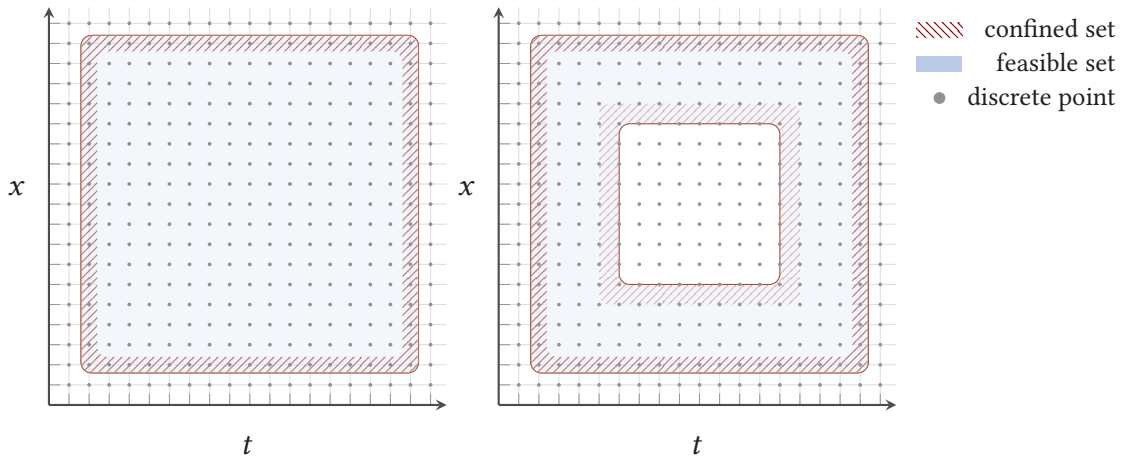


Figure 4.4: *Illustration of the reduction of the feasible set if the infinity convention is employed in approximate dynamic programming. For a convex feasible set (left) and a nonconvex feasible set (right).*

In Figure 4.4 this behaviour is exemplified for a convex and nonconvex feasible set. Let us stress, that the severity of this effect concerning the nonconvex case, compared to the convex case, can possibly be worse, to the extent that the feasible set is virtually empty.

As already mentioned above, the second term of (4.21) needs to be approximated using an interpolation method, as the end point of the evolution of the dynamics not necessarily resides on a grid point. Thus, the value function at this point is effectively unknown. Therefore, we introduce an approximated value function $\tilde{\vartheta}_h$, reflecting the necessity of interpolation. Further, to provide a generic framework, the characteristic of the interpolation is defined by a convex combination of the neighbouring grid points. For an arbitrary non-grid point x^ν, these points are denoted by $\Pi_j(x^\nu)$ for $j = 1, \dots, 2^{n_x}$. Hence, if the values on all grid points are known, we set

$$\tilde{\vartheta}_h(t_\ell, z^\nu) := \sum_{j=1}^{2^{n_x}} \alpha_j \tilde{\vartheta}_h(t_\ell, \Pi_j[z^\nu]), \tag{4.22}$$

for suitable weights $\alpha_j \geq 0$ and $\sum_{j=1}^{2^{n_x}} \alpha_j = 1$. With this setting, the weights can be chosen deliberately. For example, in terms of a nearest neighbour method the entire weight can be assigned to the closest grid point. Alternatively, a linear (or multilinear) interpolation scheme can be adopted, as it will be used as the preferred method in the course of this work. Accordingly, higher order interpolation methods, like spline interpolation can be applied, which however exert greater computational effort. Since, in practical applications the computation time plays a critical role and usually is required to be kept low, in particular for real time application, the choice of linear interpolation provides a good balance between computational effort and accuracy. Moreover, if any of the components of the point x^ν is already on the grid, we consider only the neighbours not related to these components, while setting the remaining weights to zero.

Nevertheless, due to the adoption of an interpolation scheme, which constitutes an approximation, accordingly a recursively propagating error is introduced. The aberration induced by this approximation is addressed in the subsequent section.

4.6.2 Convergence of the Approximate Value Function

The concept of ADP is derived from the definition of the dynamic programming principle, which states that the decisions at dates $\ell, \ell + 1, \dots, M$ are independent of the preceding decision. Thus, implicating that the optimal control problem can be solved in stages, where the optimality of the trajectory of each stage is independent of the previous decisions, compare Figure 4.5. Each stage then is designated by the temporal grid points. This principle is applicable as long as the respective optimal control problem can be split into separate stages.

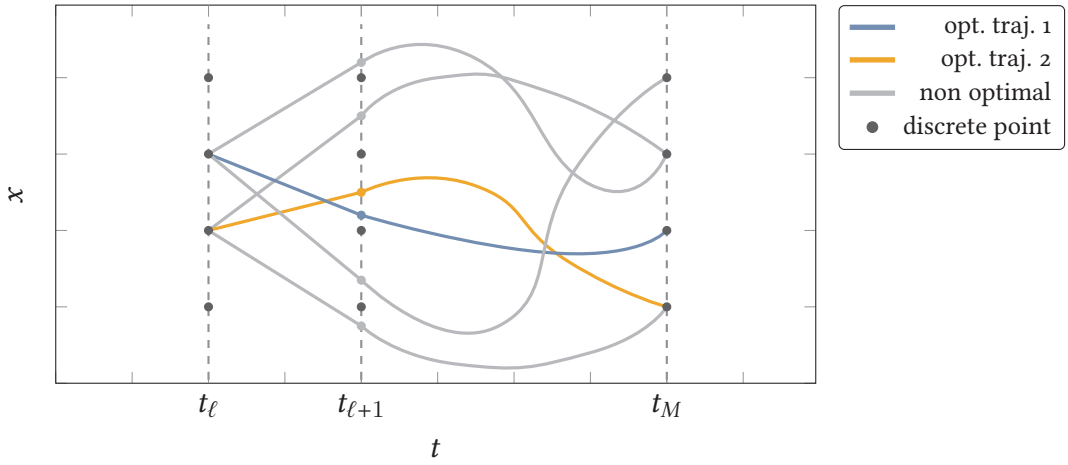


Figure 4.5: *Illustration of the recursion of the optimal value function emanating from the dynamic programming principle. Commencing from two different initial states, leading to distinct non optimal (gray) and optimal trajectories (orange and blue),*

Given an approximate value function defined on a spatial grid $\mathbb{G}_x^\nu$ the essential idea of the ADP approach is to compute at all grid points the values of the approximate value function. Therein, it can be exploited that the value function at time t_M is known and defined by

$$\tilde{\vartheta}_h(t_M, z_M^\nu) := \varphi^\nu(z_M^\nu) \tag{4.23}$$

with $z_M^\nu \in \mathbb{G}_x^\nu$. Suppose $\tilde{\vartheta}_h(t_{\ell+1}, \tilde{z}^\nu)$ is known for an arbitrary $\tilde{z}^\nu \in X_\nu$ then for all $z_\ell^\nu \in \mathbb{G}_x^\nu$, $\ell = M-1, \dots, 0$, with (4.21) and (4.22) we can recursively define:

$$\tilde{\vartheta}_h(t_\ell, z_\ell^\nu) := \min_{\varkappa_h^\nu \in X_\nu^h(t_\ell, z_\ell^\nu)} \left[P_\ell^\nu(x_h^\nu, x_h^{-\nu}) + \sum_{j=1}^{2^{n_x}} \alpha_j \tilde{\vartheta}_h(t_{\ell+1}, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) \right]. \tag{4.24}$$

Commencing from t_M, (4.23) and (4.24) together allow to compute the value function backward in time.

Before we state the algorithm, let us elaborate on the approximation error introduced by the interpolation in the second term of (4.24). To estimate the approximation error, we first prove Lipschitz continuity of the exact value function $\vartheta_h(t_\ell, \cdot)$.

Remark 4.15. *Here, the notion of exact is to be understood in the sense of not interpolated, in order to distinguish the definitions of the value functions.*

Lemma 4.16. *In the setting of this section and with Assumptions 5 to 7, the value function $\vartheta_h(t_\ell, \cdot)$ is Lipschitz continuous for all $\ell = 0, \dots, M$.*

Proof. For $\ell = M$ the Lipschitz continuity follows from φ^ν being Lipschitz continuous by Assumption 5. Now assume, that $\vartheta_h(t_{\ell+1}, \cdot)$ is Lipschitz continuous for some $\ell \in \{0, \dots, M-1\}$.

The objective functions in the definition of the value function are Lipschitz continuous, because the functions $\xi_\nu(\cdot)$ are Lipschitz continuous on a compact set, and by chain rule this also holds for

$$P_\ell^\nu(x_h^\nu, x_k^{-\nu}) := \sum_{j=1}^{m_\nu} \max\{0, g_j^\nu(x_h(t_{\ell+1}))\}.$$

Hence, we get for two distinct initial points $z_\ell^\nu \neq \tilde{z}_\ell^\nu$ and

$$\begin{aligned}(u_h^\nu, x_h^\nu) &= \operatorname*{argmin}_{\varkappa^\nu \in X_\nu^h(t_\ell, z_\ell^\nu)} \left[P_\ell^\nu(x_h^\nu, x_h^{-\nu}) + \vartheta_h(t_{\ell+1}, \xi_\nu(\varkappa_h^\nu))\right], \\ (\tilde{u}_h^\nu, \tilde{x}_h^\nu) &= \operatorname*{argmin}_{\varkappa^\nu \in X_\nu^h(t_\ell, \tilde{z}_\ell^\nu)} \left[P_\ell^\nu(x_h^\nu, x_h^{-\nu}) + \vartheta_h(t_{\ell+1}, \xi_\nu(\varkappa_h^\nu))\right],\end{aligned}$$

a constant $L > 0$ such that

$$|\vartheta_h(t_\ell, z_\ell^\nu) - \vartheta_h(t_\ell, \tilde{z}_\ell^\nu)| \leq L\|(u_h^\nu, x_h^\nu) - (\tilde{u}_h^\nu, \tilde{x}_h^\nu)\|.$$

It remains to establish an estimation of $\left\|(u_h^\nu, x_h^\nu) - (\tilde{u}_h^\nu, \tilde{x}_h^\nu)\right\|$ in terms of $\|z_\ell^\nu - \tilde{z}_\ell^\nu\|$. By definition, we get for $z_\ell^\nu \neq \tilde{z}_\ell^\nu$

$$\|u_h^\nu - \tilde{u}_h^\nu\| \leq M_u^\nu h_u = \frac{M_u^\nu h_u}{\min\{h_x^\nu\}} \min\{h_x^\nu\} \leq \frac{M_u^\nu h_u}{\min\{h_x^\nu\}} \|z_\ell^\nu - \tilde{z}_\ell^\nu\|.$$

Further, by definition we have for all $t \in [t_\ell, t_{\ell+1}]$

$$\begin{aligned}\|x_h^\nu(t) - \tilde{x}_h^\nu(t)\| &= \|x_h^\nu(t_\ell) + (t - t_\ell) f(u_h^\nu(t_\ell), x_h^\nu(t_\ell)) - \tilde{x}_h^\nu(t_\ell) - (t - t_\ell) f(\tilde{u}_h^\nu(t_\ell), \tilde{x}_h^\nu(t_\ell))\| \\ &\leq \|x_h^\nu(t_\ell) - \tilde{x}_h^\nu(t_\ell)\| + h\|f(u_h^\nu(t_\ell), x_h^\nu(t_\ell)) - f(\tilde{u}_h^\nu(t_\ell), \tilde{x}_h^\nu(t_\ell))\| \\ &\leq \|z_\ell^\nu - \tilde{z}_\ell^\nu\| + h \max\{L_{f,1}, L_{f,2}\} \left[\|u_h^\nu(t_\ell) - \tilde{u}_h^\nu(t_\ell)\| + \|x_h^\nu(t_\ell) - \tilde{x}_h^\nu(t_\ell)\|\right] \\ &\leq \left(1 + h \max\{L_{f,1}, L_{f,2}\}\right) \|z_\ell^\nu - \tilde{z}_\ell^\nu\| + h \max\{L_{f,1}, L_{f,2}\} \frac{M_u^\nu h_u}{\min\{h_x^\nu\}} \|z_\ell^\nu - \tilde{z}_\ell^\nu\| \\ &= \left[1 + h \max\{L_{f,1}, L_{f,2}\} \left(1 + \frac{M_u^\nu h_u}{\min\{h_x^\nu\}}\right)\right] \|z_\ell^\nu - \tilde{z}_\ell^\nu\|.\end{aligned}$$

All these inequalities together prove Lipschitz continuity of $\vartheta_h(t_\ell, \cdot)$ and the assertion follows by backward induction. □

By this lemma the value function $\vartheta_h(t_\ell, \cdot)$ is Lipschitz continuous for all $\ell = 0, \ldots, M$. Now, we can turn to the estimation of the approximation error.

Lemma 4.17. *Let $L_\vartheta > 0$ be the maximum over all the Lipschitz constants of $\vartheta_h(t_\ell, \cdot)$ for $\ell = 0, \dots, M$. Let an arbitrary $t_\ell \in \{0, \dots, M\}$ be given. Then, we have for any grid point $z_\ell^\nu \in \mathbb{G}_x^\nu$*

$$|\tilde{\vartheta}_h(t_\ell, z_\ell^\nu) - \vartheta_h(t_\ell, z_\ell^\nu)| \leq L_V(M-\ell)\|h_x^\nu\| \tag{4.25}$$

Further, we have for any $z_\ell^\nu \in [x_{\min}^\nu, x_{\max}^\nu]$

$$|\tilde{\vartheta}_h(t_\ell, z_\ell^\nu) - \vartheta_h(t_\ell, z_\ell^\nu)| \leq L_\vartheta(M+1-\ell)\|h_x^\nu\|. \tag{4.26}$$

Proof. We prove the assertion by backward induction. For $\ell = M$ we do not have any approximation error at a grid point $z_M^\nu \in \mathbb{G}_x^\nu$, since

$$|\tilde{\vartheta}_h(t_M, z_M^\nu) - \vartheta_h(t_M, z_M^\nu)| = |\varphi_\nu(z_M^\nu) - \varphi_\nu(z_M^\nu)| = 0.$$

Hence, (4.25) holds for $\ell = M$. For a non grid point $z_M^\nu \in [x_{\min}^\nu, x_{\max}^\nu]$, we exploit Lipschitz continuity of ϑ_h and $\sum_{j=1}^{2^{n_x}} \alpha_j = 1$ to get

$$\begin{aligned}
|\tilde{\vartheta}_h(t_M, z_M^\nu) - \vartheta_h(t_M, z_M^\nu)| &\overset{(4.22)}{=} \left| \sum_{j=1}^{2^{n_x}} \alpha_j \tilde{\vartheta}_h(t_M, \Pi_j[z_M^\nu]) - \vartheta_h(t_M, z_M^\nu) \right| \\
&\overset{(4.25)}{=} \left| \sum_{j=1}^{2^{n_x}} \alpha_j (\vartheta_h(t_M, \Pi_j[z_M^\nu]) - \vartheta_h(t_M, z_M^\nu)) \right| \\
&\leq \sum_{j=1}^{2^{n_x}} \alpha_j L_\vartheta \|\Pi_j[z_M^\nu] - z_M^\nu\| \\
&\leq \sum_{j=1}^{2^{n_x}} \alpha_j L_\vartheta \|h_x^\nu\| = L_\vartheta \|h_x^\nu\|.
\end{aligned}$$

Hence, (4.25) and (4.26) hold for $\ell = M$. Now assume that (4.25) and (4.26) hold for some $\ell \in \{1, \dots, M\}$. Then we obtain for any grid point $z_{\ell-1}^\nu \in \mathbb{G}_x^\nu$:

$$\begin{aligned}
\tilde{\vartheta}_h(t_{\ell-1}, z_{\ell-1}^\nu) &= \min_{\varkappa_h^\nu \in X_\nu^h(t_{\ell-1}, z_{\ell-1}^\nu)} \left[P_{\ell-1}^\nu(x_h^\nu, x_h^{-\nu}) + \sum_{j=1}^{2^{n_x}} \alpha_j \tilde{\vartheta}_h(t_\ell, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) \right] \\
&= \min_{\varkappa_h^\nu \in X_\nu^h(t_{\ell-1}, z_{\ell-1}^\nu)} \Bigg[P_{\ell-1}^\nu(x_h^\nu, x_h^{-\nu}) + \sum_{j=1}^{2^{n_x}} \alpha_j \vartheta_h(t_\ell, \xi_\nu(\varkappa_h^\nu)) \\
&\quad + \sum_{j=1}^{2^{n_x}} \alpha_j \Big(\vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) - \vartheta_h(t_\ell, \xi_\nu(\varkappa_h^\nu)) \\
&\quad + \tilde{\vartheta}_h(t_\ell, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) - \vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) \Big) \Bigg]
\end{aligned}$$

Now we can use Lipschitz continuity of ϑ_h to estimate

$$|\vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) - \vartheta_h(t_\ell, \xi_\nu(\varkappa_h^\nu))| \leq L_\vartheta \|\Pi_j[\xi_\nu(\varkappa_h^\nu)] - \xi_\nu(\varkappa_h^\nu)\| \leq L_\vartheta \|h_x^\nu\|.$$

Further, by assumption we have (4.25) for ℓ and hence,

$$|\tilde{\vartheta}_h(t_\ell, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) - \vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa_h^\nu)])| \leq L_\vartheta (M - \ell) \|h_x^\nu\|.$$

This together with $\sum_{j=1}^{2^{n_x}} \alpha_j = 1$ implies on the one hand

$$\begin{aligned}
&\tilde{\vartheta}_h(t_{\ell-1}, z_{\ell-1}^\nu) \\
&\quad \leq \min_{\varkappa_h^\nu \in X_\nu^h(t_{\ell-1}, z_{\ell-1}^\nu)} \left[P_{\ell-1}^\nu(x_h^\nu, x_h^{-\nu}) + \vartheta_h(t_\ell, \xi_\nu(\varkappa_h^\nu)) + L_\vartheta(M + 1 - \ell) \|(h_x)\| \right] \\
&\quad = \vartheta_h(t_{\ell-1}, z_{\ell-1}^\nu) + L_\vartheta(M - (\ell - 1)) \|h_x^\nu\|,
\end{aligned}$$

and on the other hand

$$\begin{aligned}
&\tilde{\vartheta}_h(t_{\ell-1}, z_{\ell-1}^\nu) \\
&\quad \geq \min_{\varkappa_h^\nu \in X_\nu^h(t_{\ell-1}, z_{\ell-1}^\nu)} \left[P_{\ell-1}^\nu(x_h^\nu, x_h^{-\nu}) + \vartheta_h(t_\ell, \xi_\nu(\varkappa_h^\nu)) - L_\vartheta(M + 1 - \ell) \|(h_x)\| \right] \\
&\quad = \vartheta_h(t_{\ell-1}, z_{\ell-1}^\nu) - L_\vartheta(M - (\ell - 1)) \|h_x^\nu\|.
\end{aligned}$$

Together these two inequalities prove (4.25) for $\ell - 1$.

Now, let a non-grid point $z_{\ell-1}^\nu \in [x_{\min}^\nu, x_{\max}^\nu]$ be given. Then, we get

$$\begin{aligned}
& |\tilde{\vartheta}_h(t_{\ell-1}, z_{\ell-1}^\nu) - \vartheta_h(t_{\ell-1}, z_{\ell-1}^\nu)| \\
= \;& \left| \sum_{j=1}^{2^{n_x}} \alpha_j \tilde{\vartheta}_h(t_{\ell-1}, \Pi_j[z_{\ell-1}^\nu]) - \vartheta_h(t_{\ell-1}, z_{\ell-1}^\nu) \right| \\
\overset{(4.25)}{\leq} \;& \left| \sum_{j=1}^{2^{n_x}} \alpha_j \vartheta_h(t_{\ell-1}, \Pi_j[z_{\ell-1}^\nu]) - \vartheta_h(t_{\ell-1}, z_{\ell-1}^\nu) \right| + L_\vartheta(M - (\ell - 1)) \|h_x^\nu\| \\
\leq \;& \sum_{j=1}^{2^{n_x}} \alpha_j L_\vartheta \|\Pi_j[z_{\ell-1}^\nu] - z_{\ell-1}^\nu\| + L_\vartheta(M - (\ell - 1)) \|h_x^\nu\| \\
\leq \;& L_\vartheta(M + 1 - (\ell - 1)) \|h_x^\nu\|.
\end{aligned}$$

This proves (4.26) for $\ell - 1$, and completes the proof. □

By Lemma 4.17 we know that the exact value function ϑ_h is well approximated by the approximate value function $\tilde{\vartheta}_h$ for all sufficiently small $\|h_x^\nu\|$. Hence, we can solve problem (3.27) based on the approximate value function. Now, we are ready to state the corresponding algorithm:

Algorithm 6: Approximate Dynamic Programming Algorithm

(S.1) Computation of the approximate value function:
For all $z_M^\nu \in \mathbb{G}_x^\nu$ set

$$\tilde{\vartheta}_h(t_M, z_M^\nu) := \varphi^\nu(z_M^\nu)$$

For $\ell = M-1, \dots, 0$,

for all $z_\ell^\nu \in \mathbb{G}_x^\nu$ compute

$$\tilde{\vartheta}_h(t_\ell, z_\ell^\nu) = \min_{\varkappa_h^\nu \in X_\nu^h(t_\ell, z_\ell^\nu)} \left[P_\ell^\nu(x_h^\nu, x_h^{-\nu}) + \sum_{j=1}^{2^{nx}} \alpha_j \tilde{\vartheta}_h(t_{\ell+1}, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) \right].$$

(S.2) Computation of global minimizer:
Set the initial state to $\hat{x}_h^\nu(t_0) := x_0^\nu$.
For $\ell = 0, \dots, M-1$ compute

$$\hat{u}_h^\nu(t_\ell) = \operatorname*{argmin}_{\varkappa_h^\nu \in X_\nu^h(t_\ell, x_\ell^\nu)} \left[P_\ell^\nu(x_h^\nu, x_h^{-\nu}) + \sum_{j=1}^{2^{nx}} \alpha_j \tilde{\vartheta}_h(t_{\ell+1}, \Pi_j[\xi_\nu(\varkappa_h^\nu)]) \right],$$

and set

$$\hat{x}_h^\nu(t_{\ell+1}) := \xi_\nu(\hat{u}_h^\nu(t_\ell), \hat{x}_h^\nu(t_\ell)).$$

It now remains to show that the above algorithm admits a global optimum of the subproblem (3.27) in Algorithm 5 and that within the setting of this section, the algorithm terminates in finite time for a sufficiently small spatial discretization.

Theorem 4.18. *Let Assumptions 5 to 7 hold, then the Approximate Dynamic Programming Algorithm computes a global minimizer of problem (3.27) in (S.3) of Algorithm 5 in a finite time for all sufficiently small $\|h_x^\nu\|$.*

Proof. In (S.1) of Algorithm 6 the approximate value function $\tilde{\vartheta}_h$ is computed in finite time at all time steps and in all grid points in $\mathbb{G}_h^\nu$ and $\mathbb{G}_x^\nu$. In (S.2) the discrete optimization problems are solved for all time points $\ell = 0, \dots, M-1$. We have to evaluate the functions at $M_u^\nu + 1$ possible values for u^ν and choose the minimal value. Then, we evaluate the dynamics at the solution. Therefore, the entire algorithm terminates in a finite time.

Now, if we consider problem (3.27), we only have a finite number of strategies $\varkappa_h^\nu \in X_\nu^h$. Hence, we have only a finite number of possible values $f_\nu(\varkappa_h^\nu) + \rho P(x_h^\nu, x_h^{-\nu})$, from which we have to choose the minimal one. Then, the set of its minimizers leads to the optimal function value $\vartheta_h(t_0, x_0^\nu)$. Further, there is an $\varepsilon > 0$ such that each non-optimal function value is larger than $\vartheta_h(t_0, x_0^\nu) + \varepsilon$. Choosing $\|h_x^\nu\| < \frac{\varepsilon}{2L_\vartheta(M+1)}$ we obtain from Lemma 4.17, that one of the exact

minimizers must also be a minimizer for the approximate problem with function value $\tilde{\vartheta}_h(t_0, x_0^\nu)$. Thus, Algorithm 6 finds a global minimizer of problem (3.27). □

Let us summarize the results so far. In order to acquire a solution of (GNEP), after a penalty reformulation together with a suitable discretization, which recasts (GNEP) as a semi-discrete (NEP$_h$) we can use the decomposition algorithm, Algorithm 5, that terminates by Theorem 3.26 after a finite number of iterations. In each iteration of this algorithm a (semi-discrete) optimal control problem needs to be solved globally. Therefore a dynamic programming approach, which enables the application of an Approximate Dynamic Programming Algorithm is employed due to its suitable global solution properties. By Theorem 4.18, the computation of the exact global minimum is possible in finite time, if the lattice $\mathbb{G}_x^\nu$ of the state variables is sufficiently fine enough. Thus, we proved convergence of an algorithmic framework to solve the original (GNEP). The main assumption in this theoretical and algorithmic framework is the existence of a feasible point for (GNEP).

4.6.3 Error Estimate of the Continuous Value Function

After establishing an error estimate of the (numerically) approximated value function and the interpolated value function in the previous section, the purpose of this section is to derive an estimation of the error introduced by approximating the continuous value function by a higher order backward Semi-Lagrangian method. In addition, we show that the estimated error dissolves by refining the discretization. For general consideration, an appropriate setting and notation is adopted. We consider a grid $\mathbb{G}_h = \{I_i\}_{i\in\{0,\dots,M-1\}}$ where $\bar{\Omega} := \bigcup_{\ell=0}^{M-1} I_\ell$ and let

$$Q_h^p := \left\{ v_h \in C^p\left(\bar{\Omega}\right) \mid \forall i \in \{0,\dots,M-1\}, v_{h|I_i} \in \mathbb{Q}_p \right\}$$

denote the space of continuous piecewise interpolating functions, with $\mathbb{Q}_p$ the space of functions characterizing an interpolation scheme of degree p. Further we define the interpolation operator by

$$\mathcal{I}_p^h : C^p(\bar{\Omega}) \ni v \to q(v) \in Q_h^p.$$

Next, suppose $u \in C^{p+1}\left([t_0, t_f]\right)$, with $p \in \mathbb{N}_0$ and let $\tilde{u}_h(t) := \mathcal{I}_p^h[u]$ denote the interpolated continuation of $u : [t_0, t_f] \to \mathbb{R}^{n_u}$ at the grid points $t \in \mathbb{G}_h$. Let us stress, that an arbitrary interpolation method, e.g. polynomial, spline or trigonometric, can be applied. Next, we define,

$$u_h(t_i) := \Xi_{\{u_{\min}+jh_u \mid j=0,\dots,M_u\}}(\tilde{u}_h(t_i))$$

as the projection of $\tilde{u}_h(t_i)$ onto a grid point in $\mathbb{U} := \left\{u \in \mathbb{R}^{n_u} \mid u_{\min}^\nu \le u \le u_{\max}^\nu\right\}$ with the respective interpolation such that $u_h(t) := \mathcal{I}_p^h[u]$. Furthermore, in the following we denote by

$$\|\cdot\|_\infty := \operatorname*{ess\,sup}_{t\in[a,b]} \|\cdot\|$$

the essential supremum norm. Additionally, we require the auxiliary error estimate

$$\|u - \tilde{u}_h\|_\infty \leq O(h^{p+1})$$

which emanates from an interpolation of order p, see for instance [Hall and Meyer, 1976] for the respective estimate regarding spline interpolation and [Stoer, 2005] for polynomial interpolation. For the proof of the error estimate of the value function some intermediate technical estimates are required, which are stated in the following lemmata.

Lemma 4.19. *Suppose $u \in C^{p+1}\left([t_0, t_f]\right)$ and $\tilde{u}_h$ is the interpolated continuation at $t \in \mathbb{G}_h$ of order p, such that*

$$\|u - \tilde{u}_h\|_\infty \leq O(h^{p+1}) \tag{4.27}$$

Then the following estimates

$$\|\tilde{u}_h - u_h\|_\infty \leq O(h_u), \tag{4.28}$$
$$\|u - u_h\|_\infty \leq O(h^{p+1}) + O(h_u), \tag{4.29}$$

hold with positive h, h_u.

Proof. The first approximation in (4.28), can be proven by exploiting the definition of $\tilde{u}_h$ together with the definition of the projection onto the grid yielding that the approximation error of $u_h(t)$ and $\tilde{u}_h(t)$ is at most $\frac{h_u}{2}$. Hence, for any $t \in [t_\ell, t_{\ell+1}]$, we have

$$\|\tilde{u}_h - u_h\|_\infty \leq O(h_u).$$

It remains to prove (4.29). Eventually, using the previous estimate together with (4.27) we obtain for arbitrary $t_\ell \leq t \leq t_{\ell+1}$

$$\begin{aligned}\|u(t) - u_h(t)\| &= \|u(t) - \tilde{u}_h(t) + \tilde{u}_h(t) - u_h(t)\| \\ &\leq \|u(t) - \tilde{u}_h(t)\| + \|\tilde{u}_h(t) - u_h(t)\| \\ &\leq O(h^{p+1}) + O(h_u),\end{aligned}$$

concluding the proof. □

Next, consider a controlled system governed by the ordinary differential equation

$$x'(t) = f(u(t), x(t)), \qquad x(t_0) = x_0. \tag{4.30}$$

where the exact solution is denoted by $x(t)$. Furthermore, the approximate solution is defined by an arbitrary one step method, compliant with Definition 3.15. Assume that $x_h(t_0) = x(t_0)$, with $t \in [t_0, t_f]$.

Lemma 4.20. *Consider an arbitrary one step method of order $p \geq 1$. Let $f \in C^p\left([t_0, t_f] \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x}\right)$ and suppose Assumption 6 holds. Moreover, let the control $u \in C^p\left([t_0, t_f]\right)$ be approximated by a discrete control $u_h \in \{u_{\min} + jh_u \mid j = 0, \dots, n_u\}$. Then, the local approximation error of the time and control discrete one step method estimates to*

$$\|x_h - x\|_\infty \leq O(h^{p+1}) + O(hh_u). \tag{4.31}$$

Proof. Quintessentially, we consider the controls as parameters of the initial value problem. Therefore, we introduce the notation $x(t; u)$ which indicates the dependency of the state at time t to the control u. Then, we have

$$\begin{aligned}
\|x(t_{\ell+1}; u) - x_h(t_{\ell+1}; u_h)\| &\leq \|x(t_{\ell+1}; u) - x(t_{\ell+1}; u_h)\| + \|x(t_{\ell+1}; u_h) - x_h(t_{\ell+1}; u_h)\| \\
&\leq \|x(t_\ell) - x(t_\ell)\| + \|\int_{t_\ell}^{t_{\ell+1}} f(u(\tau), x(\tau)) - f(u_h(\tau), x(\tau))\, d\tau\| \\
&+ \|x(t_{\ell+1}; u_h) - x_h(t_{\ell+1}; u_h)\| \\
&\leq \|x(t_\ell) - x(t_\ell)\| + L_{f,1} h \|u - u_h\|_\infty \\
&+ \|x(t_{\ell+1}; u_h) - x_h(t_{\ell+1}; u_h)\|
\end{aligned}$$

By means of the above analysis, together with $x(t_0) = x_h(t_0)$, the error bound of one step methods, cf. [Stoer and Bulirsch, 2002], as well as the estimates of Lemma 4.19 yields,

$$\|x_h - x\|_\infty \leq O(h^{p+1}) + O(hh_u).$$

□

With these estimates we can now turn to the investigation of the convergence of the value function. Therefore we require some additional notation. To this end, let

$$\begin{aligned}
F(x) &:= \sup_{u \in \mathcal{U}} \nabla \vartheta(t, z) f(u, x) \\
H(x) &:= P(x) = \sup_{u \in \mathcal{U}} P(x).
\end{aligned}$$

Moreover the following assumptions are required:

Assumption. 9 (Lipschitz Hamiltonian). Let $F(x)$ and $H(x)$ be bounded and Lipschitz continuous, i.e.,

$$\begin{aligned}
\|F(x)\|_\infty \leq M_F, \qquad & |F(x_1) - F(x_2)| \leq L_{F,1}|x_1 - x_2| \\
\|H(x)\|_\infty \leq M_H, \qquad & |H(x_1) - H(x_2)| \leq L_{H,1}|x_1 - x_2|
\end{aligned}$$

Assumption. 10 (Consistency). Suppose, for $0 < h < h_0$, that

$$\lim_{h \to 0} \Phi_F(x, h) = F(x)$$

$$\lim_{h\to 0} \Phi_H(x,h) = H(x)$$

and let,

$$|\Phi(x_1,h) - \Phi(x_2,h)| \leq L_\Phi |x_1 - x_2|$$

for all $u \in \mathcal{U}$ and a constant L_Φ independent of u.

Note that the continuity assumption on the control, i.e., $u \in C^p\left([t_0,t_f]\right)$, in the subsequent assertion is usually not viable as far as optimal control problems are concerned, yet the assumption is retained at this point for the sake of universality. The following result then is an extension of [Falcone and Ferretti, 1998, Theorem 2.2], with respect to a backward Semi-Lagrangian scheme.

Theorem 4.21. *Let Assumption 9 and Assumption 10 hold and let $u \in C^p\left([t_0,t_f]\right)$ and $f \in C^p\left([t_0,t_f]\times\mathbb{R}^{n_u}\times\mathbb{R}^{n_x}\right)$. Then for any $t_\ell \in [t_0,t_f]$, there exists a constant $C > 0$ independent of h an h_u such that*

$$|\vartheta(t_\ell,z) - \vartheta_h(t_\ell,z)|_\infty \leq C t_\ell (h^p + h_u). \tag{4.32}$$

Proof. Recalling that under Lemma 4.17 the value function is Lipschitz continuous, with Lipschitz constant L_ϑ. Hence, we have

$$\begin{aligned}
|\vartheta(t_\ell,z) - \vartheta_h(t_\ell,z)| &\leq |H(x) - H(x_h)| + |\vartheta(t_{\ell+1},x(t_{\ell+1})) - \vartheta(t_{\ell+1},x_h(t_{\ell+1}))| \\
&\leq L_{H,1}|x(t_\ell) - x_h(t_\ell)| + |\vartheta(t_{\ell+1},x(t_{\ell+1})) - \vartheta(t_{\ell+1},x_h(t_{\ell+1}))| \\
&+ |\vartheta(t_{\ell+1},x_h(t_{\ell+1})) - \vartheta_h(t_{\ell+1},x_h(t_{\ell+1}))| \\
&\leq L_{H,1}C(h^{p+1} + hh_u) + |\vartheta(t_{\ell+1},x(t_{\ell+1})) - \vartheta(t_{\ell+1},x_h(t_{\ell+1}))| \\
&+ |\vartheta(t_{\ell+1},x_h(t_{\ell+1})) - \vartheta_h(t_{\ell+1},x_h(t_{\ell+1}))|,
\end{aligned}$$

which, when employing the Lipschitz continuity of the value function yields,

$$\begin{aligned}
\|\vartheta(t_\ell,z) - \vartheta_h(t_\ell,z)\|_\infty \leq & L_{\vartheta,2}|x(t_{\ell+1}) - x_h(t_{\ell+1})| + \|\vartheta(t_{\ell+1},x_h(t_{\ell+1})) - \vartheta_h(t_{\ell+1},x_h(t_{\ell+1}))\|_\infty \\
&+ L_{H,1}\tilde{C}(h^{p+1} + hh_u).
\end{aligned}$$

Adopting the results of Lemma 4.20 and under consideration that $\|\vartheta(t_M,\cdot) - \vartheta_h(t_M,\cdot)\|_\infty = 0$ and $\ell = \frac{t_\ell}{h}$, the error estimates to

$$\|\vartheta(t_\ell,z) - \vartheta_h(t_\ell,z)\|_\infty \leq C t_\ell (h^p + h_u), \tag{4.33}$$

concluding the proof. □

As a consequence of Theorem 4.21 the time and control discrete value function ϑ_h converges for $h, h_u \to 0$ to the continuous value function ϑ, even for higher order Semi-Lagrangian schemes. Intriguingly, let us stress, that by reducing the order $\tilde{p} < p$ of the definition of the

control u, the order of the convergence rate reduces in this case from h^p to $h^{\tilde{p}}$. Hence, the order of the approximation of the control is not negligible and impacts the convergence rate severely. Moreover, for $p = 1$, the results show similarities with the findings in Chapter 3.

4.7 A Large Scale Parallel Algorithm for Dynamic Programming

In the previous section we established, that the standard approach to solve a nonconvex problem, by means of dynamic programming, is to formulate the respective value function, which is then solved recursively. Where the problem is not in bounds the value function is then set to some infinitely high value. By preceding in this manner, the interpolation leads to a reduced feasible set, due to the propagation of infinite values to neighbouring non-grid points.

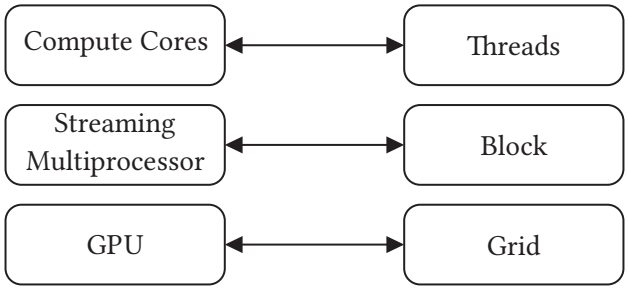
Consequently, an infinite value might not be the best strategy. Therefore, a penalty reformulation was introduced, where the constraints contribute to the objective function by means of a penalty term. Despite the suitable convergence properties and the possibility to compute global optima, an essential impediment of dynamic programming is the *curse of dimension*. That is the memory requirements and the computational effort grows exponentially with the number of states and discretization points. Thus, increasing the approximation error for small h, h_u and h_x. The cause for this exponential growth is based in the concept of dynamic programming itself, by striding through all state and control combinations. The repetitiveness of the tasks in the algorithm, however, imbues the idea of executing these tasks in parallel. The standard ADP in Algorithm 6 usually can only be parallelized to some degree, since interdependencies of the interpolation as well as the state and control loops prevent a fully immersive integration of parallelism. The purpose of this section is to present an algorithm which is fully parallelizable and due to the large matrices emanating from the dynamic programming principle and the problem dimensions, is favourably suited for computation on a GPU. First, a brief and general overview of the data structures and process flows by example of a NVIDIA GPU is given, together with some notes on techniques utilized to achieve efficient occupancy and therefore performance of the GPU architecture, where we refer to [Nvidia Corporation, 2020] for a more in depth discussion. Before stating the algorithm, a restructuring of the value function and the required state grid are introduced. Additionally, we introduce a fully parallelizable multi linear interpolation algorithm.

4.7.1 GPU Data and Process Structure

Due to major advances in hardware architecture, computational parallelism has become an indispensable asset for industry and research in recent years. For example in the area of neural networks and big data science to name a few applications. In general, there exist two branches of parallel computing distinct by the respective hardware on which the computation is conducted. More precisely, either a parallel execution on the CPU or the GPU. Each compute architecture

has its justification and entails its own advantages and disadvantages. It is well known that CPUs are comprised of a low number of compute cores and threads, while GPUs inhibit thousands of compute cores (threads). However, the single core performance of each CPU core is higher compared to the performance of a single GPU core. On that note, while each operation on a core of the CPU is processed faster the number of concurrent operations is much lower compared to a GPU, which is designed for massive parallelism. Historically this is reflected in the execution models. While CPUs followed a Single Instruction Multiple Data (SIMD) model, i.e., applying a single instruction on multiple data sets simultaneously, GPUs on the other hand follow a Single Instruction Multiple Threads (SIMT) model, where multiple threads work on different data sets, thus reducing latencies. Nowadays, with the introduction of multithreading a CPU can also employ SIMT executions to some extend. In conclusion, GPU parallel computation is the preferred method for large data structures, while low dimensional problems are more suitable for CPUs.

Since in dynamic programming naturally large data structures emanate, inevitably leading to the course of dimension, together with the repetitiveness of the operations, the transfer to a GPU parallelization stands to reason. Therefore, in the following we give a general overview of the rudimentary structures and process of a GPU before developing the algorithm. The GPU can be addressed using Nvidia's CUDA library, which provides high level instructions to ease the development and deployment of GPU parallel programs. Furthermore, CUDA provides an abstract model for the programmer to address different levels of the GPU. From the hardware point of view, a GPU is comprised of several streaming multiprocessor (SM), which in turn contain the compute cores (or CUDA cores), control units, registers and caches to name the most important parts. The abstracted CUDA equivalents are:



Before a kernel is launched the data needs to be transferred from host to device memory carefully. When the CPU then instructs the GPU to launch a kernel, the number of blocks and threads are automatically spawned. Moreover, blocks are divided into so called warps, where each warp is comprised of 32 threads. Consequently, the instructions are distributed to the threads, handling the concurrent computation. Herein, blocks can be executed concurrently or sequentially and only can share data through the global device memory. Threads of different blocks however can not cooperate. More precisely, threads of the same block can only cooperate using shared memory which needs to be instantiated manually, compare Figure 4.6.

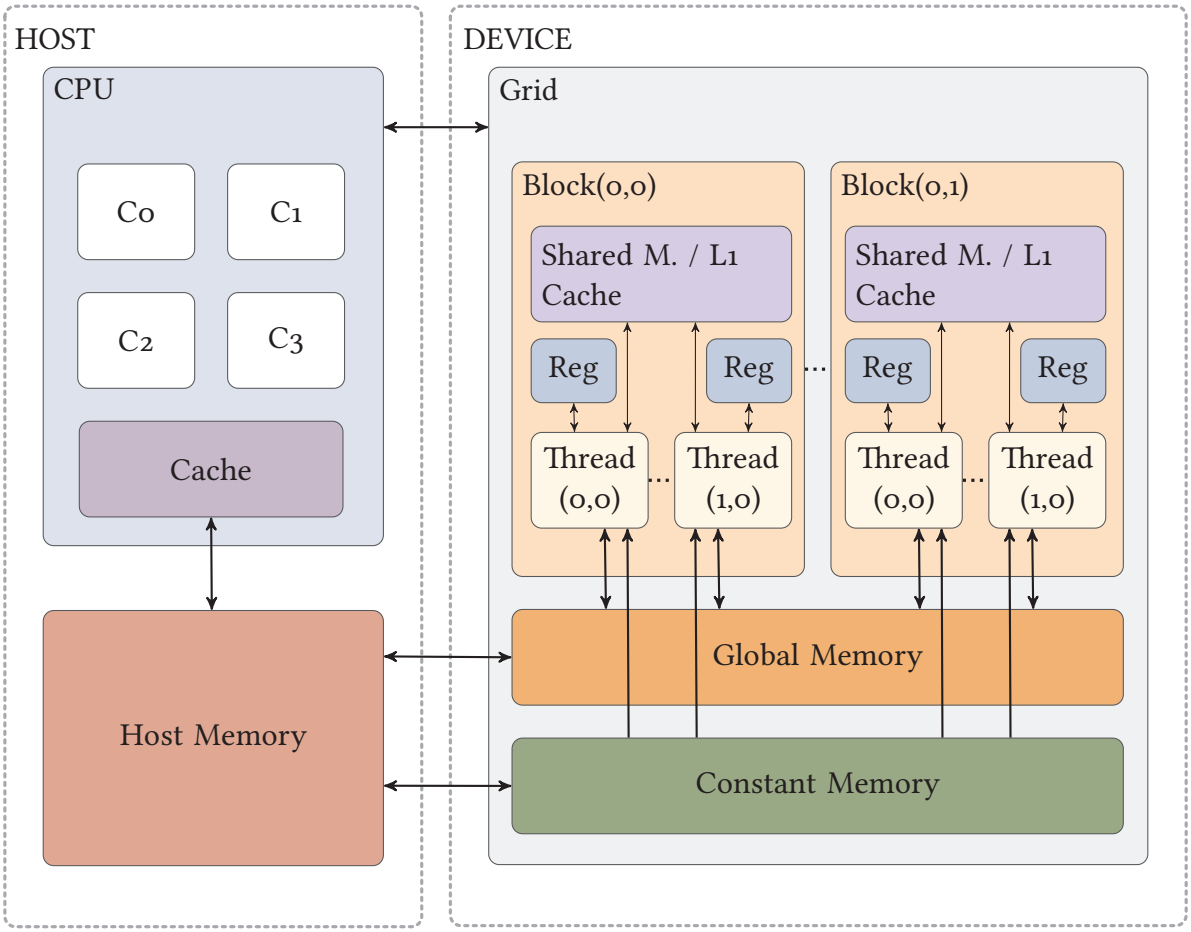


Figure 4.6: *Illustration of an abstracted memory and instruction flow between CPU (Host) and GPU (Device), cf. [Nvidia Corporation, 2020]. Memory is relayed via constant or global memory to streaming multiprocessors abstracted through blocks. Each block is composed of registers (Reg), threads and shared memory. CPU issues instructions directly to the GPU (Grid); Memory is exchanged from host memory.*

Contrary to the intuition that as many parallel threads can be executed as there are compute cores on the whole GPU, the number of parallel threads might be lower. Since within each block 32 threads are grouped to so called warps, and depending on the available number of cores per SM warps might be executed sequentially. A similar allocation is possible to occur for blocks. When the computation is finished the result needs to be transferred back to the host memory. Let us stress, that while the GPU is processing, by using asynchronous instructions, the control can be returned to the host, which in turn can launch further instructions.

Consequently, a CUDA program cannot function independent of the CPU. It is always embedded into a C++ or Python program, where the CPU provides instructions to the GPU to launch kernels, which are the equivalent to function calls. Compared to plain C or C++ programs, where most of the memory management is done by the compiler, on GPUs there exist different types of

memory, in particular, global, local, texture, shared memory and registers. The access of memory needs to be handled carefully to maximize the occupancy, minimize latencies and ensure validity of the computation or more concisely to maximize efficiency of the tasks.

In general, both the CPU (host) and the GPU (device) have their distinct Dynamic Random Access Memory (DRAM), i.e., host and device memory respectively. To execute a kernel on the device the data first has to be transferred from the host memory to the device memory via the PCIe Bus. As already established, on the device there exist different types of memory, here a high-level distinction is drawn between on-chip and off-chip memory. Global memory is off-chip memory and resides in the DRAM and is initialised from the host memory. It can be accessed by transactions of $32, 64$ or 128 bytes, which must be aligned to be read or written properly, otherwise the bandwidth is reduced. Overall, the more transactions calls are necessary to acquire the data for a specific computation the more unused data is transferred by each thread.

Example 4.22. Suppose a 128 byte access is issued by each thread of 4 bytes, then the effective bandwidth is divided by $\frac{128}{4} = 32$. Since the required 4 byte access entails an additional 124 byte transaction.

Consequently, a coalesced memory access is required for performance and validity of the result. The simplest way to achieve this is to access consecutive data points by consecutive threads, provided the memory is aligned, which is the case for variables in global memory up to 256 bytes. Let us stress that a strided access with an offset > 1 inevitably reduces the bandwidth, compare Figure A.6. To further improve the bandwidth of the data transfer between host and device it is possible to use pinned or managed memory. Once declared pinned memory behaves like a single coherent memory image, where the address space is shared. Consequently reducing the latency otherwise induced by copying to device memory and back. However, pinned memory is limited and needs to be handled with care.

Another important technique which requires pinned memory is to employ asynchronous and overlapping memory transfers with computation. By using an asynchronous transfer once the transaction is issued the control of the program immediately returns to the host thread which initiated the call. Hence, the host code commences to be executed. This together with multiple stream structures enables to interleave and overlap device and host computation as well as data transfer operations, hiding the major latencies of the data transfers to some extend.

Furthermore, we can use shared memory which operates at bandwidth speeds similar to registers. Shared memory is unique to blocks, and allows different threads of one block to cooperate and share data. It is divided into memory modules of equal size, which are called banks, and can be addressed simultaneously. For compute capabilities of version 5.x or higher there are usually 32 banks, where per bank one address request can be handled. In the case where multiple requests try to access the same bank, the accesses are serialized and a bank conflict emanates. This in return reduces the effective bandwidth. Thus, the memory requests need to be scheduled according to the address mapping of the banks. A bank conflict can for example be avoided by either padding the data or by using linear indexing.

Example 4.23. Let us illustrate the concept of a bank conflict by means of a 4x2 matrix:

$$\begin{pmatrix} 4 & 5 & 6 & 7 \\ 1 & 3 & 2 & 8 \end{pmatrix}$$

Assume further for the sake of simplicity that we only have four banks and the matrix is stored consecutively in shared memory as shown in Table 4.1

Table 4.1: *Distribution of a Matrix with respect to Banks in shared memory.*

bank	0	1	2	3
value	4	5	6	7
	1	3	2	8
address	0-3	4-7	8-11	12-15
	16-19	20-23	24-27	28-31

If we want to access each element by linear indexing, i.e, the k-th thread accesses the k-th element we get no bank conflicts. However if we access the data across threads in a warp, for example by indexing with the thread identifiers times 2, a bank conflict occurs. More precisely, thread 0 accesses the address 0 which resides in bank 0, but at the same time thread 8 wants concurrent access to address 16 which also is located in bank 0. Thus, two threads want to access the same bank at the same time.

A benchmark of different implementations of a matrix-matrix multiplication can be found in Appendix A.5, emphasizing the impact on computation times if the GPU is not efficiently occupied.

4.7.2 Algorithm

As established in the previous section, it is suitable to provide larger matrices to the GPU, to ensure occupancy and capitalize on the massive parallelism. More precisely, larger matrices allow for an efficient distribution of data on the GPU, by incorporating the techniques presented in the previous section, which is complemented by a batching strategy in the software package implementing the provided parallel dynamic programming algorithm. The purpose of this section is to develop an algorithm exploiting the structure emanating from ADP. Therefore, the emphasis in this section will rest on the optimal control problem rather than on the game structure. Thus,

to alleviate the notation, the index of the individual player v is neglected. To this end let us recall the penalized problem,

$$\min_{\varkappa} \quad \varphi(\varkappa_h(t_M)) + \rho P(x_h) \qquad \text{s.t.} \quad \varkappa_h \in X_h$$

The problem structure, i.e, the recursion in approximate dynamic programming, can be taken advantage of to render the computation speed essentially dependent on the speed of element wise matrix-matrix and matrix-vector multiplications. With the structure exploitation presented here together with the possible use of parallelism, the impact of the curse of dimensions on the computational effort can be reduced vastly. This is achieved by two major modifications:

(1) First, by reformulating the problems recursion such that the state grid operations are pooled in large matrix structures, which favorably can be handled with massive parallelization. Let us stress, that additionally the resulting matrix can be completely precomputed and as such does not significantly contribute to the runtime of the algorithm later on. On that note, evoking that M_x^i indicates the number of discrete points of the i-th state, we then can define a substitute

$$\overline{p} := \bigtimes_{i=1}^{n_x} M_x^i = M_x^1 \times M_x^2 \times \ldots \times M_x^{n_x}$$

for the state dimensions and

$$\overline{q} := \bigtimes_{i=1}^{n_u} M_u^i = M_u^1 \times M_u^2 \times \ldots \times M_u^{n_u}$$

accordingly for the controls. Furthermore, let the function of the right hand side of the dynamics be restructured by $f_i : \mathbb{R}^{\overline{p}\times\overline{q}} \mapsto \mathbb{R}^{\overline{p}\times\overline{q}}$, where $\mathbf{X} := (\mathbf{X}_1, \ldots, \mathbf{X}_{n_x})$ is a vector of matrices associated with the state grid (state-grid matrices). Each state-grid matrix $\mathbf{X}_i = (x^l) \in \mathbb{R}^{\overline{p}\times\overline{q}}$, with the index vector $l = (l_1, \ldots, l_{n_x})$, contains the state grid point vectors. That is for every $i = 1, \ldots, n_x$,

$$x_h^{\{l|_j^i\}} = x_{\min}^i + jh_x^i, \qquad j = 0, \ldots, (M_x^i - 1).$$

Herein, the expression $\{l|_j^i\}$ describes the matrix element at the position given by the multi-index vector l, where the entries along the i-th dimension are set by the propagation of j. Suppose t_ℓ indicates the initial date, in conjunction with the state-grid matrices, the evolution of each state can be computed for the whole permuted and discrete state space and all dates $\ell = 0, \ldots, M$, by employing the Hadamard product:

$$\mathbf{X}^+ = \mathbf{X}(t_\ell) + h \circ f(\mathbf{X}(t_\ell), \mathbf{u}). \tag{4.34}$$

In particular, the Hadamard product is an element-wise multiplication, i.e., given arbitrary matrices $A = (a_{ij}) \in \mathbb{R}^{m\times n}$ and $B = (b_{ij}) \in \mathbb{R}^{m\times n}$, the product is defined as

$$A \circ B = (a_{ij} \cdot b_{ij})$$

for all $1 \leq i \leq m,\ 1 \leq j \leq n$ and $m, n \in \mathbb{N}$.
To further, unburden the notation, the explicit time dependency of the initial date is now indicated by the respective index $\mathbf{X}(t_\ell) \coloneqq \mathbf{X}_\ell$. Thus, (4.34) yields the vector of the propagated states,

$$\mathbf{X}^+ \coloneqq (\mathbf{X}_1^+, \dots, \mathbf{X}_{n_x}^+)^T = (\mathbf{X}_1^\ell, \dots, \mathbf{X}_{n_x}^\ell)^T + h \circ \boldsymbol{\mathcal{F}}(\mathbf{X}_\ell, \mathbf{u}),$$

where $\boldsymbol{\mathcal{F}}(\mathbf{X}_\ell, \mathbf{u}) \coloneqq (f^1(\mathbf{X}_\ell, \mathbf{u}), \dots, f^{n_x}(\mathbf{X}_\ell, \mathbf{u}))^T$. Hence, attaining a large and dense matrix containing the information of all possible propagations of the discrete state-space. More importantly this matrix can be precomputed and is independent of the iterative alterations of the dynamic programming algorithm. On the downside, however, when implementing this structure the required storage increases proportionally to the number of grid points.

(2) Second, exploiting the penalty problem formulation, which can be reformulated to have an identical structure as the state grid matrices, to establish a parallel interpolation scheme and value function evaluation. Consequently, concurrency solely remains in the temporal recursion. As established in Section 4.6.2, the value function is recursively computed on the grid points, by minimization with respect to the control. By implication, we can define the respective matrix of the value function

$$\Theta \coloneqq \left(\vartheta_{ij}\right) \in \mathbb{R}^{\bar{p}\times M}, \qquad \text{for} \quad i = 1, \dots, n_x,\ j = 0, \dots, M$$

which, similar to the state propagation matrix, is large and densely populated. Moreover, at time instance ℓ, let the ℓ-th dimension be expressed by $\Theta(t_\ell, \mathbf{X}) = (\vartheta_i) \in \mathbb{R}^{\bar{p}}$. Accordingly, the matrix corresponding to the approximate value function in (4.24) is represented by $\overline{\Theta}$ and respectively $\overline{\Theta}(t_\ell, \mathbf{X})$. The approximate value function $\overline{\vartheta}$ in (4.22), is defined as the convex combination of the 2^{n_x} grid points neighbouring a non-grid point $x = (x_1, \dots, x_{n_x})^\top$. In this setting, the weights $\alpha_j \geq 0$ and $\sum_{j=1}^{2^{n_x}} \alpha_j = 1$ can be chosen arbitrarily, thus enfolding various interpolation methods.
In the face of online computation, which requires low latencies, linear interpolation is the method of choice, due to its low computational complexity. Let us elaborate in more detail on the interpolation. Therefore, we introduce some functions which we will utilize to compute, for an arbitrary number of dimensions, a linear interpolation. First, however let us state the GPU algorithm for the multi linear interpolation before we discuss its components:

Algorithm 7: Multi-dimensional linear interpolation

(S.0) Compute for each state $i = 1, \ldots, n_x$ lowest neighbouring indices $\mathcal{I}_i$ in Θ, and the corresponding weights $\mathcal{W}_i$

$$\mathcal{I}_i = \max\left\{0, \min\left\{\left\lfloor \frac{\mathbf{X}_i - x^i_{\min}}{h^i_x} \right\rfloor, M^i_x\right\}\right\}, \quad \mathcal{W}_i = \frac{\mathbf{X}_i - x^i_{\min}}{h^i_x} - \mathcal{I}_i$$

(S.1) Compute the n_x-dimensional permutation of the 2-tuple $\{0, 1\}$,

$$p(n_x) = \mathrm{perm}[\{0, 1\}, n_x].$$

(S.2) Transform the index set to linear indices,

$$\mathcal{I}_{lin} = \mathcal{I}_1 + \sum_{k=2}^{2^{n_x}} (\mathcal{I}_k - 1) \prod_{i=1}^{k-1} M^i_x$$

(S.3) For every $k = 1, \ldots, 2^{n_x}$,

$$\Omega_k := \prod_{n=1}^{n_x} (\mathcal{W}_n)^{p_{kn}} \circ (1 - \mathcal{W}_n)^{1-p_{kn}}.$$
$$\overline{\Theta} = \overline{\Theta} + \Omega_k \circ \Theta(\mathcal{I}_{lin})$$

Reset weights, $\Omega_k \leftarrow \mathbb{1}$.

Interpolation is necessary if the evolution of the dynamics leads to a non-grid point, i.e., $y \notin \mathbb{G}_x$. Since, the value function is only known at all $x_h \in \mathbb{G}_x$. Hence, the indices of the lowest proximate neighbour of a given non-grid point can be computed by,

$$\mathcal{I}_i = \max\left\{0, \min\left\{\left\lfloor \frac{\mathbf{X}_i - x^i_{\min}}{h^i_x} \right\rfloor, M^i_x\right\}\right\}, \qquad \text{for each} \quad i = 1, \ldots, n_x.$$

The contribution of each grid-point to the interpolated value is weighted by its linear distance,

$$\mathcal{W}_i = \frac{\mathbf{X} - x_{\min}}{h^i_x} - \mathcal{I}_i, \qquad \text{for each} \quad i = 1, \ldots, n_x.$$

Further, let us introduce the mapping $p : \mathbb{N} \to \mathcal{P}_{n_x \times 2^{n_x}}(\{0, 1\})$, which maps to a matrix $\mathcal{P}$ containing all n_x-dimensional permutations of elements $\lambda \in \{0, 1\}$, such that

$$p(n_x) := \mathrm{perm}\left[\{0, 1\}, n_x\right]. \tag{4.35}$$

As an example, to put the behaviour of (4.35) into a more comprehensible context, suppose $n_x = 3$, then,

$$p(3) = \operatorname{perm}[\{0,1\},3] = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}^\top.$$

Using (4.35), for arbitrary dimensions n_x, the cumulative weights for all $k = 1, \dots, 2^{n_x}$ are defined by,

$$\Omega_k := \prod_{n=1}^{n_x} (\mathcal{W}_n)^{p_{kn}} \circ (1 - \mathcal{W}_n)^{1-p_{kn}}.$$

The interpolated value function at the non grid-point then becomes,

$$\overline{\Theta}(t_\ell, \Gamma_k[y]) := \sum_{k=1}^{2^{n_x}} \Omega_k \Theta(\mathcal{I}). \tag{4.36}$$

Consequently, establishing a large dense matrix, containing the values of the interpolated value function. Eventually, it remains to minimize (4.36) over the control and reconstruct the solution by means of value iteration.

Remark 4.24. *Computationally it is more convenient to use linear indexing. For an arbitrary number of dimensions the linear index can be computed by*

$$\mathcal{I}_{lin} := \mathcal{I}_1 + \sum_{k=2}^{2^{n_x}} (\mathcal{I}_k - 1) \prod_{i=1}^{k-1} M_x^i.$$

Offering the possibility of consecutive memory access, which is concomitant with the GPUs intended storage architecture. Moreover, the matrix structures are additionally reordered to have a tridiagonal structure on the GPU to leverage the memory structure.

In compliance with Lemma 4.17 for any grid point $z_\ell \in \mathbb{G}_x$ at $t_\ell \in \{0, \dots, M\}$ the approximation error of the value function estimates to

$$|\tilde{\vartheta}_h(t_\ell, z_\ell^v) - \vartheta_h(t_\ell, z_\ell^v)| \leq L_\vartheta (M + 1 - \ell) \|h_x\|.$$

A detailed proof for the approximation error using multi linear interpolation can be found in Appendix A.3.

Using the reformulations, we can now state the *Parallel Semi-Lagrangian Dynamic Programming* algorithm (PARSEL-DP).

Algorithm 8: Parallel Semi-Lagrangian Dynamic Programming [PARSEL-DP]

(S.0) Precompute the evolution of the state grid matrices $\mathbf{X}_{\ell+1}$, for every $i = 1, \dots, n_x$

$$\mathbf{X}^i_{\ell+1} = \mathbf{X}^i_\ell + h \circ \mathcal{F}(\mathbf{X}^i_\ell, u).$$

(S.1) Set $\overline{\Theta}(t_M, \mathbf{X}) := \Phi(\mathbf{X}(t_M))$

(S.2) For every $k = M-1, \dots, 0$, compute the minimum over the interpolated value function

$$\overline{\Theta}(t_\ell, \mathbf{X}) := \min_{(\mathbf{X}) \in X(t_\ell, \mathbf{X})} \{P_\ell(\mathbf{X}) + \overline{\Theta}(t_{\ell+1}, \Pi_j[\mathcal{F}(\mathbf{X}, u)])\}.$$

(S.3) Set $\tilde{x}(0) = x_0$, and $k \leftarrow 0$, then for $k = 0, \dots, M-1$ compute

$$u_\ell = \operatorname*{argmin}_{(\tilde{x}, u) \in X(t_\ell, \tilde{x}_\ell)} \left[P_\ell(\tilde{x}) + \overline{\Theta}(t_{\ell+1}, \Pi_j[\theta(\tilde{x}_\ell, u_\ell)]) \right]$$

With Algorithm 8 the speed of the computation only relies on the speed of matrix multiplications which now can as well as the preprocessing of the dynamics be parallelized, since the interdependences of neighbouring time instances and constraints are no longer present. Even the interpolation can now be computed in parallel. Regarding the computation on GPUs, with a Nvidia GTX 1060, which provides 1280 CUDA cores, each core can operate in parallel, hence, we can achieve a significant faster computation, which will be evaluated in the subsequent section. Moreover, this allows to consider problems of higher dimension to be computed efficiently with dynamic programming.

4.7.3 Performance Evaluation

The purpose of this subsection is to vindicate that the performance of the proposed Algorithm 8 excels for larger dimensions compared to a common implementation on a CPU. Therefore, the well-known Lotka-Volterra fishing problem is used as an exemplary model problem:

$$\begin{aligned}
\min_{x,u} \quad & \int_{t_0}^{t_f} (x_0(\tau) - 1)^2 + (x_1(\tau) - 1)^2 \; d\tau \\
\text{s.t.} \quad & x_0'(t) = x_0(t)\,(1 - x_1(t) - c_0 u(t))\,, \\
& x_1'(t) = x_1(t)\,(x_0(t) - c_1 u(t) - 1)\,, \\
& x(0) = (0.5, 0.7)^\top\,, \\
& u(t) \in \mathbb{U}.
\end{aligned}$$

The intention of the Lotka-Volterra fishing problem is to identify an optimal fishing strategy for a fixed time horizon, leading the biomasses of predator and prey, i.e., the differential states x_0 and x_1 respectively, to a steady state. Suppose we have a fleet of fishing vessels, then the control u of the problem indicates if the fishermen have to fish or not. Indeed, $u \in \mathbb{U}$ can be treated as a relaxed or binary control. In the discrete problem, the relaxed control $u_h \in \breve{\mathbb{U}} := \{u_{\min} + jh_u \mid j = 0, \dots, M_u\}$ indicates which contingent of the fleet has to fish, whereas a binary control $u_h \in \{0, 1\}$ either indicates that all vessels fish or none.

Next, to render the problem formulation coherent with (NEP$_h$), the objective function is transformed into a Mayer term, by introducing an additional state x_2. Hence, obtaining the discrete optimal control control problem.

$$\min_{x,u} \quad \varphi_h(x(t_f)) = x_2(t_f) \qquad \text{s.t.} \quad (u_h, x_h) \in X^h$$

where the discrete state vector is defined by $x_h = (x_0, x_1, x_2)^\top$ and u_h denotes the discrete control. Note that, the state space has order $n_x = 3$ and the control space is of order $n_u = 1$. Then, the discrete feasible set is accordingly defined by

$$\begin{aligned} X^h = \big\{ (u_h, x_h) \mid\ & x(0) = (0.5, 0.7, 0)^\top,\ u(t_\ell) \in \{0, 1\}, \quad \text{and } \forall \ell = 0, \dots, M-1 : \\ & x_0(t) = x_0(t_\ell) + (t - t_\ell) x_0(t_\ell)\,(1 - x_1(t_\ell) - c_0 u(t_\ell)), \forall t \in [t_\ell, t_{\ell+1}] \\ & x_1(t) = x_1(t_\ell) + (t - t_\ell) x_1(t_\ell)\,(x_0(t_\ell) - c_1 u(t_\ell) - 1), \forall t \in [t_\ell, t_{\ell+1}] \\ & x_2(t) = x_2(t_\ell) + (t - t_\ell)\left[(x_0(t_\ell) - 1)^2 + (x_1(t_\ell) - 1)^2\right], \forall t \in [t_\ell, t_{\ell+1}] \big\}. \end{aligned}$$

Moreover, we have the positive constants $c_0 = 0.5$ and $c_1 = 0.7$. All calculations were performed on a machine with the specifications given in Table A.1.

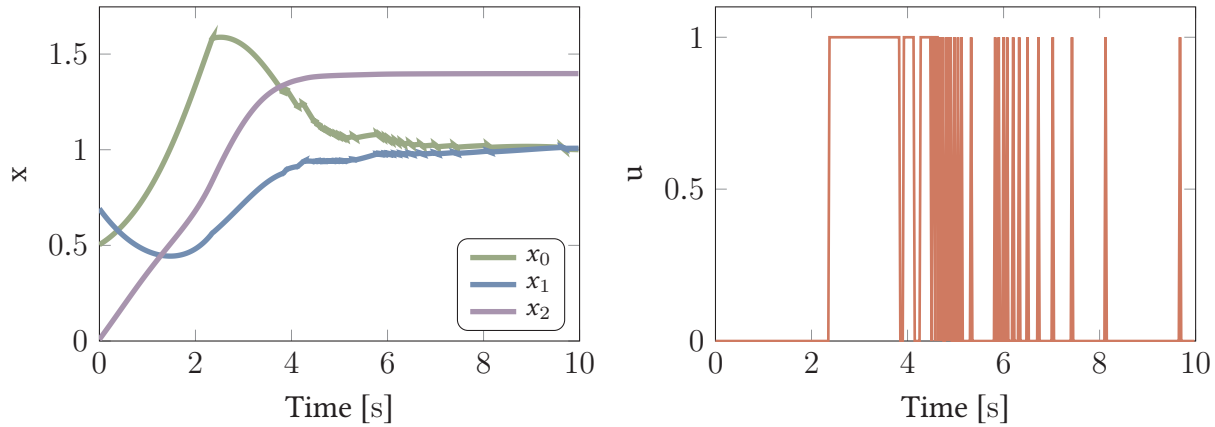


Figure 4.7: *Optimal solution of the state (left) and the control (right) trajectories of the Lotka-Volterra problem using Algorithm 8.*

The time horizon is set to $t_f = 10\,\mathrm{s}$ with initial time $t_0 = 0\,\mathrm{s}$. For the discretization we choose $M = 400$, and for all $i = 0, 1, 2$ the differential states are discretized in $M_x^i = 100$ steps. In this

setting, the optimal objective value reached is $x_2(t_f) = 1.39798$ and the steady state values are $x_0(t_f) = 1.00019$ and $x_0(t_f) = 1.00769$ respectively. The respective solutions, computed by the GPU implementation, are shown in Figure 4.7.

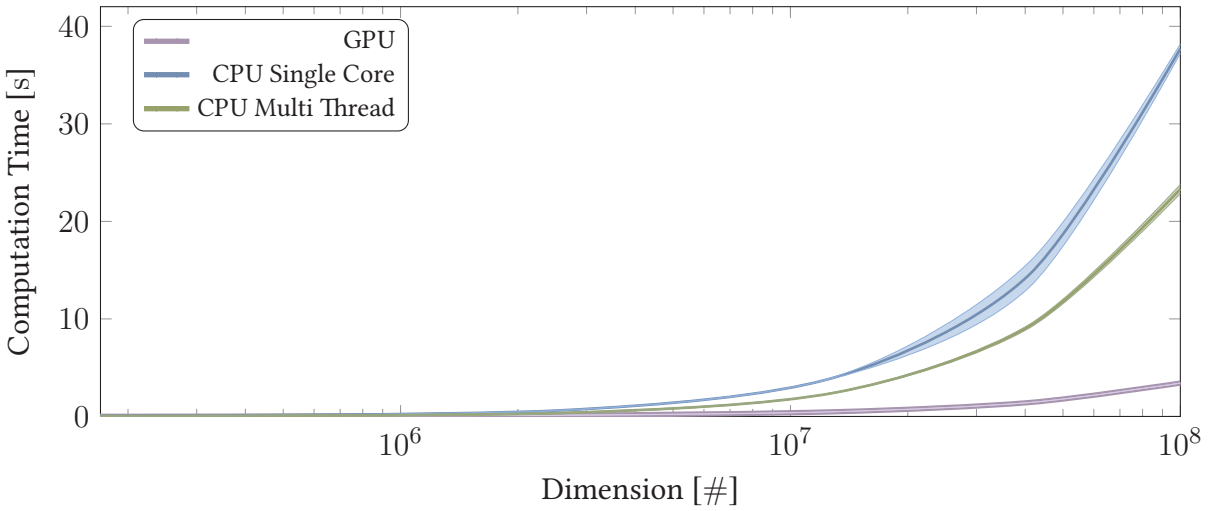


Figure 4.8: *Comparison of the computation times for Algorithm 8, for increasing problem dimensions, executed on GPU and CPU with multithreading (MT) and single core (SC) execution. The standard deviation is displayed as a shaded area around the mean value.*

To evaluate the computational performance, Algorithm 8 is implemented for deployment on a GPU using CUDA as well as in a C++ program for CPU. Computations on the CPU were performed under both single-core and multi-threaded (6 cores and 12 threads) conditions. The computation times for increasing problem dimensions are depicted in Figure 4.8. It becomes apparent that with increasing dimension the GPU dynamic programming implementation yields better performance by an order of magnitude. Even though the GPU implementation also is faster by a factor of approximately 6 compared to the multi-core CPU implementation, for lower dimensions the difference is not as significant, especially regarding the multi-threaded implementation (Table 4.2). Moreover, it is discernible that the standard deviation of the GPU implementation remains consistently low, indicating stability of the performance.

Table 4.2: *Performance metrics of Algorithm 8 for CPU and GPU execution.*

Dimension	Computation Times			Standard Deviation		
	CPU SC	CPU MT	GPU	CPU SC	CPU MT	GPU
16000	0.053	0.037	0.0115	0.0223	0.0205	5.744×10^{-4}
2560000	0.619	0.348	0.125	0.0254	0.0565	0.0149
13000000	3.993	2.429	0.481	0.0606	0.0412	0.0014
41000000	14.53	9.244	1.425	1.2964	0.1849	0.0049
100000000	37.71	23.315	3.420	0.4828	0.4613	7.264×10^{-4}

5 Application to vehicle coordination

> *However beautiful the strategy, you should occasionally look at the results.*
>
> — *Winston S. Churchill*

Conflict avoidance employing multiple autonomous robotic vehicles, in particular collision avoidance, is one of the most crucial aspects of automation, to ensure the safety and security of those involved. Motivating the formulation of coordination and path following problems. Conflicts, consequently arise when the paths or workspaces of individual agents intersect. A prominent example is the intersection management problem, where every vehicle attempts to cross the intersection as fast as possible while avoiding collisions. Consequently rendering the problem of every individual vehicle dependent on the states and strategies of the other vehicles.

Given that the vehicles are able to communicate with each other, which is attributed to the premise of autonomy, either a centralized or decentralized coordination approach can be followed. In the first approach a central controller unit coordinates the vehicles enforcing prescribed rules, resulting in a hierarchy. Whereas the decentralized approach engages the problem on the vehicle level, emanating in a differential game. Here, the vehicles act as players in pursuit of an equilibrium, where no one can further improve if the others stand by their strategies. Depending on the particular traffic scenario the differential game can be viewed as either antagonistic, cooperative or leader-follower game. If the vehicles behave selfish and do not communicate with others, the antagonistic model is most suitable. On the contrary in the cooperative case a Pareto optimal point is often the desired solution. By design the leader-follower game leads to prescribed hierarchies among the players, which however, is desirable in platooning applications. In comparison with prescribed rules, using a game theoretical approach or even adaptive traffic rules, potentially yields lower costs and ameliorate efficiency. Indeed, prescribing hierarchies, regardless of the current states and strategies, implicates a biased solution of the coordination problem. Eventually, impeding particular agents.

The main intricacy, however, is imposed by the nonconvexities emanating from the collision constraints. In [Katriniok et al., 2017] , semi definite programming is used to handle the collision

constraint, resulting in a rank constraint which is dropped in the relaxed problem. A different approach is suggested in [Assellaou, 2015, Altarovici et al., 2013]. Here an auxiliary value function of the continuous problem is considered to handle the state constraints. In the present chapter the penalty approach presented in Section 3.2 is used to shift the nonconvex constraints into the cost function, assuring that the suproblems of the players are always solvable. Due to the communication among the vehicles we have a game of perfect information. To solve the coordination problem and render online computation possible a model predictive control framework is utilized. As suggested in [Dreves and Gerdts, 2018] in each step of the MPC framework the problem can be modeled as a GNEP. Hence, Algorithm 5 is used to solve the emanating generalized potential game. Therein, each vehicle needs to solve an optimal control problem. Dynamic programmming is employed to solve the individual players problem, which are coupled through the nonconvex anti-collision constraints. A Nash equilibrium then is a strategy where a collision free transition of all vehicles through the intersection is achieved and no individual vehicle can improve its objective by unilaterally deviating from its strategy, while the others stick to theirs.

The purpose of this chapter is to fruitfully combine the penalty reformulation and the decomposition method for nonconvex GNEP with approximate dynamic programming and nonlinear model predictive control (NMPC) and apply it to the traffic coordination problem. Therefore, we give a brief outline of MPC. Then, we will introduce two optimal control problems. One associated with a low dimensional vehicle model, which will be used to compute the velocity profiles the other with a truck type vehicle model. Next, we will introduce two penalty terms associated with the anti collision constraints. Furthermore, we show that the convergence theory of the previous sections also applies to the coordination OCP. The chapter will be concluded by numerical simulations and their evaluations. The results of this chapter were published first as a numerical study in [Britzelmeier et al., 2019], followed by the convergence proofs in [Britzelmeier and Dreves, 2020].

5.1 Model Predictive Control

As implied by its name, model predictive control (MPC), cf. [Grüne and Pannek, 2011], is based on the computation of optimal controls using a mathematical substitute model—linear or nonlinear—of a process to predict and optimize the future system behaviour. Thereby the computation of optimal controls results from repeatedly solving the optimal control problem, for a given preview horizon $M \in \mathbb{N}$ and the associated path segment. Thus, in every iteration of the MPC scheme, a predictive control trajectory consisting of M control signals is obtained. A partition, i.e., $m < M$, of the obtained controls are eventually adapted to the real system. In the classic MPC scheme only the first control of the computed sequence, i.e., $m = 1$ is applied to the system. Let us now elaborate in more detail on this. To this end, we adduce our model problem (NEP_h), representing a control system in discrete time, where the time instances are denoted by $t_\ell = \ell h$, with a fixed sampling frequency $f = \frac{1}{h}$. Herein, the state $x_h(t_\ell)$ (e.g. velocity, position etc.) of

the dynamic system can be influenced by the control input $u_h(t_\ell)$, to achieve a desired behavior. Therefore, the MPC scheme requires to solve the OCP on a moving time horizon $[t_\ell, t_{\ell+M}]$. The resulting control sequence $\hat{u}_h = \{u_h(t_\ell), \dots, u_h(t_{\ell+M-1})\}$, then allows us to define the feedback control law $\pi(x_h(t_\ell)) = u_h(t_\ell)$, where $\pi : X \to \mathcal{U}$.

Remark 5.1. *For tracking problems or in general for nonautonomous optimal control problems, the feedback control law is defined by $\pi(t_\ell, x_h(t_\ell)) = u_h(t_\ell)$, where $\pi : \{t_\ell, \dots, t_{\ell+M}\} \times X \to \mathcal{U}$, i.e., explicitly dependent on time.*

Applying the feedback control to the dynamic system consequently yields a closed loop system.

$$\begin{aligned} x(t_0) &= x_0 \\ x(t_{\ell+1}) &= f(\pi(x_h(t_\ell)), x_h(t_\ell)). \end{aligned}$$

Subsequently, by shifting the temporal horizon by τ, together with the current state measurement as the new initial value, the process is repeated. Thus, a new control input is determined in each iteration, allowing the MPC algorithm to advance in time with the system. Additionally, due to the closed loop control, the system is able to manage state perturbations. Moreover, in the context of autonomous vehicle coordination, the principle of MPC control allows the sudden appearance of obstacles, such as non-cooperating vehicles, to be considered and acted on. Next, let us state the algorithm:

Algorithm 9: Nonlinear Model-Predictive Control

(S.0) Measure or estimate current state $x(t_\ell)$ at time t_ℓ.

(S.1) Set $x_0 := x(t_\ell)$. On the time horizon $[t_\ell, t_{\ell+M}]$ solve the discrete time optimal control problem: Let $\hat{u}(t_\ell), \dots, \hat{u}(t_{\ell+M-1})$ delineate an optimal control sequence.

(S.2) Define the feedback control law $\pi_M(x(t_\ell)) := \hat{u}(t_\ell)$ and apply it:

$$x(t_{\ell+1}) = f\left(\pi_M(x(t_\ell)), x(t_\ell)\right)$$

(S.3) Set $\ell \leftarrow \ell + 1$ and go to (S.0).

Remark 5.2. *Depending on the chosen preview horizon and quality of the model representing the physical system, the accuracy of the system in tracking the computed trajectory can vary significantly. Another factor influencing the control performance is the computation time. Solving the OCP in each iteration of the MPC is expensive and requires time, thus resulting in a delay in deploying the next control input to the system. Another related problem arises if solving the OCP fails. For both of these problems a fallback solution needs to be provided. In [Grüne and Pannek, 2011] and [Rawlings and Mayne, 2009] various modifications of the classical MPC scheme are discussed to ensure the provision of the next control input to the system in due time. The application*

of a multi step MPC is one such approach. Herein, without new measurements $m \leq M$, with $m > 1$ controls are applied to the system, shifting the system into an open loop control for m steps. Additionally, re-optimization on the remaining intervals $[\ell + j, \ell + M]$ under consideration of new state measurement can be used to improve the accuracy of the multi step MPC, cf. [Gerdts, 2018]. Alternatively, parametric sensitivity updates can be incorporated, avoiding the successive solution of the OCP, see [Zavala et al., 2008]. Moreover, a reduction of the sampling time can be achieved employing the real time iteration scheme introduced in [Diehl et al., 2005], which is based on direct multiple shooting and exploits the serration of the solution iterations along the evolution of the process.

An important virtue of this control scheme is that the OCP only needs to be solved on the complete temporal horizon T, consequently reducing the computation time. With regard to dynamic programming, which is burdened by the course of dimensions, this implicates favorable synergistic effects. The state of a dynamic system is inevitably dependent on time and likewise, for a given time period T, the reachable states are bounded. Accordingly, within each iteration of the MPC scheme, a reduced state space can be considered. Therefore, in the discrete problem, to establish the same sampling resolution of the state space, less discretization points M_x are required. Thus, attenuating the curse of dimensions to some degree and resulting in faster computation times. Model predictive control, hence, can be beneficially used in combination with dynamic programming, even though dynamic programming is usually considered to slow for online control.

5.2 Vehicle Dynamics

In this section, vehicle models are introduced that are suitable for the control of planar vehicle motions and are applied in this thesis. When coordinating vehicles, the centre of attention is focused on collision avoidance, which can be established in one of two ways. Either by means of spatial evasion using path planning schemes or otherwise temporally in terms of velocity profiles. The required vehicle models for the respective tasks differ significantly. Whilst path planning is indubitably connected to lateral motion control, coordination by means of velocity control can be realised using longitudinal models. The kinematic models discussed here can be found in [Mitschke, 2014] [Rill, 2011] and for the motion in curvilinear coordinates see [Lot and Biral, 2014]. For the longitudinal point-mass model we refer to [Frego et al., 2016]. Additionally, for a comprehensive and in-depth survey on vehicle dynamics we refer to [Gillespie, 1992], [Jazar, 2017] and [Rajamani, 2012].

5.2.1 Equations of Motion

In the literature, a primary distinction is made between three vehicle models of increasing complexity, the kinematic model, the single-track model and the full vehicle model, see [Rajamani,

2012],[Gillespie, 1992],[Mitschke, 2014]. A simple kinematic model generally provides sufficient accuracy for online computations, cf. [Gerdts, 2018]. As its name suggests, the kinematic model only takes into account kinematic characteristics rather than dynamic aspects of the vehicle's behavior, such as slip or tire forces. These in turn are reflected in the single-track and full-vehicle models. The full vehicle is considered as a multi body system considering not only the lateral and longitudinal but also the vertical dynamic. This allows the investigation of dynamic loads and vibrations, at the cost of a higher dimensional problem, compare [Rill, 2011, Gerdts, 2003].

In the context of dynamic programming, we are interested in developing a low dimensional model to keep the effect of the curse of dimensions contained. Therefore, in this section we will derive a kinematic model, both for a passenger car and a truck. By further assumptions it is demonstrated in Section 5.2.2, how the passenger car model, henceforth denoted as vehicle model, can be further simplified to a simple point-mass model for the computation of velocity profiles.

The kinematic model, following [Rill, 2011], provides a mathematical description of the vehicle motion but does not take into account the external forces that may afflict the motion. Hence, the equations of motion are derived from the geometric coherences of the mechanical system. In general the vehicle motion can be described in various coordinate systems, like polar coordinates, curvilinear coordinates or Cartesian coordinates to name a few. However, the most commonly used are Cartesian coordinates as a reference system. More precisely, the position and orientation of the vehicle are described in an earth fixed system, which provides an inertial reference frame (x_I, y_I, z_I). By convention the z_I-axis of the inertial system is oriented upward, i.e., the opposing direction of the gravity vector g. Relative to the inertial coordinate system, the car's fixed, local coordinate system (x_C, y_C, z_C) resides in it's center of mass. In accordance with *DIN ISO 8855:2013-11*, cf. [DIN 8505, 2013], the x-axis of the local system is oriented forward in the direction of the longitudinal motion, the y-axis perpendicular to the left and the z-axis points upwards as in the inertial system. The orientation between the two reference frames can be described by the yaw, pitch and roll motion angles, respectively ψ, θ and ϕ.

Remark 5.3. *Let us elaborate in more detail on the choice of origin of the local reference system:*

- *Any point on the chassis can be used as the origin of the local system. Generally, choosing the center of mass allows to take advantage of symmetry properties. Through the choice of a different origin, for example, the load distribution of a vehicle can be taken into account.*
- *Alternatively, the vehicle can be considered as a multi-body system, enabling the individual subsystems, as for example suspension, tires, etc., to be investigated.*

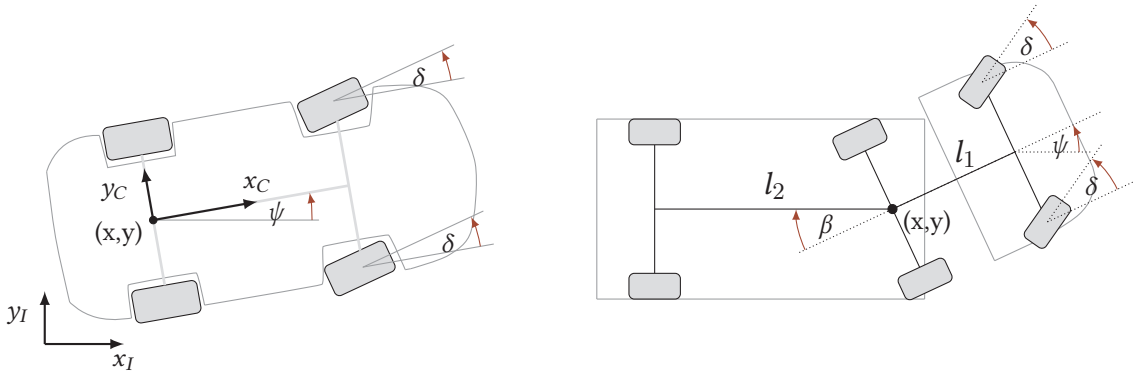


Figure 5.1: *Metrics of a planar truck model (right) and a simplified planar vehicle model (left) in a Cartesian coordinate system, with steering angle δ and heading angle ψ.*

Henceforth, the vehicle is considered as a rigid body, implying that no elastic or plastic deformations or phenomena are accounted for. Otherwise this would lead to high dimensional and complex equations of motion. Furthermore, the model is based on the assumption that the motion of the vehicle is restricted to a planar plane. Consequently, the elevation z as well as the angular pitch and roll motions can be neglected. Thus, the motion and orientation in the inertial system is defined by three coordinates x, y and ψ. With these simplifications and setting the origin of the local reference system in the center of the rear axle (x, y), the transformations between the intertial and the local reference system can be described by the following rotation and translation:

$$T = \begin{pmatrix} x & y \end{pmatrix}^\top, \tag{5.1}$$

$$A(\psi) = \begin{pmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{pmatrix}. \tag{5.2}$$

By considering the geometric relations in Figure 5.1 in conjunction with the assumption of low lateral acceleration and negligible lateral tyre forces, the midpoint position and velocity of the vehicle is described by,

$$r_R = T, \tag{5.3}$$

$$v_R = \begin{pmatrix} x' \\ y' \end{pmatrix}. \tag{5.4}$$

Remark 5.4. *As the vehicle traverses a curve, lateral forces are generated on the tire as a result of lateral acceleration. To keep the vehicle on track, it is imperative that these forces and the associated lateral acceleration remain small. Otherwise, this will lead to slip, in other words a deviation between the distance covered by each revolution of the wheel compared with its actual circumference, attributable to the effects of sliding between the surfaces in contact with each other (road and tires) [Rajamani, 2012]. Under typical municipal driving conditions at low longitudinal velocities, such an assumption is justified.*

Accordingly, for the front axle position r_F and velocity v_F using (5.1) and (5.2) we get,

$$r_F = \begin{pmatrix} x + l\cos\psi \\ y + l\sin\psi \end{pmatrix} \quad \text{and} \quad v_F = \begin{pmatrix} x' - l\psi'\cos\psi \\ y' + l\psi'\sin\psi \end{pmatrix},$$

where l is the distance between the front and rear axle. Suppose, in accordance with our assumption on negligible lateral tyre forces, that the lateral velocities at the front and rear axle vanish, then by transforming (5.3) to the local reference system employing (5.2), only the longitudinal component of the velocity v at the rear axle remains. Thus, yielding the system of first order differential equations for the vehicles position,

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = v \begin{pmatrix} \cos\psi \\ \sin\psi \end{pmatrix}. \tag{5.5}$$

Next, a differential equation for the yaw angle, i.e., the orientation, is to be determined. Expressing the front axle velocity v_F in the vehicles local reference system yields,

$$v_{F,L} = A(\psi)^\top v_F = \begin{pmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{pmatrix} \begin{pmatrix} x' - l\psi'\cos\psi \\ y' + l\psi'\sin\psi \end{pmatrix},$$

and by substituting x' and y' with (5.5) we get,

$$v_{F,L} = \begin{pmatrix} v\cos^2\psi + l\psi'(\cos\psi\sin\psi - \cos\psi\sin\psi) + v\sin^2\psi \\ -v\cos\psi\sin\psi + l\psi'\left(\sin^2\psi + \cos^2\psi\right) + v\cos\psi\sin\psi \end{pmatrix} = \begin{pmatrix} v \\ l\psi' \end{pmatrix}. \tag{5.6}$$

On the assumption of vanishing lateral velocities we obtain,

$$0 = \begin{pmatrix} -\sin\delta \\ \cos\delta \end{pmatrix} v_{F,L}, \tag{5.7}$$

where δ denotes the steering angle.

Remark 5.5. *In general the steering angles on the right and left front wheel can differ. For the sake of simplicity we assume here that they are identical.*

Then, combining (5.6) with (5.7) yields the differential equation of the yaw angle,

$$\psi' = \frac{v(t)}{l}\tan\delta. \tag{5.8}$$

In conclusion, the motion of the vehicle is characterized by the following system of ordinary differential equations:

$$x' = v(t)\cos\psi, \tag{5.9.1}$$
$$y' = v(t)\sin\psi, \tag{5.9.2}$$
$$\psi' = \frac{v(t)}{l}\tan\delta. \tag{5.9.3}$$

On one hand, the velocity in this model may be thought of as a time-varying function, while on the other hand, it may be obtained from a longitudinal vehicle model, as shown in the upcoming section.

The truck model is composed of a carrier vehicle, which is identical to the passenger car model, and a semi-trailer. For a complete description of the orientation and position of the truck model, the equations of motion from the kinematic model can be applied. These only need to be complemented by an additional differential equation, specifying the orientation of the trailer relative to the carrier vehicle.

First we introduce the position vector of the trailer in the local system, which, under retention of the above assumptions, leads to

$$r_{T,L} = \begin{pmatrix} -l_1 \\ 0 \end{pmatrix} + A(\beta) \begin{pmatrix} -l_2 \\ 0 \end{pmatrix} = \begin{pmatrix} -l_1 - l_2 \cos\beta \\ l_2 \sin\beta \end{pmatrix}.$$

Subsequently, the position vector in the inertial system is to be described. Applying (5.2), we obtain,

$$r_{T,I} = \begin{pmatrix} x \\ y \end{pmatrix} + A(\psi) r_{T,L} = \begin{pmatrix} x - l_1 \cos\psi - l_2 \cos(\psi - \beta) \\ y - l_1 \sin\psi - l_2 \sin(\psi - \beta) \end{pmatrix}. \tag{5.10}$$

Derivation of (5.10) yields the velocity of the trailer in the inertial reference system:

$$v_{T,I} = \begin{pmatrix} x' + l_1\psi' \sin\psi + l_2(\psi' - \beta')\sin(\psi - \beta) \\ y' - l_1\psi' \cos\psi - l_2(\psi' - \beta')\cos(\psi - \beta) \end{pmatrix},$$

and respectively the backwards transformation, in combination with the substitution $x' = v$ and the assumption that the lateral velocities vanish leads to,

$$v_{T,L} = A(\psi)^\top v_{T,I} = \begin{pmatrix} v - l_2(\psi' - \beta')\sin(\beta) \\ -l_1\psi' - l_2(\psi' - \beta')\cos(\beta) \end{pmatrix}.$$

As there is no slip between road and tire, the velocity in the direction of the y-axis of the local reference system has to be zero and thus,

$$0 = \begin{pmatrix} \sin\beta \\ \cos\beta \end{pmatrix} \cdot v_{T,L} = v\sin\beta - l_1\psi'\cos\beta - l_2(\psi' - \beta').$$

Eventually, solving for $\beta'(t)$ yields the differential equation describing the orientation of the trailer with respect to the carrier vehicle:

$$\beta'(t) = \frac{1}{l_2}\left(-v\sin\beta + \psi'(t)\left(l_2 + l_1\cos\beta\right)\right).$$

Additionally, one can substitute the differential equation for the yaw angle, resulting in:

$$\beta'(t) = \frac{1}{l_2}\left(-v\sin\beta + \frac{l_2}{l_1}v\tan\delta(t) + v\tan\delta(t)\cos\beta\right).$$

Summarizing, the motion of the truck model is characterized by the following system of ordinary differential equations:

$$x'(t) = v(t)\cos\psi(t), \tag{5.11.1}$$
$$y'(t) = v(t)\sin\psi(t), \tag{5.11.2}$$
$$\psi'(t) = \frac{v(t)}{l_1}\tan\delta. \tag{5.11.3}$$
$$\beta'(t) = \frac{1}{l_2}\left(-v(t)\sin\beta(t) + \psi'(t)\left(l_2 + l_1\cos\beta(t)\right)\right). \tag{5.11.4}$$

Remark 5.6. *An essential advantage of the truck model proposed here is that it only requires four states to provide a complete description of the motion. It is therefore an appealing choice for the use with dynamic programming in the solution of, e.g., path planning problems.*

5.2.2 Curvilinear Coordinates

For some applications it may be beneficial to characterize the vehicular motion in terms of an alternative reference system - the curvilinear coordinates, compare [Lot and Biral, 2014, Anderson et al., 2016]. A particular subset of these applications is characterized by the presence of a reference trajectory. Considering municipal traffic scenarios, where the trajectories of the vehicles are dictated through the lanes of the road, the midline of such lanes can be adopted as a reference curve without imposing significant restrictions on the vehicle movement. With respect to autonomous driving we assume that such a reference curve is provided by, e.g., a navigation system, for each vehicle.

Thus, let $\gamma_m : [0, L] \to \mathbb{R}^2$ denote the reference curve of length L (e.g., midline of a road) where,

$$\gamma_m(s) := \begin{pmatrix} x_m(s) \\ y_m(s) \end{pmatrix} \quad \text{and} \quad \psi_m = \arctan\left(\frac{y_m'(s)}{x_m'(s)}\right) \tag{5.12}$$

which is parametrized with respect to its arc length s and let $\|v\| \neq 0$. Here the index m indicates the affiliation of the variable to the reference curve. An illustration of the coherence between Cartesian and curvilinear coordinates is provided in Figure 5.2.

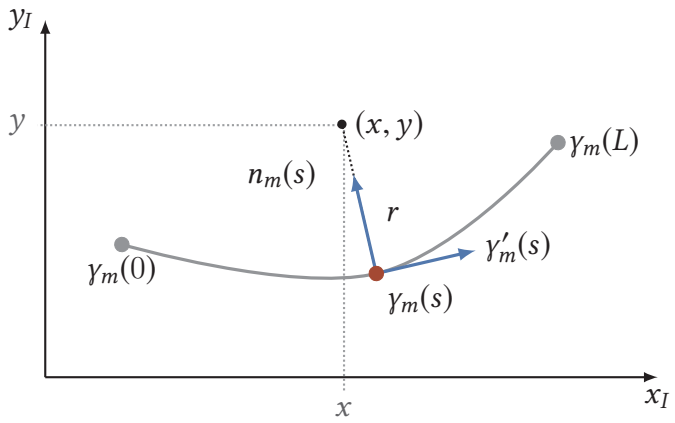


Figure 5.2: *Curvilinear Coordinates.*

An arbitrary point (x, y) in the Cartesian coordinate system then can be expressed in terms of the curvilinear coordinates $s \in [0, L]$ and $r \in \mathbb{R}$, respectively the arc length along the curve and the lateral distance from the curve, such that

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_m(s) \\ y_m(s) \end{pmatrix} + n_m(s)r, \tag{5.13}$$

whereby the normal vector is defined by

$$n_m(s) = \gamma''_m(s) = \begin{pmatrix} -y'_m(s) \\ x'_m(s) \end{pmatrix} = \begin{pmatrix} -\sin \psi_m(s) \\ \cos \psi_m(s) \end{pmatrix},$$

which is oriented perpendicular to velocity vector of the curve

$$\gamma'_m(s) = \begin{pmatrix} x'_m(s) \\ y'_m(s) \end{pmatrix} = \begin{pmatrix} \cos \psi_m(s) \\ \sin \psi_m(s) \end{pmatrix} \tag{5.14}$$

with $\|\gamma'_m(s)\| = 1$ for all $s \in [0, L]$. The curvature at arclength s computes to,

$$\kappa_m(s) = \psi'_m(s) = x'_m(s)y''_m(s) - x''_m(s)y'_m(s). \tag{5.15}$$

Remark 5.7. *In the above definitions, the normal vector $n_m(s)$ was declared to be oriented to the left. A definition in the opposite direction, that is $-n_m(s)$ is also possible. The equivalence of the two definitions can be established by substitution of r with $-r$. Furthermore, within the MPC scheme a model parametrization in terms of the arc length facilitates a more natural definition of the preview horizon in terms of a length, rather than time.*

5.2.3 Motion in Curvilinear Coordinates

Having established the curvilinear coordinates in the preceding section, we now can derive the equations of motion in terms of the (s, r) reference system. In [Lot and Biral, 2014] a similar model was developed. To this end, let the position and orientation of the vehicle be delineated by a curve $\gamma : [t_0, t_f] \to \mathbb{R}^2$, which is parameterized as a function of time using the transformation defined in Appendix A.1, i.e., $s^{-1}(\xi) = t$.

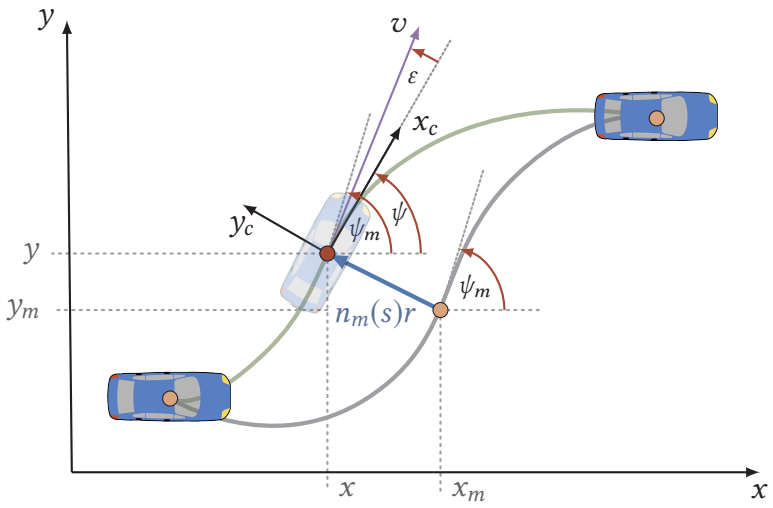


Figure 5.3: *Curvilinear coordinates defined with respect to spline trajectory. Showing the spatial relation of a reference track (gray) and the driven track (green), as well as the respective positions and angles.*

The lateral deviation of the vehicle position and its direction from this curve is indicated by r and n respectively, compare Figure 5.3. Let the absolute velocity of the vehicle be $v(t)$ and let $\psi(s(t))$ denote the vehicle's yaw angle with respect to the x-axis of the Cartesian coordinate system. Moreover, $\chi(t) = \psi(s(t)) - \psi_m(s(t))$ delineates the orientation of the x_c-axis of the local vehicle reference system, with respect to the tangent of the reference line. It is common to model the reference trajectory by means of a clothoid or a cubic spline, due to their continuous derivatives. Within the scope of this thesis a cubic spline curve will be adopted.

Applying the geometric correlations in (5.12) to (5.14), allows to express the position of the vehicle with respect to the reference curve in the Cartesian coordinate system,

$$x(t) = x_m(s(t)) - r(t)y'_m(s(t)) = x_m(s(t)) - r(t)\sin\psi_m(s(t)), \tag{5.16.1}$$

$$y(t) = y_m(s(t)) + r(t)x'_m(s(t)) = y_m(s(t)) + r(t)\cos\psi_m(s(t)). \tag{5.16.2}$$

Computing the derivatives of (5.16) in conjunction with (5.15) yields the subsequent system of differential equations:

$$x'(t) = x'_m(s(t))s'(t) - r(t)y''_m(s(t))s'(t) - r'(t)y'_m(s(t)) \tag{5.17}$$
$$y'(t) = y'_m(s(t))s'(t) + r(t)x''_m(s(t))s'(t) + r'(t)x'_m(s(t)) \tag{5.18}$$
$$\psi'_m(s(t)) = \kappa_m(s(t)) = x'_m(s(t))y''_m(s(t)) - y'_m(s(t))x''_m(s(t)) \tag{5.19}$$

Sustaining the assumptions of the kinematic model, that the lateral velocity components $y'(t) = v_c^y(t)$ vanish and the longitudinal velocity is perfectly aligned with the x-axis of the local reference system, i.e., $\varepsilon = 0$. We have the auxiliary definitions:

$$r'(t) = v_c^x(t)\sin(\chi(t)) = v(t)\sin(\chi(t)) \tag{5.20}$$
$$x'(t) = \cos\psi(s(t)) \tag{5.21}$$
$$y'(t) = \sin\psi(s(t)) \tag{5.22}$$

With (5.20) to (5.22) plugged into (5.17) to (5.18) we get,

$$\begin{aligned} v(t)\cos\psi(s(t)) &= \cos\psi_m(s(t))s'(t) - r'(t)\sin\psi_m(s(t)) \\ &\quad - r(t)\kappa_m(s(t))s'(t)\cos\psi_m(s(t)) \\ v(t)\sin\psi(s(t)) &= \sin\psi_m(s(t))s'(t) + r'(t)\cos\psi_m(s(t)) \\ &\quad - r(t)\kappa_m(s(t))s'(t)\sin\psi_m(s(t)). \end{aligned}$$

Next, we need to solve this system for the derivative of $s(t)$. Hence, we get,

$$\begin{aligned} v(t)[\cos\psi(s(t)) + \sin\psi(s(t))] &= s'(t)[\cos\psi_m(s(t)) + \sin\psi_m(s(t))] \\ &\quad + v(t)[-\sin\chi(s(t))\sin\psi_m(s(t)) + \sin\chi(s(t))\cos\psi_m(s(t))] \\ &\quad - r(t)\kappa_m(s(t))s'(t)[\sin\psi_m(s(t)) + \cos\psi_m(s(t))]. \end{aligned}$$

Reorganizing together with $\psi = \chi + \psi_m$ and some trigonometric relations leads to,

$$\begin{aligned} &v(t)[\cos\chi(s(t))\cos\psi_m(s(t)) - \sin\chi(s(t))\sin\psi_m(s(t)) + \sin\chi(s(t))\sin\psi_m(s(t)) \\ &\quad + \cos\chi(s(t))\sin\psi_m(s(t)) + \sin\chi(s(t))\cos\psi_m(s(t)) - \sin\chi(s(t))\cos\psi_m(s(t))] \\ &\quad = s'(t)[\cos\psi_m(s(t)) + \sin\psi_m(s(t))](1 - r(t)\kappa(s(t))). \end{aligned}$$

Eventually, the differential equations in the (s, r) coordinate system at time t read

$$s'(t) = \frac{v(t)\cos\chi(t)}{1 - r(t)\kappa(s(t))}, \tag{5.23}$$
$$r'(t) = v(t)\sin\chi(t), \tag{5.24}$$
$$\chi'(t) = v(t)\kappa(t) - \frac{v(t)\kappa_m(s(t))\cos\chi(t)}{1 - r(t)\kappa(s(t))}. \tag{5.25}$$

As the vehicle is bound to its lane in conventional road traffic, the midline of its lane can be adopted as a reference curve. Furthermore, small lateral deviations can be assumed to emanate. Thus, the vehicle dynamics problem can be split into two subproblems. The first subproblem is concerned with the lateral motion, i.e., a trajectory tracking control which ensures that the lateral deviations to the reference trajectory remain negligible. The second subproblem focuses on the longitudinal motion and thus, on the control of the velocity, assuming that the vehicle is always on the reference trajectory. While the tracking problem is discussed in more detail in Section 5.6, for the remainder of this section we focus on the second subproblem. Therefore, with the introduction of supplementary premises, the kinematic model can be further simplified and reduced to a longitudinal model, see [Frego et al., 2016].

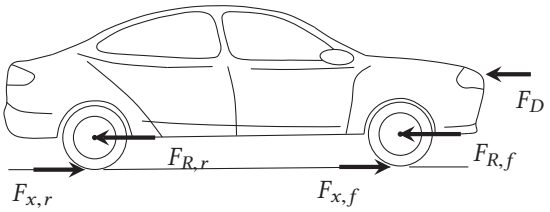


Figure 5.4: *Forces affecting the longitudinal vehicle motion.*

As a model reduction is always accompanied by a loss of information, it is important to understand the external forces that affect the longitudinal motion of the vehicle, compare Figure 5.4. In general, the accelerating force of the vehicle is impaired by gravitational forces F_N, longitudinal tyre forces $F_{x,\cdot}$, rolling resistance $F_{R,\cdot}$ and aerodynamic drag F_D, cf. [Mitschke, 2014]. Due to the fact that only a planar vehicle model is assumed, inclinations along the z-axis and thus the impact of the normal force on the forward motion of the car can be neglected. The balance of the remaining forces then becomes,

$$mx''(t) = F_{x,f} + F_{x,r} - F_D - F_{R,f} - F_{R,r} \tag{5.26}$$

where m indicates the mass of the vehicle, $x''(\cdot)$ its acceleration and the indices r and f allude to the rear and front wheels, where the respective force affects the vehicle. Due to the rigid body assumption and the restriction to longitudinal motion, an additional abstraction can be made and the vehicle can be considered as a point mass at its geometric centre of gravity. Hence, all forces can be considered as acting on the point mass and the notation can be simplified to the extent that a distinction between front and rear wheel is redundant.

Next, we will discuss the individual forces in more detail, commencing with the tire forces. The longitudinal tyre forces emanate, for small slip rations, from the friction of the ground with the tyres. Thus, the force is proportional to the slip ratios:

$$F_{x,\cdot} = c_{tyre}\sigma_x \qquad \text{with} \quad \sigma_x = \begin{cases} \frac{r_{eff}\omega - v}{v}, & \text{brake} \\ \frac{r_{eff}\omega - v}{r_{eff}\omega}, & \text{accelerate,} \end{cases}$$

where c_{tyre} is the tyre stiffness and σ_x denotes the slip ratio, where r_{eff} denotes the effective tyre radius and ω the angular rate. [Rajamani, 2012]

Another force acting on the tyre is the rolling resistance, which results from the elasticity of the tyre material. As a consequence of the stiffness of the road, the tyre is deformed vertically, resulting in an asymmetrical contact surface with the road. the resulting asymmetrical distribution of the normal force leads to a part of the normal force acting against the movement of the vehicle, see [Mitschke, 2014]:

$$F_R = c_R mg,$$

with g the gravitational constant and the rolling resistance coefficient c_R, which is a dimensionless constant. A typical value for c_R for passenger cars is 0.013 given by [Wong, 2001, p. 18].

The aerodynamic force is composed of the drag force and laminar friction. While the latter primarily takes effect at lower velocities, the impact of aerodynamic drag in particular rises significantly with increasing velocities. The vehicle is affected by aerodynamic forces, due to the air exerting a force on the vehicle in the opposing direction of the relative velocity as it moves as an unobstructed mass on the ground level through the atmosphere. Where the relative velocity is low, it is referred to as laminar friction, which is free of turbulence and directly proportional to the velocity. The laminar friction computes to,

$$F_{D,lam} = c_F v,$$

with c_F the laminar friction coefficient. At higher velocities, the aerodynamic drag,

$$F_{D,drag} = \frac{1}{2}\rho A \tilde{c}_D v^2,$$

which is proportional to the square of the velocity and takes into account the cross-section A of the vehicle must be considered. Herein, $\tilde{c}_D$ is the drag coefficient. A typical value for modern passenger cars is around 0.3. The air density is denoted by ρ , which at road level is approximately $1.226\mathrm{kg\,m^{-3}}$ and can be assumed to be constant, compare [Mitschke, 2014, p. 55-71].

As a consequence of the rigid body assumption no deformations are considered. Thus, the longitudinal tyre force as well as the rolling resistance which emanate from deformations, can be neglected. If we further consider normalized forces, that is free of the vehicle mass, (5.26) reduces to

$$x''(t) = -c_F v(t) - c_D v(t)^2, \tag{5.27}$$

where the constant $c_D = \frac{1}{2}\rho A \tilde{c}_D$. Furthermore, often it is beneficial to employ acceleration as a control instead of a direct control of the velocity. Consequently, an additional differential equation for the velocity is required, which in combination with (5.27) is

$$v'(t) = u(t) - c_F v(t) - c_D v(t)^2 \tag{5.28}$$

where $u(t)$ denotes the acceleration. Returning to the premise that no lateral deviations occur, directly implies $r(t) = 0$. Moreover, $\chi = 0$, since the yaw angles of the vehicle trajectory ψ and the reference trajectory ψ_m coincide. Therefore, the vector of the velocity is always tangent to the reference curve. Applying these modifications to (5.23) leads to the longitudinal point mass model:

$$s'(t) = v(t), \tag{5.29.1}$$
$$v'(t) = u(t) - c_F v(t) - c_D v(t)^2. \tag{5.29.2}$$

Remark 5.8. *Let us comment on the derived model:*

- *The resulting longitudinal vehicle model takes into account environmental influences, as of the laminar friction and the aerodynamic drag, indicated by their respective coefficients c_F,$\tilde{c}_D > 0$. Furthermore the influence of the longitudinal acceleration $u(t)$ is considered.*
- *Even though, using the acceleration as a control rather than the velocity, introduced an additional differential equation. Due to the splitting of the control task into two separate problems, the differential equations for the lateral deviation $r(t)$ as well as for the yaw angle $\psi(t)$ vanish. Thus, leaving us with a two dimensional system.*
- *Note that the coefficients of environmental influences c_F, $\tilde{c}_D$ are usually very low. Especially considering low velocity's the aerodynamic drag can be neglected without loss of accuracy.*

5.3 Coordination Problem and Obstacle Avoidance

5.3.1 Control and State Constraints

Within this subsection, various physical and structural control and state constraints are introduced. These serve on one hand to provide a more detailed characterization of the physical properties and conditions while the vehicle is in motion, and on the other hand to assure the collision-free coordination of vehicles. Dynamic collision avoidance constraints consistently depend on the current, albeit, evolving state of the other vehicles, e.g., position and velocity. As a consequence, the problems of the individual vehicles become coupled. To distinguish the information of the vehicles, we revert to the game-theoretic notation and henceforth, consider N players $\nu = 1, \dots, N$, where each of the vehicles is a player.

In this context, the physical characteristics and boundaries of the vehicle are characterized by the following constraints:

⟨Velocity⟩ Let the velocity $v^\nu(t)$ be constrained by an upper and lower limit. For the lower limit, we assume that the velocity is always non negative, i.e., $0 \leq v^\nu_{\min}$. Hence, the vehicle is able to come to a halt, however, it is not able to drive in reverse, which would correspond to a negative velocity. The upper bound is subject to two constraints. On the one hand, imposed by the construction of the vehicle, we have a fixed physical maximum velocity

$v^\nu_{\max}$. On the other, the maximum velocity is limited by $v^\nu_{\max}(s(t))$, which characterizes a velocity limit correlating with the current position of the vehicle on the reference path. In particular, employing Kamm's circle [Mitschke, 2014], which models the dynamic interaction between road and tire, from the equilibrium of forces the following expression for $v^\nu_{\max}(s(t))$ can be derived

$$v^\nu_{\max}(s^\nu(t), u^\nu(t)) := \min\left\{v^\nu_{\max}, \sqrt[4]{\frac{\mu_H^2 g^2 - (u^\nu(t))^2}{\kappa(s)^2}}\right\}.$$

Herein, μ_H is the friction coefficient, g is the gravitational acceleration and $\kappa(s)$ the curvature at the current position. For simplicity one can use the approximation for non-slippery dry roads

$$v^\nu_{\max}(s^\nu(t)) := \min\left\{v^\nu_{\max}, \sqrt{\frac{a_{\max}}{|\kappa(s)|}}\right\} \tag{5.30}$$

with a maximum lateral acceleration $a_{\max}$ in order to render the upper bound independent of the control u^ν.

⟨Arc Length⟩ The arc length $s(t)$ is by definition of the curve strictly positive and is supposed to satisfy

$$s^\nu \in \left[s^\nu_{\min}, s^\nu_{\max}\right],$$

where $s^\nu_{\min}$ and $s^\nu_{\max}$ denote the upper and lower limits, which indicate the initial position on the reference curve and the maximum reach respectively.

⟨Angles⟩ Considering the kinematic truck model, the rotational angles are inherently confined by upper and lower limits, imposed by physical restrictions due to the design of the vehicle. Thus, the steering angle is constrained by the minimum and maximum wheel angle,

$$\delta^\nu \in \left[\delta^\nu_{\min}, \delta^\nu_{\max}\right],$$

accordingly, β^ν is physically limited by the maximum lock angle of the trailer,

$$\beta^\nu \in \left[\beta^\nu_{\min}, \beta^\nu_{\max}\right].$$

By definition, the yaw angle is within $[-\pi, \pi]$.

Collision avoidance constraints can be defined in a variety of ways, yet are in general derived from a geometric proximity principle.

Radial Distance Approximation

A natural way to formulate a collision avoidance condition is to constrain the distance between potential collision parties by a critical threshold. Thus, the physical representation of the players

is approximated by a circular boundary in the most simplified formulation. Suppose, we have two players ν and μ which are on a collision course and their current positions are given by $(x^\nu, y^\nu), (x^\mu, y^\mu) \in \mathbb{R}^2$. Then, the distance between the two players is obtained by employing the Euclidean norm,

$$d(x, y) = \sqrt{(x^\nu - x^\mu)^2 + (y^\nu - y^\mu)^2}.$$

In order to prevent a collision, a threshold $d_{\nu\mu}$ is defined which must not be violated, as otherwise this will be considered a collision. Hence, the anti-collision constraint reads,

$$d(x, y) \geq d_{\nu\mu} \tag{5.31}$$

Remark 5.9. *The threshold often is defined in terms of the dimensions of the vehicles. In particular, for a car its length l^ν is chosen as the radius of the circular approximation. Then, commonly the threshold is defined as the sum of the radii of the vehicles, i.e., $d_{\nu\mu} = l^\nu + l^\mu$. However, this often leads to a very conservative approach and can render the problem unfeasible. Indeed, an approximation with a more suitable shape (e.g. an ellipse) can certainly ameliorate this constraint approach.*

Distance to Point of Conflict

An alternative approach, proposed in [Britzelmeier and Dreves, 2020], exploits the reference curves, which are known to all players. More precisely, rather than the relative distance between the current player positions, the distance to the intersection points with the trajectories of every opposing player is used as a measure, to establish, if a collision ensues. Within our setting, each player is following a precomputed spline curve. Furthermore, with the premise of perfect information, the trajectories are known to all players. Consequently, the intersection points can be computed a priori. Hence, let the intersection point $p_{\mu\nu}$ at which the trajectories of players μ and ν intersect be defined in terms of the arc length along the trajectory of player μ. Moreover, we have the index set J^ν, which contains all players $\mu \in \{1, \dots, N\} \setminus \nu$, whose trajectories intersect with the trajectory of player ν. Then, the anti-collision constraint for players ν and μ is defined by the functions,

$$g_{\nu\mu}(s^\nu(t), s^\mu(t)) := d_{\nu\mu} - \max\left\{|s^\nu(t) - p_{\nu\mu}|, |s^\mu(t) - p_{\mu\nu}|\right\} \leq 0, \tag{5.32}$$

where $d_{\nu\mu} > 0$ is a safety distance and $\mu \in J^\nu$. To put it another way, at any time only one vehicle is permitted to be within the safety perimeter — defined by $d_{\nu\mu}$ — around the point of conflict $p_{\mu\nu}$ and $p_{\nu\mu}$.

Remark 5.10. *With this collision condition, the admissible space is not as restricted as if the relative distance between the vehicles would be considered. Furthermore, (5.32) provides a much better representation of genuine conflicts arising from road traffic, such as crossing an intersection. Even rear-end collisions can be considered, i.e., for vehicles with shared paths an equidistant grid*

of conflict points can be established with distance $d_{v\mu}$. However, for paths with long distances this might yield a vast increase in collision conditions and therefore is considered inefficient.

Weighted Residency Function

So far, the collision condition has been measured in terms of spatial distances, however, it is equally possible to quantify the anti-collision constraint using temporal correlations. To this end, let us define the indicator function,

$$\mathcal{X}_{[p_{\mu\nu}-d_{\nu\mu},p_{\mu\nu}+d_{\nu\mu}]}(s^{\nu}(t)) := \begin{cases} 1, & \text{if } s^{\nu}(t) \in \left[p_{\mu\nu}-d_{\nu\mu}, p_{\mu\nu}+d_{\nu\mu}\right] \\ 0, & \text{if } s^{\nu}(t) \notin \left[p_{\mu\nu}-d_{\nu\mu}, p_{\mu\nu}+d_{\nu\mu}\right] \end{cases}, \tag{5.33}$$

where $s^{\nu}(t)$ indicates the arc length along the ν-th players trajectory and $d_{v\mu}$ again denotes the safety distance. Inherently, the function suggests, if for two vehicles at any time in $[t_0, t_f]$ at most one of the indicator functions is one, the vehicles drive collision free. Then, we can define the anti-collision constraints as

$$g_{v\mu}(s^{\nu}, s^{\mu}) := \frac{1}{T}\int_0^T \mathcal{X}_{[p_{\nu\mu}-d_{\nu\mu},p_{\nu\mu}+d_{\nu\mu}]}(s^{\nu}(t)) \cdot \mathcal{X}_{[p_{\mu\nu}-d_{\nu\mu},p_{\mu\nu}+d_{\nu\mu}]}(s^{\mu}(t))\, dt = 0 \tag{5.34}$$

Note, that we have equations instead of inequalities to define the anti-collision constraints. The collision criterion defined above, measures the overlap of the intervals of the players, if they simultaneously are within the safety perimeter. Hence, a weighted measure of residency in the collision zone is used to establish the state of collision.

Remark 5.11. *Compared to (5.32), (5.34) takes implicitly into account the proportional distance to the point of conflict.*

5.3.2 Objectives

To simulate the coordination of the players, a suitable optimal control problem needs to be formulated for each player. While the constraints of the individual optimal control problems are comprised of one of the previously defined vehicle models as well as the respective physical constraints and a suitable collision constraint, it remains to provide an appropriate driver model. For the purpose of the simulation, an ideal driver is assumed, implying that the driver is able to accurately realize the computed trajectories free of any deviations or time delays. Within the framework of the optimal control problem, the driver itself is characterized by the objective function. There exist a number of ways to express the objective function and thus the driver's intention.

Observing the traffic in urban areas or on highways, it becomes evident that there exists a multitude of distinct driving styles, which reflect directly or indirectly the intentions of the driver. While some drivers are keen to drive economically, by trying to reduce their fuel consumption,

e.g., to protect the environment, others aim to arrive at their destination as quickly as possible. Within the framework of this thesis the latter intention is considered, which allows the modelling of a selfish behaviour of the players within the coordination problem of the vehicles through the computation of time-minimal velocity profiles.

Essentially, given an initial time t_0 and a final time t_f, such that $t \in [t_0, t_f]$ and assume that the vehicle moves on a given planar reference curve $\gamma^\nu : [t_0, t_f] \to \mathbb{R}^2$, with $\gamma^\nu(t) = (x^\nu(t), y^\nu(t))^\top$ for $t_0 \leq t \leq t_f$, then each player $\nu = 1, \ldots, N$ is interested in reaching his destination within a minimum time frame. Thus, the optimality criterion is to

$$\text{Minimize} \quad t_f^\nu = \int_{t_0}^{t_f} 1 \, d\tau, \tag{5.35}$$

where the initial and terminal position are respectively defined by the reference curve through $\gamma_0^\nu = \gamma^\nu(0) = (x_0^\nu, y_0^\nu)^\top$ and $\gamma_f^\nu = \gamma^\nu(t_f) = (x_f^\nu, y_f^\nu)^\top$.

Furthermore, the optimality criterion can be parametrized with respect to the arc length. To this end, let $\zeta \in [s_0, s_f]$ denote the arc length of the vehicles current position at time t. Then, applying the transformation (A.1) in combination with $\frac{d\zeta}{dt} = v$ and the identity $s(t) = \zeta$ yields

$$t_f = \int_{t_0}^{t_f} 1 \, dt = \int_{s_0}^{s_f} \frac{1}{v(\zeta)} d\zeta \tag{5.36}$$

where s_f describes the target arc length at time t_f and correspondingly s_0 the arc length of the initial position at time t_0. Hence, the optimality criterion can be expressed as a function of the velocity, implying that a time minimal objective function is equivalent to maximizing the velocity. Conversely, (5.29.1) suggests that for a fixed time horizon, the maximization of the travelled distance is equivalent to the time minimal optimality criterion. Therefore, the progress along the reference track in curvilinear coordinates is considered. Then, the objective is to

$$\text{Maximize} \quad s(t_f) \Leftrightarrow \text{Minimize} \quad -s(t_f). \tag{5.37}$$

It remains to show that (5.35) is equivalent to (5.37).

Theorem 5.12. *Let a reference curve $\gamma : [0, T] \to \mathbb{R}^2$, with $\gamma(t) = (x(t), y(t))^\top$ be given. Furthermore, there exists a strictly monotone and continuous function $s : [0, T] \to [0, L]$, with $s(t) = \zeta$ and $\zeta \in [0, L]$ denoting the arc length along the curve. Then, the inverse function*

$$s^{-1} : [0, L] \to [0, T],$$

with $s^{-1}(\zeta) = t$ exists and it holds for $v > 0$ that,

$$\int_0^T 1 \, dt = -s(T). \tag{5.38}$$

Proof. To prove the equality in (5.38), we first consider the two objective functions,

$$\min -s(T) \qquad \text{and} \qquad \min \int_0^L \frac{1}{v(s)}\, ds,$$

Then, for the first function we have,

$$-s(T) = -\int_0^T s'(t)\, dt.$$

and with $s'(t) = v(t)$, the identity, $s^{-1}(s(t)) = t$ and the derivative of the identity $(s^{-1})'(s(t)) = \frac{1}{s'(s^{-1}(s(t)))}$, which implies $t'(s) = \frac{1}{v(s)}$, we can state that,

$$\int_0^L \frac{1}{v(s)}\, ds = \int_0^L (s^{-1})'(t)\, ds.$$

Then, since for $v > 0$ the function $s(t)$ is bijective and strictly monotonically increasing, moreover continuous, we can apply Theorem A.3. Inserting the equations in the geometric cohesion together with the boundaries,

$$\begin{aligned} a &= 0, \qquad & c &= s'(0) = 0, \\ b &= T, \qquad & d &= s'(T) = 0, \end{aligned}$$

yields,

$$\int_0^L (s^{-1})'(t) ds = (bd - ac) - \int_0^T s'(t) dt. \tag{5.39}$$

Hence, we get,

$$\int_0^L (s^{-1})'(t) ds = -\int_0^T s'(t) dt. \tag{5.40}$$

The assertion then follows from the equality in (5.36). □

Remark 5.13. *Objectives characterizing an economic driving behavior and the presented minimal time objectives are not mutually exclusive to each other. On the contrary, suitably combining such contradicting targets can be used to model a moderate driver. Besides, there exist many more approaches in modeling a virtual driver, however, to remain within the scope of the theory presented in Chapter 3 and Chapter 4, and due to the reasons stated above we chose the minimal time objective.*

5.4 Optimal Control for a Truck Manoeuvring Problem

After establishing the vehicle models, constraints and optimality criteria in the preceding sections, we can now state the problem formulation. In particular, employing the truck model in (5.11) with a suitable collision constraint a truck manoeuvring problem subject to obstacle avoidance conditions is defined subsequently. Thereafter, the problem is evaluated numerically in order to validate the performance of Algorithm 8 for large scale problems of higher dimension. Further, Algorithm 8 is embedded into Algorithm 9, with a fixed time horizon $[0, T]$. The optimal controls for this time horizon are computed and subsequently applied for a particular time $\Delta < T$.

5.4.1 Problem: Truck Collision Avoidance

In order to evaluate the performance impact of a higher order problem on the GPU parallel dynamic programming implementation and the feasibility of real time path planning with collision avoidance, we consider the following problem.

$$
\begin{aligned}
\min_{(v,\delta,u,x,y,\psi,\beta)} \quad & (x_c(T) - x_d)^2 + (y_c(T) - y_d)^2 + (\psi(T) - \psi_d)^2 \\
\text{s.t.} \quad & x_c(0) = x_0, \quad y_c(0) = y_0, \\
& \psi(0) = \psi_0, \quad \beta(0) = \beta_0, \\
& x'(t) = v(t)\cos\psi(t), && \text{a.e. } t \in [0,T], \\
& y'(t) = v(t)\sin\psi(t), && \text{a.e. } t \in [0,T], \\
& \psi'(t) = \frac{v(t)}{l_1}\tan\delta(t), && \text{a.e. } t \in [0,T], \\
& \beta'(t) = \frac{1}{l_2}\left(-v(t)\sin\beta(t) + \psi'(t)\left(l_2 + l_1\cos\beta(t)\right)\right), && \text{a.e. } t \in [0,T], \\
& v(t) \in [v_{\min}, v_{\max}], && \forall t \in [0,T], \quad \text{(T-OCP)} \\
& \delta(t) \in [\delta_{\min}, \delta_{\max}], && \forall t \in [0,T], \\
& \psi(t) \in [\psi_{\min}, \psi_{\max}], && \forall t \in [0,T], \\
& \beta(t) \in [\beta_{\min}, \beta_{\max}], && \forall t \in [0,T], \\
& g(x_c(t), y_c(t)) \leq 0, && \forall t \in [0,T].
\end{aligned}
$$

Herein, x_d, y_d and ψ_d denote the desired destination and orientation at time T. Moreover, the position of the tractor is characterized by x_c and y_c, to preclude any confusion with the state vector. Thus, the objective is to plan a path to a certain destination while avoiding collisions. To this end, let the collision constraint be defined by (5.31), i.e., a radial safety distance approximation.

Thus, we have for each $i = 1, \dots, n_g$,

$$g_i(x_c(t), y_c(t)) := d - \sqrt{\left(x_c(t) - x_p^i\right)^2 + \left(y_c(t) - y_p^i\right)^2},$$

where the vectors x_p and y_p describe the position of the obstacles and d is the safety distance. Let us stress that for $N = 1$ this can be interpreted as a one player game of problem type (GNEP), with $n_x = 4$ and $n_u = 2$. Hence, let $u := (v, \delta)$ denote the cumulative control vector and $x := (x_c, y_c, \psi, \beta)$ the cumulative state vector, with $u : [0, T] \to \mathbb{R}^2$ and $x : [0, T] \to \mathbb{R}^4$. Accordingly, let the lower and upper bounds on the states and controls be aggregated by the vectors,

$$\begin{aligned} x_{\min} &= \left(x_{c,\min}, y_{c,\min}, \psi_{\min}, \beta_{\min}\right)^\top, & x_{\max} &= \left(x_{c,\max}, y_{c,\max}, \psi_{\max}, \beta_{\max}\right)^\top, \\ u_{\min} &= \left(v_{\min}, \delta_{\min}\right)^\top, & u_{\max} &= \left(v_{\max}, \delta_{\max}\right)^\top. \end{aligned}$$

To solve (T-OCP) by means of Algorithm 8, i.e., approximate dynamic programming, a discrete approximation of the problem is required. To this end, on the lattice $\mathbb{G}_h$, the discrete states x_h are approximated piecewise linearly by an explicit Euler method, and the discrete controls u_h piecewise constant, respectively. Moreover, to handle the intricate anti-collision constraints, the penalty approach presented in Section 3.2.1 is adopted. Thus, we define the penalty term

$$\tilde{P}(x_h) = \sum_{i=1}^{n_g} \sum_{\ell=0}^{M-1} \max\left\{0, g_i(x_h(t_\ell))\right\},$$

where the constraint is assumed to hold point wise at each point of $\mathbb{G}_h$.

For a more compact formulation let the function

$$\tilde{\xi}(u_h, x_h) = \begin{pmatrix} v(t)\cos\psi(t) \\ v(t)\sin\psi(t) \\ \frac{v(t)}{l_1}\tan\delta(t) \\ \frac{1}{l_2}\left(-v(t)\sin\beta(t) + \psi'(t)\left(l_2 + l_1\cos\beta(t)\right)\right) \end{pmatrix},$$

define the dynamics in one time step. Then, with initial time $t_\ell \in \{t_0, \dots, t_M\}$ and initial states $z_\ell = \left(x_{c,\ell}, y_{c,\ell}, \psi_\ell, \beta_\ell\right)^\top$, the remaining constraints can be aggregated in the admissible set for one time step :

$$\begin{aligned} \tilde{X}_h(t_\ell, z_\ell) = \Big\{ (u_h, x_h) \mid\ & x_h(t_\ell) = z_\ell; \quad \forall t \in [t_\ell, t_{\ell+1}] : \\ & x_h(t) = x_h(t_\ell) + (t - t_\ell)\tilde{\xi}(u_h(t_\ell), x_h(t_\ell)), \\ & x(t_{\ell+1}) \in [x_{\min}, x_{\max}], \\ & u(t_\ell) \in [u_{\min}, u_{\max}] \Big\}. \end{aligned}$$

Hence the discrete penalty problem reads,

$$\min_{(u_h,x_h)\in\tilde{X}_h(t_\ell,z_\ell)} \quad \varphi(x_h(T)) + \rho\tilde{P}(x_h). \tag{PT-DOCP}$$

with the Mayer term defined by

$$\varphi(x_h(T)) = (x_h(T) - x_d)^2 + (y_h(T) - y_d)^2 + (\psi_h(T) - \psi_d)^2 .$$

By Definition 4.14 the approximate value function, (4.23) to (4.24), defined on a spatial grid $\mathbb{G}_x$, with step size h_x^i for each $i = 1, \ldots, n_x$, is acquired by means of the backward Semi-Lagrangian scheme in Section 4.5.2 in conjunction with a suitable interpolation. Thus, at time t_ℓ and the initial state z_ℓ the approximate value function of (PT-DOCP) is recursively defined by

$$\begin{aligned}
\tilde{\vartheta}_h^{Truck}(t_M, z_M) &= \varphi(z_M), \\
\tilde{\vartheta}_h^{Truck}(t_\ell, z_\ell) &= \min_{x\in\tilde{X}(t_\ell,z_\ell)} \left[\tilde{P}_\ell(x_h) + \sum_{j=1}^{2^{n_x}} \alpha_j \tilde{\vartheta}_h^{Truck}(t_{\ell+1}, \Pi_j[\tilde{\xi}(x)])\right].
\end{aligned}$$

5.4.2 Numerical Evaluation: Truck Collision Avoidance

At first in order to evaluate and legitimate the performance of Algorithm 8 we consider (PT-DOCP), namely the truck path planning problem with obstacle avoidance. While the Lotka-Volterra problem discussed in Section 4.7.3 provided a useful benchmark to establish the performance gain over a CPU parallel computation, the complexity of the unconstrained problem was low. Therefore, we consider the path planning problem, where the task of the truck is to avoid collision with an obstacle and eventually assume a parking position and orientation which requires the truck to revert his orientation and drive backwards. Moreover, the considered problem is of sixth dimension, which for standard dynamic programming is extremely challenging. Algorithm 8 is realized in a software package, with the source code written in C++ and CUDA. Furthermore, the software package envelopes, a model predictive control setting, where Algorithm 8 is embedded in (S.1) in Algorithm 9. Due to the model predictive control approach in each iteration a shorter time interval, i.e., the prediction horizon, can be considered, which allows us to further reduce the computational load, since less discretization points are required to maintain a certain temporal resolution, as compared to solving the problem on the whole time interval.

As penalty parameter a fixed value $\rho = 30$ is used and the safety distance to avoid collision with the obstacle is set to $d = 8$ m. The time horizon is set to $t_f = 3$ s with an initial time $t_0 = 0$ s. For the temporal discretization we choose $M = 100$. The spatial domain is chosen to be symmetric for $x, y \in [200, 400]$. The trailer angle β is discretized on $[-0.2\pi, 0.2\pi]$ the heading angle φ on $[-\pi, \pi]$, the velocity on $[-5.0, 5.0]$ $\mathrm{m\,s^{-1}}$ and the steering angle δ on $[-0.3\pi, 0.3\pi]$. Accordingly, the discretization parameters are set to $\Delta = h = 0.1$ s, $h_x = (2.0, 2.0, 0.13, 0.13)^\top$ and

$h_u = (0.18, 0.18)^\top$. The truck is supposed to reach a final position of $x_d = 290$ and $y_d = 305$, with an orientation of $\varphi_d = 0.75\pi$, while avoiding an obstacle positioned at $(x_p, y_p)^\top = (265, 280)^\top$. The initial configuration is given by $(x_0, y_0, \beta_0, \varphi_0) = (250, 280, 0.5\pi, 0.5\pi)$.

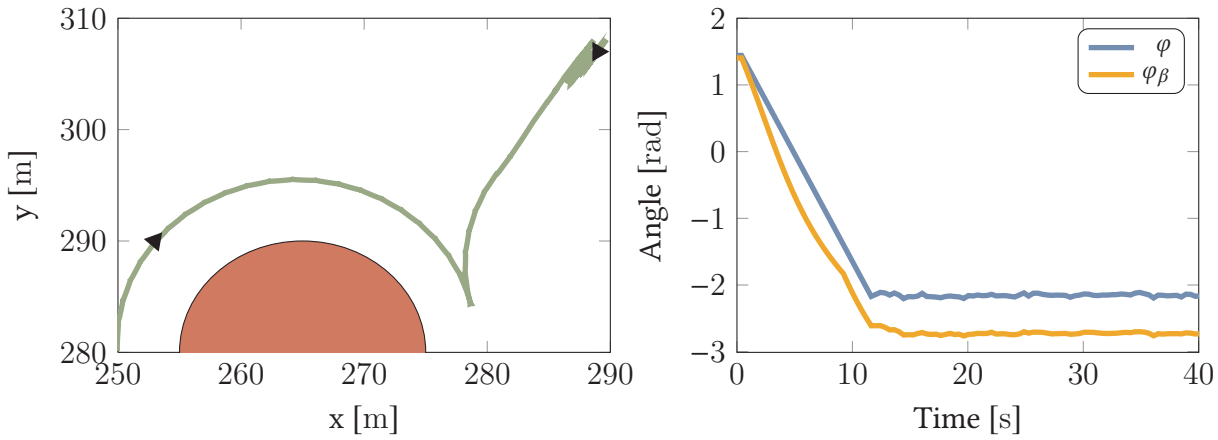


Figure 5.5: *Optimal solution trajectories of the position (left) and the angles (right) of the Truck manoeuvring problem using Algorithm 8. The orientation of the truck is indicated by a triangle (black).*

The computed state trajectories are shown in Figure 5.5. A collision with the obstacle could be successfully avoided by the truck and the final position and orientation is reached by a reverse parking manoeuvre. Herein, the trailer angle is represented by its heading angle to ease the interpretation, where $\varphi_\beta \in [-\pi, \pi]$. We can see that the trailer is almost always aligned with the tractor. Respectively the optimal controls are at display in Figure 5.6, yet reveal no peculiarities.

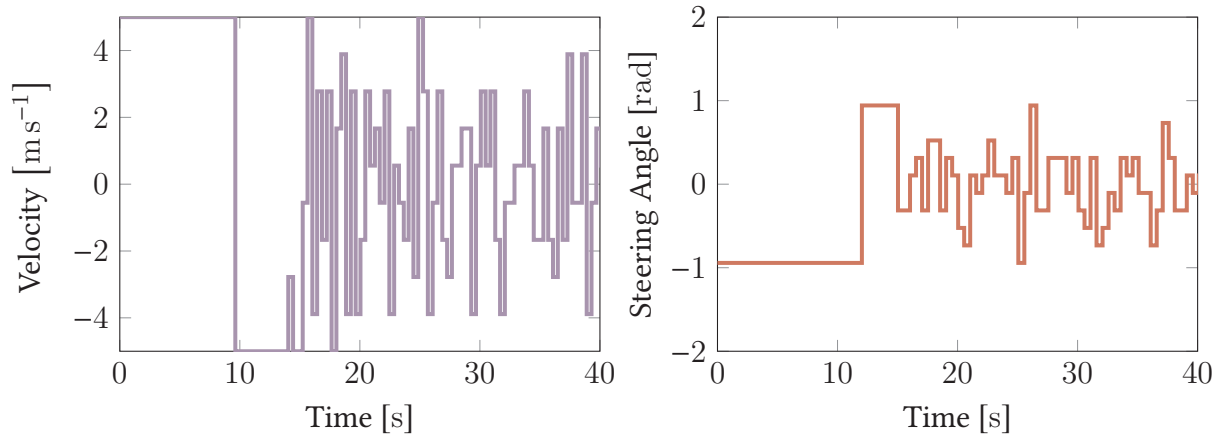


Figure 5.6: *Optimal solution of the control trajectories, i.e., the velocity (left) and steering angle (right), of the Truck problem using Algorithm 8.*

In order to measure the computational performance of the algorithm, the computation times for each MPC iteration are considered, see Figure 5.7. It is noticeable that, with the exception of a peak of 188.87 ms at the initialisation stage, using Algorithm 8 each MPC step can in average be computed in 148.18 ms. In conclusion, Algorithm 8 in combination with a MPC setting suggests to be a suitable choice in order to achieve real time computations for the control of high dimensional problems, in particular path planning, using dynamic programming.

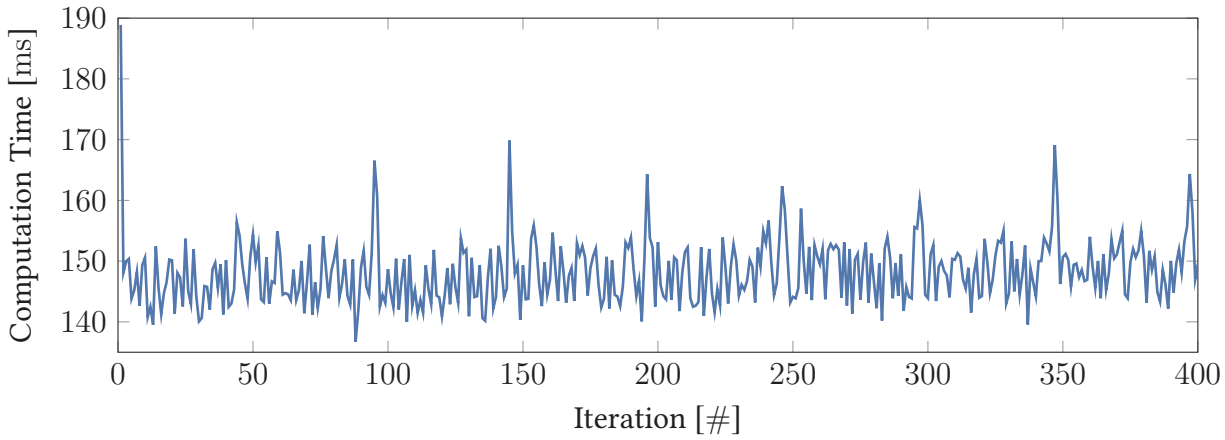


Figure 5.7: *Computation times in milliseconds for each MPC step of the truck path planning problem, executed on GPU.*

5.5 Optimal Control in Terms of Generalized Nash Equilibrium Problems

In this section, a multi-agent coordination problem is formulated as an N player generalized Nash equilibrium. Further, we show that the coordination problem is in compliance with the setting and theory established in Section 3.2 and Section 4.6. Therefore, first the respective continuous GNEPs are defined, which, employing the penalty approach introduced in Section 3.2.1 results in a standard Nash equilibrium problem. The discrete value function is then defined applying the Semi-Lagrangian scheme in conjunction with a state space discretization. Some parts of this section have already been published in [Britzelmeier and Dreves, 2020].

5.5.1 Problem: Multi-Agent Coordination

In the context of autonomous driving, traffic control, in particular the coordination of several vehicles, constitutes a complex yet intriguing control problem. On the premise of autonomous vehicles, we assume that the vehicles communicate with each other in order to avoid collisions and that each vehicle follows a precomputed trajectory, obtained from a navigation system.

When in conflict, the individual drivers are interested in establishing a consensus in order to resolve it. Game theoretically, such a problem can be interpreted as the players striving to find an equilibrium where nobody can improve if the others stick to their driving strategies. Therefore, the coordination problem can be expressed as a N player differential game $\Gamma = \{X_\nu(\varkappa^{-\nu}), \theta_\nu\}_{\nu=1}^N$ of perfect information. Furthermore, to reflect the selfishness of individual players, let the driver be characterized according to (5.37), where each vehicle tries to reach its destination in minimal time. Moreover, to eventually solve the coordination problem, optimal velocity profiles, subject to the vehicle dynamics given by (5.29), are computed. Therein, each player has the acceleration as a single control with $u = (u^1, \dots, u^N)$, where each $u^\nu : [0,T] \to \mathbb{R},\ \nu = 1, \dots, N$ are measurable functions. The states, are defined by the velocity $v = (v^1, \dots, v^N)$ and the arc-length $s = (s^1, \dots, s^N)$, where each $(v^\nu, s^\nu) : [0,T] \to \mathbb{R}^2,\ \nu = 1, \dots, N$, are absolutely continuous functions in $W^{1,1}([0,T])$. Furthermore, as in the previous section, a NMPC scheme is applied, where in each step of the NMPC, for a fixed time horizon $[0,T]$, the coordination problem is modelled as a GNEP. Thus, within the GNEP every player $\nu = 1, \dots, N$ has to solve an optimal control problem

$$
\begin{aligned}
\min_{(u^\nu, v^\nu, s^\nu)} \quad & -s^\nu(T) \\
\text{s.t.} \quad & v^\nu(0) = v_0^\nu, \quad s^\nu(0) = s_0^\nu, \\
& (v^\nu)'(t) = u^\nu(t) - c_1 v^\nu(t) - c_2 v^\nu(t)^2, && \text{a.e. } t \in [0,T], \\
& (s^\nu)'(t) = v^\nu(t), && \text{a.e. } t \in [0,T], \\
& k^\nu(v^\nu(t), s^\nu(t)) \le 0, && \forall t \in [0,T], \qquad \text{(MAC-GNEP)} \\
& u^\nu(t) \in \mathbb{U}_\nu, \\
& v^\nu(t) \le v_{\max}^\nu(s^\nu(t)), && \forall t \in [0,T], \\
& g_{\nu\mu}(s^\nu(t), s^\mu(t)) \le 0, \qquad \forall \mu \in J^\nu, && \forall t \in [0,T].
\end{aligned}
$$

Herein and in compliance with the definitions in 3.2.1, let the individual constraints,

$$
k^\nu(v^\nu(t), s^\nu(t)) := \begin{pmatrix} v^\nu(t) - v_{\max}^\nu \\ v_{\min}^\nu - v^\nu(t) \end{pmatrix}
$$

impose boundary conditions on the state variables. Moreover, let g^ν aggregate the shared and troublesome constraints such that

$$
g^\nu(v, s) := \begin{pmatrix} v(t) - v_{\max}^\nu(s(t)) \\ g_{\nu\mu}(s^\nu(t), s^\mu(t)) \end{pmatrix}
$$

where the collision avoidance constraint $g_{\nu\mu}(s^\nu(t), s^\mu(t))$ are defined by (5.34) with $s^\mu(t)$ characterizing the arc length of the opposing players trajectory $\mu \neq \nu$. Furthermore, the index set J^ν contains all players $\mu \in \{1, \dots, N\} \setminus \nu$, whose trajectories have a possible point of conflict, i.e., an intersection, with the trajectory of player ν. The individual constraints imposed on the problem with respect to the velocity have been defined in Section 5.3.1, where the upper bound

on v is characterized by $v^\nu_{\max}$, the physical limit of the vehicle, as well as $v^\nu_{\max}(s(t))$, which is a velocity limit that correlates to the current position of the vehicle on the path. For instance, the latter may be defined as road dependent speed limits or a lateral stability condition of the vehicle, as in (5.30), i.e., not to exceed the lateral acceleration. For the lower limit $0 \leq v^\nu_{\min}$ we suppose that the vehicles cannot drive in reverse.

Now, in order to assure the existence of a feasible point, suppose that $0 \leq v^\nu_{\min} \leq v^\nu_{\max}(s)$ for all $s \in [s^\nu(0), s^\nu(T)]$ and for all $\nu = 1, \ldots, N$. Moreover, for all players $\nu = 1, \ldots, N$ we assume that the functions $v^\nu_{\max}(s)$ are Lipschitz continuous, more precisely, for all $\nu = 1, \ldots, N$ and all $s, \tilde{s} \in [s^\nu_0, s^\nu_0 + T v^\nu_{\max}]$ there exists a common constant $L > 0$, such that

$$|v^\nu_{\max}(s) - v^\nu_{\max}(\tilde{s})| \leq L|s - \tilde{s}|. \tag{5.41}$$

Further, we assume the existence of some constant $c_3 > 0$ such that for all $s \in [s^\nu(0), s^\nu(T)]$

$$u^\nu_{\min} < c_1 \min\{v^\nu_{\max}, v^\nu_{\max}(s^\nu)\} + c_2 \min\{v^\nu_{\max}, v^\nu_{\max}(s^\nu)\}^2 - c_3, \tag{5.42}$$

$$u^\nu_{\max} > c_1 v^\nu_{\min} + c_2 (v^\nu_{\min})^2 + c_3. \tag{5.43}$$

Considering (MAC-GNEP), we have a potential game of type (GPG), as the objective function is independent of the opposing players strategies. Thus, instead of solving (MAC-GNEP), the structure of the potential game allows us to consider the single optimization problem

$$\min_{(u,v,s)} \quad -\sum_{\nu=1}^{N} s^\nu(T) \quad \text{s.t. } (u,v,s) \in \bigtimes_{\nu=1}^{N} X^{MAC}_\nu, \quad (v,s) \in G^{MAC}(v,s) \tag{MAC-GPG}$$

where we defined the set holding the individual constraints by,

$$\begin{aligned} X^{MAC}_\nu := \big\{(u^\nu, v^\nu, s^\nu) \mid\ & v^\nu(0) = v^\nu_0, \quad s^\nu(0) = s^\nu_0, \\ & (v^\nu)'(t) = u^\nu(t) - c_1 v^\nu(t) - c_2 v^\nu(t)^2, && \text{a.e. } t \in [0,T], \\ & (s^\nu)'(t) = v^\nu(t), && \text{a.e. } t \in [0,T], \\ & k^\nu(v^\nu(t), s^\nu(t)) \leq 0, && \forall t \in [0,T], \\ & u^\nu(t) \in \mathbb{U}_\nu \big\} \end{aligned}$$

and the set of the remaining state constraints is defined by,

$$G^{MAC}(v,s) := \{(v,s) \mid g^\nu(v,s) \leq 0,\ \forall t \in [0,T],\ \forall \mu \in J^\nu,\ \forall \nu = 1, \ldots, N\}.$$

Now, since the problem at hand is a potential game, the equilibrium solution can be obtained through the application of Algorithm 5, that is, by means of our decomposition method with penalty selection. Therefore, at first, to cope with the intricacy imposed by the collision avoidance constraints, the penalty reformulation presented in Section 3.2.1 is applied to shift the nonconvex and state dependent velocity constraints in G^{MAC} to the objective function. Hence, we define a

penalty function,

$$P(v,s) := \sum_{\nu=1}^{N} \sum_{\mu \in J^\nu} \frac{1}{T} \int_0^T \max\{0, g^\nu(v,s)\}\,. \tag{5.44}$$

With this we obtain the penalized (continuous) NEP:

$$\min_{(u^\nu, v^\nu, s^\nu)} -s^\nu(T) + \rho P(v,s) \qquad \text{s.t.} \qquad (u^\nu, v^\nu, s^\nu) \in X_\nu^{MAC},$$

where the positive constant ρ is defined as in Section 3.2.1, respectively

$$\rho = \frac{1}{(h+h_u)^p} \quad \text{with} \quad p \le 1.$$

Remark 5.14. *It is easily possible to introduce further (Lipschitz continuous) shared constraints, for example for vehicles that are driving behind each other in the same direction, by treating them as the $g_{\nu\mu}$ functions. For notational simplicity, we restrict our presentation to the common constraints $g_{\nu\mu}$, since they already have the problematic property of being nonconvex.*

Next, let us define the discrete approximation on the lattice $\mathbb{G}_h$, with $t_f = T$ and step size,

$$h = \frac{T}{M}.$$

Moreover, let the discrete controls u_h^ν attain values on an equidistant grid defined by (3.20) and therefore be piecewise constant approximations, i.e., $u_h^\nu \in \mathbb{U}_\nu^{h_u^\nu}$. Additionally, the step size h_u^ν shall be defined by the maximum stride length of all players

$$h_u := \max_{\nu=1,\dots,N} h_u^\nu,$$

and accordingly let the bound on the control be characterized through

$$B := \max_{\nu=1,\dots,N} \max\{|u_{\min}^\nu|, |u_{\max}^\nu|\}.$$

The discrete states (v_h^ν, s_h^ν) are approximated piecewise linearly through an explicit Euler method. As for the discrete approximation of the velocity constraints we set

$$v_{\max} := \max_{\nu=1,\dots,N} v_{\max}^\nu$$

that is, the maximum velocity among all players, as a homogeneous upper physical limit. Further, for all $i = 0, \dots, M-1$,

$$v_h^\nu(t_{i+1}) \le v_{\max}^{\nu,h}(s_h^\nu(t_i)) := \min_{s \in [s_h^\nu(t_i), s_h^\nu(t_i) + v_{\max}^\nu h]} v_{\max}^\nu(s).$$

By this definition, the velocity $v_h^\nu(t_{i+1})$ is limited from above by the smallest maximal feasible velocity along the path section $[s_h^\nu(t_i), s_h^\nu(t_i) + v_{\max}^\nu h]$. The anti-collision constraint (5.32) and the velocity constraint (5.30) are assumed to hold point-wise at each time point of the grid. Therefore, we set

$$g_{\nu\mu}^h(s_h^\nu, s_h^\mu) := d_{\nu\mu} - \max\left\{|s_h^\nu(t_{i+1}) - p_{\nu\mu}|, |s_h^\mu(t_{i+1}) - p_{\mu\nu}|\right\}, \tag{5.45}$$

and hence, the discrete approximation of the penalty term yields,

$$\begin{aligned} P_{MAC}(v_h, s_h) :=& \frac{1}{2}\sum_{\nu=1}^{N}\sum_{\mu\in J^\nu}\sum_{i=0}^{M-1}\max\{0, g_{\nu\mu}(s_h^\nu(t_{i+1}), s_h^\mu(t_{i+1}))\} \\ &+ \sum_{\nu=1}^{N}\sum_{i=0}^{M-1}\max\{0, v_h^\nu(t_{i+1}) - v_{\max}^{\nu,h}(s_h^\nu(t_i))\}. \end{aligned}$$

Then, for the individual contribution of each player we have

$$\begin{aligned} P_{MAC}^\nu(v_h, s_h) :=& \sum_{\mu\in J^\nu}\sum_{i=0}^{M-1}\max\{0, g_{\nu\mu}(s_h^\nu(t_{i+1}), s_h^\mu(t_{i+1}))\} \\ &+ \sum_{i=0}^{M-1}\max\{0, v_h^\nu(t_{i+1}) - v_{\max}^{\nu,h}(s_h^\nu(t_i))\}. \end{aligned} \tag{5.46}$$

Note that the anti-collision constraints for the vehicles ν and μ are symmetric in our setting, and appear once for player ν and once for player μ, which is the reason for the factor $\frac{1}{2}$ in P_{MAC}. Equivalently, using the collision measure in (5.34) yields

$$P(v, s) := \sum_{\nu=1}^{N}\left[\sum_{\mu\in J^\nu}|g_{\nu\mu}(s^\nu, s^\mu)| + \frac{1}{T}\int_0^T \max\left\{0, v^\nu(t) - v_{\max}^\nu(s^\nu(t))\right\}\right]. \tag{5.47}$$

Further, assuming again the constraints hold point-wise at each time point of the grid we have,

$$g_{\nu\mu}^h(s_h^\nu, s_h^\mu) := \frac{1}{M}\sum_{i=1}^{M}\chi_{[p_{\nu\mu}-d_{\nu\mu}, p_{\nu\mu}+d_{\nu\mu}]}(s_h^\nu(t_i))\cdot\chi_{[p_{\mu\nu}-d_{\nu\mu}, p_{\mu\nu}+d_{\nu\mu}]}(s_h^\mu(t_i))$$

and respectively the discrete penalty term is defined by,

$$P_h(v_h, s_h) := \sum_{\nu=1}^{N}\left[\sum_{\mu\in J^\nu}|g_{\nu\mu}^h(s_h^\nu, s_h^\mu)| + \frac{1}{M}\sum_{i=1}^{M}\max\left\{0, v_h^\nu(t_i) - v_{\max}^{\nu,h}(s_h^\nu(t_i))\right\}\right].$$

Accordingly, we define the individual contribution of each player

$$P_h^\nu(v_h, s_h) := \sum_{\mu \in J^\nu} |g_{\nu\mu}^h(s_h^\nu, s_h^\mu)| + \frac{1}{M} \sum_{i=1}^{M} \max\left\{0, v_h^\nu(t_i) - v_{\max}^{\nu,h}(s_h^\nu(t_i))\right\}. \tag{5.48}$$

Remark 5.15. *Due to the collision constraint in (5.34) being defined as an equality constraint the considered setting of (MAC-GNEP) needs to be adjusted. Therefore, we set*

$$g^\nu(v, s) := g_{\nu\mu}(s^\nu, s^\mu) = 0$$

as an equality constraint and aggregate the state dependent velocity constraint together with the individual constraints such that,

$$k^\nu(v^\nu(t), s^\nu(t)) := \begin{pmatrix} v^\nu(t) - v_{\max}^\nu \\ v_{\min}^\nu - v^\nu(t) \\ v^\nu(t) - v_{\max}^\nu(s(t)) \end{pmatrix}.$$

Then, conducting the penalty reformulation and discretization yields (5.47) to (5.48). Note however, that the resulting penalty function is not Lipschitz continuous and therefore, no longer conforms to the theoretical framework of Section 3.2. Hence, subsequently we adhere to the penalty formulation in (5.46).

Adopting the state and control approximations, a discrete approximation of the admissible set on the grid $\mathbb{G}_h$ is obtained for each player $\nu = 1, \dots, N$:

$$\begin{aligned} X_\nu^{MAC,h} := \big\{(u_h^\nu, v_h^\nu, s_h^\nu) \in L_{1,h} \times W_{1,1,h}^2 \mid\ & v_h^\nu(0) = v_0^\nu, \quad s_h^\nu(0) = s_0^\nu, \ \text{ and } \forall i = 0, \dots, M-1: \\ & v_h^\nu(t_{i+1}) = v_h^\nu(t_i) + h(u_h^\nu(t_i) - c_1 v_h^\nu(t_i) - c_2 v_h^\nu(t_i)^2), \\ & s_h^\nu(t_{i+1}) = s_h^\nu(t_i) + h v_h^\nu(t_i), \\ & k^\nu(v_h^\nu(t_{i+1}), s_h^\nu(t_{i+1})) \le (h + h_u)^p, \\ & u_h^\nu(t_i) \in \mathbb{U}_\nu^{h_u^\nu}\big\}. \end{aligned}$$

With these definitions, and a constant $p < 1$, we define the discrete Nash equilibrium problem

$$\min_{(u_h^\nu, v_h^\nu, s_h^\nu)} -s_h^\nu(T) + \frac{1}{(h + h_u)^p} P((v_h^\nu, v_h^{-\nu}), (s_h^\nu, s_h^{-\nu})) \quad \text{s.t.} \quad (u_h^\nu, v_h^\nu, s_h^\nu) \in X_\nu^{MAC,h} \tag{MAC-NEP$_h$}$$

for all $\nu = 1, \dots, N$.

Let us stress, that in compliance with Remark 3.20 we only have a finite number of possible strategies in (MAC-NEP$_h$), since the piecewise constant controls u^ν are only defined on a grid, and hence through the equations also v^ν and s^ν may attain only a finite number of piecewise linear functions.

Besides, the subproblems in $(S.3)$ of Algorithm 5 are to be solved by approximate dynamic programming, to acquire a global solution of (MAC-GNEP). Therefore, we adapt the notation of Section 4.6. Hence, for each player $\nu = 1, \dots, N$ let $t_\ell \in \mathbb{G}_h$ denote the initial date and $z_\ell^\nu = (v_\ell^\nu, s_\ell^\nu)$ the initial state. Then, the discrete feasible set for one time step reads,

$$\begin{aligned} X_\nu^{MAC,h}(t_\ell, z_\ell^\nu) := \big\{(u_h^\nu, v_h^\nu, s_h^\nu) \in L_{1,h} \times W_{1,1,h}^2 \;|&(v_h^\nu(t_\ell), s_h^\nu(t_\ell))^\top = y_\ell^\nu, \\ &v_h^\nu(t_{\ell+1}) = v_h^\nu(t_\ell) + h(u_h^\nu(t_\ell) - c_1 v_h^\nu(t_\ell) - c_2 v_h^\nu(t_\ell)^2), \\ &s_h^\nu(t_{\ell+1}) = s_h^\nu(t_\ell) + h v_h^\nu(t_\ell), \\ &k^\nu(v_h^\nu(t_{\ell+1}), s_h^\nu(t_{\ell+1})) \le (h + h_u)^p\,, \\ &u_h^\nu(t_\ell) \in \mathbb{U}_\nu^{h_u^\nu}\big\}. \end{aligned}$$

Accordingly, for $\ell = 0, \dots, M-1$, the penalty contribution in (5.46) of the ν-th player in the ℓ-th step translates to

$$P_{MAC,\ell}^\nu(x^\nu, x^{-\nu}) := \max\{0, v_h^\nu(t_{\ell+1}) - v_{\max}^{\nu,h}(s_h^\nu(t_\ell))\} + \sum_{\mu \in J^\nu} \max\{0, g_{\nu\mu}(s_h^\nu(t_{\ell+1}), s_h^\mu(t_{\ell+1}))\}.$$

Remark 5.16. *Similarly, for the weighted residency penalty function in (5.48) we have in one time step:*

$$P_\ell^\nu(v_h, s_h) := \sum_{\mu \in J^\nu} \chi_{[p_{\nu\mu}-d, p_{\nu\mu}+d]}(s_h^\nu(t_\ell)) \cdot \chi_{[p_{\mu\nu}-d, p_{\mu\nu}+d]}(s_h^\mu(t_\ell)) + \max\left\{0, v_h^\nu(t_\ell) - v_{\max}^{\nu,h}(s_h^\nu(t_\ell))\right\},$$

as the contribution of the individual player.

Moreover, we define a function for the dynamics in one step to alleviate the subsequent notation:

$$\theta_\nu^{MAC}(u_h^\nu, v_h^\nu, s_h^\nu) := \begin{pmatrix} s_h^\nu \\ v_h^\nu \end{pmatrix} + h \begin{pmatrix} v_h^\nu \\ u_h^\nu - c_1 v_h^\nu - c_2 (v_h^\nu)^2 \end{pmatrix}.$$

Consider the k-th iteration of Algorithm 5 with the abbreviation $x^\nu = (u_h^\nu, v_h^\nu, s_h^\nu)$ and for a fixed strategy of the opposing players $x_k^{-\nu}$. Then, by Definition 4.14 the value function $V : \mathbb{G}_h \times \mathbb{R}^2 \to \mathbb{R}$ of (MAC-GNEP) at time t_ℓ and initial state z_ℓ^ν is recursively defined by

$$V(t_M, z_M^\nu) = \varphi_h(z_M^\nu) = -s_h^\nu, \tag{5.49}$$

$$V(t_\ell, z_\ell^\nu) = \min_{x^\nu \in X_\nu^{MAC,h}(t_\ell, z_\ell^\nu)} \left[P_{MAC,\ell}^\nu(x^\nu, x_k^{-\nu}) + V(t_{\ell+1}, \theta_\nu^{MAC}(x^\nu))\right], \ \forall \ell = M-1, \dots, 0. \tag{5.50}$$

Since we want to employ Algorithm 6 we define the following discretization of the spatial domain for each player ν:

$$h_v^\nu = \frac{v_{\max}^\nu - v_{\min}^\nu}{M_v^\nu}, \qquad \text{and} \qquad h_s^\nu = \frac{T v_{\max}^\nu}{M_s^\nu},$$

with the step sizes of the respective states $h_s^\nu,\ h_v^\nu > 0$ and the number of grid points $M_s^\nu,\ M_v^\nu$. Let us mention, that $T v_{max}^\nu =: s_{\max}$ constitutes the maximum possible reach of a vehicle within one MPC iteration, where we set $s_{\min} = 0$. This leads to the lattice

$$\mathbb{G}_x^{MAC,\nu} := \{v_{\min}^\nu + j h_v^\nu \mid j = 0, \dots, M_v^\nu\} \times \{s_0^\nu + j h_s^\nu \mid j = 0, \dots, M_s^\nu\}.$$

Considering that the second term $V(t_{\ell+1}, \theta_\nu^{MAC}(x^\nu))$ of the value function defined through the Semi-Lagrangian scheme is generally unknown and, therefore, needs to be approximated using an interpolation method. In compliance with (4.22) for $n_x = 2$, the approximate value function at all grid points is recursively defined for all $z_\ell^\nu \in \mathbb{G}_x^{MAC,\nu}$ and $\ell = M, \dots, 0$:

$$\tilde{V}(t_M, z_M^\nu) := \varphi_\nu(z_M^\nu), \tag{5.51}$$

$$\tilde{V}(t_\ell, z_\ell^\nu) := \min_{x^\nu \in X_\nu^h(t_\ell, z_\ell^\nu)} \left[P_{MAC,\ell}^\nu(x^\nu, x_k^{-\nu}) + \sum_{j=1}^{4} \alpha_j \tilde{V}(t_{\ell+1}, \Pi_j[\theta_\nu(x^\nu)]) \right]. \tag{5.52}$$

For the convergence analysis, we will use (5.46) as a penalty function.

Theorem 5.17. *Assume that* (MAC-GNEP) *has a feasible point* (u, v, s). *Further, let* (5.41) *and* (5.42) *to* (5.43) *hold. Then,*

(i) (MAC-NEP$_h$) *has a solution* $(\hat{u}_h, \hat{v}_h, \hat{s}_h)$ *and there exists a constant* $C_{MAC} > 0$ *such that*

$$P_{MAC}(\hat{v}_h, \hat{s}_h) \le C_{MAC}\,(h + h_u)^p$$

for all sufficiently small h *and* h_u.

(ii) for $\|h, h_u\| \to 0$ *the sequence* $\{(\hat{v}_h, \hat{s}_h)\}$ *has an accumulation point* $(\hat{v}, \hat{s})$ *with respect to uniform convergence and there exists a control* $\hat{u}$ *which together with* $(\hat{v}, \hat{s})$ *is a generalized Nash equilibrium of* (MAC-GNEP).

(iii) if X_ν^{MAC} *is nonempty, with a finite number of strategies, Algorithm 5 terminates with a Nash equilibrium of* (MAC-NEP$_h$) *in a finite number of iterations.*

Proof. Assertions (i) and (ii) follow from the general Theorem 3.25 if we verified that all assumptions are met. Note that Assumption 4 directly follows from the setting of the problem. The proof is divided into three steps:

1. First, we show that the required Lipschitz conditions in Assumptions 5 to 6 as well as the monotonicity condition imposed by Assumption 7 are satisfied.
By definition, for all player $\nu = 1, \dots, N$, we have $v^\nu(t) \in W^{1,1}([0,T])$ together with $v^\nu(t) \in \left[v^\nu_{\min}, v^\nu_{\max}\right]$ and $u^\nu(t) \in \left[u^\nu_{\min}, u^\nu_{\max}\right]$ bounded. Hence, the derivative of $(v^\nu)'(t)$ is bounded. Thus, there exists a constant L_v, with

$$\begin{aligned}\operatorname*{ess\,sup}_{t\in[0,T]} |(v^\nu)'(t)| &= \operatorname*{ess\,sup}_{t\in[0,T]} |u^\nu - c_1 v^\nu(t) - c_2(v^\nu(t))^2| \\ &\leq |u^\nu_{\max}| + c_1 v^\nu_{\max} + c_2 (v^\nu_{\max})^2 =: L_v.\end{aligned}$$

Therefore, $v^\nu(t)$, for all player $\nu = 1, \dots, N$ is Lipschitz continuous with a constant $L_v > 0$. Equivalently, for $s^\nu(t) \in \left[s^\nu_0, s^\nu_0 + T v^\nu_{\max}\right]$ and $v^\nu(t) \in \left[v^\nu_{\min}, v^\nu_{\max}\right]$, with

$$|(s^\nu)'(t)| = |v^\nu(t)| \leq v^\nu_{\max},$$

$s^\nu(t)$ is Lipschitz continuous for all $\nu = 1, \dots, N$. Next, we address the Lipschitz assertion of

$$f(u^\nu, v^\nu, s^\nu) := \begin{pmatrix} v^\nu(t) \\ u^\nu(t) - c_1 v^\nu(t) - c_2(v^\nu(t))^2 \end{pmatrix}.$$

Consider, two points $(u^\nu_1, v^\nu_1, s^\nu_1), (u^\nu_2, v^\nu_2, s^\nu_2) \in X^{MAC}_\nu$. Then, due to linearity of f in the first component, it immediately follows,

$$|f_1(u^\nu, v^\nu_2, s^\nu) - f_1(u^\nu, v^\nu_1, s^\nu)| \leq |v^\nu_2 - v^\nu_1|$$

with a constant $L_{\theta,s} := 1$. Accordingly, for the second component, Lipschitz continuity in u^ν follows from the linear dependency of $\theta^{MAC}(u^\nu, v^\nu, s^\nu)$ in u^ν with a constant $L_{\theta,u} := 1$. Then, for the remainder we have

$$\begin{aligned}|f_2(u^\nu, v^\nu_2, s^\nu) - f_2(u^\nu, v^\nu_1, s^\nu)| &= |-c_1 v^\nu_2(t) - c_2(v^\nu_2(t))^2 + c_1 v^\nu_1(t) + c_2(v^\nu_1(t))^2| \\ &\leq c_1 |v^\nu_2(t) - v^\nu_1(t)| + c_2 |(v^\nu_2(t))^2 - (v^\nu_1(t))^2| \\ &\leq c_1 |v^\nu_2(t) - v^\nu_1(t)| + c_2 |(v^\nu_2(t) - v^\nu_1(t))(v^\nu_2(t) + v^\nu_1(t))| \\ &\leq |v^\nu_2(t) - v^\nu_1(t)| \left(c_1 + 2c_2 v^\nu_{\max}\right).\end{aligned}$$

Hence, with $L_{\theta,v} := \left(c_1 + 2c_2 v^\nu_{\max}\right) > 0$ combining the results yields,

$$\|f(u^\nu_2, v^\nu_2, s^\nu_2) - f(u^\nu_1, v^\nu_1, s^\nu_1)\| \leq \max\left\{1, L_{\theta,v}\right\} \|(u^\nu_2, v^\nu_2, s^\nu_2) - (u^\nu_1, v^\nu_1, s^\nu_1)\|,$$

and therefore implies Lipschitz continuity of f in its arguments.
Consequently, Assumption 6 holds. Moreover, Assumption 5 holds since the Mayer term $\varphi(s(T)) = -s(T)$ is Lipschitz with a constant $L_\varphi > 0$. The monotonicity assertion in Assumption 7 follows directly from the linearity of $f(u^\nu, v^\nu, s^\nu)$ in u^ν.

2. Second, we set $x^\nu = (v^\nu, s^\nu)$ and $L_x := \max\{L_s, L_v\} > 0$. Then, with h, h_u sufficiently small and for some constants $C_3, C_4 > 0$, employing Lemma 3.22 yields the estimates for the states:

$$\left\| \begin{pmatrix} v - v_h \\ s - s_h \end{pmatrix} \right\| \leq C_3 (h + h_u),$$

and for the penalty function,

$$P_{MAC}(v_h, s_h) \leq C_4 (h + h_u)$$

Moreover, for sufficiently small h and h_u there exists a feasible point (u_h, v_h, s_h) for (MAC-NEP$_\mathrm{h}$). Then, by Theorem 3.25, since we have a potential game (MAC-GPG), together with Lemma 3.23 and a constant $p < 1$, for $\|h, h_u\| \to 0$ the sequence $\{(\hat{v}_h, \hat{s}_h)\}$ omits an accumulation point $(\hat{v}, \hat{s})$ with respect to uniform convergence and there exists a control $\hat{u}$ such that $(\hat{u}, \hat{v}, \hat{s})$ is a generalized Nash equilibrium of (MAC-GNEP). Moreover, there exists a constant $C_{MAC} > 0$ such that

$$P_{MAC}(\hat{v}_h, \hat{s}_h) \leq C_{MAC} (h + h_u)^p$$

holds. Thus, proving assertions (i) and (ii).

3. Third, since X_v^{MAC} is nonempty, together with Remark 3.20, we can apply Theorem 3.26. Hence, for (MAC-NEP$_\mathrm{h}$), Algorithm 5 terminates in a finite number of iterations with a Nash equilibrium. Eventually, verifying assertion (iii) and thus concluding the proof.

□

This Theorem establishes that the coordination problem (MAC-NEP$_\mathrm{h}$) has a solution, which by application of Algorithm 5 can be obtained in a finite number of iteration. Moreover, if h and h_u tend to zero, said solution converges uniformly to a generalized Nash equilibrium. Hence, to a solution of (MAC-GNEP). With this it remains to show that solving (MAC-NEP$_\mathrm{h}$) in ($S.3$) of Algorithm 5 by means of approximate dynamic programming yields a global minimizer.

Corollary 5.18. *Let the Assumptions 5 to 7 hold. Algorithm 6 omits a global minimizer of* (MAC-NEP$_\mathrm{h}$) *in a finite time for all sufficiently small* $\|(h_v, h_s)\|$.

Proof. As a consequence of the first step of the proof of Theorem 5.17, the conditions imposed by the Assumptions 5 to 7 are satisfied. Consequently, employing Lemma 4.16, for all $\ell = 0, \ldots, M$ the value function $V(t_\ell, \cdot)$ is Lipschitz continuous with a constant $L_V > 0$.

Further, since within the setting of this section, with $n_x = 2$ the approximation of the value function is equivalent to (4.22). Hence, for an arbitrary $t_\ell = \{0, \ldots, M\}$ and any initial state $z_\ell^\nu = (v_\ell^\nu, s_\ell^\nu) \in \mathbb{G}_x^\nu$, Lemma 4.17 yields the following error estimates:

$$|\tilde{V}(t_\ell, y_\ell^\nu) - V(t_\ell, y_\ell^\nu)| \leq L_V (M - \ell) \|(h_v, h_s)\|,$$

and for any $(v_\ell^\nu, s_\ell^\nu) \in [v_{\min}^\nu, v_{\max}^\nu] \times [s_0^\nu, s_0^\nu + T v_{\max}^\nu]$

$$|\tilde{V}(t_\ell, y_\ell^\nu) - V(t_\ell, y_\ell^\nu)| \leq L_V(M + 1 - \ell)\|(h_v, h_s)\|.$$

These estimates together with Theorem 4.21 proves the assertions and thus completes the proof. □

Recapitulating, in this section a multi-agent coordination problem (MAC-GNEP) was modeled as a differential game. Moreover, with the penalty reformulation of Section 3.2.1 of the non-convex GNEP and a discretization in time and control, we obtained the discrete (MAC-NEP$_h$), which was characterized as a potential game. Furthermore, utilizing the results of Section 3.2.4, we have shown that an accumulation point of a sequence of solutions of (MAC-NEP$_h$) yields a solution of (MAC-GNEP). The solution of (MAC-GNEP) can be obtained using the decomposition method in Algorithm 5, which as a consequence of Theorem 5.17 terminates in a finite number of iterations. Therein, (MAC-NEP$_h$) needs to be solved globally in every iteration. Given a suitable approximate value function, (5.51), in combination with Algorithm 6 and a sufficiently small discretization of the spatial grid, by Corollary 5.18 the computation of a global minimizer of (MAC-NEP$_h$) in finite time is possible. Thus, we have shown that the algorithmic and theoretical framework derived in Section 3.2 and Section 4.6 can be fruitfully applied to establish criteria for convergence to a solution of (MAC-GNEP). Now, it remains to validate the theoretical results numerically and experimentally.

5.5.2 Numerical Evaluation: Multi-Agent Coordination

In everyday traffic, provided there are no traffic rules, vehicles are inclined to stop in the event of a potential conflict, for example before entering an intersection or merging into another road. Eventually, causing the flow of traffic to subside, which should be avoided at all cost. To do so, in the game theoretical approach established in previous section, a consensus among the vehicles is sought by solving (MAC-GNEP), i.e., the generalized Nash equilibrium problem which characterizes the conflict among the vehicles. To this end, Algorithm 5 is embedded in a model predictive control setting. More precisely in (S.1) of every iteration of Algorithm 9, Algorithm 5 is evaluated and therein the subproblem of each player (MAC-NEP$_h$) is solved using Algorithm 8. To this end, the preview horizon is set to $t_f = T = 5.0\ \mathrm{s}$ and the initial date is $t_0 = 0.0\ \mathrm{s}$. As a penalty parameter, without impairing the convergence theory, a fixed value of $\rho = 10^6$ is chosen. The safety distance $d_{\nu\mu} = 8\ \mathrm{m}$ is selected to be roughly twice the length of a car in order to ensure a realistic spacing behavior of the vehicles. Within our algorithm, the initial position of each vehicle on its a priori assigned path, with respect to the preview horizon, provides at least one iteration free of conflicts, which can be used as an initial guess for the decomposition algorithm with penalty selection. Where this is uncertain, as a starting guess, the vehicles can solve their individual unrestricted (MAC-NEP$_h$). The resulting trajectories then can be used as starting point to compute a collision free Nash equilibrium coordination of the system.

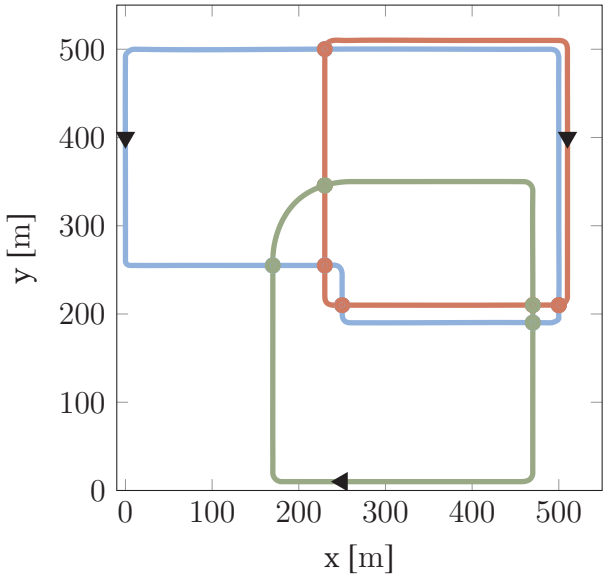


Figure 5.8: *Traffic scenario for the model predictive coordination via Algorithm 5, showing the a priori assigned paths of the vehicles (Vehicle 1: blue; Vehicle 2: red; Vehicle 3: green). Circular markers indicate possible conflict points and starting positions are indicated by triangles.*

The algorithmic framework is implemented in a C++ program, which later allows for a better comparison with the experimental results since the vehicles are not equipped with a GPU. The environmental parameters influencing the velocity are set to $c_F = 0.01$ and $c_D = 0.0024$. The computations are executed on a machine with the hardware settings given in Table A.1. The exemplary scenario is illustrated in Figure 5.8, with the possible points of conflict $p_{\nu\mu}$ and $p_{\mu\nu}$ precomputed by spline intersection. The discretization parameters are summarized in Table 5.1.

Table 5.1: *Domain and Discritization Parameters for Vehicle Coordination.*

	States and Controls			
	Velocity $\mathrm{m\,s^{-1}}$	Arc Length m	Acceleration $\mathrm{m\,s^{-2}}$	Time s
min	0.0	0.0	-5.0	0.0
max	25.0	1000.0	5.0	5.0
h	0.4	5.0	0.2	0.2

Within this scenario $N = 3$ vehicles, respectively player are considered. Through their physical constraints, the vehicles can be configured individually, leading to a heterogeneous system and allowing to model different types of vehicles. To this end, the configuration for the initial position and maximum velocity are given in Table 5.2, and the initial velocity for all vehicles is set to $5.0\ \mathrm{m\,s^{-1}}$.

Table 5.2: *Model parameters.*

	Vehicle			
	1	2	3	Unit
$v_{\max}$	20.0	20.0	15.0	$\mathrm{m\,s^{-1}}$
s_0	80.0	350 .0	500.0	m

The system is simulated for $T = 60$ s, with $\Delta = 0.2$ s. Where the driving strategies are not adjusted or regulated by traffic rules, the vehicles approaching a conflict point would indubitably collide. Moreover, in actual traffic, deadlocks can occur if for instance in a four way crossing a vehicle arrives at each lane at exactly the same time. On that note, in Figure 5.9 the computed optimal driving strategies characterized by the velocity profiles and arc length with respect to time are displayed. While the vehicles are far away from any point of conflict, i.e, within the preview horizon no conflict and therefore no penalty emanates, it can be observed that the vehicles accelerate in order to gain velocity and maximize their reach. At some point all vehicles reach their maximum velocity and proceed to drive with approximately constant speed.

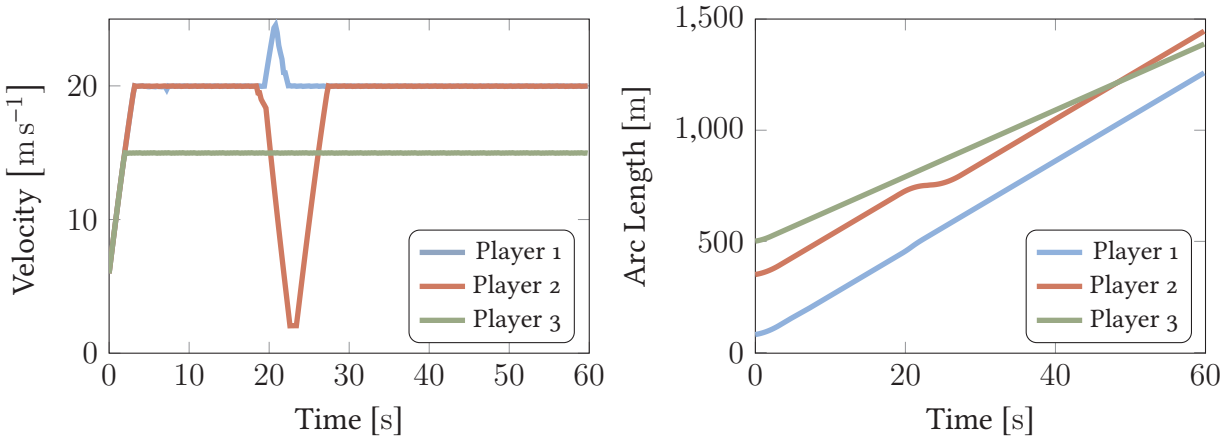


Figure 5.9: *Velocity profiles (left) and arc length (right) progression for the collision free, model predictive vehicle coordination using Algorithm 5.*

Due to the setting, vehicle 1 encounters a conflict with vehicle 3 and then immediately afterwards with vehicle 2, in the center of the map. Moreover, if vehicle 2 would pass the collision points with vehicle 1, a conflict emanates with vehicle 3. Consequently, either of the involved conflicting players has to break or suitably accelerate to avoid a collision. To resolve the multitude of conflicts, vehicle 1 is able to avoid a collision with vehicle 3 by accelerating and therefore violating its velocity constraint. Let us elaborate on that. Since the objective is to maximize the reach, an additional acceleration which in turn increases the velocity and therefore the reach balances the scale with a penalty for the violation of the velocity constraint. Therefore, an acceleration is the more attractive violation then a collision in this scenario. On the other

hand, the conflict with vehicle 2 awaits. Peculiarly, since vehicle 1 now has greater speed it is able to pass the collision point with vehicle 2 much faster. Therefore, it is suitable for both that vehicle 2 decelerates, compare Figure 5.10, and therefore resolves the conflict. Moreover, this deceleration entails a conflict avoidance of vehicle 2 with vehicle 3 in the preceding collision point.

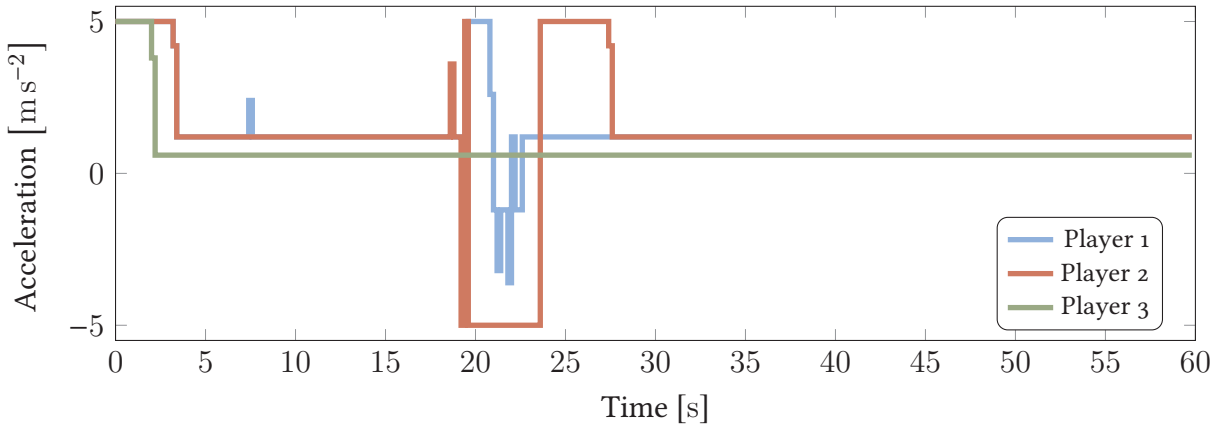


Figure 5.10: *Control trajectories as a result of the model predictive vehicle coordination using Algorithm 5.*

Let us further address the optimality condition of our problem. On the premise that not all vehicles are stationary, if neither a collision nor a violation of the velocities remain, which is equivalent to a zero penalty, a Nash equilibrium is found. Indeed, as can be observed in Figure 5.11, which shows the penalty evolution over several MPC iterations of Algorithm 5, the penalty goes to zero for all player $\nu = 1, 2, 3$.

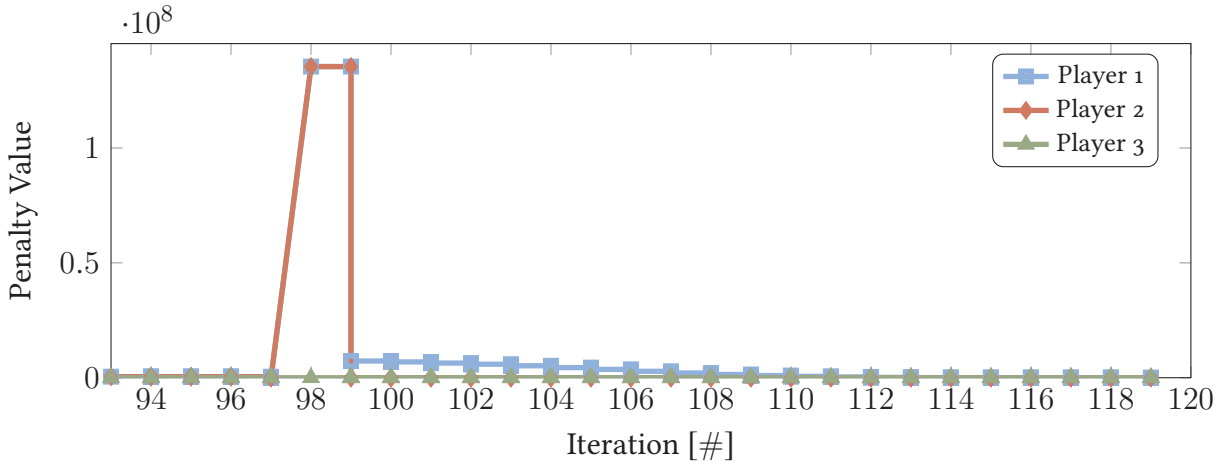


Figure 5.11: *Excerpt of the evolution of the penalty values, with penalty parameter $\rho = 10^6$ over the course of the combined iterations of Algorithm 5 and Algorithm 9 are displayed.*

Quintessentially, in each evaluation of Algorithm 5 the players try to resolve their inflicted penalties by adjusting their velocity profiles accordingly. All possible velocity profiles, sampled by the respective players are displayed for the given preview horizon in Figure 5.12. Peculiarly, it can be observed that all players pursued various acceleration and deceleration strategies. Moreover, it is a game of balance in it self. Revisiting the penalty evolution in Figure 5.11 it becomes apparent that the velocity penalty of vehicle 1 is resolved only after a view iterations. As explained above, as long as the gain in reach balances the scale with the magnitude of violation, a minor penalty is tolerated by the individual player. Yet at some point this scale tips and the player is forced to resolve this minor penalty, which in the end was the case.

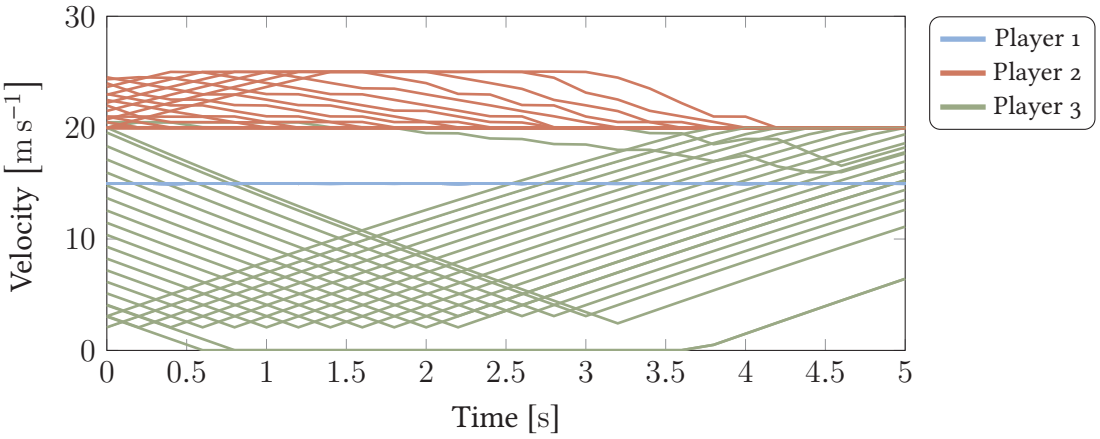


Figure 5.12: *Potential velocity profiles of the players to reduce their penalty contribution. Computed over multiple MPC steps.*

Remark 5.19. *Let us comment on the evaluation:*

- *Note, that the adaption of a driving strategy of one vehicle inherently affects the penalty of the conflicting parties. Therefore, as the name coupling constraints aptly describes reducing the collision penalty term of one vehicle also reduces the penalty term of at least one other vehicle. Let us mention that in general, one iteration of Algorithm 5 can yield a reduction of the penalty for some players, whereas it grows for others. In particular, if a conflict is resolved a driving strategy for one vehicle is adapted which yields a reduction of the penalty contribution associated with the collision of the involved vehicles. Concurrently the driving strategy can introduce a new conflict which induces a gain in the penalty of another vehicle, if the resulting penalty of the adopting vehicle is lower than its previous penalty.*
- *We numerically demonstrated that the algorithmic and theoretical framework presented in this thesis can be applied to resolve conflicts arising in traffic management scenarios by means of Nash equilibrium coordination.*
- *Likewise, scenarios with $N > 3$ player are possible. However, in order to retain a conceivable discussion of the interleaving mechanics the scenario was limited to $N = 3$ player.*

5.6 Experimental Validation

The algorithmic framework has so far been investigated on the premise of an idealized environment. In particular, an exact conversion of the numerically computed velocity profiles, accurate tracking of the prescribed paths, an immaculate exchange of information among the players, as well as a perturbation free measurement of the vehicle's position and velocity. Although these assumptions are both appropriate and customary for numerical simulation, when it comes to practical applications, they are unlikely to sustain. Therefore, by means of experimental investigations conducted with mobile robots, we will validate whether the developed framework is able to guarantee collision-free coordination under perturbations induced by measurement inaccuracies and imperfect actuation.

5.6.1 Implementation and Testbench

A number of intricacies and challenges emanate if a practical implementation and realization of the coordination problem in Section 5.5.1 is pursuit. In this context, the challenges can be associated with different levels of control and consequently be addressed independently. In particular, the approach combines a high-level controller concerned with the generation of trajectories, which ensure the collision free coordination of the vehicles, with a low-level controller for the tracking of prescribed paths as well as for the computed velocity profiles.

Challenges on the high-level controller:

- online computation of collision free trajectories in sufficient time
- robustness with respect to imperfect information and communication

Challenges on the low-level controller:

- accurate tracking of the prescribed paths
- timely realization of the computed velocity profiles
- robustness with respect to measurement inaccuracies or imperfect information

The control structure of the experimental setup and the interaction of the higher and lower level are schematically visualized in Figure 5.13. In the higher level a generalized Nash equilibrium of (MAC-GNEP) is computed by online execution of Algorithm 5 embedded into a model predictive control framework. At first each vehicle commences with an initialization phase, where driving paths are prescribed and for the current (starting) position a penalty distribution based on free driving, i.e., solving (MAC-NEP$_h$) by neglecting the collision constraints, is computed. Once the initial penalty distribution is acquired the online optimization is initiated, which provides collision free velocity profiles in terms of a Nash equilibrium solution. The computed MPC trajectories are then relayed to the lower level, the path tracking controller. The measured vehicle metrics are transmitted to the respective subsystems simultaneously and the online MPC is

executed with a fixed frequency using the current states of all vehicles. Let us stress, that the higher and lower level are executed with different frequencies, which allows to deploy them on distinct hardware platforms. The high level coordination control runs on a dedicated server, with the specifications given in Table A.2, while the lower level controller is executed on a real-time operating system directly on the individual vehicles. In order to provide a heterogeneity of the vehicles and therefore a higher degree of authenticity, two different type of path tracking controllers are implemented on the vehicles, in particular a linear model predictive control (LMPC) and a Dynamic Inversion controller, which track the prescribed path with respect to its curvature. The path tracking controllers are discussed in more detail in the respective publications, i.e., [Britzelmeier and Gerdts, 2020] and [Britzelmeier et al., 2020a], produced in the course of this work.

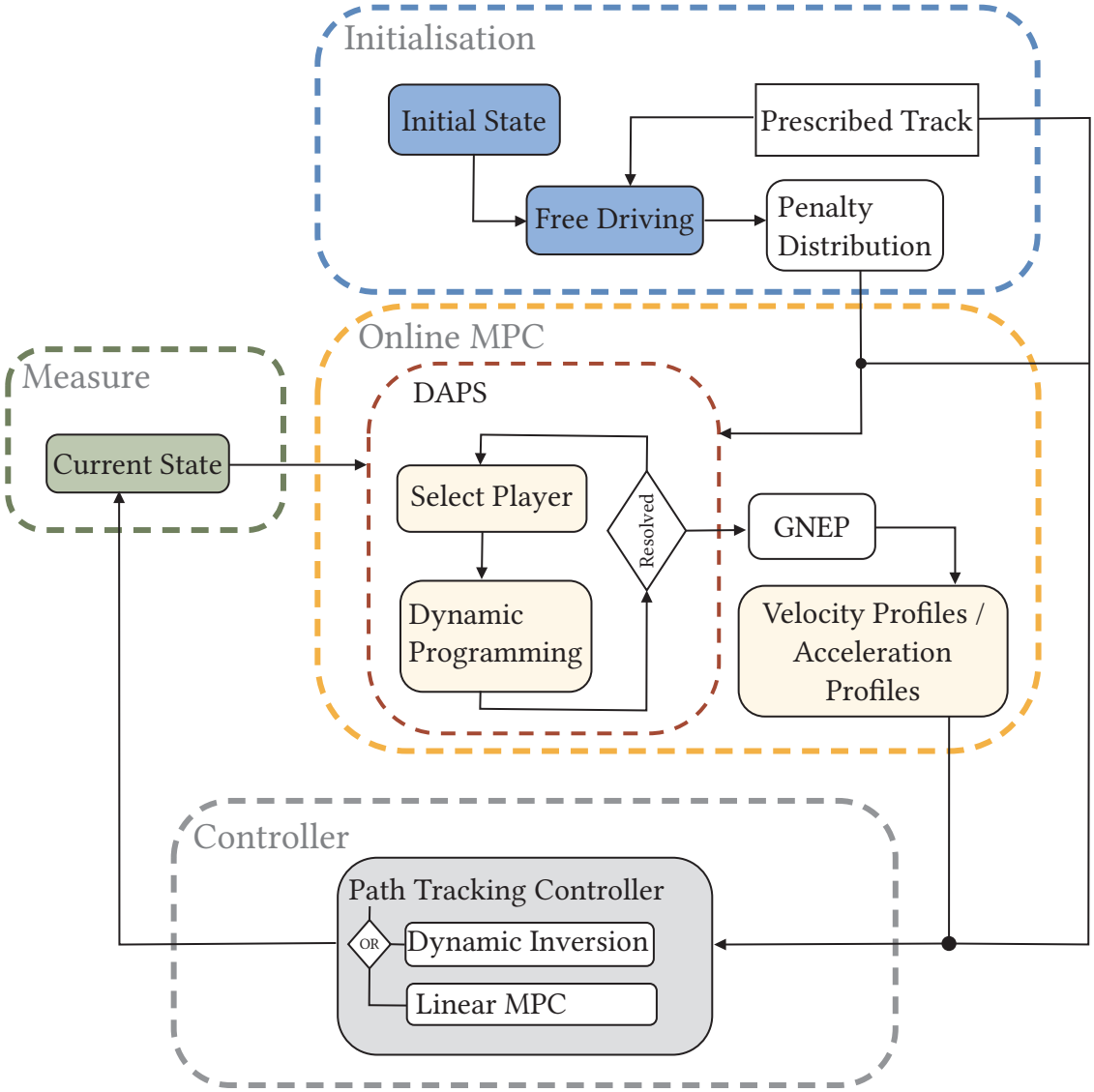


Figure 5.13: *Illustration of the schematic control structure for the experimental testing.*

For the implementation and experimental validation of the algorithmic framework nonholonomic robots in a laboratory testing environment are employed. The test bench covers an area of $3.5\ \mathrm{m}$ by $3.2\ \mathrm{m}$. Further, a *HTC Vive VR System* is installed in order to track the metrics of the vehicles, such as position and velocity. Therefore, in order to cover the testing area two *HTC Vive Lighthouses rev. 2.0* are mounted in opposing corners with a down facing angle. The tracking in the x-y plane is accurate up to $1\ \mathrm{cm}$ according to the findings of [Niehorster et al., 2017]. The configuration of the test bench is illustrated in Figure 5.14.

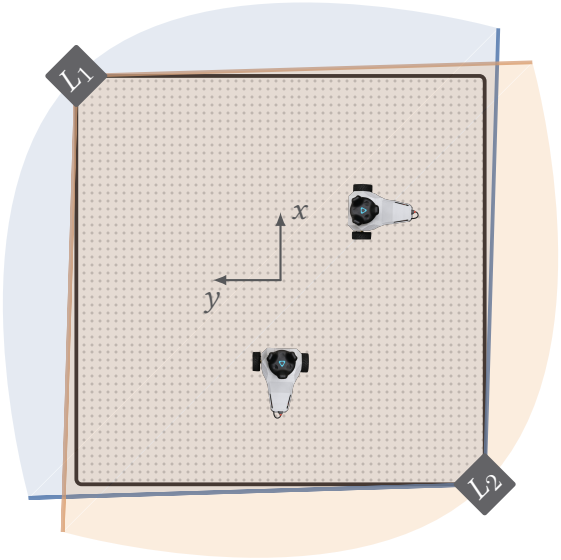


Figure 5.14: *Experimental setup showing the testing area (dotted), vehicles and the lighthouses L_1, L_2 with their respective measuring cones and orientation.*

A vehicle in the cloud approach is pursuit to aggregate the tracking data and manage the communication and control of the system. In particular, the implemented vehicle cloud (High Level Controller) is interfaced to the positioning system using Steam VR in conjunction with Open VR to acquire the tracking data with a frequency of $50\ \mathrm{Hz}$. The system control and communication is realized through a primary and secondary transmission control protocol (TCP) server in the cloud. Each vehicle needs to register in the cloud in order to acquire position data and control instruction. The primary server is tasked with the registration and monitoring of the connection status of each vehicle. Moreover, additional control instructions, e.g., abort signals, can be issued to the vehicles through the primary server. The secondary server on the other hand is responsible for data distribution and acquisition among the registered vehicles. Essentially, a request and response message exchange pattern is implemented to manage the communication. In case a registered vehicle requests a set of data, the secondary server provides the latest measured positions, angles and velocities. Moreover, the algorithmic framework, i.e.,

Algorithm 5 embedded intoAlgorithm 9 is implemented in the vehicle cloud and provides collision free trajectories to coordinate the vehicles by means of Nash equilibria. The MPC trajectories obtained by this procedure will then be relayed to the vehicles through the secondary server.

The nonholonomic robots developed for the experimental validation are driven by two 6 V direct current (DC) motors equipped with a transmission and a rotary sensor. Hence, the dual-wheeled robots are steered with respect to the difference of the velocities on their right and left wheel, respectively $v_r(t)$ and $v_l(t)$. Hence, we have for the longitudinal velocity,

$$v(t) = \frac{v_r(t) + v_l(t)}{2}.$$

The path tracking and velocity controller (lower level controller) are running on a *Raspberry Pi 3B+* compute unit with a patched Raspbian operating system to achieve real time capabilities. The control of the desired velocity computed by the high level controller is implemented as a proportional controller. Additionally, to the distinct path tracking controllers, the mobile robots are equipped with different motor controller boards. A *Phidget* DC motor controller board which is connected to the Raspberry Pi through a universal serial bus (USB) and a *Adafruit DC Stepper Motor HAT*, which uses the inter-integrated circuit (I^2C)-bus of the compute unit and is physically connected by general purpose input/output (GPIO) pins. Consequently, different motor controller boards lead to distinct acceleration properties and therefore evoke individual driving characteristics.

Remark 5.20. *The data and communication stream and the state monitoring stream have been separated to provide a modular system, which allows for flexibility and simple adaption. Essentially, by means of this communication structure, the protocols of each stream can be changed independently from TCP to user datagram protocol (UDP) and the position services can be substituted by services like global positioning system (GPS) or Galileo.*

Despite Algorithm 5 terminating in a finite number of iterations, the computation time in online control plays a significant role. More precisely, while the optimization is executed the vehicles progress on their track according to the control input from the previous MPC step. If the successive MPC step is not completed in a time frame, which is shorter than or equal to the time frame for which the previously applied control is valid, discontinuities in the trajectories occur and the collision avoidance might no longer be guaranteed. As a fall back mechanism, if the computation time exceeds the length of the MPC step, the previously computed trajectories are continued and Algorithm 5 is restarted with the current vehicle states.

5.6.2 Evaluation

The theory developed in Chapter 3 and Chapter 4 is set under the assumption of a perfect environment. Hence, influences of perturbations and inaccuracies are not considered. The challenges on the higher and lower level controller discussed in the previous section therefore

may lead to failure of the algorithmic framework. A scenario with $N = 2$ vehicles is considered to evaluate whether the algorithmic framework and its implementation can cope with these challenges. The experimental scenario with the a priori assigned tracks, initial positions and possible collision points is illustrated in Figure 5.15.

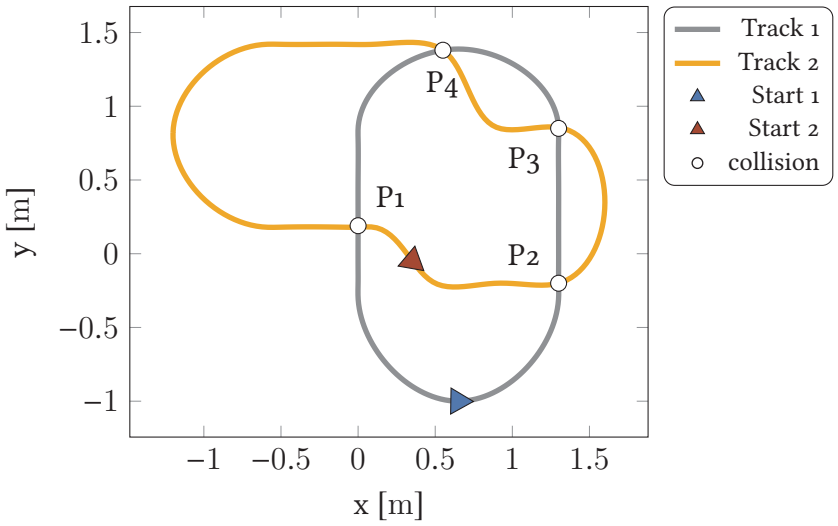


Figure 5.15: *Illustration of the tracks, starting positions (triangles) and points of conflicts (P1–P4) of the respective players.*

The first vehicle starts at an arc length $s_0 = 2.0$ m, while the second starts at $s_0 = 0.0$ m. The tracks intersect at four possible points of conflict $p_{\nu\mu}$ and $p_{\mu\nu}$ with $\nu, \mu \in \{1, 2\}$ and $\nu \neq \mu$, which are listed in Table 5.3.

Table 5.3: *Arclength of the points of conflict with respect to the track of the player.*

	Points of Conflict			
	P1	P2	P3	P4
p_{12}	3.02 m	4.32 m	5.37 m	1.55 m
p_{21}	1.13 m	2.21 m	3.20 m	4.55 m

The safety distance $d_{\nu\mu}$ is set to 10 cm for the collision points P1 and P2 and $d_{\nu\mu} = 20$ cm for P3 and P4, which equals the length of the mobile robots. Employing two different safety distances allows to emulate adaptive driving scenarios and to analyse if the algorithm is able to cope with alternating safety distances between two collision points within a short range. Moreover, a preview horizon of $t_f = T = 3.0$ s is chosen, with a step size of $\Delta = h = 0.1$ s. To synchronize the time with the position of the two vehicles a joint initial time $t_0 = 0.0$ s is set. Note that the vehicles not necessarily need to start with the same initial date, this merely eases

the comparison of the data later on. Further, the parameters characterizing the domain and the discretization of the vehicles are summarized in Table 5.4. In compliance with the convergence theory, for $p = 0.99$ the penalty parameter utilized in the subsequent experiment amounts to

$$\rho = \frac{1}{(h + h_u)^p} = \frac{1}{(0.1 + \min\{0.04, 0.24\})^{0.99}} \approx 7.00.$$

As a consequence of the combination of different controller boards and tracking controllers, compared to the offline simulation in Section 5.5.2, a heterogeneous system of vehicles emanates naturally. Here, the first vehicle is equipped with the *Adafruit* motor control HAT together with the LMPC tracking controller. The *Phidget* motor control board is installed on the second vehicle which runs the dynamic inversion controller.

Table 5.4: *Discretization parameters of the mobile robots for the experimental setting.*

		States and Controls			
		Velocity $\mathrm{m\,s^{-1}}$	Arc Length m	Acceleration $\mathrm{m\,s^{-2}}$	Time s
Vehicle 1	min	0.0	0.0	-1.0	0.0
	max	0.2	10.0	1.0	3.0
	h	0.01	0.2	0.04	0.1
Vehicle 2	min	0.0	0.0	-10.0	0.0
	max	0.2	10.0	2.0	3.0
	h	0.01	0.2	0.24	0.1

Altogether, each vehicle completed at least three laps on the test course. Due to the emergence of two successive conflicts at the onset of the first lap, whose unravelling led to a shift in the relative positions of the vehicles to each other, hardly any collisions or significant control interventions ensued in the subsequent laps. Therefore, we will concentrate the ensuing discussion on the conflicts that arose at the beginning. In Figure 5.16, a segment of the online computed collision free driving strategies of the aforementioned conflicts, which are characterized by the velocity profiles in contrast to the tracked velocities is displayed with respect to time. While the vehicles are far away from any point of conflict, i.e., within the preview horizon of the first couple MPC iterations no penalty due to a potential collision emanates. The vehicles proceed to drive with their maximum velocity in order to maximize their reach. While both vehicles manage to track on average the constant maximum velocity, substantial oscillation in the resulting velocity profiles can be observed. Moreover, the first vehicle seems to fall slightly short of the commanded velocity. In comparison, while the second vehicle fares far better in tracking the commanded velocity, it also shows larger magnitudes within the oscillations, thus emphasizing the influence and differences in hardware.

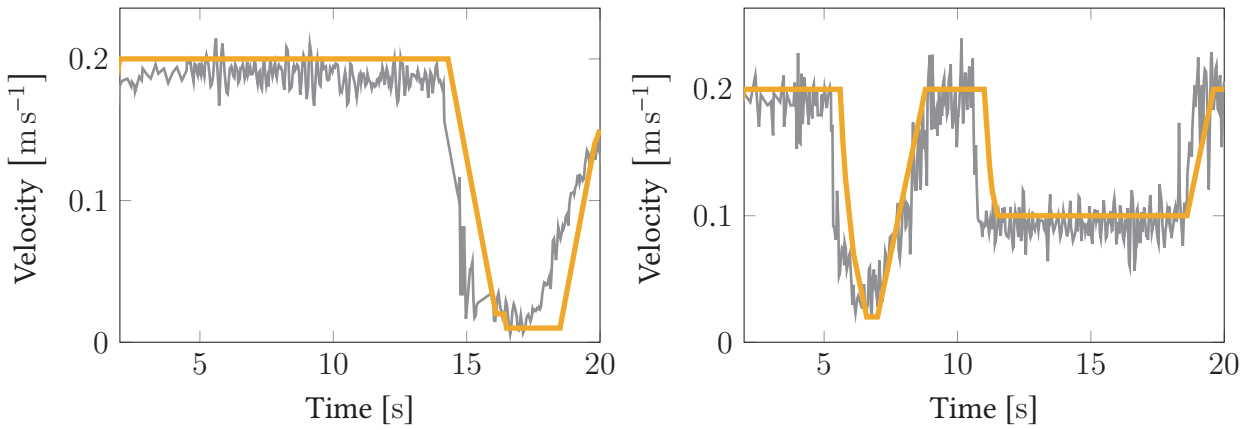


Figure 5.16: *Comparison of the online computed velocity profiles (orange) and the actual tracked results (gray) for vehicle 1 (left) and vehicle 2 (right) and a time period of* 20.0 s. *The velocity profiles show the behaviour of the two players around collision points P2 and P3. For a preview horizon of* $T = 3.0$ s *and a time shift of* $\Delta = 0.1$ s.

The first conflict emanates at collision point P2 and then immediately afterwards at P3. Both vehicles approach the point of conflict with roughly the same speed. Consequently, either of the involved players needs to break or suitably accelerate in order to avoid a collision. Due to the greater acceleration and deceleration capabilities of the second vehicle together with the shorter safety distance a brief but strong breaking manoeuvre followed by an extensive acceleration allows to resolve the conflict. Since the objective of both vehicles is to maximize their reach without violating the velocity constraints, the cost for the first vehicle to decelerate is greater compared to a breaking manoeuvre of the second vehicle due to its greater dynamic range. In particular, the second vehicle is able to return faster to its maximum speed after a breaking manoeuvre. However, the second conflict at P3 immediately ensues. Considering, the larger safety distance a reaction of either of both vehicles is required sooner. Peculiarly, as a consequence of the strategy to resolve the first conflict in conjunction with the shorter distance the first vehicle has to pass to reach P3, it is already closer to the point of conflict than the second vehicle. Therefore, the only reasonable strategy is for the second vehicle to decelerate again. The influence of the larger safety distance on the dynamic behaviour is reflected in the velocity profile of the second vehicle. While the magnitude of deceleration is far lower compared to the first conflict, the second vehicle remains longer at a lower velocity to abide the larger safety distance.

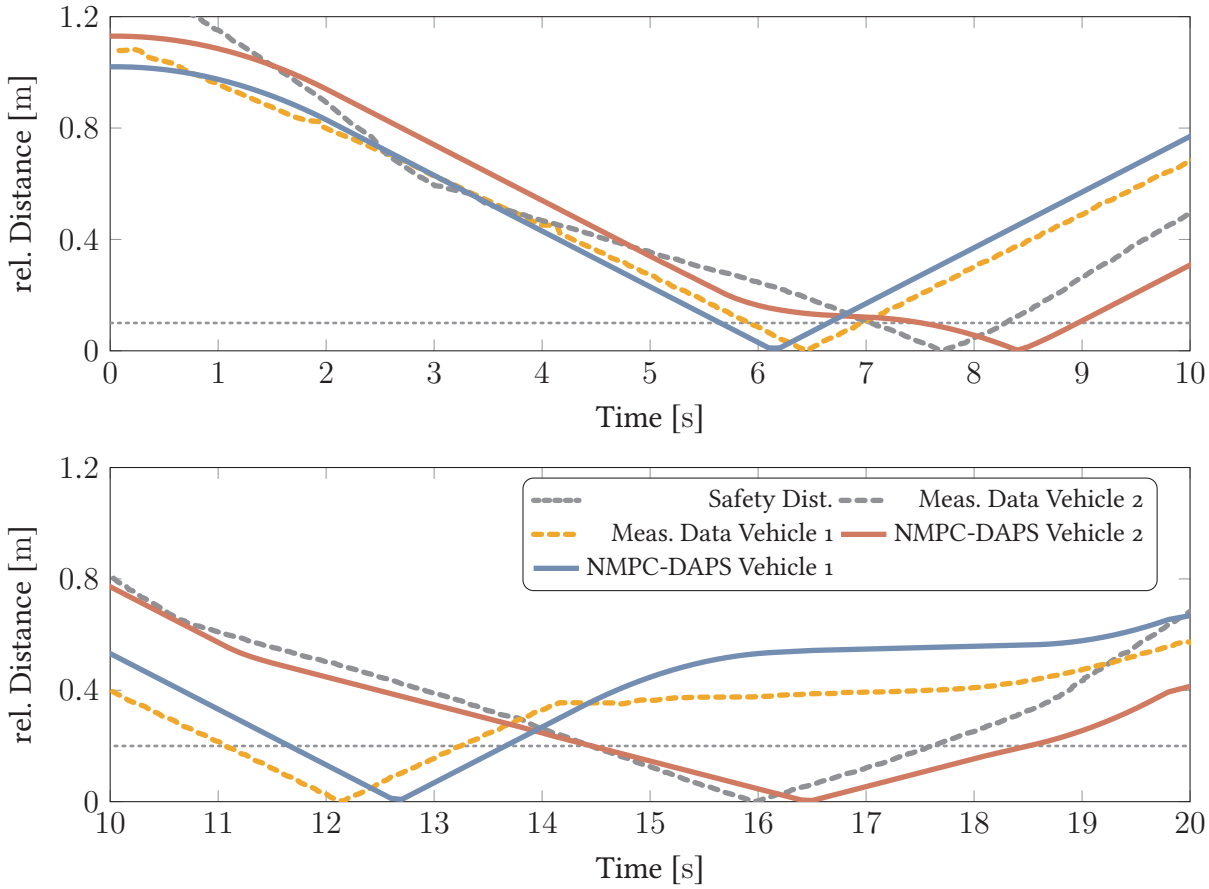


Figure 5.17: *The relative distances in terms of the arc length of the vehicles to collision points P2 (top) and P3 (bottom). The data shows the change in distance to the collision point of the computed MPC trajectory and the tracked positions over time.*

In Figure 5.17, the relative distances to the collision points, respectively P2 and P3, in terms of the arc length of each vehicle is depicted. It can be observed that despite perturbations the respective safety distances (dotted horizontal line) to the collision points P2 and P3 are upheld and therefore no collision occurs. This is supported by the vanishing penalties in Figure 5.18. Moreover, a deviation at the collision points between the computed solution and the measured vehicle data can be noticed. For once, it can be observed that at the start of the considered time frame a slight deviation in the position of the vehicles already exists. Additionally, inaccurate measurements of the velocity and position contribute further to a delay in the real driving path.

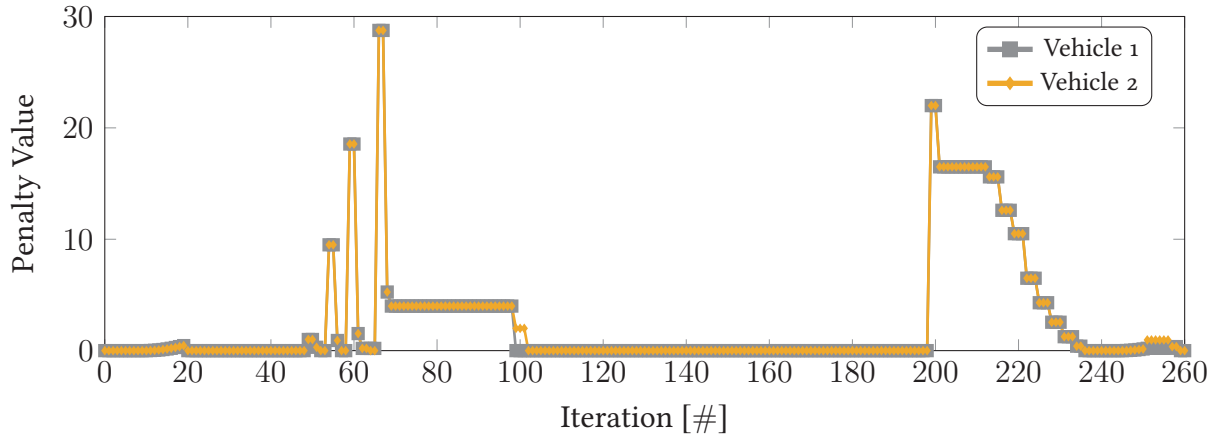


Figure 5.18: *Evolution of the penalty values over the course of the model predictive control iterations. The number of iterations exceed the number of MPC iterations (200), since the iterations within the execution of Algorithm 5 are included.*

Let us now elaborate on the performance of the control structure. To this end, the computation times of the individual MPC steps are collected in Figure 5.19. One of the major challenges discussed above is concerned with the capability of the algorithmic coordination framework to provide a solution within sufficient time. The high level controller is executed with a frequency of 10 Hz, while the low level controller runs with 50 Hz. Analyzing the computation times, we notice the time limit of 0.1 s to be exceeded only once. Throughout the other iterations, the computation times remain well below this threshold. Moreover, in the singular case where no solution is found in time, the subsequent iteration is solved within about 0.084 s. Consequently, the proposed algorithmic framework seems to be able to compute collision free trajectories in sufficient time despite imperfect information.

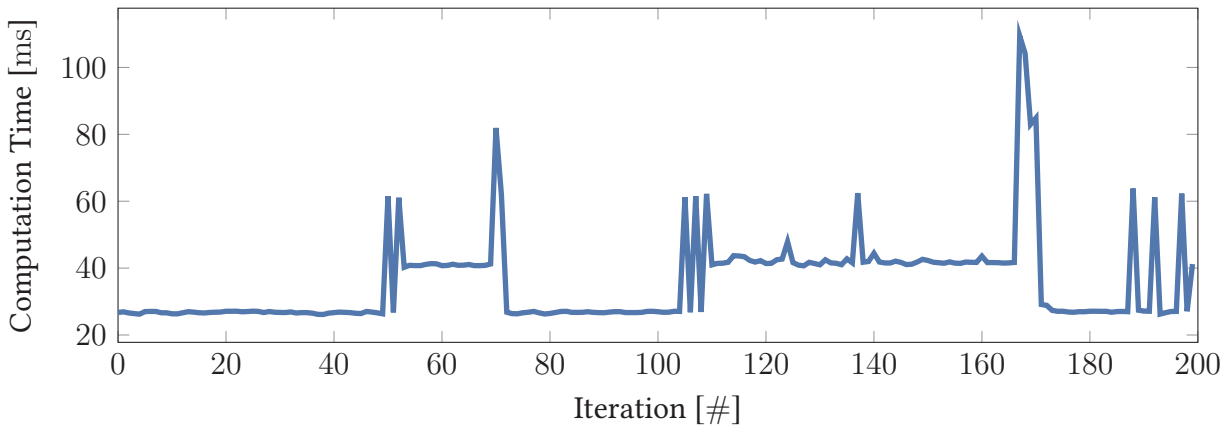


Figure 5.19: *Computation time of each model predictive control iteration. The execution times represent the cumulative times, i.e., the sum over all vehicles and iterations of Algorithm 5.*

With respect to the low level controller, we have established that it is able to accurately track the commanded velocities. Despite, measurement inaccuracies and oscillations the proportional controller is able to realize timely accelerations and deceleration. Moreover, both path tracking controllers are able to track the prescribed paths with sufficient accuracy and at a peak lateral deviation of 2.64 cm, see Figure 5.20.

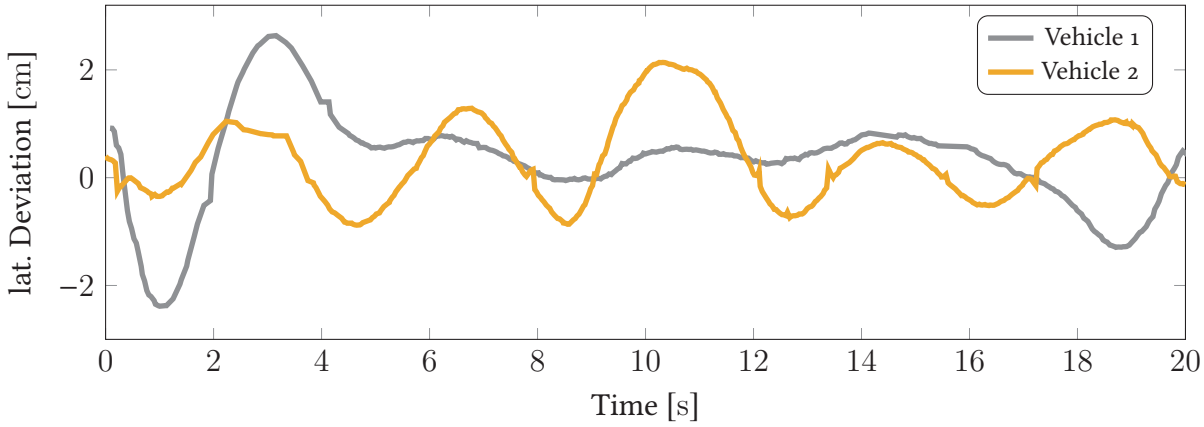


Figure 5.20: *Lateral deviation from the tracked path with respect to the time, for vehicle 1 (gray) and vehicle 2 (orange).*

Remark 5.21. *The evaluation of the conducted experiments together with the numerical analysis have shown that embedding Algorithm 5 and Algorithm 6 into a model predictive setting, provided by Algorithm 9, yields an efficient and robust algorithmic framework. Even though the theoretical foundation does not account for perturbations of any kind, we established that our implementation is suited to meet the requirements and challenges of online computation despite measurement inaccuracies and loss of information. Moreover, the benchmarks of the GPU implementation indicate that the performance for online computation can be vastly improved, by employing (PARSEL-DP), provided the respective hardware is available. In addition, the performance of (PARSEL-DP) with respect to online path planning indicates that our algorithmic framework could be extended with a low frequency path planning algorithm, thus, rendering the a priori assignment of paths obsolete. We conclude, that with respect to their performance and robustness, the presented algorithms are well suited for the collision-free online coordination of (autonomous) vehicles on the basis of Nash equilibria.*

6 | Conclusion

> *Every day you may make progress. Every step may be fruitful. Yet there will stretch out before you an ever-lengthening, ever-ascending, ever-improving path. You know you will never get to the end of the journey. But this, so far from discouraging, only adds to the joy and glory of the climb.*
>
> — *Winston S. Churchill*

In this thesis, we studied a class of differential generalized Nash equilibrium problems subject to nonconvex shared constraints and contrived general methods for their numerical solution based on a framework which fruitfully combines partial penalization, decomposition methods and dynamic programming, thereby extending results in the literature on decomposition methods and the convergence analysis of generalized Nash equilibrium problems.

In Chapter 3 the nonconvex differential generalized Nash equilibrium problem (GNEP) was inaugurated and a relation to generalized potential games (GPG) is established. Then, a partial penalization reformulation was proposed to handle the nonconvex constraints imposed on (GNEP), which led to a penalized standard Nash equilibrium problem (NEP). Convergence of a sequence of solutions of the semi-discrete standard Nash equilibrium problem (NEP_h) to the solution of (GNEP) in the state variables was proven under standard assumptions. Leveraging the structure of the generalized potential game (GPG), we devised a decomposition method, whose subproblems are the penalized standard Nash equilibrium problems and where the order of the players to optimize is determined through their individual contribution to the common penalty term. We were able to show that the algorithm converges in a finite number of iterations. It is conceivable that the proposed partial penalization reformulation in combination with a decomposition method poses a viable framework for other finite dimensional generalized Nash equilibrium problems, which are subject to nonconvex shared constraints.

In Chapter 4, we consider a Hamilton-Jacobi-Bellman approach for solving standard penalized Nash equilibrium problems. Accordingly, a backward Semi-Lagrangian method for the numerical solution of the Hamilton-Jacobi-Bellman equation has been derived. Combined with a suitable state space discretization and interpolation method, an approximate value function

for penalized Nash equilibrium problems, satisfying the principle of dynamic programming (DPP), was constructed. Then, an iterative dynamic programming algorithm was presented, which synthesis the optimal control in feedback form. An error estimate for the approximate value function is devised, based on which the convergence to a global minimizer in finite time is shown. Moreover, an estimation of the convergence rate for a control and time discrete higher order (backward) Semi-Lagrangian scheme was established, evidencing a severe repercussion of the order of the control approximation on the convergence rate. Eventually, exploiting the recursive problem structure of dynamic programming a parallel dynamic programming algorithm (PARSEL-DP), suitable for large-scale problems was developed and implemented together with a batching strategy in a software package for parallel computation on the GPU. A benchmark with standard CPU implementations confirms the algorithm's efficiency and compelling scalability properties.

In Chapter 5, we applied the algorithmic framework towards two tangible problems. A path planning problem concerning a truck parking manoeuvre and a coordination problem involving autonomous vehicles. For this purpose, two curvilinear vehicle models with appropriate collision avoidance metrics were inaugurated. The path planning problem was solved using the PARSEL-DP solver. With respect to the scalability, the numerical results evidenced a compensation of the curse of dimensions to some degree. Moreover, the computation times and the quality of the trajectories were indicative on its viability for online path planning.
Next, we considered the coordination problem, which was modelled as a generalized Nash equilibrium problem subject to nonconvex shared anti-collision constraints. Based on the partial penalty reformulation the GNEP was recast as a penalized Nash equilibrium problem. By adaptation of the theoretical results from Chapter 3 and Chapter 4 we were able to establish criteria for the convergence to a solution of the GNEP in the state variables. Then, the decomposition method (DAPS) together with the approximate dynamic programming algorithm were embedded in a model predictive control scheme. The nested algorithmic scheme was implemented in a C++ routine. Eventually, numerical results evidenced its viability and efficiency in resolving potential collisions by means of Nash equilibrium coordination.

Complementing the numerical analysis, experimental investigations of the coordination problem were conducted. For this purpose, a test bench equipped with a positioning system was set up and nonholonomic dual-wheeled mobile robots were developed. Heterogeneity of the vehicles was established through distinct hardware platforms and tracking controller on the vehicle level. Moreover, a vehicle cloud was implemented, which managed the communication, handled the negotiations by means of our algorithmic framework and provided the control instructions for the vehicles. Experimental results were reported, which demonstrated the robustness and adaptivity of our implementation with respect to perturbations and imperfect information. Even at relatively dense discretization, online computation proved to be feasible in almost all instances.

Outlook

Ensuing from the methods and theoretical analyses developed in this thesis, paths which indubitably lead to future lines of research unravel before us and may contribute in answering questions which emanated along the way.

Chapter 3 combines a partial penalization with a decomposition method by leveraging the potential game structure which was inspired by [Facchinei et al., 2011], to solve nonconvex differential GNEP with shared constraints. An issue that remains unanswered is the convergence of the control variables. Therefore, it would be desirable to directly solve the generalized potential game and characterize a global solution of the potential function. Implicitly this is coherent with selecting a Nash equilibrium, which best fits the potential function. Moreover, it would be intriguing to consider the potential game as a multi-objective optimal control problem, as indicated in [Britzelmeier et al., 2020a], invoking naturally a connection between Nash equilibria and Pareto optimal solutions. Moreover, [Sagratella, 2017] suggests to reformulate the multi-objective optimal control problems as a mixed-integer nonlinear problem with switching cost, by introducing continuous and binary variables to obtain Pareto efficient solutions.

In Chapter 4 we explored a Hamilton-Jacobi-Bellman approach and developed a dynamic programming solver. The inclusion of stochastic uncertainties into the constraint and nonconvex differential game, by extending the framework developed in [Murano and Poznyak, 2004], may result in increased robustness and possibly better solutions, especially with respect to perturbations that occur in the practical implementation. Devising criteria when a Nash equilibrium of a system of second order HJB equations exists is challenging yet an interesting line of future research. With respect to the CUDA implementation of our dynamic programming solver, most of the computation time is lost in the interpolation step. Therefore, one could investigate machine learning algorithms such as supervised learning and convolutional neural networks [Llanas and Sainz, 2006, Niklaus et al., 2017], to further improve the performance and accuracy. Herein, the discretization of the state space plays an important role. Devising an approximation scheme with adaptive step size is a topic for future research.

An application towards collision-free coordination of vehicles was numerically and experimentally investigated in Chapter 5. Adapting alternative objectives such as comfort of the driver or energy consumption can be considered in conjunction with the presented time minimal approach to diversify the spectrum of drivers. Moreover, an implementation and experimental validation of our algorithmic framework in conjunction with the adoption of the developed GPU parallel optimization algorithm towards the coordination of actual vehicles or trucks is of particular interest. Furthermore, path planning under consideration of statistical uncertainties can be considered. In particular, the use of machine learning approaches based on Q-learning or deep reinforcement learning as in [Bertsekas, 2019, Schulman et al., 2015] can be pursued which pose a viable extension to dynamic programming by formulating a Markov decision problem in combination with approximate policy iteration methods. In fact, such an approach would alleviate the constraints of an a priori fixed path and complement our algorithmic framework.

A | Appendix

A.1 Curves

Curves play a significant role in the definition of path tracking and planning problems. Therefore in this section, the essential definitions with respect to curves and a parametrization of a dynamical system in terms of the arc length are discussed. For a more detailed discussion on the topic of curves we refer to [Rudin, 1976, Kühnel, 2013].

Definition A.1. *Suppose $\gamma : I \to \mathbb{R}^n$ is a continuous function defined on the interval $I = [a, b]$ with $a, b \in \mathbb{R}$. Then, $\gamma(I)$ is called a curve.*

A subclass of differentiable curves can be distinguished where γ is a continuously differentiable function. The length of such a curve then follows from

Definition A.2 ([Rudin, 1976, p. 136])**.** *Suppose $\mathcal{X}$ is a metric space with a metric d, then the length of the curve $\gamma : I \to \mathcal{X}$ is defined by*

$$L(\gamma) \coloneqq \sup \left\{ \sum_{i=1}^{n} d(\gamma(t_i), \gamma(t_{i-1})) \mid n \in \mathbb{N}, a = t_0 < t_1 < \ldots < t_n = b \right\},$$

where the supremum is taken over all $n \in \mathbb{N}$ and all partitions t_i, with $i = 1, \ldots, n$ of the intervall $[a, b]$. Moreover if L is finite the curve is called rectifiable.

If in particular $\mathcal{X} = \mathbb{R}^n$ and $\gamma : I \to \mathbb{R}^n$ is injective and continuously differentiable, the arc length s of the curve can accordingly be defined as an integral,

$$s(t) \coloneqq \int_a^t \|\gamma'(\tau)\| \, d\tau,$$

with $t \in [a, b]$. Consequently the total length of the curve amounts to

$$L = s(b) = \int_a^b \|\gamma'(\tau)\| \, d\tau.$$

Furthermore, curves can be parameterized arbitrarily, for example by means of the arc length of the curve. Henceforth, we will consider a curve $\gamma : [0, T] \to \mathbb{R}^n$. For $\xi \in [0, L]$ let $t = s^{-1}(\xi)$. Thus, the arc length parameterized curve $\tilde{\gamma} : [0, L] \to \mathbb{R}^n$ is given by $\gamma(\xi) := \gamma(s^{-1}(\xi))$.

If the corresponding time $t = s^{-1}(\xi)$ is to be calculated for an arbitrary arc length ξ, which is equivalent to reverting to the parameterization in t, the following equation needs to be solved:

$$\xi - s(t) = 0.$$

To calculate the zero, either bisection or Newton's method can be employed.

Transformation of Dynamic Systems

Generally, in optimal control tasks the dynamical systems are time variant functions, yet in some use cases with given reference curves, it is convenient to parameterize the dynamical system in terms of the arc length of the associated reference curve. Therefore, the ordinary differential equations (3.2) describe the time variant evolution of a dynamical system. Further, let the transformation be given by the mapping $s : [t_0, t_f] \to [0, L]$, where $s(t) = \xi$. Suppose the function $s(\cdot)$ is strictly monotonically increasing, then there exists the inverse function $s^{-1} : [0, L] \to [t_0, t_f]$, where $s^{-1}(\xi) = t$.

The derivative of the inverse function s^{-1} then can be acquired through the derivation of the identity $t = s^{-1}(s(t))$, compare [Lot and Biral, 2014]:

$$(s^{-1})'(s(t)) = \frac{1}{s'(t)}. \tag{A.1}$$

Substituting $s(t) = \xi$ yields,

$$(s^{-1})'(\xi) = \frac{1}{s'(s^{-1}(\xi))}.$$

Setting $\tilde{x}(\xi) := x(s^{-1}(\xi))$ and $\tilde{u}(\xi) := u(s^{-1}(\xi))$ together with derivative of the identity leads to the transformed differential equation:

$$\tilde{x}'(\xi) = \frac{1}{s'(s^{-1}(\xi))} f(s^{-1}(\xi), \tilde{u}(\xi), \tilde{x}(\xi)), \tag{A.2}$$

for any $\xi \in [0, L]$.

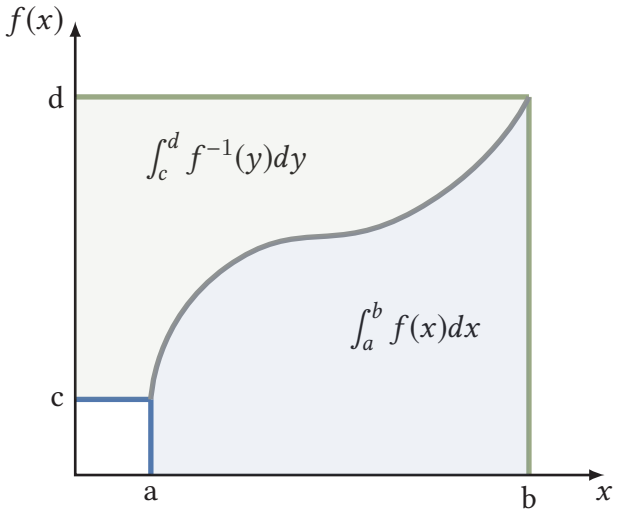


Figure A.1: *Geometric relation between the integrals of a monotone function $f(x)$ and its inverse function.*

In this context, consider a function $f : [a, b] \to [c, d]$, where $f(a) = c$, $f(b) = d$. The geometric correlation between the integral of a function $f(\cdot)$ and its inverse function $f^{-1}(\cdot)$, compare Figure A.1, was originally stated by [Laisant, 1905] under the condition that $f(\cdot)$ is required to be continuously differentiable. In [Key, 1994, Theorem 1], this condition was relaxed, leading to the following theorem:

Theorem A.3. *[Key, 1994, Theorem 1] Suppose that $f : [a, b] \to [c, d]$ is (strictly) monotone and continuous. Then,*

$$\int_c^d f^{-1}(y)dy + \int_a^b f(x)dx = bd - ac. \tag{A.3}$$

A.2 Bilinear Interpolation

As the name aptly reveals bilinear interpolation is a linear interpolation method of bivariate functions, see for instance [Kirkland, 2010]. Even though the interpolation is linear in the sampled values as well as the position, in general and with respect to the sampling location, the method is of quadratic order rather than linear. Suppose we want to find the value of an unknown function $f : [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \to \mathbb{R}$ at an arbitrary point (x, y). The values of f are only known at some sampled data points $V_{k\ell} := (x_k, y_\ell)$, defined through an equidistant grid

$$\mathbb{G}_{(x,y)} := \{x_{\min} + kh_x \mid k = 0, \ldots, M_x\} \times \{y_{\min} + \ell h_y \mid \ell = 0, \ldots, M_y\}$$

with the step sizes defined by,

$$h_x := \frac{x_{\max} - x_{\min}}{M_x} \quad \text{and} \quad h_y := \frac{y_{\max} - y_{\min}}{M_y}$$

with the number of grid points denoted by $M_x, M_y > 0$. The interpolation scheme is illustrated with respect to a subset of the grid in Figure A.2.

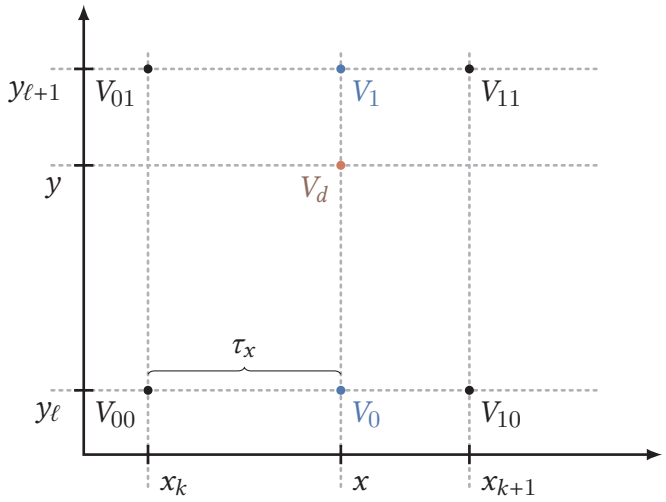


Figure A.2: *Interpolation scheme for bilinear interpolation for a desired point* (x, y).

To be precise, the method relies on the successive application of two linear interpolations, which in conjunction with the convention $V_{k,\ell} = V(x_k, y_\ell) = f(x_k, y_\ell)$ can be expressed along the x-axis as

$$V(x, y_\ell) = \frac{x_{k+1} - x}{x_{k+1} - x_k} V(x_k, y_\ell) + \frac{x - x_k}{x_{k+1} - x_k} V(x_{k+1}, y_\ell) \tag{A.4.1}$$

$$V(x, y_{\ell+1}) = \frac{x_{k+1} - x}{x_{k+1} - x_k} V(x_k, y_{\ell+1}) + \frac{x - x_k}{x_{k+1} - x_k} V(x_{k+1}, y_{\ell+1}). \tag{A.4.2}$$

Proceeding along the y-axis together with (A.4.1) to (A.4.2), yields

$$\begin{aligned} V(x, y) &= \frac{y_{\ell+1} - y}{y_{\ell+1} - y_\ell} V(x, y_\ell) + \frac{y - y_\ell}{y_{\ell+1} - y_\ell} V(x, y_{\ell+1}) \\ &= \frac{y_{\ell+1} - y}{y_{\ell+1} - y_\ell} \left[\frac{x_{k+1} - x}{x_{k+1} - x_k} V(x_k, y_\ell) + \frac{x - x_k}{x_{k+1} - x_k} V(x_{k+1}, y_\ell) \right] \\ &+ \frac{y - y_\ell}{y_{\ell+1} - y_\ell} \left[\frac{x_{k+1} - x}{x_{k+1} - x_k} V(x_k, y_{\ell+1}) + \frac{x - x_k}{x_{k+1} - x_k} V(x_{k+1}, y_{\ell+1}) \right]. \end{aligned} \tag{A.5}$$

Next, we define the weights with respect to the grid points,

$$\tau_x = \frac{x - x_k}{x_{k+1} - x_k} \qquad \text{and} \qquad \tau_y = \frac{y - y_\ell}{y_{\ell+1} - y_\ell}. \tag{A.6}$$

Due to the weights along each grid dimension accumulating to one, we can rewrite (A.5) as,

$$\begin{aligned} V(x,y) &= (1-\tau_y)\cdot V(x,y_\ell) + \tau_y \cdot V(x,y_{\ell+1}) \\ &= (1-\tau_y)\cdot \Big((1-\tau_x)\cdot V(x_k,y_\ell) + \tau_x \cdot V(x_{k+1},y_\ell)\Big) \\ &+ \tau_y \cdot \Big((1-\tau_x)\cdot V(x_k,y_{\ell+1}) + \tau_x \cdot V(x_{k+1},y_{\ell+1})\Big) \end{aligned} \tag{A.7}$$

which characterizes the bilinear interpolation scheme.

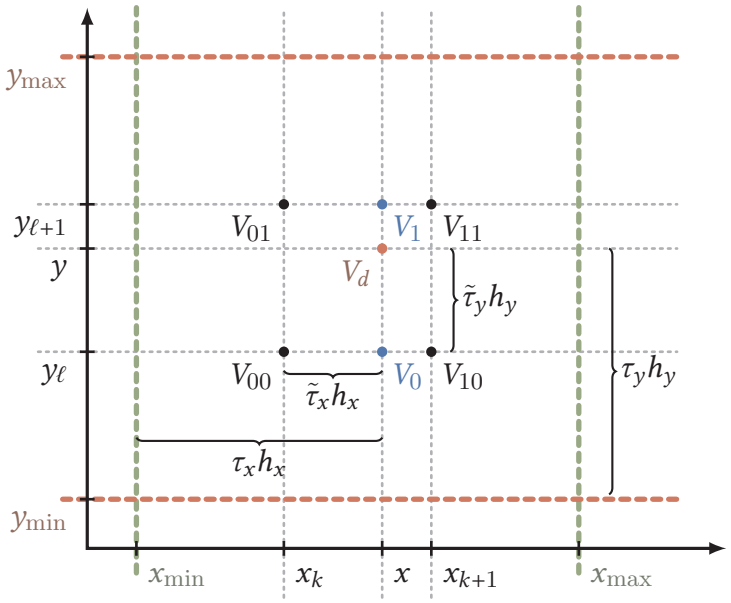


Figure A.3: *Interpolation scheme for bilinear interpolation algorithm.*

Moreover, by exploiting the grid structure with respect to the bounded intervals an algorithm subject to index matching can be derived, compare [Bertsekas, 2007]. The index of the nearest lowest neighbour on the grid with respect to the desired point (x, y) estimates to

$$i_x := \left\lfloor \frac{x - x_{\min}}{h_x} \right\rfloor \qquad \text{and} \qquad i_y := \left\lfloor \frac{y - y_{\min}}{h_y} \right\rfloor$$

for all $k = 0, \dots, M_x - 1$ and $\ell = 0, \dots, M_y - 1$. Then, given the step sizes, the weights for each dimension accumulates to

$$\tilde{\tau}_x = \frac{x_{k+1} - (x_{\min} + i_x h_x)}{h_x} \quad \text{and} \quad \tilde{\tau}_y = \frac{y_{\ell+1} - \left(y_{\min} + i_y h_y\right)}{h_y}.$$

Let further $V(k, \ell)$ denote the (known) value at the grid point (x_k, y_ℓ). Eventually, we can state the bilinear interpolation algorithm with index matching, i.e., Algorithm 10.

Algorithm 10: Bilinear Matching Interpolation

(S.0) Compute the index of the nearest lowest neighbour of the desired point (x, y) in the grid.

$$i_x := \left\lfloor \frac{x - x_{\min}}{h_x} \right\rfloor \qquad \text{and} \qquad i_y := \left\lfloor \frac{y - y_{\min}}{h_y} \right\rfloor$$

(S.1) Compute the weights of the nearest lowest grid points,

$$\tilde{\tau}_x = \frac{x_{k+1} - (x_{\min} + i_x h_x)}{h_x} \qquad \text{and} \qquad \tilde{\tau}_y = \frac{y_{\ell+1} - \left(y_{\min} + i_y h_y\right)}{h_y}.$$

(S.2) Acquire the function values at the nearest grid points :

$$\begin{aligned} V_{00} &= V(i_x, i_y) \\ V_{01} &= V(i_x, \min\{i_y, M_y\} + 1) \\ V_{10} &= V(\min\{i_x, M_x\} + 1, i_y) \\ V_{11} &= V(\min\{i_x, M_x\} + 1, \min\{i_y, M_y\} + 1) \end{aligned}$$

(S.3) Compute the interpolated function value at (x, y),

$$\begin{aligned} V_0 &= V_{00} \cdot (1 - \tilde{\tau}_x) + V_{01} \cdot \tilde{\tau}_x \qquad , V_1 = V_{10} \cdot (1 - \tilde{\tau}_x) + V_{11} \cdot \tilde{\tau}_x \\ V(x, y) &= V_0 \cdot (1 - \tilde{\tau}_y) + V_1 \cdot \tilde{\tau}_y \end{aligned}$$

A.3 Error Estimate n-dimensional Linear Interpolation

The purpose of this section is to establish a validation of the error estimate provided in Lemma 4.17 with respect to a n-dimensional linear interpolation characteristic. To this end, we consider the approximate value function in one step, defined in (4.24). Herein, we substitute the generic

interpolation scheme. Within the setting of Section 4.6 the state dimension is denoted by n_x. Consequently, let the weight of the n-th state dimension be defined by ω_n. The weights are to be construed as in (A.6). Thus, for an arbitrary non-grid point x^v, with initial date t_ℓ the approximate value function culminates to

$$\bar{\vartheta}_h(t_\ell, z^v) := \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} \bar{\vartheta}_h(t_\ell, \Pi_j[z^v]) \tag{A.8}$$

with the weights $\sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} = 1$. Further, the permuted indexing matrix $p = p(n_x)$ is characterized by (4.35), where p_{jn} indicates the respective matrix element.

Lemma A.4. *Let $L_\vartheta > 0$ be the maximum over all the Lipschitz constants of $\vartheta_h(t_\ell, \cdot)$ for $\ell = 0, \dots, M$. Let an arbitrary $t_\ell \in \{0, \dots, M\}$ be given. Then, we have for any grid point $z_\ell^v \in \mathbb{G}_x^v$*

$$|\bar{\vartheta}_h(t_\ell, z_\ell^v) - \vartheta_h(t_\ell, z_\ell^v)| \leq L_V(M-\ell)\|h_x^v\| \tag{A.9}$$

Further, we have for any $z_\ell^v \in [x_{\min}^v, x_{\max}^v]$

$$|\bar{\vartheta}_h(t_\ell, z_\ell^v) - \vartheta_h(t_\ell, z_\ell^v)| \leq L_\vartheta(M+1-\ell)\|h_x^v\|. \tag{A.10}$$

Proof. We prove the assertion by backward induction. For $\ell = M$ we do not have any approximation error at a grid point $z_M^v \in \mathbb{G}_x^v$, since

$$|\bar{\vartheta}_h(t_M, z_M^v) - \vartheta_h(t_M, z_M^v)| = |\varphi_v(z_M^v) - \varphi_v(z_M^v)| = 0.$$

Hence, (A.9) holds for $\ell = M$. For a non grid point $z_M^v \in [x_{\min}^v, x_{\max}^v]$, we exploit Lipschitz continuity of ϑ_h and $\sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} = 1$ to get

$$\begin{aligned}
|\bar{\vartheta}_h(t_M, z_M^v) - \vartheta_h(t_M, z_M^v)| &\overset{(A.8)}{=} \left| \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} \bar{\vartheta}_h(t_M, \Pi_j[z_M^v]) - \vartheta_h(t_M, z_M^v) \right| \\
&\overset{(A.9)}{=} \left| \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} (\vartheta_h(t_M, \Pi_j[z_M^v]) - \vartheta_h(t_M, z_M^v)) \right| \\
&\leq \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} L_\vartheta \|\Pi_j[z_M^v] - z_M^v\| \\
&\leq \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} L_\vartheta \|h_x^v\| \\
&= L_\vartheta \|h_x^v\|.
\end{aligned}$$

Hence, (A.9) and (A.10) hold for $\ell = M$. Now assume that (A.9) and (A.10) hold for some $\ell \in \{1, \dots, M\}$. Then we obtain for any grid point $z^\nu_{\ell-1} \in \mathbb{G}^\nu_x$:

$$\begin{aligned}
\bar{\vartheta}_h(t_{\ell-1}, z^\nu_{\ell-1}) &= \min_{\varkappa^\nu_h \in X^h_\nu(t_{\ell-1}, z^\nu_{\ell-1})} \left[P^\nu_{\ell-1}(x^\nu_h, x^{-\nu}_h) + \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} \bar{\vartheta}_h(t_\ell, \Pi_j[\xi_\nu(\varkappa^\nu_h)]) \right] \\
&= \min_{\varkappa^\nu_h \in X^h_\nu(t_{\ell-1}, z^\nu_{\ell-1})} \Bigg[P^\nu_{\ell-1}(x^\nu_h, x^{-\nu}_h) + \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} \vartheta_h(t_\ell, \xi_\nu(\varkappa^\nu_h)) \\
&\quad + \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} \Big(\vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa^\nu_h)]) - \vartheta_h(t_\ell, \xi_\nu(\varkappa^\nu_h)) \\
&\quad + \bar{\vartheta}_h(t_\ell, \Pi_j[\xi_\nu(\varkappa^\nu_h)]) - \vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa^\nu_h)]) \Big) \Bigg]
\end{aligned}$$

Now we can use Lipschitz continuity of ϑ_h to estimate

$$|\vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa^\nu_h)]) - \vartheta_h(t_\ell, \xi_\nu(\varkappa^\nu_h))| \le L_\vartheta \|\Pi_j[\xi_\nu(\varkappa^\nu_h)] - \xi_\nu(\varkappa^\nu_h)\| \le L_\vartheta \|h^\nu_x\|.$$

Further, by assumption we have (A.9) for ℓ and hence,

$$|\bar{\vartheta}_h(t_\ell, \Pi_j[\xi_\nu(\varkappa^\nu_h)]) - \vartheta_h(t_\ell, \Pi_j[\xi_\nu(\varkappa^\nu_h)])| \le L_\vartheta (M-\ell) \|h^\nu_x\|.$$

This together with $\sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} = 1$ implies on the one hand

$$\begin{aligned}
&\bar{\vartheta}_h(t_{\ell-1}, z^\nu_{\ell-1}) \\
&\quad \le \min_{\varkappa^\nu_h \in X^h_\nu(t_{\ell-1}, z^\nu_{\ell-1})} \left[P^\nu_{\ell-1}(x^\nu_h, x^{-\nu}_h) + \vartheta_h(t_\ell, \xi_\nu(\varkappa^\nu_h)) + L_\vartheta (M+1-\ell) \|(h_x)\| \right] \\
&\quad = \vartheta_h(t_{\ell-1}, z^\nu_{\ell-1}) + L_\vartheta (M-(\ell-1)) \|h^\nu_x\|,
\end{aligned}$$

and on the other hand

$$\begin{aligned}
&\bar{\vartheta}_h(t_{\ell-1}, z^\nu_{\ell-1}) \\
&\quad \ge \min_{\varkappa^\nu_h \in X^h_\nu(t_{\ell-1}, z^\nu_{\ell-1})} \left[P^\nu_{\ell-1}(x^\nu_h, x^{-\nu}_h) + \vartheta_h(t_\ell, \xi_\nu(\varkappa^\nu_h)) - L_\vartheta (M+1-\ell) \|(h_x)\| \right] \\
&\quad = \vartheta_h(t_{\ell-1}, z^\nu_{\ell-1}) - L_\vartheta (M-(\ell-1)) \|h^\nu_x\|.
\end{aligned}$$

Together, these two inequalities prove (A.9) for $\ell-1$. Now, let a non-grid point $z^\nu_{\ell-1} \in [x^\nu_{\min}, x^\nu_{\max}]$ be given. Then, we get

$$\begin{aligned}
&|\bar{\vartheta}_h(t_{\ell-1}, z^\nu_{\ell-1}) - \vartheta_h(t_{\ell-1}, z^\nu_{\ell-1})| \\
= &\left| \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} \bar{\vartheta}_h(t_{\ell-1}, \Pi_j[z^\nu_{\ell-1}]) - \vartheta_h(t_{\ell-1}, z^\nu_{\ell-1}) \right|
\end{aligned}$$

$$\begin{aligned}
&\overset{(A.9)}{\leq} \left| \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} \vartheta_h(t_{\ell-1}, \Pi_j[z_{\ell-1}^v]) - \vartheta_h(t_{\ell-1}, z_{\ell-1}^v) \right| + L_\vartheta(M-(\ell-1))\|h_x^v\| \\
&\leq \sum_{j=1}^{2^{n_x}} \prod_{n=1}^{n_x} \omega_n^{p_{jn}} \cdot (1-\omega_n)^{(1-p_{jn})} L_\vartheta \|\Pi_j[z_{\ell-1}^v] - z_{\ell-1}^v\| + L_\vartheta(M-(\ell-1))\|h_x^v\| \\
&\leq L_\vartheta(M+1-(\ell-1))\|h_x^v\|.
\end{aligned}$$

This proves (A.10) for $\ell - 1$, and completes the proof. □

Consequently, the estimate shows that the generic error estimate in Lemma 4.17 and the error estimate with respect to n-dimensional interpolation Lemma A.4 coincide.

A.4 Specifications for Computations

In this section, we aggregate the specifications of the hardware used to conduct the computations and benchmarks in the course of this thesis. The parameters of the hardware employed for offline and GPU computations is listed in Table A.1

Table A.1: *Hardware Specifications for offline and GPU computations.*

	Manufacturer	Specification Model	Cores/Modules	Memory	Clock
CPU	Intel	i7-8700	6/12T	12288 KB	3.2/4.6 GHz
DRAM	Kingston	KM0VW4-MIB	2	816 GB	2400 MHz
GPU	Nvidia	GTX 1060	1280	6 GB	1708 MHz
OS	Ubuntu	18.04.5 LTS	-	-	-

The cloud system established in the experimental setup is implemented on a machine with the hardware specifications summarized in Table A.2.

Table A.2: *Vehicle in the Cloud specifications.*

	Manufacturer	Specification Model	Cores/Modules	Memory	Clock
CPU	Intel	i7-7700K	4/8T	8192 KB	4.2/4.5 GHz
DRAM	Samsung	M378A2K43BB1-CPB	4	64 GB	2133 MHz
OS	Windows	10 64-bit	-	-	-

Let us mention, that with respect to parallel computing on the CPU level, the core and thread count as well as the clock frequency have a significant impact on the ensuing performance.

Similarly, the same applies to GPU parallel computations, although the bandwidth and the available memory of the device are of considerably increasing importance.

A.5 GPU Performance and Occupancy Benchmarks

In this section, the influence of different memory accesses patterns on the efficiency of computational operations on a GPU will be investigated. Therefore, we consider a simple matrix-matrix multiplication operation at increasing matrix dimensions. A sequential addressing pattern of the individual matrix elements, a coalesced access and a shared memory access pattern are compared.

At first, we investigate the influence of the implementations on the bandwidth, which constitutes a measure of the memory throughput in the GPU and thus directly affects the computation time. The resulting bandwidth over increasing dimensions is shown in Figure A.4

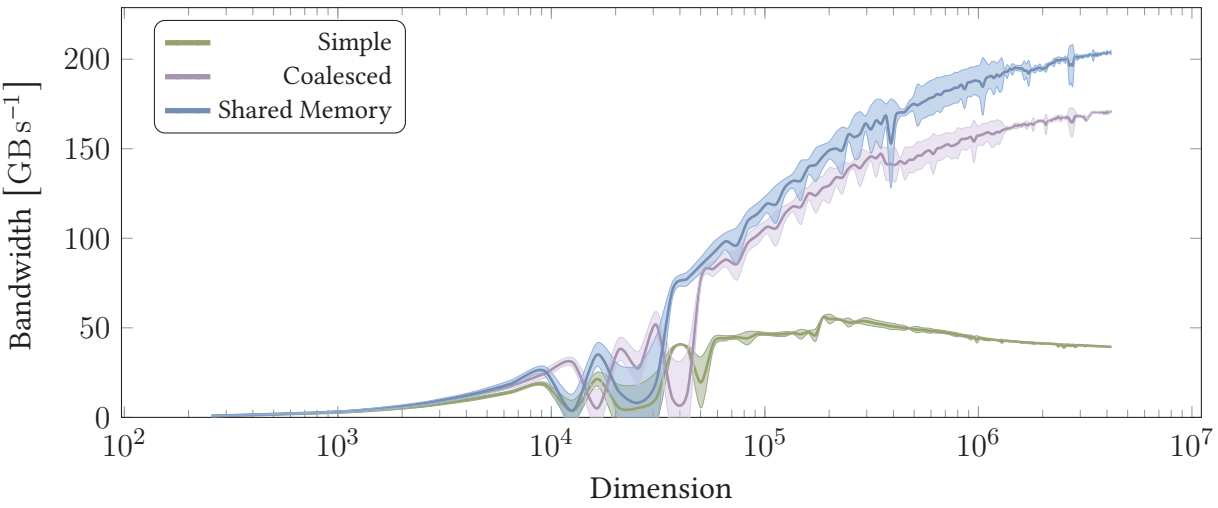


Figure A.4: *Comparison of the effective bandwidth of different implementations of a matrix (square) multiplication, for increasing matrix dimensions. The standard deviation, with respect to 100 executions, is displayed as a shaded area around the mean value.*

The effect on the computation times is presented in Figure A.5. A correlation with the bandwidth is clearly visible.

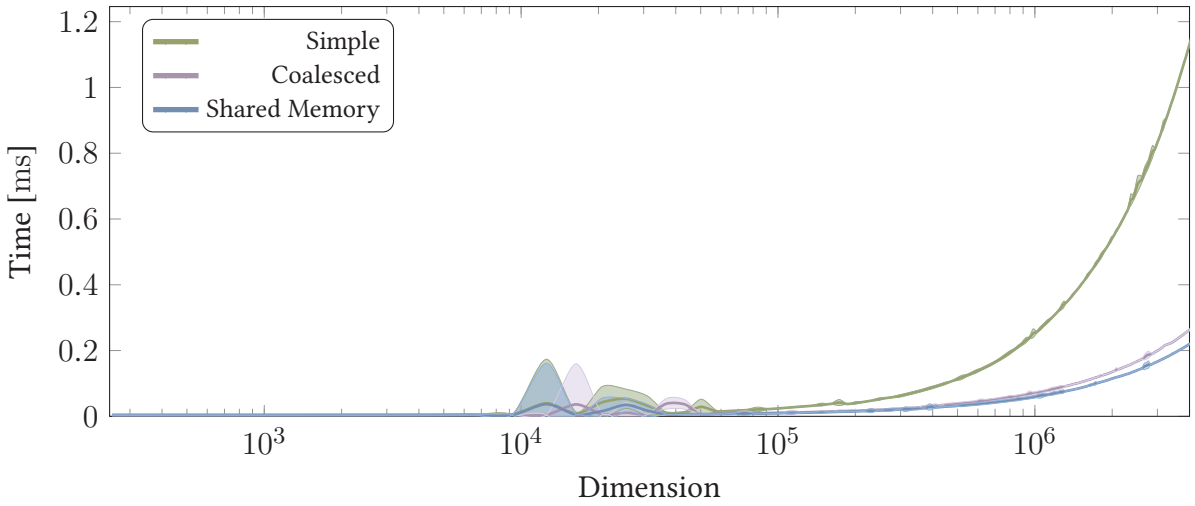


Figure A.5: *Comparison of the GPU computation times of different implementations of a matrix multiplication. The standard deviation, with respect to 100 executions, is displayed as a shaded area around the mean value.*

Eventually, we address the aspect of a stride offset memory access, i.e., with an offset distributed to different threads.

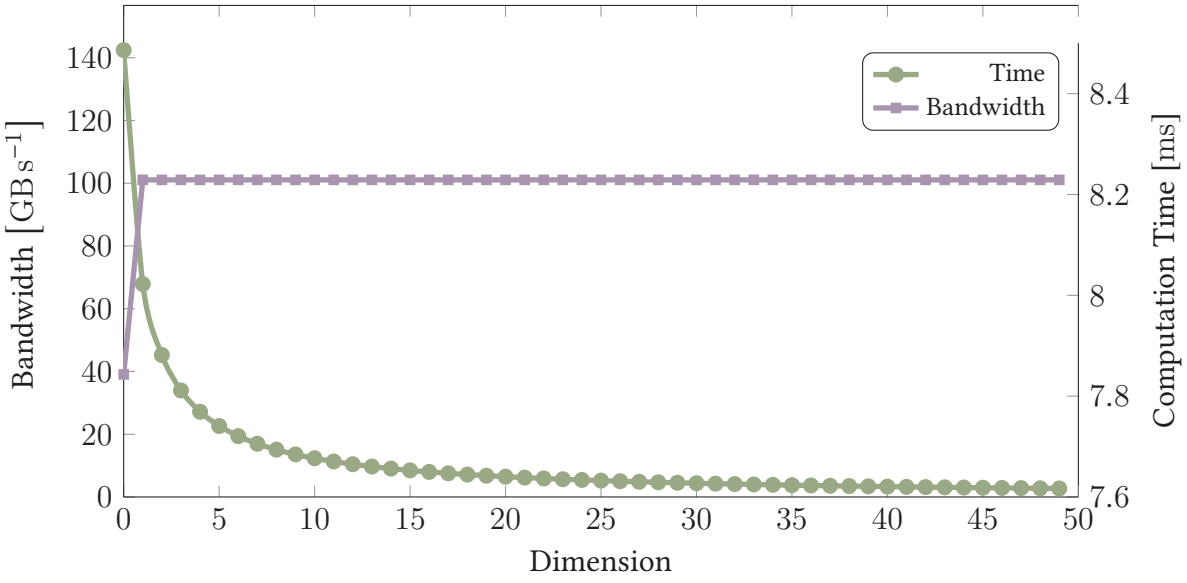


Figure A.6: *Effect of increasing stride offset memory access on the bandwidth and computation time*

Let us, stress that an increasing offset no longer constitutes a correct result and this is reflected in lower computation times, since it is no longer guaranteed that all elements are accessed, cf. Figure A.6.

A.6 Auxiliary Results

For the sake of intelligibility, in this section some established results are presented, which find application within this thesis and therefore complement the preceding chapters.

First, we provide an important existence result concerning initial value problems which is based on differential inclusion, namely the prominent Lemma of *Filippov*, whose proof can be found in [Macki and Strauss, 1982].

Lemma A.5 (Filippov's Lemma, [Macki and Strauss, 1982, Chapter IV, p. 94]). *Let $\Omega \subset \mathbb{R}^m$ be compact and $\varphi : \mathbb{R} \times \Omega \to \mathbb{R}^{n+1}$ be a continuous function. Suppose $\Psi : \mathbb{R} \to \mathbb{R}^{n+1}$ is a bounded measurable function, with $\Psi(t) \in \varphi(t, \Omega)$. Then there is a measurable $x(\cdot)$ with $x(t) \in \Omega$ for all t, such that $\Psi(t) = \varphi(t, x(t))$.*

Before we can state the next result, we require some additional definitions. To this end, let $x \in W^n_{1,1}([a, b])$ be an absolutely continuous function and let $x(\cdot)$ denote the solution of

$$x'(t) = f(t, x(t)) \quad \text{with} \quad x'(t) \in \mathcal{F}(t, x(t))$$

and let the set valued function $\mathcal{F}$ be defined on a tube $\Omega := \{(t, \bar{x}) \mid a \leq t \leq b, \bar{x} \in x(t) + \varepsilon \mathbb{B}^n(0))\}$. Now, for set valued functions we recall the sequential compactness theorem of [Clarke, 1983, Theorem 3.1.7, p. 118].

Theorem A.6. *Let $\mathcal{F}$ be $L \times B$ measurable and upper semicontinuous, and let $\{x_j\}$ be a sequence of arcs on $[a, b]$ satisfying:*

(i) $x_j(t) \in X(t)$, with $X : [a, b] \rightrightarrows \mathbb{R}^n$ and $|\dot{x}_j(t)| \leq \Phi(t)$ for almost all $t \in [a, b]$, where $\Phi(t)$ denotes an integrable function.

(ii) $\dot{x}_j(t) \in \mathcal{F}(t, x_j(t) + y_j(t)) + r_j(t)B$ for $t \in A_j$, with B the open unit ball, where $\{y_j\}$, $\{r_j\}$ are sequences of measurable functions on $[a, b]$ which converge uniformly to zero, and where $\{A_j\}$ is a sequence of measurable subsets of $[a, b]$ such that measure $(A_j) \to (b - a)$.

(iii) The sequence $\{x_j(a)\}$ is bounded.

Then there is a subsequence of $\{x_j\}$ which converges uniformly to an arc x which is a trajectory for $\mathcal{F}$.

Bibliography

[Alessandretti and Aguiar, 2020] Alessandretti, A. and Aguiar, A. P. (2020). An Optimization-Based Cooperative Path-Following Framework for Multiple Robotic Vehicles. *IEEE Transactions on Control of Network Systems*, 7(2):1002–1014.

[Alt, 2012] Alt, H. W. (2012). *Lineare Funktionalanalysis: Eine anwendungsorientierte Einführung*. Masterclass. Springer-Verlag, Berlin Heidelberg, sixth edition.

[Altarovici et al., 2013] Altarovici, A., Bokanowski, O., and Zidani, H. (2013). A general Hamilton-Jacobi framework for non-linear state-constrained control problems. *ESAIM: Control, Optimisation and Calculus of Variations*, 19(2):337–357.

[Anderson et al., 2016] Anderson, J. R., Ayalew, B., and Weiskircher, T. (2016). Modeling a professional driver in ultra-high performance maneuvers with a hybrid cost MPC. In *2016 American Control Conference (ACC)*, pages 1981–1986.

[Anderson, 1976] Anderson, L. B. (1976). Antagonistic Games. Technical report, Institue for Defense Analyses, 400 Army-Navy Drive, Arlington, Virginia 22202.

[Ardagna et al., 2011] Ardagna, D., Panicucci, B., and Passacantando, M. (2011). A game theoretic formulation of the service provisioning problem in cloud systems. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 177–186, New York, NY, USA. Association for Computing Machinery.

[Arrow and Debreu, 1954] Arrow, K. J. and Debreu, G. (1954). Existence of an Equilibrium for a Competitive Economy. *Econometrica*, 22(3):265–290.

[Assellaou, 2015] Assellaou, M. (2015). *Hamilton Jacobi Bellman Approach for Some Applied Optimal Control Problems*. Thesis, Ensta ParisTech, Paris.

[Aubin and Cellina, 1984] Aubin, J.-P. and Cellina, A. (1984). *Differential Inclusions: Set-Valued Maps and Viability Theory*. Springer Berlin Heidelberg.

[Aumann, 1961] Aumann, R. J. (1961). Almost Strictly Competitive Games. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):544–550.

[Ba and Pang, 2020] Ba, Q. and Pang, J.-S. (2020). Exact Penalization of Generalized Nash Equilibrium Problems. *Operations Research*, pages 1–17.

[Bardi and Capuzzo-Dolcetta, 2008] Bardi, M. and Capuzzo-Dolcetta, I. (2008). *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*. Modern Birkhäuser Classics. Birkhäuser, Boston.

[Bates and McDonald, 1982] Bates, J. R. and McDonald, A. (1982). Multiply-Upstream, Semi-Lagrangian Advective Schemes: Analysis and Application to a Multi-Level Primitive Equation Model. *Monthly Weather Review*, 110(12):1831–1842.

[Bellman, 1957] Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton, New Jersey.

[Bellman, 2003] Bellman, R. E. (2003). *Dynamic Programming*. Dover Publications Inc., Mineola, N.Y, reprint edition edition.

[Bensoussan, 1974] Bensoussan, A. (1974). Points de Nash Dans le Cas de Fonctionnelles Quadratiques et Jeux Differentiels lineaires a N Personnes. *SIAM Journal on Control*, 12(3):460–499.

[Bertsekas, 2005] Bertsekas, D. P. (2005). *Dynamic Programming & Optimal Control*. Athena Scientific, Belmont, Mass, third edition.

[Bertsekas, 2007] Bertsekas, D. P. (2007). *Dynamic Programming and Optimal Control, Vol. II*. Athena Scientific, third edition.

[Bertsekas, 2019] Bertsekas, D. P. (2019). Feature-based aggregation and deep reinforcement learning: A survey and some new implementations. *IEEE/CAA Journal of Automatica Sinica*, 6(1):1–31.

[Betts, 2020] Betts, J. T. (2020). *Practical Methods for Optimal Control Using Nonlinear Programming*. Advances in Design and Control. Society for Industrial and Applied Mathematics, third edition.

[Bokanowski et al., 2010] Bokanowski, O., Forcadel, N., and Zidani, H. (2010). Reachability and Minimal Times for State Constrained Nonlinear Problems without Any Controllability Assumption. *SIAM Journal on Control and Optimization*, 48(7):4292–4316.

[Bokanowski et al., 2011] Bokanowski, O., Forcadel, N., and Zidani, H. (2011). Deterministic state-constrained optimal control problems without controllability assumptions. *ESAIM: Control, Optimisation and Calculus of Variations*, 17(4):995–1015.

[Britzelmeier et al., 2018] Britzelmeier, A., De Marchi, A., and Gerdts, M. (2018). An Iterative Solution Approach for a Bi-level Optimization Problem for Congestion Avoidance on Road Networks. In Falcone, M., Ferretti, R., Grüne, L., and McEneaney, W. M., editors, *Numerical Methods for Optimal Control Problems*, Springer INdAM Series, pages 23–38. Springer International Publishing, Cham.

[Britzelmeier and Dreves, 2020] Britzelmeier, A. and Dreves, A. (2020). A decomposition algorithm for Nash equilibria in intersection management. *Optimization*, pages 1–38.

[Britzelmeier et al., 2019] Britzelmeier, A., Dreves, A., and Gerdts, M. (2019). Numerical solution of potential games arising in the control of cooperative automatic vehicles. In *2019 Proceedings of the Conference on Control and Its Applications*, Proceedings, pages 38–45. Society for Industrial and Applied Mathematics.

[Britzelmeier and Gerdts, 2018] Britzelmeier, A. and Gerdts, M. (2018). Non-linear Model Predictive Control of Connected, Automatic Cars in a Road Network Using Optimal Control Methods. *IFAC-PapersOnLine*, 51(2):168–173.

[Britzelmeier and Gerdts, 2020] Britzelmeier, A. and Gerdts, M. (2020). A Nonsmooth Newton Method for Linear Model-Predictive Control in Tracking Tasks for a Mobile Robot With Obstacle Avoidance. *IEEE Control Systems Letters*, 4(4):886–891.

[Britzelmeier et al., 2020a] Britzelmeier, A., Gerdts, M., and Rottmann, T. (2020a). Control of interacting vehicles using model-predictive control, generalized Nash equilibrium problems, and dynamic inversion. *IFAC-PapersOnLine*, 53(2):15146–15153.

[Britzelmeier et al., 2020b] Britzelmeier, A., Gerdts, M., and Rottmann, T. (2020b). Control of interacting vehicles using model-predictive control, generalized Nash equilibrium problems, and dynamic inversion. In *21st IFAC World Congress 2020*, Berlin.

[Cacace et al., 2014] Cacace, S., Cristiani, E., and Falcone, M. (2014). Two Semi-Lagrangian Fast Methods for Hamilton-Jacobi-Bellman Equations. In Pötzsche, C., Heuberger, C., Kaltenbacher, B., and Rendl, F., editors, *System Modeling and Optimization*, IFIP Advances in Information and Communication Technology, pages 74–84, Berlin, Heidelberg. Springer.

[Capuzzo-Dolcetta and Lions, 1990] Capuzzo-Dolcetta, I. and Lions, P.-L. (1990). Hamilton-Jacobi Equations with State Constraints. *Transactions of the American Mathematical Society*, 318(2):643–683.

[Cardellini et al., 2016] Cardellini, V., De Nitto Personé, V., Di Valerio, V., Facchinei, F., Grassi, V., Lo Presti, F., and Piccialli, V. (2016). A game-theoretic approach to computation offloading in mobile cloud computing. *Mathematical Programming*, 157(2):421–449.

[Chan and Pang, 1982] Chan, D. and Pang, J. S. (1982). The Generalized Quasi-Variational Inequality Problem. *Mathematics of Operations Research*, 7(2):211–222.

[Clarke, 1983] Clarke, F. H. (1983). *Optimization and Nonsmooth Analysis*. Wiley, New York.

[Contreras et al., 2013] Contreras, J., Krawczyk, J. B., Zuccollo, J., and García, J. (2013). Competition of Thermal Electricity Generators with Coupled Transmission and Emission Constraints. *Journal of Energy Engineering*, 139(4):239–252.

[Courant et al., 1928] Courant, R., Friedrichs, K., and Lewy, H. (1928). Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, 100(1):32–74.

[Courant et al., 1952] Courant, R., Isaacson, E., and Rees, M. (1952). On the solution of nonlinear hyperbolic differential equations by finite differences. *Communications on Pure and Applied Mathematics*, 5(3):243–255.

[Cournot, 1838] Cournot, A.-A. (1838). *Recherches sur les principes mathématiques de la théorie des richesses, par Augustin Cournot*. L. Hachette, Paris.

[Crandall et al., 1984] Crandall, M. G., Evans, L. C., and Lions, P.-L. (1984). Some properties of viscosity solutions of Hamilton-Jacobi equations. *Transactions of the American Mathematical Society*, 282(2):487–502.

[Crandall and Lions, 1983] Crandall, M. G. and Lions, P.-L. (1983). Viscosity solutions of Hamilton-Jacobi equations. *Transactions of the American Mathematical Society*, 277(1):1–42.

[Crandall and Lions, 1984] Crandall, M. G. and Lions, P. L. (1984). Two Approximations of Solutions of Hamilton-Jacobi Equations. *Mathematics of Computation*, 43(167):1–19.

[de Campos et al., 2013] de Campos, G. R., Falcone, P., and Sjöberg, J. (2013). Autonomous cooperative driving: A velocity-based negotiation approach for intersection crossing. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 1456–1461.

[Debreu, 1952] Debreu, G. (1952). A Social Equilibrium Existence Theorem. *Proceedings of the National Academy of Sciences of the United States of America*, 38(10):886–893.

[Diehl et al., 2005] Diehl, M., Bock, H. G., and Schlöder, J. P. (2005). A Real-Time Iteration Scheme for Nonlinear Optimization in Optimal Feedback Control. *SIAM Journal on Control and Optimization*, 43(5):1714–1736.

[DIN 8505, 2013] DIN 8505 (2013). Straßenfahrzeuge - Fahrzeugdynamik und Fahrverhalten - Begriffe (ISO 8855:2011).

[Dresner and Stone, 2008] Dresner, K. and Stone, P. (2008). A Multiagent Approach to Autonomous Intersection Management. *Journal of Artificial Intelligence Research*, 31:591–656.

[Dreves, 2011] Dreves, A. (2011). *Globally Convergent Algorithms for the Solution of Generalized Nash Equilibrium Problems.* PhD thesis, Universität Würzburg.

[Dreves, 2016] Dreves, A. (2016). Improved error bound and a hybrid method for generalized Nash equilibrium problems. *Computational Optimization and Applications*, 65(2):431–448.

[Dreves et al., 2014] Dreves, A., Facchinei, F., Fischer, A., and Herrich, M. (2014). A new error bound result for generalized Nash equilibrium problems and its algorithmic application. *Computational Optimization and Applications*, 59(1):63–84.

[Dreves et al., 2011] Dreves, A., Facchinei, F., Kanzow, C., and Sagratella, S. (2011). On the solution of the KKT conditions of generalized Nash equilibrium problems. *SIAM Journal on Optimization*, 21(3):1082–1108.

[Dreves and Gerdts, 2018] Dreves, A. and Gerdts, M. (2018). A generalized Nash equilibrium approach for optimal control problems of autonomous cars. *Optimal Control Applications and Methods*, 39(1):326–342.

[Dreves and Kanzow, 2011] Dreves, A. and Kanzow, C. (2011). Nonsmooth optimization reformulations characterizing all solutions of jointly convex generalized Nash equilibrium problems. *Computational Optimization and Applications*, 50(1):23–48.

[Ehrgott, 2005] Ehrgott, M. (2005). *Multicriteria Optimization.* Springer Science & Business Media.

[Facchinei et al., 2007] Facchinei, F., Fischer, A., and Piccialli, V. (2007). On generalized Nash games and variational inequalities. *Operations Research Letters*, 35(2):159–164.

[Facchinei et al., 2009] Facchinei, F., Fischer, A., and Piccialli, V. (2009). Generalized Nash equilibrium problems and Newton methods. *Mathematical Programming*, 117(1):163–194.

[Facchinei and Kanzow, 2010a] Facchinei, F. and Kanzow, C. (2010a). Generalized Nash Equilibrium Problems. *Annals of Operations Research*, 175(1):177–211.

[Facchinei and Kanzow, 2010b] Facchinei, F. and Kanzow, C. (2010b). Penalty Methods for the Solution of Generalized Nash Equilibrium Problems. *SIAM Journal on Optimization*, 20(5):2228–2253.

[Facchinei et al., 2014] Facchinei, F., Kanzow, C., and Sagratella, S. (2014). Solving quasi-variational inequalities via their KKT conditions. *Mathematical Programming*, 144(1):369–412.

[Facchinei and Lampariello, 2011] Facchinei, F. and Lampariello, L. (2011). Partial penalization for the solution of generalized Nash equilibrium problems. *Journal of Global Optimization*, 50(1):39–57.

[Facchinei and Pang, 2003] Facchinei, F. and Pang, J.-S. (2003). *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer Series in Operations Research and Financial Engineering, Finite-Dimensional Variational Inequalities and Complementarity Problems. Springer-Verlag, New York.

[Facchinei and Pang, 2006] Facchinei, F. and Pang, J.-S. (2006). Exact penalty functions for generalized Nash problems. In Di Pillo, G. and Roma, M., editors, *Large-Scale Nonlinear Optimization*, Nonconvex Optimization and Its Applications, pages 115–126. Springer US, Boston, MA.

[Facchinei and Pang, 2009] Facchinei, F. and Pang, J.-S. (2009). Nash equilibria: The variational approach. In Palomar, D. P. and Eldar, Y. C., editors, *Convex Optimization in Signal Processing and Communications*, pages 443–493. Cambridge University Press, Cambridge.

[Facchinei et al., 2011] Facchinei, F., Piccialli, V., and Sciandrone, M. (2011). Decomposition algorithms for generalized potential games. *Computational Optimization and Applications*, 50(2):237–262.

[Falcone, 1991] Falcone, M. (1991). Corrigenda: A numerical approach to the infinite horizon problem of deterministic control theory. *Applied Mathematics and Optimization*, 23(1):213–214.

[Falcone and Ferretti, 1998] Falcone, M. and Ferretti, R. (1998). Convergence Analysis for a Class of High-Order Semi-Lagrangian Advection Schemes. *SIAM Journal on Numerical Analysis*, 35(3):909–940.

[Falcone and Ferretti, 2002] Falcone, M. and Ferretti, R. (2002). Semi-Lagrangian Schemes for Hamilton-Jacobi Equations, Discrete Representation Formulae and Godunov Methods. *Journal of Computational Physics*, 175(2):559–575.

[Falcone and Giorgi, 1999] Falcone, M. and Giorgi, T. (1999). An Approximation Scheme for Evolutive Hamilton-Jacobi Equations. In McEneaney, W. M., Yin, G. G., and Zhang, Q., editors, *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming*, Systems & Control: Foundations & Applications, pages 289–303. Birkhäuser, Boston, MA.

[Fischer et al., 2014] Fischer, A., Herrich, M., and Schönefeld, K. (2014). Generalized Nas Equilibrium Problems - Recent Advances and Challenges. *Pesquisa Operacional*, 34(3):521–558.

[Fleming and Soner, 2006] Fleming, W. H. and Soner, H. M. (2006). *Controlled Markov Processes and Viscosity Solutions*. Stochastic Modelling and Applied Probability. Springer-Verlag, New York, second edition.

[Frankowska and Plaskacz, 2000] Frankowska, H. and Plaskacz, S. (2000). Semicontinuous Solutions of Hamilton–Jacobi–Bellman Equations with Degenerate State Constraints. *Journal of Mathematical Analysis and Applications*, 251(2):818–838.

[Frankowska and Vinter, 2000] Frankowska, H. and Vinter, R. B. (2000). Existence of Neighboring Feasible Trajectories: Applications to Dynamic Programming for State-Constrained Optimal Control Problems. *Journal of Optimization Theory and Applications*, 104(1):20–40.

[Frego et al., 2016] Frego, M., Bertolazzi, E., Biral, F., Fontanelli, D., and Palopoli, L. (2016). Semi-analytical minimum time solutions for a vehicle following clothoid-based trajectory subject to velocity constraints. In *2016 European Control Conference (ECC)*, pages 2221–2227.

[Fukushima, 2007] Fukushima, M. (2007). A Class of Gap Functions for Quasi-Variational Inequality Problems1. *Journal of Industrial and Management Optimization*, 3(2):165–171.

[Fukushima, 2011] Fukushima, M. (2011). Restricted generalized Nash equilibria and controlled penalty algorithm. *Computational Management Science*, 8(3):201–218.

[Geiger and Kanzow, 2002] Geiger, C. and Kanzow, C. (2002). *Theorie und Numerik restringierter Optimierungsaufgaben*. Masterclass. Springer-Verlag, Berlin Heidelberg.

[Gerdts, 2003] Gerdts, M. (2003). A moving horizon technique for the simulation of automobile test-drives. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 83(3):147–162.

[Gerdts, 2011] Gerdts, M. (2011). *Optimal Control of ODEs and DAEs*. De Gruyter, Berlin.

[Gerdts, 2018] Gerdts, M. (2018). Numerical experiments with multistep model-predictive control approaches and sensitivity updates for the tracking control of cars. *arXiv:1809.00577*.

[Gerdts and Lempio, 2011] Gerdts, M. and Lempio, F. (2011). *Mathematische Optimierungsverfahren des Operations Research*. De Gruyter, Berlin.

[Gillespie, 1992] Gillespie, T. D. (1992). *Fundamentals of Vehicle Dynamics*. Society of Automotive Engineers, Warrendale, PA.

[Grüne and Pannek, 2011] Grüne, L. and Pannek, J. (2011). *Nonlinear Model Predictive Control: Theory and Algorithms*. Communications and Control Engineering. Springer-Verlag, London.

[Hall and Meyer, 1976] Hall, C. A. and Meyer, W. W. (1976). Optimal error bounds for cubic spline interpolation. *Journal of Approximation Theory*, 16(2):105–122.

[Han et al., 2012] Han, Z., Niyato, D., Saad, W., Baar, T., and Hjrungnes, A. (2012). *Game Theory in Wireless and Communication Networks: Theory, Models, and Applications*. Cambridge University Press, USA, 1st edition.

[Harker, 1991] Harker, P. T. (1991). Generalized Nash games and quasi-variational inequalities. *European Journal of Operational Research*, 54(1):81–94.

[Harten et al., 1987] Harten, A., Engquist, B., Osher, S., and Chakravarthy, S. R. (1987). Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71(2):231–303.

[Harten et al., 1986] Harten, A., Osher, S., Engquist, B., and Chakravarthy, S. R. (1986). Some results on uniformly high-order accurate essentially nonoscillatory schemes. *Applied Numerical Mathematics*, 2(3):347–377.

[Houska et al., 2016] Houska, B., Frasch, J., and Diehl, M. (2016). An Augmented Lagrangian Based Algorithm for Distributed NonConvex Optimization. *SIAM Journal on Optimization*, 26(2):1101–1127.

[Hult et al., 2015] Hult, R., Campos, G. R., Falcone, P., and Wymeersch, H. (2015). An approximate solution to the optimal coordination problem for autonomous vehicles at intersections. In *2015 American Control Conference (ACC)*, pages 763–768.

[Hult et al., 2018] Hult, R., Zanon, M., Gras, S., and Falcone, P. (2018). An MIQP-based heuristic for Optimal Coordination of Vehicles at Intersections. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 2783–2790.

[Hult et al., 2019] Hult, R., Zanon, M., Gros, S., and Falcone, P. (2019). Optimal Coordination of Automated Vehicles at Intersections: Theory and Experiments. *IEEE Transactions on Control Systems Technology*, 27(6):2510–2525.

[Isaacs, 1999] Isaacs, R. (1999). *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization.* Dover Publications Inc., Mineola, N.Y, new ed edition edition.

[Ishii and Koike, 1996] Ishii, H. and Koike, S. (1996). A new formulation of state constraint problems for first-order PDES. *SIAM Journal on Control and Optimization*, 34(2):554–571.

[Jazar, 2017] Jazar, R. N. (2017). *Vehicle Dynamics.* Springer, Cham, third edition.

[Jiang and Peng, 2000] Jiang, G.-S. and Peng, D. (2000). Weighted ENO Schemes for Hamilton–Jacobi Equations. *SIAM Journal on Scientific Computing*, 21(6):2126–2143.

[Kanzow and Schwartz, 2018] Kanzow, C. and Schwartz, A. (2018). *Spieltheorie: Theorie und Verfahren zur Lösung von Nash- und verallgemeinerten Nash-Gleichgewichtsproblemen.* Mathematik Kompakt. Birkhäuser Basel.

[Karush, 1939] Karush, W. (1939). *Minima of Functions of Several Variables with Inequalities as Side Conditions.* PhD thesis, University of Chicago, Department of Mathematics, Chicago, Illinois.

[Katriniok et al., 2017] Katriniok, A., Kleibaum, P., and Joševski, M. (2017). Distributed Model Predictive Control for Intersection Automation Using a Parallelized Optimization Approach. *IFAC-PapersOnLine*, 50(1):5940–5946.

[Key, 1994] Key, E. (1994). Disks, Shells, and Integrals of Inverse Functions. *The College Mathematics Journal*, 25(2):136–138.

[Kirk, 2004] Kirk, D. E. (2004). *Optimal Control Theory: An Introduction.* Dover Publications Inc., Mineola, N.Y, illustrated edition edition.

[Kirkland, 2010] Kirkland, E. J. (2010). *Advanced Computing in Electron Microscopy.* Springer, New York, second edition.

[Krabs, 2005] Krabs, W. (2005). *Spieltheorie: Dynamische Behandlung von Spielen.* Vieweg+Teubner Verlag, Wiesbaden.

[Kubota and Fukushima, 2010] Kubota, K. and Fukushima, M. (2010). Gap Function Approach to the Generalized Nash Equilibrium Problem. *Journal of Optimization Theory and Applications*, 144:511–531.

[Kuhn and Tucker, 1951] Kuhn, H. W. and Tucker, A. W. (1951). Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, California. University of California Press.

[Kühnel, 2013] Kühnel, W. (2013). *Differentialgeometrie: Kurven - Flächen - Mannigfaltigkeiten*. Springer-Verlag, Wiesbaden.

[Lã et al., 2016] Lã, Q. D., Chew, Y. H., and Soong, B.-H. (2016). *Potential Game Theory: Applications in Radio Resource Allocation*. Springer, New York, NY, first edition.

[Laisant, 1905] Laisant, C.-A. (1905). Intégration des fonctions inverses. *Nouvelles annales de mathématiques : journal des candidats aux écoles polytechnique et normale*, 5:253–257.

[Lions, 1982] Lions, P. L. (1982). *Generalized Solutions of Hamilton-Jacobi Equations*. Pitman, Boston.

[Llanas and Sainz, 2006] Llanas, B. and Sainz, F. J. (2006). Constructive approximate interpolation by neural networks. *Journal of Computational and Applied Mathematics*, 188(2):283–308.

[Lot and Biral, 2014] Lot, R. and Biral, F. (2014). A Curvilinear Abscissa Approach for the Lap Time Optimization of Racing Vehicles. *IFAC Proceedings Volumes*, 47(3):7559–7565.

[Luo et al., 2016] Luo, L.-h., Ge, Y.-e., Chen, J.-h., and Zhang, F.-w. (2016). Real-time routing control design for traffic networks with multi-route choices. *Journal of Central South University*, 23(7):1807–1816.

[Macki and Strauss, 1982] Macki, J. and Strauss, A. (1982). *Introduction to Optimal Control Theory*. Undergraduate Texts in Mathematics. Springer-Verlag, New York.

[Makarem and Gillet, 2013] Makarem, L. and Gillet, D. (2013). Model predictive coordination of autonomous vehicles crossing intersections. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 1799–1804.

[Mitschke, 2014] Mitschke, M. (2014). *Dynamik {der} Kraftfahrzeuge*. Springer Fachmedien, Wiesbaden, fifth edition.

[Mochaourab et al., 2012] Mochaourab, R., Zorba, N., and Jorswieck, E. (2012). Nash equilibrium in multiple antennas protected and shared bands. In *2012 International Symposium on Wireless Communication Systems (ISWCS)*, pages 101–105.

[Monderer and Shapley, 1996] Monderer, D. and Shapley, L. S. (1996). Potential Games. *Games and Economic Behavior*, 14(1):124–143.

[Murano and Poznyak, 2004] Murano, D. A. and Poznyak, A. S. (2004). Nash Equilibrium Strategies for a Class of Non-Linear Stochastic Constrained Differential Games. *IFAC Proceedings Volumes*, 37(21):609–614.

[Nabetani et al., 2011] Nabetani, K., Tseng, P., and Fukushima, M. (2011). Parametrized variational inequality approaches to generalized Nash equilibrium problems with shared constraints. *Computational Optimization and Applications*, 48:423–452.

[Nash, 1951] Nash, J. (1951). Non-Cooperative Games. *Annals of Mathematics*, 54(2):286–295.

[Nash, 1950] Nash, J. F. (1950). Equilibrium Points in N-Person Games. *Proceedings of the National Academy of Sciences of the United States of America*, 36(1):48–49.

[Nasiri and Zaccour, 2010] Nasiri, F. and Zaccour, G. (2010). Renewable Portfolio Standard Policy: A Game-theoretic Analysis. *Information Systems and Operational Research*, 48(4):251–260.

[Niehorster et al., 2017] Niehorster, D. C., Li, L., and Lappe, M. (2017). The Accuracy and Precision of Position and Orientation Tracking in the HTC Vive Virtual Reality System for Scientific Research. *i-Perception*, 8(3):2041669517708205.

[Nikaidô and Isoda, 1955] Nikaidô, H. and Isoda, K. (1955). Note on non-cooperative convex games. *Pacific Journal of Mathematics*, 5(Suppl. 1):807–815.

[Niklaus et al., 2017] Niklaus, S., Mai, L., and Liu, F. (2017). Video Frame Interpolation via Adaptive Convolution. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2270–2279.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer-Verlag, New York, second edition.

[Nvidia Corporation, 2020] Nvidia Corporation (2020). CUDA Toolkit Documentation. https://-docs.nvidia.com/cuda/index.html.

[Osborne and Rubinstein, 1994] Osborne, M. J. and Rubinstein, A. (1994). *A Course in Game Theory*, volume 1 of *MIT Press Books*. The MIT Press.

[Osher and Shu, 1991] Osher, S. and Shu, C.-W. (1991). High-Order Essentially Nonoscillatory Schemes for Hamilton–Jacobi Equations. *SIAM Journal on Numerical Analysis*, 28(4):907–922.

[Pang and Fukushima, 2009] Pang, J.-S. and Fukushima, M. (2009). Quasi-variational inequalities, generalized Nash equilibria, and multi-leader-follower games. *Computational Management Science*, 6(3):373–375.

[Pang and Scutari, 2011] Pang, J.-S. and Scutari, G. (2011). Nonconvex Games with Side Constraints. *SIAM Journal on Optimization*, 21(4):1491–1522.

[Pang et al., 2008] Pang, J.-S., Scutari, G., Facchinei, F., and Wang, C. (2008). Distributed Power Allocation With Rate Constraints in Gaussian Parallel Interference Channels. *IEEE Transactions on Information Theory*, 54(8):3471–3489.

[Petrosyan and Zaccour, 2016] Petrosyan, L. A. and Zaccour, G. (2016). Cooperative Differential Games with Transferable Payoffs. In Basar, T. and Zaccour, G., editors, *Handbook of Dynamic Game Theory*, pages 1–38. Springer International Publishing, Cham.

[Powell, 1973] Powell, M. J. D. (1973). On search directions for minimization algorithms. *Mathematical Programming*, 4(1):193–201.

[Rajamani, 2012] Rajamani, R. (2012). *Vehicle Dynamics and Control*. Mechanical Engineering Series. Springer US, second edition.

[Rawlings and Mayne, 2009] Rawlings, J. B. and Mayne, D. Q. (2009). *Model Predictive Control: Theory and Design*. Nob Hill Pub.

[Reinl and von Stryk, 2007] Reinl, C. and von Stryk, O. (2007). Optimal control of multi-vehicle-systems under communication constraints using mixed-integer linear programming. In *Proceedings of the 1st International Conference on Robot Communication and Coordination*, RoboComm '07, pages 1–8, Athens, Greece. IEEE Press.

[Riegger et al., 2016] Riegger, L., Carlander, M., Lidander, N., Murgovski, N., and Sjöberg, J. (2016). Centralized MPC for autonomous intersection crossing. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1372–1377.

[Rill, 2011] Rill, G. (2011). *Road Vehicle Dynamics*. Ground Vehicle Engineering Series. CRC Press, Taylor & Francis, Boca Raton, Florida.

[Rockafellar and Wets, 1998] Rockafellar, R. T. and Wets, R. J.-B. (1998). *Variational Analysis*. Grundlehren {der} {mathematischen} Wissenschaften. Springer-Verlag, Berlin Heidelberg.

[Rosen, 1965] Rosen, J. B. (1965). Existence and Uniqueness of Equilibrium Points for Concave N-Person Games. *Econometrica*, 33(3):520–534.

[Rosenthal, 1973] Rosenthal, R. W. (1973). A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2(1):65–67.

[Rudin, 1976] Rudin, W. (1976). *Principles of Mathematical Analysis*. McGraw-Hill Education Ltd, New York, 3rd edition edition.

[Sagratella, 2017] Sagratella, S. (2017). Algorithms for generalized potential games with mixed-integer variables. *Computational Optimization and Applications*, 68(3):689–717.

[Schulman et al., 2015] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France. PMLR.

[Shu, 2007] Shu, C.-W. (2007). High order numerical methods for time dependent hamilton-jacobi equations. In *Mathematics and Computation in Imaging Science and Information Processing*, volume 11 of *Lecture Notes Series, Institute for Mathematical Sciences, National University of Singapore*, pages 47–91. World Scientific.

[Soner, 1986a] Soner, H. M. (1986a). Optimal Control with State-Space Constraint I. *SIAM Journal on Control and Optimization*, 24(3):552–561.

[Soner, 1986b] Soner, H. M. (1986b). Optimal Control with State-Space Constraint. II. *SIAM Journal on Control and Optimization*, 24(6):1110–1122.

[Staniforth and Côté, 1991] Staniforth, A. and Côté, J. (1991). Semi-Lagrangian Integration Schemes for Atmospheric Models—A Review. *Monthly Weather Review*, 119(9):2206–2223.

[Stoer, 2005] Stoer, J. (2005). *Numerische Mathematik 1: Eine Einführung - unter Berücksichtigung von Vorlesungen von F.L. Bauer*. Springer-Lehrbuch. Springer-Verlag, Berlin Heidelberg, ninth edition.

[Stoer and Bulirsch, 2002] Stoer, J. and Bulirsch, R. (2002). *Introduction to Numerical Analysis*. Texts in Applied Mathematics. Springer-Verlag, New York, third edition.

[Taji, 2008] Taji, K. (2008). On Gap Functions for Quasi-Variational Inequalities. *Abstract and Applied Analysis*, 2008:1–7.

[Tröltzsch, 2009] Tröltzsch, F. (2009). *Optimale Steuerung partieller Differentialgleichungen: Theorie, Verfahren und Anwendungen*. Vieweg+Teubner Verlag, second edition.

[Uryas'ev and Rubinstein, 1994] Uryas'ev, S. and Rubinstein, R. (1994). On relaxation algorithms in computation of noncooperative equilibria. *IEEE Transactions on Automatic Control*, 39(6):1263–1267.

[von Heusinger and Kanzow, 2009] von Heusinger, A. and Kanzow, C. (2009). Optimization reformulations of the generalized Nash equilibrium problem using Nikaido-Isoda-type functions. *Computational Optimization and Applications*, 43(3):353–377.

[von Heusinger et al., 2012] von Heusinger, A., Kanzow, C., and Fukushima, M. (2012). Newton's method for computing a normalized equilibrium in the generalized Nash game through fixed point formulation. *Mathematical Programming*, 132(1):99–123.

[von Neumann, 1928] von Neumann, J. (1928). Zur Theorie der Gesellschaftsspiele. *Mathematische Annalen*, 100:295–320.

[von Neumann et al., 1944] von Neumann, J., Morgenstern, O., and Rubinstein, A. (1944). *Theory of Games and Economic Behavior*. Princeton University Press.

[Wang et al., 2018] Wang, Z., Zheng, Y., Li, S. E., You, K., and Li, K. (2018). Parallel Optimal Control for Cooperative Automation of Large-scale Connected Vehicles via ADMM. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1633–1639.

[Wong, 2001] Wong, J. Y. (2001). *Theory of Ground Vehicles*. John Wiley & Sons.

[Zanon et al., 2017] Zanon, M., Gros, S., Wymeersch, H., and Falcone, P. (2017). An Asynchronous Algorithm for Optimal Vehicle Coordination at Traffic Intersections. *IFAC-PapersOnLine*, 50(1):12008–12014.

[Zavala et al., 2008] Zavala, V. M., Laird, C. D., and Biegler, L. T. (2008). A fast moving horizon estimation algorithm based on nonlinear programming sensitivity. *Journal of Process Control*, 18(9):876–884.

[Zhang et al., 2010] Zhang, J., Qu, B., and Xiu, N. (2010). Some projection-like methods for the generalized Nash equilibria. *Computational Optimization and Applications*, 45(1):89–109.

Symbols

ν	player indication
$\mathbb{Z}$	set of integer numbers
Z_0^+	$\{x \in \mathbb{Z} \mid x \geq 0\}$, set of positive integer numbers
$\mathbb{Z}^+$	$\{x \in \mathbb{Z} \mid x > 0\}$, set of strictly positive integer numbers
$\mathbb{N}$	set of natural numbers
$\mathbb{N}_0$	$\mathbb{N} \cup 0$ set of natural numbers including zero
$\mathbb{R}$	set of real numbers
$\mathbb{R}^n$	n-dimensional real vector space
$\mathbb{R}^n_+$	$\{(x_1, \ldots, x_n) \in \mathbb{R}^n \mid x_i \geq 0 \text{ for all } i = 1, \ldots, n\}$
$\mathbb{R}^n_{++}$	$\{(x_1, \ldots, x_n) \in \mathbb{R}^n \mid x_i > 0 \text{ for all } i = 1, \ldots, n\}$
$\mathbb{B}^n(x)$	unit ball of dimension n and centre x
$\mathbb{B}(x; r)$	closed ball with radius r and centre x
ϑ	value function
$\mathbb{1}$	Unit Matrix (all entries are one)
$AC\left([t_0, t_f]\right)$	set of absolutely continuous functions
$\|\cdot\|_\infty$	essential supremum norm
$\|\cdot\|$	Euclidean vector norm or the 2-matrix norm
$C_k^n\left([t_0, t_f]\right)$	space of k-times continuously differentiable functions
$L_{f,\alpha}$	Lipschitz constant of a function f with respect to argument α
$\operatorname{dom} f$	$\{x \in \mathbb{R}^n \mid f(x) \neq\}$, domain of the point-to-set map $f : \mathbb{R}^n \to \mathbb{R}^m$
$\operatorname{cl}(\mathcal{D})$	closure of a set $\mathcal{D}$
$\operatorname{int}(\mathcal{D})$	interior of a set $\mathcal{D}$
$\delta\mathcal{D}$	boundary of a set $\mathcal{D}$

Acronyms

a.e.	almost everywhere
ADP	approximate dynamic programming
CFL	Courant-Friedrich-Lewy
CPU	central processing unit
CUDA	Compute Unified Device Architecture
DC	direct current
DPP	dynamic programming principle
DRAM	Dynamic Random Access Memory
ENO	essentially non-oscillatory
GNEP	generalized Nash equilibrium problem
GPG	generalized potential game
GPIO	general purpose input/output
GPS	global positioning system
GPU	graphics processing unit
HJ	Hamilton-Jacobi
HJB	Hamilton-Jacobi-Bellman
I^2C	inter-integrated circuit
KKT	Karush-Kuhn-Tucker
LMPC	linear model predictive control
MPC	model predictive control
NEP	Nash equilibrium problem
NMPC	nonlinear model predictive control
OCP	optimal control problem
ODE	ordinary differential equation
PDE	partial differential equation
QVI	Quasi-Variational Inequality
s.t.	subject to
SIMD	Single Instruction Multiple Data
SIMT	Single Instruction Multiple Threads

SM	streaming multiprocessor
TCP	transmission control protocol
UDP	user datagram protocol
USB	universal serial bus
VI	Variational Inequality
WENO	weighted essentially non-oscillatory