Dipl.-Ing. Zlatan Ajanović, MSc

Towards SuperHuman Autonomous Vehicles

DOCTORAL THESIS

to achieve the university degree of
Doktor der technischen Wissenschaften

submitted to
Graz University of Technology

Supervisor
Univ.-Prof. Dipl.-Ing. Dr. techn. Martin Horn

Institute of Automation and Control

Faculty of Electrical and Information Engineering

Graz, July 2020

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present doctoral thesis.

.....	.....
date	signature

Abstract

This thesis presents behavior and motion planning method that enables Autonomous Vehicles (AV) to achieve SuperHuman driving performance in terms of safety, efficiency and comfort. The developed method enables synergy of research in behavior and motion planning for Automated Driving, with research in eco-driving community, which target mainly complementary problem variations. Established approach in eco-driving is considering long planning horizons and multiple constraints (i.e. traffic lights, speed limits, etc.), but exclusively single lane driving. On the other hand, motion planning for Automated Driving considers multilane driving, but short planning horizons and decoupled (or hierarchical) solutions, focused on effectively reacting to the changing situations, and not on the long-term optimal behavior.

As a result of the synergy, developed search-based optimal motion planning (SBOMP) solution enables optimal Automated Driving scalable to various challenging scenarios in urban, rural and highway environment. As a highlight, SBOMP enables, what is believed to be, the first demonstration of optimal multilane diving in dense traffic with traffic lights, while achieving SuperHuman driving performance. Even though, this scenario is pretty common in everyday driving, it was not tackled by any of these research communities before.

The presented SBOMP framework is also extended to the third use-case, Performance Driving. By considering a more detailed vehicle model, SBOMP enables minimum lap-time driving on a slippery road, effectively entering and exiting drifting maneuvers and switching between right and left turns.

The presented work is extensively tested in simulation, benchmarked with human driving behavior acquired in driving simulator study and in-vehicle testing on proving ground. The results show that in challenging urban driving scenario with traffic lights, AV outperforms even the best human drivers in terms of safety, efficiency and comfort. While human drivers violate traffic rules and even cause crashes, by using predictive planning, AV manages to drive smoothly through the traffic.

Hopefully, this work contributes to the effort that Autonomous Vehicles become the first mass product of intelligent mobile robots in our society.

Keywords: Autonomous Vehicles, Motion Planning, eco-driving, Automated Driving, Performance Driving, SuperHuman Autonomous Vehicles

Kurzfassung

In dieser Arbeit werden Methoden zur Verhaltens- und Bewegungsplanung präsentiert, die es autonomen Fahrzeugen (autonomous vehicles, AV) ermöglicht, eine SuperHumane Fahrperformanz im Sinn von Sicherheit, Effizienz und Komfort zu erreichen. Die entwickelten Methoden ermöglichen Synergien der Forschungsgebiete Verhaltens- und Bewegungsplanung mit den oft komplementären Ansätzen aus dem Bereich der Fahreffizienz. Die Methode für energieoptimales Fahren basiert auf einem langen Planungshorizont und berücksichtigt Nebenbedingungen wie z. B. Ampelanlagen und Geschwindigkeitsbeschränkungen auf einspurigen Fahrbahnen. Im Gegensatz dazu basiert der Ansatz zur Bewegungsplanung für automatisiertes Fahren auf einem kurzen Planungshorizont, berücksichtigt mehrspurige Straßen, und legt den Fokus auf die Reaktion auf sich ändernde Verkehrssituationen, anstatt dem energieoptimalen Verhalten.

Als Resultat der oben angesprochenen Synergien ermöglicht die entwickelte suchbasierte Bewegungsplanung (engl. search-based optimal motion planning, SBOMP) optimales automatisiertes Fahren im Sinn von z.B. Energie oder Zeit, skalierbar auf verschiedene herausfordernde Szenarien in ländlichen und urbanen Umgebungen bzw. auf Autobahnen. Hervorzuheben ist, dass SBOMP die vermutlich erste Umsetzung von optimalem, mehrspurigem Fahren in dichtem Verkehr mit Ampelanlagen bei Erreichen SuperHumanen Fahrverhaltens ermöglicht. Obwohl dieser Anwendungsfall eine alltägliche Fahrsituation darstellt, wurde er von den oben genannten Forschungskreisen bisher nicht betrachtet.

Das vorgestellte SBOMP-Framework wird schließlich auf den dritten Anwendungsfall "Fahren am physikalischen Limit" erweitert. Unter Verwendung eines detaillierteren Fahrzeugmodells ermöglicht SBOMP Fahrten mit minimaler Rundenzeit auf rutschiger Fahrbahn, gleichmäßige Übergänge zu bzw. aus Driftmanövern und Wechsel zwischen Rechts- und Linkskurven.

Die vorgestellten Algorithmen wurden ausgiebig in Simulationen sowie auf der Teststrecke getestet und basierend auf Daten aus einer Fahrsimulatorstudie zu menschlichem Fahrverhalten bewertet. Die Resultate zeigen, dass das AV die fähigsten menschlichen Fahrer in einem anspruchsvollen urbanen Fahrszenario mit Ampelanlagen im Sinn von Sicherheit, Effizienz und Komfort übertrifft. Während menschliche Fahrer gegen Verkehrsregeln verstoßen oder sogar Unfälle verursachen, ermöglicht das AV durch prädiktive Planung eine reibungslose Fahrt.

Es bleibt zu hoffen, dass diese Arbeit zu den Anstrengungen beiträgt, autonome

Fahrzeuge zum ersten Massenprodukt intelligenter mobiler Roboter in unserer Gesellschaft zu machen.

Schlagwörter: autonome Fahrzeuge, Bewegungsplanung, energieoptimales Fahren, automatisiertes Fahren, Fahren am physikalischen Limit, SuperHuman autonome Fahrzeuge

Acknowledgment

To achieve a PhD, personal effort alone is not sufficient. It is necessary to be surrounded by people who motivate you, support you in your struggles, and celebrate your successes. Thank God I was in such an environment.

Therefore, I would like to thank all my colleagues from VIRTUAL VEHICLE Research Center, ITEAM project and Graz University of Technology who surrounded me during these years.

I especially want to thank my academic supervisor Prof. Martin Horn (TU Graz) for your encouragement and feedback throughout my PhD, my former group leader Michael Stolz for your advice and guidance during the first steps of my research career, my current group leader Georg Stettinger for your support and encouragement for finishing the PhD, and my colleagues Georg Nestlinger and Kailin Tong for proofreading this manuscript.

From ITEAM colleagues, I especially want to thank our project coordinator Prof. Valentin Ivanov (TU Ilmenau) for creating such an extraordinary environment for all of us, our department leader Prof. Daniel Watzenig for providing me the opportunity to pursue PhD in this project in the first place, and colleagues Halil Beglerović (AVL List GmbH), Enrico Regolin (University of Pavia), Tushar Chugh (Volvo Cars), Prof. Barys Shyrokau (TU Delft) and Prof. Bakir Lačević (University of Sarajevo) for your collaboration during secondments. I would like to thank Prof. Bakir Lačević, additionally, for your continuous support, advices and many fruitful discussions that raised the level of my work.

As a proof that we never know which moments will mark our future, I would like to thank my undergraduate professor Samim Konjicija, for teaching us Dynamic Programming with a big passion and a deep intuition, which helped me in fast acquiring of further knowledge and in a way marked the direction of my research.

Last but not the least, I would like to thank my wonderful family and friends, who motivate me to fulfill my ambitions. Especially, my wife Edna and sister Dž who also proofread my work with their eyes for details.

I would like to take this opportunity to express my gratitude and appreciation for the financial support provided by numerous entities, without which this effort would not have been possible. The project leading to this PhD thesis has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 675999, ITEAM project. This thesis

was written at VIRTUAL VEHICLE Research Center in Graz and partially funded by the COMET K2 – Competence Centers for Excellent Technologies Programme of the Federal Ministry for Transport, Innovation and Technology (bmvit), the Federal Ministry for Digital, Business and Enterprise (bmdw), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG). The validation work on proving ground was one of the Open Research projects at AstaZero, partially financed by RISE and Chalmers and led by SAFER.

Graz, November 2019

Zlatan Ajanović

Contents

1

Introduction

An autonomous vehicle is definitely more than a pure application for Control Theory, Artificial Intelligence or Robotics. It makes use of all existing methods and aggressively pushes development of new ones. The multidisciplinary interest in autonomous vehicles makes it challenging to introduce this topic concisely. Therefore, the introductory chapter of this PhD thesis aims to provide a concise overview of the thesis objective, contributions and structure. A more comprehensive insight into autonomous vehicle technology and theoretical foundations and related work are provided in following chapters.

Nowadays, it is hard to find a person who did not hear about Autonomous Vehicle (AV), Automated Driving (AD), Advanced Driver Assistance System (ADAS), Self-Driving Car, Driverless car, Robot Taxi, Intelligent Vehicle, etc. The naming diversity indicates the interdisciplinary interest and relatively low maturity level of this technology. Although people are exposed to news about AVs and used to everyday interaction with some automated systems such as elevators, metros, etc., AVs are mainly perceived as technology of the future, with some even questioning if it is realistic that it will be achieved at all.

Besides the public, in industry it has been more than 50 years since the first deployments of robots (in automotive assembly lines of General Motors) and 50 years since Shakey, the first intelligent mobile research robot that employed full sense-plan-act cycle, was developed. Still, currently deployed robots are typically manually programmed (i.e. industrial robots), teleoperated (i.e. medical robots) or "simple" rule based (i.e. robot lawnmowers or warehouse robots). Autonomous Vehicles cannot rely on "simple" rules as there is a high diversity of situations to be manually programmed. On the other hand, full benefits of autonomy cannot be achieved if teleoperated.

The challenge of automated driving is very hard, and some even compare it to the Mars mission, with the additional risk of endangering many human lives. The driving environment is not fully controlled (i.e. as in the case of elevators), and *understanding situation* around the vehicle is challenging as the environment is constantly changing, objects come in different sizes and colors, weather and light conditions vary a lot. On the other hand, the vehicle has to provide *timely, rational response* to any situation. It has to continuously adapt it's behavior and avoid collisions with other participants,

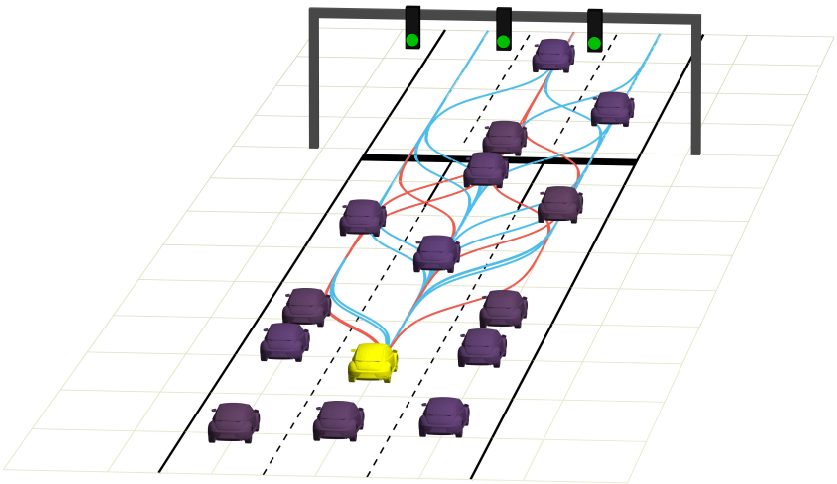

Figure 1.1: *Illustrative problem, multilane driving with traffic lights.*

obey traffic rules and comply with vehicle dynamics and factors like comfort, safety and efficiency. This is extremely challenging as the environment is usually highly dynamic with many participants and the full information is not available. Also there is a high degree of interaction with other drivers who may behave unpredictably. Additionally, the road conditions may change (i.e. slippery road). Increased research interest combined with modern computer technology (and respective computational power) as well as novel sensing technologies and advanced algorithm (for perception and rational decision making and control) might make this human dream finally possible. This achievement would make AVs the first commercial mass deployment of robotic systems with autonomous features.

Motivated by the example of human drivers who drive successfully both in good and bad conditions, appropriately adapting their behavior (i.e. reduce the speed on icy roads or during low visibility) to the changing conditions, we may conclude that the planning system is overall responsible for AV performance. However, most of the research in AVs is focused on getting the maximum performance from sensing and perception, as perception is the first problem we face. However, a lot of potential improvement which could be gained trough improved planning is unexploited. The current planning systems cannot efficiently cope with some common situations (i.e. multilane driving with traffic lights) since most of the solutions are crafted only for specific situations. This thesis focuses on expanding capabilities of planning systems of autonomous vehicles in a hope to finally achieve SuperHuman performance.

Achieving favorable behavior of the system by manipulating the system is of major interest of two fields, *Control Theory* and *Artificial Intelligence*. The approaches in these fields vary a lot and are rather complementary. They can be roughly grouped as *Control*, *Planning* and *Learning*. *Control* approaches try to devise the general mechanisms or a rule, which can be applied to bring the system from current to the desired state. Rules and mechanisms should be as simple but as general as possible, and should

provide strong theoretical guarantees on performance, such as stability, optimality, etc. They generally assume that the desired behavior description is available (i.e. reference state trajectory) and are hard to devise for complex systems. *Planning* approaches rely on deliberative reasoning about the current state and the sequence of future reachable states to solve the problem. In many cases planning approaches reason about what the desired state trajectory should be. However, they might be unusable if situation deviates from the planned trajectory. *Learning* is focused on improving performance based on experience or available data. Classic (model-free) learning problems start from no encoded information, tabula rasa, and learn a favorable behavior. However, it is hard to provide any guarantees for learning systems, since they can behave unexpectedly in seemingly simple, similar situations. Therefore, the approach adopted in this thesis is in a direction of highly needed reconciliation and cross-fertilization of these fields.

Control, Planning and Learning share many common benchmarks such as cart-pole balancing or acrobot. Having a good benchmark is essential as the greatest contributions come from applications. For example, Shakey robot inspired the development of A* Search, a basic algorithm for heuristic search, vehicle kinematics was the challenge of kinodynamic motion and automated driving fostered developments in motion planning in dynamic environment with state-time approach. Autonomous Vehicles, tackled in this thesis, are specifically challenging benchmark putting on test all available methods in these fields and fostering new developments.

1.1 Objective and Approach

As mentioned, the major focus of the research leading to this thesis is behavior and motion planning for fully autonomous vehicles achieving SuperHuman performance. The motivating use-case is Energy-efficient Driving, so-called *eco-driving*. The eco-driving problem is being researched even since the 70s, and many solutions based on predictive planning with long horizons (i.e. full trip) were developed for energy-efficient driving considering road-slopes and "using" the gravity, passing trough multiple traffic lights without stopping and so on. However, solutions in this community exclusively consider single lane driving, which makes them rather theoretic and not so practical. In the best case, they can be used on highways with no traffic [1].

On the other hand, in the mainstream *automated driving* motion planning community, the research considers multi-lane driving problem. The focus is on effectively reacting to the changing situations and not on the optimal behavior, and computationally efficient solutions with provable guarantees. This lead to decoupled (or hierarchical), Model Predictive Control (MPC)-like re-planning solutions, with planning horizons in the order of 10 seconds, which prevents to consider energy-efficiency. To consider energy-efficiency properly, MPC solver must deal with nonlinear models (i.e look-up tables for efficiency map) and handle arbitrary road slope function. This is out of the capabilities for most MPC solvers [2][3].

Major focus of this research is unification of these fields and closing the gap between eco-driving and AD motion planning community, offering general motion planning

framework capable of dealing with eco-driving, and extending eco-driving to all driving situations such as multilane driving with traffic lights. The proposed motion planning framework is scalable, so it can be applied for other criteria as well, and detailed vehicle models necessary for high performance driving can be considered.

The general approach adopted is to solve the hardest challenge first (i.e. fully autonomous vehicle - Level 5), in all situations, with some assumptions (i.e. assuming full information is available). This requires planning (behavior and motion) to provide decision making and control of the vehicle in all driving environments (i.e.highway, rural and urban driving) and all different scenarios like merging, overtaking, crossing intersection, etc. A further step would be to strengthen assumptions (i.e. limited perception). The specific approach adopted during development of the solution was to efficiently balance model expressiveness with computational complexity (i.e. vehicle lateral model). To avoid problems of too specific cost functions (i.e including obstacle avoidance in cost function), so called *reward hacking*, the cost function is kept very simple and complex requirements are expressed as constraints.

Major methods used in this work are based on Heuristic Search and Dynamic programming, dealing well with combinatorial optimization problems. For the continuous dynamic aspect of the problem, motion primitive sampling with a grid-based pruning of theoretically infinite number of continuous trajectories was adopted, based on hybrid A* approach. Several admissible and non-admissible domain-specific heuristics were developed to focus the search and improve performance. To deal with the changing environment, Search-based planning was used in MPC manner with moving, finite horizon, considering long-term effects trough heuristic function.

1.2 Contributions

Detailed contributions of the thesis are presented after the review of related work in section 3.3, while here we present only the overview. Major contributions can be summarized as:

- Theoretical results on longitudinal energy efficient driving (eco-driving).
  - Definition and analytical formula for calculating *optimal cruising velocity*.
  - Comprehensible and concise cost function based on energy-efficiency including economic aspect such as hourly rate.
- Formalization of general planning problem for automated driving, scalable to wide range of scenarios, including:
  - comprehensive catalog of semantic features which describes requirements for driving, including speed limits, traffic lights, other vehicles, road markings, traffic rules, etc.,
  - hybrid continuous-discrete vehicle motion model capable of capturing nonholonomic constraints without increasing complexity, effectively reducing one degree of freedom.
- Resolution complete optimal kinodynamic motion planning algorithm based on heuristic search for solving mentioned problems, including:

  - provable guarantees on optimality,
  - capability to cover full range of velocity (slow and fast trajectories),
  - multiple admissible and inadmissible heuristic functions (i.e. model-based, relaxed problem-based, Machine Learning-based),
  - capability to consider nonlinearities and non-holonomic constraints.
- Solving several challenging autonomous vehicle use-cases.
  - Believed to be, the first solution for multilane eco-driving, considering overtaking possibility in eco-driving.
  - Believed to be, the first solution for optimal motion planning for automated driving in multilane traffic with traffic lights.
  - Believed to be, the first solution for minimum lap time driving on slippery roads with arbitrary shape (i.e. varying curvature, mixed right and left curves).
- A novel approach to model exploration for value function learning; to be used as heuristics to speed up the search.
- Extensive demonstration, verification and validation, including:
  - a novel approach to validation of SuperHuman automated driving performance,
  - simulation study of various scenarios,
  - driver simulator study to compare to human driving,
  - testing in the vehicle on proving ground.

1.3 Results and publications

Publications

This thesis is partially based on the following publications:

- Book chapters:

  [4] Ajanovic, Z., Stolz, M. and Horn, M., 2018. Energy-Efficient Driving in Dynamic Environment: Globally Optimal MPC-like Motion Planning Framework. In Advanced Microsystems for Automotive Applications 2017 (pp. 111-122). Springer, Cham.

  [5] Ajanovic, Z., Stolz, M. and Horn, M., 2017. Energy efficient driving in dynamic environment: considering other traffic participants and overtaking possibility. In Comprehensive Energy Management–Eco Routing & Velocity Profiles (pp. 61-80). Springer, Cham.
- Conference/Workshop papers:

  [6] Ajanovic, Z., Stolz, M. and Horn, M., 2018, June. A novel model-based heuristic for energy optimal motion planning for automated driving. Proceedings of the 15th IFAC Symposium on Control in Transportation Systems (CTS 2018). IFAC.

[7] Ajanovic, Z., Lacevic, B., Shyrokau, B., Stolz, M., Horn, M., 2018 October. Search-based optimal motion planning for automated driving. In Intelligent Robots and Systems, 2018. IROS 2018. IEEE/RSJ International Conference on. IEEE.

[8] Ajanovic, Z., Lacevic, B., Stettinger, G., Watzenig, D., Horn, M., 2018 July. Safe learning-based optimal motion planning for automated driving. In Proceedings of the ICML/IJCAI/AAMAS 2018 Workshop on Planning and Learning (PAL-18).2018.

[9] Ajanovic, Z., Beglerovic, H., Lacevic, B., 2019. A novel approach to model exploration for value function learning. In Proceedings of the RSS 2019 Workshop on Combining Learning and Reasoning – Towards Human-Level Robot Intelligence. 2019.

[10] Ajanovic, Z., Regolin, E., Stettinger, G., Horn, M., A. Ferrara, 2019. Search-Based Motion Planning for Performance Autonomous Driving. In IAVSD Vehicles on Road and Tracks 2019. IAVSD.

- Patent application:

  [11] Ajanovic, Z., Stolz, M., 2016. Predictive control system for autonomous driving vehicle. Patent application, Intellectual Property Office of the United Kingdom.

- Magazine articles:

  [12] Ajanovic, Z., Stolz, M. and Horn, M., 2017. Energy Efficient Autopilot: Energy efficient driving in dynamic environment. Virtual Vehicle Magazine (pp. 36-37). Virtual Vehicle, Graz.

Awards

During doctoral process for the work presented in this thesis author was awarded following rewards:

- June 2018: **IFAC Young Author Award**, for the paper "A novel model-based heuristic for energy optimal motion planning for automated driving", awarded during IFAC Symposium on Control in Transportation Systems in Savona, Italy.
- October 2019: **Hans List Fond scholarship**, for PhD thesis, awarded by AVL List GmbH in Graz, Austria.

1.4 Structure of the PhD thesis

The presented PhD thesis is organized in 9 chapters.

Chapter 2, AV Technology, includes concise overview of autonomous vehicle technology including motivation, history, classifications, architecture and components as well as current trends.

Chapter 3, Related work, provides comprehensive but concise overview of the related work focused on planning for Autonomous Vehicles, especially three use-cases: Energy-efficient Driving, Automated Driving and Performance Driving.

Chapter 4, Problem Definition, elaborates definitions of the driving problem and three variants tackled in this thesis (longitudinal eco-driving, multilane automated driving and performance driving on slippery road) including the challenges as well as assumptions and requirements on the solutions.

Chapter 5, Modelling, presents models used for all aspects of driving problem such as vehicle, road, traffic rules, other traffic participant and performance criteria.

Chapter 6, Motion Planning, presents solutions for mentioned problems (eco-driving, automated driving and performance autonomous driving), including development of several admissible and sub-admissible heuristics.

Chapter 7, Simulation, presents extensive simulation study on various scenarios from highway, rural and urban driving with appropriate analysis and benchmarking.

Chapter 8, Validation, presents a novel validation methodology and results of validation performed in driving simulator, in-vehicle testing on proving ground providing valuable conclusions on achieving aimed SuperHuman driving capabilities in terms of safety, efficiency and comfort.

Chapter 9, Conclusion, presents conclusion on the work and possible directions for further research.

2

Autonomous Vehicle Technology

Autonomous vehicle technology is a complex interdisciplinary challenge. It requires advancements in several fields including sensing technologies, algorithms and models for robust perception, planning and control of the vehicle. The aim of this chapter is to introduce autonomous vehicle technology. The chapter starts with motivation for developing this technology, followed by an overview of the history of the development of autonomous cyber-physical systems with focus on autonomous vehicles. Finally, the mainstream approach in delivering this technology is presented including classification of levels of automation and technology components from sense-plan-act cycle. The chapter is concluded with a short insight in the current trends in this field.

As every emerging technology, AV technology is going trough many phases explained well with Gartner hype curve [13]. Based on the report from 2018, AV (level 4) technology passed *peak of inflated expectation* as early publicity produced a number of success stories, but was accompanied also by several failures. AV is now approaching *trough of disillusionment* as several participants fail to deliver and community realizes that the problem is harder than initially expected. It is expected that it will take more than 10 years to reach the plateau of productivity, when mainstream adoption starts to take off. The relaxing fact is that AV technology in reduced form (Level 1-2) is already being introduced incrementally trough ADAS functions, which keeps the momentum of development and improves user adoption.

2.1 Motivation

Besides the hype caused by moonshot character of AV technology, sometime compared even to the Mars mission [14], there are several clear promises which motivate research and development of this technology. Promises can be roughly grouped into three clusters: safety, efficiency, and comfort [15]. Combined achievement of these promises could potentially change the mobility as we know it [16].

2.1.1 Safety

The number of the annual road traffic deaths has reached 1.35 million, making the road traffic injuries the leading killer of people aged 5-29 years [17]. As human safety is one of the major concerns of vehicle engineering, reinforced with initiatives to totally remove traffic fatalities such as *Vision Zero* [18], minimizing the occurrence and consequences of traffic collisions is one of the major goals of AV technology.

There are two complementary approaches to this problem, passive and active safety. Passive safety is focused on reducing the effect of crash when it already happens. Systems such as seat belts, airbags and safe body structures are saving thousands of lives. About 50 % of non-belted heavy goods vehicle occupants would have survived accident if they had used seat belt properly [19].

On the other hand, active safety aims to prevent the crash in the first place. Human drivers are causing 90% of crashes because of fatigue, lack off attention, driving under influence of drugs, alcohol, etc. [19]. By automating the driving, these causes could be circumvented as digital controller does not suffer from these problems like human drivers do. Additionally, autonomous vehicles could have faster response time and make more informed decisions, as they can rely on more sources of information, i.e.use additional sensors which work even in conditions where visibility is low and can get information from other vehicles or infrastructure using connectivity. For example, using additional sensors and advanced models, controller could have precise estimate of road surface friction and use it for planning driving as such that the critical situations are avoided when driving on icy roads. Introduction of active safety functions as mandatory in newly produced vehicles is already in progress. The Electronic Stability Control (ESC) is mandatory in newly produced passenger vehicles in Europe and Advanced Emergency Braking System (AEB) additionally in commercial vehicles [19]. The real-world study showed the 38% overall reduction in rear-end crashes in vehicles equipped with AEB [20].

However, active safety systems only complement human driver and improve performance when it is possible. To achieve safety levels higher than a human driver in fully AV, errors introduced by AV technology should be minimized which is still a challenging goal. Current traffic safety sets a very high bar with about 210 million km driven between two fatal accidents (about 230 years of non-stop driving with velocity 100 km/h) [21].

2.1.2 Efficiency

Transportation has a significant impact to the society in many aspects of the everyday life. The transportation sector alone accounts for 20% and 27% of the total carbon dioxide (CO2) emissions in Europe and in the USA, respectively [22][23]. Traffic congestions cause a massive waste of time, fuel, and money. In the USA alone, 3.1 billion gallons (11.7 billion liters) of fuel, 6.8 billion hours of extra time (equivalent to 47 million average summer vacations), and 153 billion dollars in delay and fuel cost [24], while in the European Union, about 1% of the total GDP annually is wasted due to congestions [25]. Increasing environmental awareness, strict regulations on

greenhouse gas emissions and constant desire to increase the range of electric vehicles as well as the big economic benefits motivated significant research interest in the field of energy-efficient driving.

Over the past few decades, many different approaches addressing this topic were developed. Some approaches are related to the vehicle design optimization, some to using alternative propulsion systems, some to the driving behavior optimization and some to the efficient traffic management. Energy-efficiency is affected by vehicle weight. By reducing probability of a collision with active safety, a part of passive safety systems could be reduced, therefore reducing the overall weight of the vehicle and improving efficiency. In the case of transportation vehicles, fully removing the driver could imply totally removing the cab enabling higher load/vehicle mass ratio. On the topic of driving behavior, in [26] authors presented a study which shows that the driving behavior has a rather big influence on energy consumption. It was shown that energy consumption may vary in a range of approx. 30% depending on driving behavior in terms of aggressiveness.

Advancements in Connected and Autonomous Vehicle (CAV) technology might enable achieving optimal driving behavior in terms of energy-efficiency, so-called *eco-driving*. By utilizing information about the upcoming driving route (i.e. the road slope profile), information from on-board sensors (i.e. the position and velocity of vehicle in front) and the ability to precisely control the vehicle's propulsion, vehicle velocity trajectory can be optimized with respect to the energy consumption to avoid unnecessary acceleration/deceleration (stop/go) cycles. This can lead to savings ranging up to 20% [1]. Efficiency could be improved by keeping a convoy of vehicles on a small inter-vehicle distances one behind the other, in a formation called *platoon*. Having fast response time and direct communication with other vehicles enables driving in a such formation without sacrificing safety. This in turn can reduce aerodynamic drag resistance up to about 30% [27], which is significant if we know that at highway speeds, half of energy is used to overcome aerodynamic drag [28]. Technologies like Vehicle-to-Anything (V2X) communication could provide information about future state of upcoming traffic lights so driving can be also optimized by utilizing to that information to arrive at the traffic light on green light phase (catch the "*green wave*"). This could potentially improve the fuel consumption by 47% and lower CO2 emissions by 56% [29].

Besides improving energy efficiency and reducing emissions on individual vehicle level, CAV technology could have effects on the level of the whole traffic system. Some studies show that implementing automation which satisfies all traffic rules can reduce traffic flow as professional human drivers usually improve traffic flow by slightly breaking the rules, but it might be expected that in the future traffic rules for automated vehicle change according to their capability [16]. If properly managed, there are many possible benefits of CAVs [30]. Driving on a shorter inter-vehicle distances can improve road capacity as more vehicles can occupy the same road. Having all vehicles equipped with V2X communication and being centrally controlled, could totally remove the need for traffic lights on intersections. Intersection could be fully automated [31], which could bring significant capacity benefits ranging even more than 100% [16]. However, it is not realistic that all vehicles will become CAVs overnight, and related benefit is significantly influenced by the penetration rate (percentage of CAVs in the

whole traffic). While the full benefit is available with 100% penetration rate of CAVs, a penetration rate of 40% can bring only about 10% of the benefits [16]. However, recent work [32] shows that it is possible to use CAVs to influence human drivers leading to potential 20% increase in road capacity which promises other benefits in mixed-autonomy traffic.

AV technology reinforces some independently developed mobility concepts (i.e. Mobility as a Service (MaaS)) where transportation users do not own the vehicle but they use it and share it with others [33]. Several companies (i.e. Uber or Lyft) are already operating platforms with human driven fleets providing MaaS, making taxi service smother and more transparent. Theoretically, it would be possible to optimize the system and maintain equal mobility with CAV fleet sizing from about 10% up to about 33% of currently used independent conventional vehicle fleets [16]. Fleets of AVs serving as Robot-taxis can be used to provide timely and convenient transportation to anybody in Automated Mobility on Demand (AMoD) system, considering even multiple occupancies in vehicles [34]. AMoD is one of the most promising use cases of AVs (besides autonomous trucking on highways), due to possible high exploitation rate of generally still expensive AV technology.

2.1.3 Comfort

Potential contribution of AV technology for improving travel experience is multifaceted ranging from lowering the effort and improving accessibility of driving to optimizing and customizing driving comfort to fit passenger's taste. If the vehicle is fully autonomous, the stress of driving can be transferred to the vehicle (i.e. in traffic jams or long rides) and driver can use added time for some other, more preferred activities (i.e. work or entertainment). Fully autonomous vehicle could also conveniently pick the passenger at the origin and drop at the destination, avoiding need to search for parking place and walking from/to the vehicle. Besides parking itself autonomously after destination is reached, the vehicle could also continue to operate as in the case of AMoD. One of the particularly important benefits of AV is improved accessibility of travel for persons who do not have access now (i.e. elderly, children, persons with disabilities, etc.)

Besides accessibility and lowering the effort of driving, the driving experience itself could be improved. As vehicle is precisely controlled, riding comfort for passengers can be improved with predictive control based on future driving situation to reduce motion sickness. Driving style could also be customized to the passenger's taste, providing consistent experience over the long drives. As a consequence of overall traffic improvement and predictive planning of the trip, travel times could be shorter in general and longer distances would be more accessible for driving.

2.2 History

General public perception is that AVs are objects of the future to that extent that many do not believe that they are even possible. The futuristic image hides the facts

that human desire to automate the work has long roots and that autonomous vehicles are an active research topic for decades.

Some of the first visions about automated driving were publicly presented on the New York World's Fair in 1939. General Motors presented Futurama: Highways & Horizons, the vision of how world could look like in the 60s. Futurama exhibited a model of future transport system with automated highways and vast suburbs [35]. In the 50s, GM together with RCA (Radio Corporation of America) did some initial work and developed Autonomous Highway System tests where a steel cable, used as the guidance, was embedded in the road and tracked by magnets mounted in the vehicle [36][37]. This, however, was not so practical, and further research on automated vehicles will require almost three decades to get active again.

In the 50s, there was a growing general awareness of the potential of *automatic control*. Based on the *feedback concept* [38], the *Sense-Act* loop could be employed to achieve self-adaptable behavior, providing desired behavior even in the presence of disturbances [39]. Conference "Automatic Control" held in 1951 in Cranfield (England) and the "Frequency Response Symposium" held in 1953 in New York, as well as numerous textbooks, brought automatic control to the attention of many engineers. Norbert Wiener's work, among others, brought concept of "feedback" to the attention of the wider community also, expanding the control from technical systems to other systems in the fields such as biology, philosophy, and the organization of society. Wiener's book [40] also popularized the term "Cybernetics" (derives from the Greek "kybernetike", which means the art of steersmanship) such that term "cyber" is actively used even today, describing almost anything that deals with computers, robots or Internet [41].

In a quest to develop thinking machines, in 1955, John McCarthy, then a young Assistant Professor of Mathematics at Dartmouth College, initiated organization of Summer Research Project. To avoid strict focus on feedback concept, as it was in Cybernetics, he picked a distinct name for this topic, "Artificial Intelligence", setting the name as it is still used [41]. In 1956, Dartmouth Summer Research Project eventually took place with participation of many notable scientist such as Ray Solomonoff, Marvin Minsky, John McCarthy, Claude Shannon, John Nash, Herbert A. Simon, Allen Newell, and others, making this a seminal event for Artificial Intelligence (AI) as a field [42]. The initial goal of developing unifying theory or paradigm that guides AI research was not achieved, as interests were quite diverse ranging from algorithms to play games and automating induction to artificial neural networks. This diversity of approaches is one of the characteristics of AI field even nowadays.

In the 1950's and 1960's, one notable stream of development of control theory was focused on achieving optimal behavior (i.e. Optimal Control). This was mainly based on the R. Bellman's discovery of principle of optimality and Dynamic Programming method based on it [43] and the discovery of Maximum Principle (MP) by L.S. Pontryagin and his students, V.G. Boltyanskii and R.V. Gamkrelidze, in 1956–58 [44]. However, this work will stay rather theoretical with application mainly focused on space and military until late 70s. In 1978, after publishing of successful applications of MPC in the publication "Model Predictive Heuristic Control" [45] interest for MPC started raising [46].

Starting from 1966, Stanford Research Institute began working on Shakey robot [47], one of the first intelligent mobile robots, utilizing the full *Sense-Plan-Act* cycle. Although the performance of Shakey was very low compared to current robots (i.e.averaging about two meters an hour), it was one of the most ambitious attempts at an autonomous robot at the time. Shakey could reason out a path using input from its variety of sensors and robustly follow it. Research on Shakey fostered several contribution widely used even today. These contributions include A* search (used for path planning [48]), Hough transform adapted for use in vision [49] (applied later for lane detection) and STRIPS rules for planning [50]. Around the same time, at Stanford Artificial Intelligence Laboratory, experimenting with outdoor Stanford Cart was ongoing [51]. Stanford Cart was initially created in order to study driving a vehicle on the Moon remotely from Earth. In 1966, it was reconfigured with the idea of self-driving vehicles. It was able to follow a bright white line under controlled lighting by utilizing stereo vision system.

In the 70s and partially the 80s as well interest for the research in AI slowly decreased, marking the first AI winter.

Perhaps one of the first works on autonomous vehicles relying only on vehicle on-board modifications was presented in 1977 by group of Sadayuki Tsugawa from Mechanical Engineering Laboratory in Tsukuba, Japan. They presented the vehicle which could drive autonomously up to a speed of 30 km/h [52]. Vehicle control was based on processed input from two cameras (i.e.road pattern recognition and obstacle detection) and preprocessed optimal controls were acquired from the look-up table based on that input.

In Europe, in 1986, after several years of development, group of Ernst Dickmanns presented VaMoRs vehicle [53]. VaMoRs managed to drive autonomously, and by 1987 it was capable of driving itself at speeds of up to 96km/h (60 mph), making it probably the first autonomous vehicle capable of driving on speeds comparable to human drivers [54]. The tests were executed at the closed stretch of newly built highway. After this success, European automotive industry became more seriously interested in the research in Autonomous Vehicles. With financial support from European Union, PROMETHEUS (Program for a European Traffic with Highest Efficiency and Unprecedented Safety) was launched in 1986 [55]. The main objective was to make driving in Europe safer, more economical, more environmentally acceptable, more comfortable, and more efficient. This was a massive research effort culminating with a demonstration in 1995, a 1600 km drive by the VaMP car, 95% of which was driven autonomously [56]. VaMP was reaching maximum speed of 180 km/h on a free stretch in the northern Germany.

Meanwhile in US, in the 1985, as a part of DARPA's Strategic Computing Program, The Autonomous Land Vehicle project demonstrated offroad vehicle ALV capable of driving 2 mph (about 3 km/h) [57]. In 1986, Carnegie Mellon University researchers built NavLab 1 vehicle, but it wasn't until the late 80s that software systems could drive the vehicle, reaching the top speed of 20 mph (about 32 km/h) [58]. In 1986, the same year the first NavLab vehicle was built, and the team of Geoff Hinton introduced approach for training neural networks using back-propagation [59]. Using Artificial

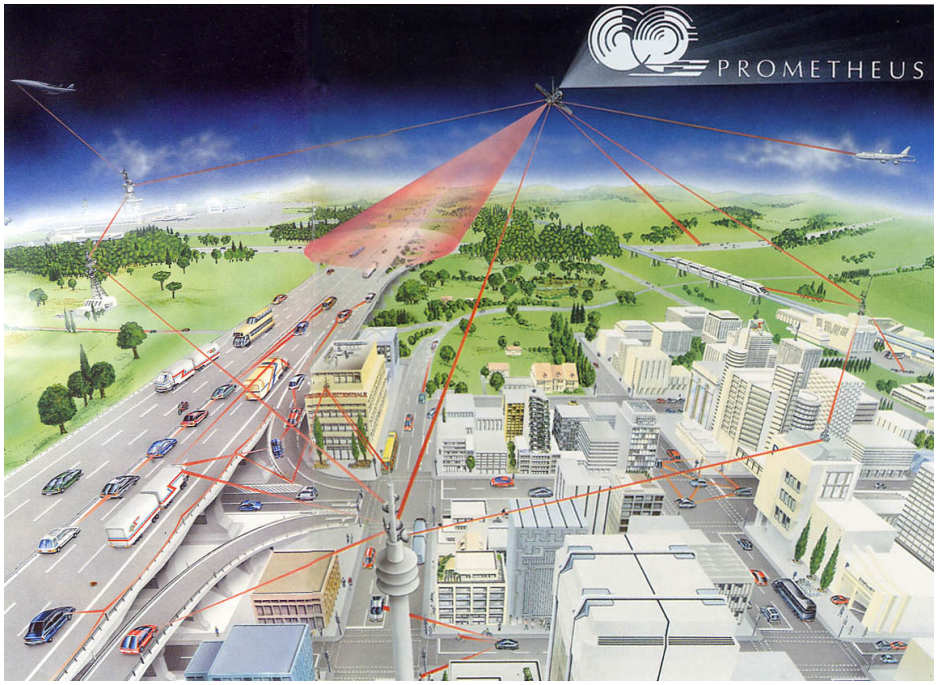

Figure 2.1: *PROMETHEUS project vision.*

Neural Networks, Dean A. Pomerlau developed ALVINN [60], capable of generating control inputs directly from camera input. Using ALVINN, the vehicle was able to drive autonomously, but at a maximum speed of about 1 mph, mainly due to low computation power needed to process the networks. This was probably the first end-to-end approach to control of the vehicle, coupling directly camera input to vehicle steering and throttle inputs (pixel-to-torque approach). In 1995, based on improved driving road detection and more modular approach, NavLAb made USA tour driving 2,850-mile from coast to coast, driving 98.2% autonomously [61].

Major efforts of these initial developments lied in perception and computer vision system. On the other hand, not so much effort was invested in intelligent decision making beyond rule based decisions.

Complementary to autonomous driving, significant effort was invested in development of Intelligent Transportation Systems (ITS). In 1986, as the first research program in North America focused on ITS, California Partners for Advanced Transit and Highways (PATH) Program was founded. PATH, as 2011 incarnation, is still running at University of California, Berkeley [62]. It aims to apply advanced technology in order to increase highway capacity and safety, and reduce traffic congestion, air pollution, and energy consumption. Besides traffic engineering related topics, PATH extensively focused on autonomous vehicles with several notable demonstrations such as four-car automated platoon in San Diego, CA, in 1994 [63].

Around the mid 90s, the reduced interest in AI research as well as the lack of funding marked the second AI winter.

Although the research in autonomous vehicles was largely reduced, it still partially

continued. Besides the work done on automating highway driving, which was focused primarily on improving capabilities of already existing passenger vehicles, rather visionary approach of developing novel vehicle concepts was very active with coordination of INRIA, France. Several projects such as PRAXITELE (mid 90s) [64][65], followed by CyberCars (2001 - 2004) [66][67] and CyberCars-2 (2006-2009) [68]), as well as some international cooperations examined ways of transforming urban mobility using a fleet of novel connected and autonomous vehicles as a complement to public transport. The approach was to develop customized "urban autonomous and electric public cars" and to efficiently operate the fleet of these cars (which can be shared among users), with a goal to minimize urban traffic congestion problems. Several notable results were demonstrated. In 1996, autonomous parking was demonstrated [69], as a work carried out within the framework of the PRAXITELE project. In 2002, Griffith University's Intelligent Control Systems Laboratory (ICSL), in cooperation with researchers from INRIA's IMARA Laboratory executed what is believed to be the world's first on-road demonstration of cooperative driving solutions in unsignalized intersection scenario and an overtaking maneuver scenario [70].

Modern interest for autonomous vehicles was kick-started in 2004 when DARPA launched Grand Challenge [71] with offered prize in money (1 million USD). Final race happened in the Mojave Desert region of the United States, along a 150-mile (240 km) route. None of the participating teams managed to drive the full course of the race. A year after, in 2005, the second DARPA Grand Challenge was organized. Five vehicles successfully completed the course, with Stanley vehicle, from Stanford University team led by Sebastian Thrun, winning the challenge [72]. In 2007, DARPA organized Urban Challenge [73], the 97 km long drive, with traffic consisting of 50 manned and unmanned vehicles, from 35 teams. The Challenge consisted of several requirements such as driving in lanes, three-point turns, parking, and maneuvering through obstacle fields. Race showed impressive progress towards motion autonomy, but there were still many issues, and even the first collision of autonomous vehicles. The BOSS vehicle from Carnegie Mellon University [74] won the race, followed by Junior vehicle from Stanford University [75], VictorTango from Virginia Tech [76] and MIT [77]. After the DARPA Challenges, Sebastian Thrun joined the Google in 2009 and started working in Google Self-Driving Car project.

In 2010, as a preparation for World Exposition to be held in Shanghai, China, VisLab decided to make VisLab Intercontinental Autonomous Challenge (VIAC). VIAC expedition was 13,000 km, three-month long trip, from Parma (Italy) to Shanghai (China). The vehicle control was based on cooperative leader–follower approach. As no complete map was available, the leader was human driven and ran autonomously only in places where no decision had to be made (i.e. on highways) with the follower fully autonomous [78]. Another notable work related to cooperative vehicle convoys was aiming to improve energy consumption by utilizing air drag reduction caused by driving in convoys, so-called platooning. EU project named SARTRE (Safe Road Trains for the Environment) [79], running from 2009 to 2012, aimed to develop strategies and technologies to enable platooning on public highways. To promote work on cooperative driving, TNO (the Netherlands Organisation for Applied Scientific Research), organized Grand Cooperative Driving Challenge (GCDC), in 2011. GCDC

was the first competition to implement realistic heterogeneous, cooperative driving scenarios [80]. Participants had to come up with control strategies which are robust to unexpected behavior of other vehicles, varying data quality or failures in communication, among others. The winning team was AnniWAY from Karlsruhe Institute of Technology (KIT), lead by Christoph Stiller [81].

In 2010, researchers from Stanford University together with researchers from Volkswagen Group tested Autonomous Audi TT on the Hill Climb course in Pikes Peak. Vehicle was able to successfully navigate 156 turns over 12.42 miles in 27 minutes without a driver [82]. In May 2012, Google's test vehicle, Toyota Prius, modified for test of autonomous driving was licensed by the Nevada Department of Motor Vehicles (DMV) with the first license in the world issued for a autonomous vehicle [83]. Soon after, many institutions followed and tests on public roads began in Nevada. In 2013, 125 years after Bertha Benz made first overland journey in automotive history, research team from Daimler, in cooperation with KIT, followed the same route from Mannheim to Pforzheim, Germany, in fully autonomous manner [84]. The route was 103 km long and covered rural roads, 23 small villages and major cities (e.g. downtown Mannheim and Heidelberg). Vehicle relied solely on vision and radar sensors in combination with accurate digital maps to obtain a comprehensive understanding of complex traffic situations.

In the end of 2015, Tesla motors rolled out version 7 of their software in the USA that included Tesla Autopilot capability, making them the first Original Equipment Manufacturer (i.e. vehicle producers) (OEM) to offer Automated Driving (Level 2) capabilities [85]. Later they added a new "summon" feature that allowed cars to self-park at parking locations without the driver in the car [86]. By October 2018, test vehicles of Waymo, Google self driving car project spin off company, had traveled over 10,000,000 miles (16,000,000 km) in automated mode, adding about 1,000,000 miles (1,600,000 kilometers) each month [87]. In December 2018, Waymo announced that they are starting to offer "Waymo One" service, the first commercial self-driving service to hundreds of early riders [88], beginning the commercialization of Level 4 AD on certain regions.

The presented overview is a sampling of a very rich history of AVs. It is almost impossible to cover the full breadth of AV research. Currently, OEMs, technology companies, start-ups and many others, invest significant efforts into development of autonomous driving technology. Landscape is very wide and acquisitions of relatively young startups are reaching billions. It is outside of the scope of this work to map all activities, but I will try to provide an overview with several milestones to see the depth and the scope of the presented challenge. Although significant research effort was invested, it is important to note that there still exists no solution that is capable to drive in new, unknown environments. Solutions are always limited either by functionality or by geographic operational coverage.

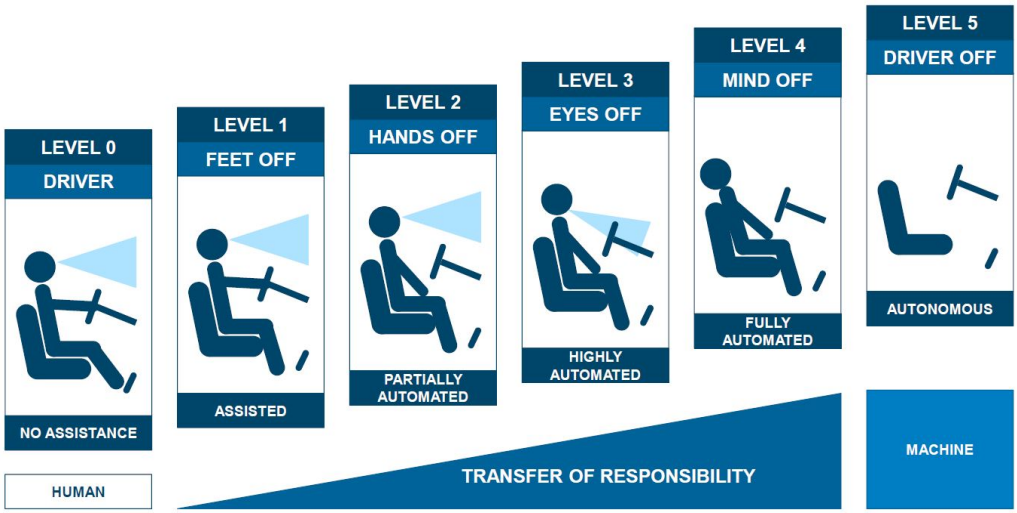

Figure 2.2: *Levels of automation.*

2.3 Technology

As AV should operate in dynamic and not fully controlled environment, the technology necessary for achieving it encompasses all other technologies present nowadays, such as aviation autopilots, industrial automation, space missions, etc. AV requires totally new sensor types (i.e. Lidars), advanced signal processing (i.e. computer vision), High Performance Computing (HPC), reliable decision making, robust and redundant actuation etc. Many components are still developing and there is still no standardized approach for many of them. Several topics which are widely accepted are levels of automation as well as rough architecture of the control system of fully autonomous vehicles.

2.3.1 Levels of vehicle automation

To make distinctions between capabilities of different systems, the Society of Automotive Engineers (SAE) identifies 6 levels of autonomous vehicles, ranging from no driving automation (Level 0) to full driving automation (Level 5), as illustrated on the Figure 2.2 [89].

- **Level 0 - No assistance:** At this level, the driver is the sole decision maker being in control of the steering wheel, brake and throttle. Driver may be supported by warning systems such as lane departure warning or forward collision warning systems but without any direct action on the vehicle.
- **Level 1 - Driver assistance:** At this level, the driver can get function specific support from vehicle with either steering or acceleration. The driver is still in control and must stay fully engaged, but one or more independent functions are automated. Examples include adaptive cruise control and lane keep assist.
- **Level 2 - Partial automation:** At this level, the vehicle controls combined

functions such as automatic cruise control and lane keeping, providing both steering and throttle/brake. The driver is still responsible for the control of the system and has to take over control at any time, if corrections are needed. The parking assistance or Tesla's autopilot feature [90] or Mercedes-Benz DISTRONIC PLUS with Steering Assist [91] are examples of the Level 2 systems.

- **Level 3 - Conditional automation:** A Level 3 system provides autonomous driving in certain conditions. All safety critical functions are automated, but in some cases, control may be returned to the driver with prior notice. Driver can be partially distracted but should be ready to take over when requested by the system to do so.
- **Level 4 - High automation:** In this level, vehicle controls all aspects of the driving, so that the driver is not expected to perform any task. Vehicle of this level might not even have the steering wheel. Vehicle autonomous operation might however be restricted to some area, so-called geofence. Example of this level is Waymo's self-driving taxi service [88].
- **Level 5 - Full automation:** This is the highest level of vehicle automation. At this level vehicle can operate autonomously in all driving conditions where human driver can drive. Currently there are no examples of Level 5 AD.

Focus of this work is on Level 5 autonomous vehicles which can operate fully autonomously in all driving conditions where human drivers operate.

2.3.2 Architecture of Autonomous Vehicles

Despite several demonstrations of end-to-end control approaches, where AV control gets as input raw data from camera and generates directly desired controls to vehicle, it is widely accepted that AV need a modular approach. Modular approach divides the problem into smaller subproblems and solves them individually [92]. This approach has several advantages including improved transparency, easier development cycles, verification and reuse. Major components of AV control system are *Sense*, *Plan* and *Act* and they form *Sensing-Planning-Acting* cycle. This is a well known approach in designing intelligent agents in AI or mobile robots [93].

Sense

The goal of Sense component is to provide timely and reliable information about state of the driver, vehicle and environment based on readings from multiple sensors. Raw or preprocessed information from sensors is combined together in a process called sensor fusion and provides outputs such as object list (i.e. from multi-object tracking) and occupancy grids representing vehicle environment.

Besides the environment perception, major tasks of sense component are ego vehicle (i.e. controlled vehicle) state estimation (i.e. localization) as well as driver/passengers monitoring. The task of environment perception is to provide vehicle with complete and reliable information (with uncertainty information) about the state of the environment (i.e. position and velocity of other vehicle participants, drivable road, etc.),

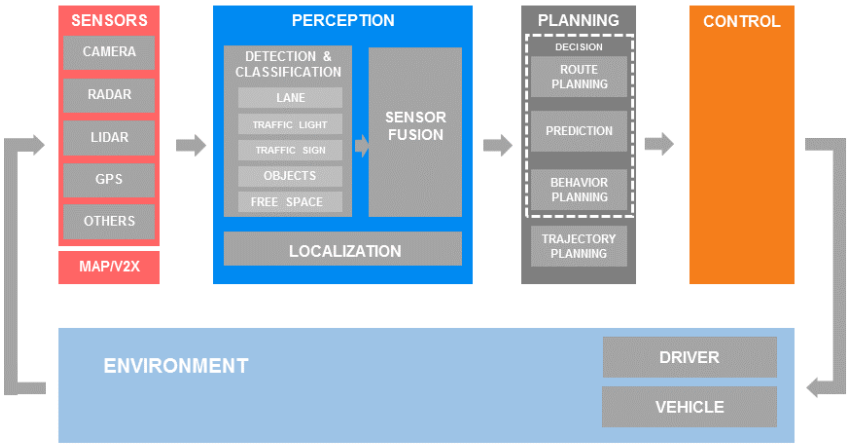

Figure 2.3: *Autonomous Vehicle architecture.*

robust to environmental changes such as bad weather conditions, low lightning etc. The task of ego vehicle state estimation is related to the environment perception, but it is focused on ego vehicle and may include ego vehicle position, velocity and orientation (i.e. localization), but also higher vehicle dynamic states such as wheel-tire forces, side slip angle and slip ratio. In-vehicle monitoring system is responsible for providing vehicle control system with information about state of the driver and the passengers. Monitoring systems like this are particularly important for lower levels of automation where responsibility is shared between the human and the vehicle.

Because of inherent disadvantages of each sensor type, various sensor types are used together in a sensor suite complementing each other. Sensors like radar, lidar, camera, etc. are used and their outputs are fused using sensor fusion to provide single source of information, decrease uncertainty, improve robustness, accuracy and availability trough redundancy [94]. One of the major challenges in Sense component is to achieve real-time, robust processing of sensor inputs. Processing is usually hierarchical and multiple sensors can be combined at a lower level (i.e. without much processing) or/and higher level (i.e. preprocessed, with objects detected). Tasks of preprocessing include object detection and classification (i.e. vehicle detection, traffic sign recognition, lane detection, etc.), semantic segmentation (i.e. free road detection), etc. [95].

On-board sensor perception can be augmented by communicating with cloud systems and other vehicles. Systems like this include Electronic Horizon, High Definition Maps and V2X communication [96].

Plan

The goal of Plan component is to provide a reasonable plan on how to drive vehicle in dynamic world based on the information provided by Dense component and a priori models (i.e. vehicle model, traffic rules, etc.). As output, plan component provides

collision free and comfortable trajectories, as sequence of vehicle states (i.e. position, orientation and velocity, timestamped). Vehicle trajectory is derived based on HD maps, environment perception, localization and desired goal provided by the user.

Major tasks of plan component are route planning, prediction of other participants (i.e. trajectories or intentions), behavior and motion planning. Planning tasks are usually organized hierarchically. Routing is the highest level problem, it answers which road to take to go from point A to point B on the map. It takes as input the map data in form of the street network and gives the output drivable path (i.e. much like Google Maps Navigation). Map information may include real-time traffic information. Navigation map is simpler than HD map as it usually represents the graph on which some graph search algorithms are applied for solving it [97]. As vehicle moves among other cars, to provide collision-free plans, vehicle needs to predict what other vehicles will do in the future. In several situations such as merging, where there is a high interaction between the vehicles, it is not possible to directly predict trajectory of other vehicles as ego vehicle actions influence actions of other participants. In these cases, the vehicle has to detect intention and evolve different situations of interactions during planning [98][99]. Behavior planning assumes higher level planning in a form of discrete decision with a smaller subset of actions-maneuvers (i.e. stay in the lane, change lane, follow, stop etc). Behavior planning was historically decoupled from motion planning and it provides motion specification for which motion planning provides feasible trajectory. Behavior and motion planning are tightly coupled as solutions of motion planning are required to evaluate certain maneuver in behavior planning. Both behavior and motion planning influence the quality of the final trajectory which includes making decisions to avoid obstacles and satisfy traffic rules, vehicle dynamics, etc. Because of the coupling, recent approaches integrate behavior and motion planning. Integrated behavior and motion planning is presented as trajectory planning in *stl* space, where maneuvers represent different homotopy classes. As the focus of this thesis is particularity on integrated behavior and motion planning, this will be further discussed in related work and modeling chapter.

Besides driving on the structured roads, distinctive task is planning in unstructured environment (i.e. parking lot navigation). This might seem like a harder problem as some simplifications from planning in structured environment are not vali. However, it is relatively easier as motion is limited to lower speeds and is less dynamic.

Act

The goal of Act component is to accurately track plans provided by the plan component and to minimize deviations caused by disturbances (i.e. unmodelled vehicle dynamics, external factors such as wind). It takes as input target trajectory from planning module as sequence of desired future vehicle states (i.e. position, velocity, orientation). Based on the error from current measured vehicle state (i.e. position, orientation and velocity or deviation from lane provided by Sense component), control component generates as outputs acceleration, braking and steering commands to the vehicle. These commands are then executed on the vehicle by drive-by-wire system which ensures robust execution and necessary redundancy for target reliability.

Major tasks of act component are longitudinal and lateral stabilizing control of the vehicle. These tasks may be decoupled (i.e. using methods like LQR, PID control) [100]. However, maximum performance is reached only with coupled control of this MIMO system (i.e. MPC) [101]. Typically, control command must be feasible to the car, and preferably smooth (i.e. minimize jerk). However, in some situations it might be also necessary to control the vehicle at the limits of handling [102]. Lower levels of driving automation (i.e. Level 1 or Level 2), functions like Lane Keeping Assistance, ACC [103] may be realized directly with *Sense-Act* cycle, avoiding use of *Planning*. However, for higher levels of automation, full *Sense-Plan-Act* cycle is necessary.

2.4 Trends

Vehicle automation is not an isolated trend and as such it influences and is influenced by many other trends such as vehicle electrification and connectivity, vehicle sharing and mobility-as-a-service. The safe approach for developing AV is to use all possibilities (i.e. connectivity) but not critically rely on them.

Across the industry two major approaches for introducing AV are present. One approach aims to introduce Level 4 AD directly on limited geographic regions and incrementally increase operational range. This is mainly by technology companies in a form of Robot-taxi [88]. The other approach is lead mainly by OEMs, who offer vehicles with iteratively increasing levels of automation. As mentioned earlier, currently commercially available are Level 2 AD.

In regards to the research and development, major efforts so far were focused on Sense part, which is reasonable since reliable perception is a prerequisite for other components. As mentioned before, lower levels of automation can be achieved only based on Sense and Act components, or enhanced with simple rule-based decision making. As these problems are being partially solved, problems in planning become notable, and it is clear that Level 3-5 AD and long-term autonomy are not possible without reliable planning which can deal with complex driving situations [104]. Therefore, the focus of this thesis is on the planning for Autonomous Vehicles which achieve Level 5 AD.

3

Related Work

Autonomous vehicle is essentially a mobile robot, or an intelligent agent in a more general sense, therefore enormous work in Control Theory, Artificial Intelligence and Robotics provide extensive theoretical foundations for solving this problem. On the other hand, autonomous vehicle builds upon more than a century of engineering experience since Karl Benz introduced the first practical automobile. Pedigree like this makes work in autonomous vehicles widely spread across disciplines.

As it is stated in Lavalle's[1] book from 2006 [105], Robotics, Artificial Intelligence and Control Theory, three traditionally distinct topics are on a collision course. We may say that they are already in collision and autonomous vehicles are one of the most challenging applications all of these methods are being applied to. Interest in autonomous vehicles is far beyond these three fields. Researchers in the field of Vehicle and Traffic engineering, among others, focus on extending capabilities of conventional vehicles and traffic system by adopting the concept of autonomous vehicles. Additionally, large effort of non-technical researchers in the field of human factors, economy, legislative, etc. is crucial for wide acceptance and adoption of this technology in the society. This makes autonomous vehicles one of the most cross-disciplinary problems of our time.

Because of the overwhelming literature available, the focus here is on three different use-cases of autonomous vehicles: Energy-efficient Driving, Automated Driving and Performance autonomous Driving. Together, these use-cases present pillars for autonomous vehicles with SuperHuman performance in terms of efficiency, comfort and safety.

3.1 Use-cases

Although all three use-cases mentioned above are dealing with autonomous vehicles, they attracted attention of different communities and have different historical developments, so mainstream approaches differ, as well as development directions.

[1] Steven M. LaValle, the author of the book "Planning algorithms" and the pioneer of RRT paradigm in sampling-based motion planning.

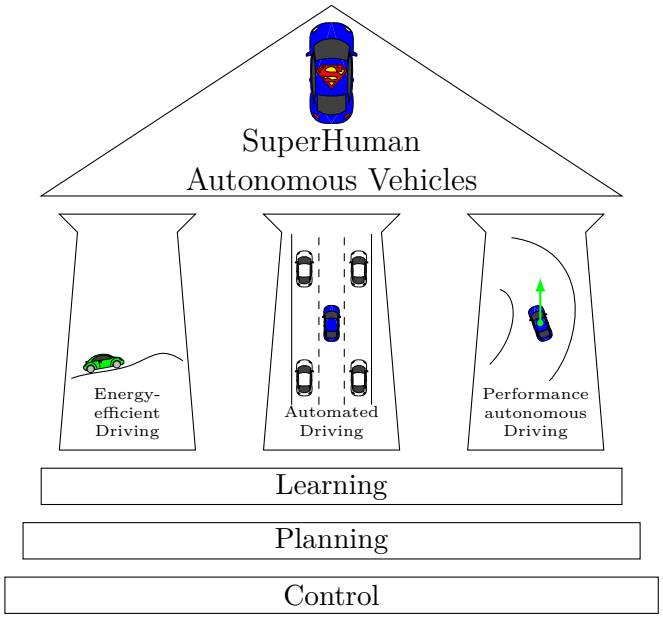

Figure 3.1: *Related work.*

Saving fuel has been a major concern ever since automobiles came to use, and represents *Energy-efficient Driving* use-case. This use-case attracts wide attention and it was addressed even in popular science shows like MythBusters [106]. Special trainings were offered to drivers to improve their driving behavior, to achieve better fuel efficiency. Research in this field goes back at least to 60s with a quest to improve *energy-efficiency* in road and rail vehicles by optimizing driving behavior. It started as a theoretical analysis of possible fuel saving, but served also as a benchmark problem for optimization methods. Although initially the results of optimization offered only an insights into efficient behavior and advice to human drivers, as vehicle automation becomes reality, they offer basis for optimal control of the vehicle.

The other concern is to free human drivers of repetitive tasks and improve safety by automating driving task. This represents *automated driving* use-case. From a perspective of Artificial Intelligence and Robotics automobile can be seen as "a bit" more complex mobile robot, having more complex dynamics, dealing with more complex environment and having higher safety requirements. The goal here is to achieve acceptable behavior while driving among other traffic in everyday driving situations (i.e. driving on multilane roads, highway, intersection, etc.) and satisfying traffic rules.

The final use-case, *Performance Driving*, considers driving on the limits of vehicle handling. This is an use-case mainly addressed by researchers interested in vehicle dynamics and motorsport enthusiast. The approach here is to model the behavior of a race driver and emulate it using control, planning and learning approaches to achieve minimum lap-time driving.

These use-cases together serve as representative examples of wide operational range of autonomous vehicles. Their complementarity motivates synergy between developments as well as need for general solution to achieve SuperHuman autonomous vehicles.

Figure 3.2: *Approaches to increase driving energy-efficiency.*

3.1.1 Energy-efficient Driving (eco-driving)

Increasing environmental awareness, strict regulations on greenhouse gas emissions and constant desire to increase the range of electric vehicles, as well as the large economic benefits, drive a lot of research in the field of energy efficient driving. As a result, there are many different approaches addressing this topic. Some approaches are related to the vehicle design optimization, some to using alternative propulsion systems and some to the driving behavior optimization. In [26] the authors present a study which shows that the driving behavior has a rather large influence on energy consumption. It is shown that energy consumption may vary in a range of approx. 30% depending on moderate or aggressive driving behavior.

Driving behavior related approaches for improving energy-efficiency can be grouped into: “eco-routing”, “using road slope information”, “traffic light assist”, “platooning” and “overtaking”, as shown in Figure 3.2.

Eco-routing

The goal of eco-routing is to determine the most energy-efficient route for the trip, which may differ from the shortest or the fastest one. Problem is set by creating road network graph, where weighted edges represent road segments and weights represent the cost of traveling that segment. Graph search algorithms are used to find the shortest path on the graph representing the route with the lowest energy consumption.

Many solutions are available in the literature, i.e. [107] which presents an example of using historical data to determine an eco-route or [108] which considers traffic lights as well. Extensive review of routing algorithms is presented in [97].

Driving on the open roads

The classic eco-driving problem is driving on the open road with a variable slope. In this problem, the goal is to determine the vehicle speed trajectory which results in optimal energy consumption for the given transportation task, while taking into account traffic rules (i.e speed limits). In the following sections classic eco-driving problem is extended with consideration of traffic lights and other traffic participants.

Knowledge about the upcoming driving route, the road conditions and the ability to control the vehicle's propulsion enables the optimization of the speed trajectory of the vehicle with respect to the energy consumption. This problem has been extensively studied. What is believed to be the first scientific research on economization of train operations was presented in 1968 [109] (overview of the optimal control of a train [110]). One of the earliest energy-efficient velocity planning for road vehicle based on Dynamic Programming was presented in 1983 in [111] and analysis is extended to the fleet in [112]. An alternative approach was presented even earlier in 1976, based on so called "pulse and glide" method, consisting of periodically accelerating vehicle in short pulses and then coasting without any engine torque [113].

Discrete dynamic programming (DP) has been used extensively in the last decade for generating energy-efficient global velocity trajectory e.g. in research focused on heavy duty vehicles [114][115]. A comparison between different optimization methods (Euler-Lagrange, Pontryagin's Maximum Principle, DP, and Direct Multiple Shooting) was presented in [116], where the reader can find an analysis on the DP grid choice, as well as some tips on backward and forward dynamic programming. Other works based on Dynamic Programming are presented in [117][118]. Major drawback of Dynamic Programming solutions is high computational requirements. Because of this drawback in [119] authors used a cloud service to compute energy efficient velocity trajectories. Other approach recently presented is iterative Dynamic Programming [120], which offers better solution quality in same computation time. Alternative approach to reduce computational requirements is exploiting heuristic function to focus search on more promising regions of the search space. A* search algorithm enables to do this optimally while keeping guarantees on optimality [121].

Explicit, closed-form, solutions usually offer much better computational performances. Several explicit solutions for global trajectory planning were also provided in the literature. However, they usually assume some simplification. For example in [122] authors assume low speed so air drag effect can be neglected, while in [123] authors present formula for calculating weight factor for travel time in optimization, to achieve desired stationary velocity. Some of many other alternatives of global trajectory planning include rule based [124] and Evolutionary Algorithms [125].

Instead of looking for a whole trip and global trajectory planning, some authors proposed using Model Predictive Control (MPC) with horizons of 5-10 seconds [126][127].

Besides predictive approaches relying on look-ahead planning, several solutions exist based on instantaneous actions. In [128] Neural Networks (NN) were used to define single action by taking lookahead slopes in account as input to NN, without lookahead planning. In [129] authors introduce Estimated minimum principle, where driving trajectory is "optimized" online by applying instantaneous control (no preview).

Previously, optimized speed trajectories were usually used to advise a human driver via an appropriate human-machine-interface (HMI). Rarely, optimized speed trajectories were used to directly provide a reference value for underlying low-level controllers such as cruise control. Increasing vehicle automation is expected to change this in the near future, and several of these mentioned works have already presented prototypes.

Driving in the presence of other traffic participants

Majority of the early eco-driving works considered driving on an empty road. If leading vehicles are neglected in the optimization, the unconstrained plan will not be fully achievable in real driving conditions, and may in some situations lead to drawbacks in energy consumption and very likely to bad driver acceptance. As in general in eco-driving, only single lane driving is assumed, the approaches that consider other traffic consider only vehicle following and not overtaking possibility. Multi-lane driving makes problem combinatorial, and therefore hard to solve for several methods.

In [130] a possible solution for a vehicle following problem is presented, showing different concepts for safe vehicle following, defining helpful concepts such as the safe distance, time-inter-vehicular and time-to-collision. In [131] authors use A* search for safe vehicle trajectory as a basis for energy oriented adaptive cruise control. Several works used MPC approach for dealing with the vehicle following problem. MPC was used to control a hybrid vehicle driving over a hill and performing vehicle following in [132]. Several methods (iterative PMP, LQR, SQP) were used in Moving Horizon approach (MPC) for non-cooperative [133] and cooperative [134] driving. Instantaneous control previously mentioned was also extended also to safe vehicle following [129].

Besides vehicle following, several works deal with optimizing overtaking execution. In these works ([135][136][137][138]), velocity trajectory planning is done by minimizing deviation from the desired velocity while the vehicle is overtaking. These approaches provide a locally optimal solution by trying to track the reference velocity that should not be reference after passing other vehicle. They introduce unnecessary braking after overtaking occurs to minimize the deviation.

Based on author's best knowledge there exists no other approach (before our work was presented) that holistically considers multilane eco-driving as such that it provides velocity trajectory, the decision if the vehicle should overtake or not, and where is the best location for potential overtaking, based on energy-efficiency. Some work appeared recently, inspired by our work [139].

Driving in the presence of Traffic Lights

Driving in the presence of traffic lights puts additional constraints on velocity trajectory. It has been widely studied how one could use knowledge of Signal Phase And Timing (SPaT) of oncoming traffic lights with the aim to improve energy-efficiency and reduce trip time. Approaches in the literature are usually hierarchical. On the top level, limits on velocity or desired velocity are determined so that the vehicle can pass one or more traffic lights without stopping. The output is then fed to MPC-based local motion planning. The approaches for top level planning vary, such as simple kinematics [29], Dijkstra's algorithm [140] or simple MPC [141], but the structure is usually the same. Local motion planning is usually based on MPC. It may consider other vehicles as well, but only vehicle following and single-lane driving is considered [29] [141]. Some approaches extend planning to the situations with incomplete knowledge about SPaT as well [142].

Several very recent works advance the state-of-the-art. In [143] authors extend by optimizing engine torque, the brake force, and the gearshift while ensuring safe distance separation and traffic speed limits. They show 8% energy savings in a realistic scenario. In [144] authors present public transit use-case with communication to Traffic Lights. Optimal velocity is displayed to the driver via mobile application and 5.5% improvement compared to regular driving without advice was shown in tests in real traffic in Beijing. In [145] authors incorporate stochastic red light duration delays in higher level planning via chance-constrained optimal control to improve robustness of tracking.

To the best of author's knowledge, none of the related works successfully tackled combined multi-lane driving in the presence of traffic lights so far (besides our work). Authors of [145] state in outlook their plans to extend the work to a multi-lane driving situations and refer to our work.

Search-based methods for eco-driving

As this PhD work relies on search-based methods (i.e. A* search, Dynamic Programming, Dijkstra, etc.), in this section, more focus is drawn on these methods applied to eco-driving problem.

Many variants of Dynamic Programming were already presented. Dijsktra's algorithm was used in [146] and [140]. In [125] authors formulated the problem as distance-based so no cycles appear, which makes it possible to implement Breadth First Search (BFS). BFS improved computational complexity to $\mathcal{O}(n)$ compared to Dijsktra's $\mathcal{O}(n\log n)$.

As mentioned earlier A* search was also used for planning energy-efficient velocity trajectory. In [121] it was used for electric bicycle. For heuristic function, authors used kinetic and potential energy, as well as a rolling resistance, but for estimation of air drag resistance, the authors used an upper instead of a lower bound. The results were suboptimal with a difference of around 1.2% from another optimal control strategy. In [147] authors introduced air drag and a time proportional cost without interdependence. For air drag, they used a tunable minimum velocity, and for the time-proportional cost, they computed a minimum time based on the maximum ve-

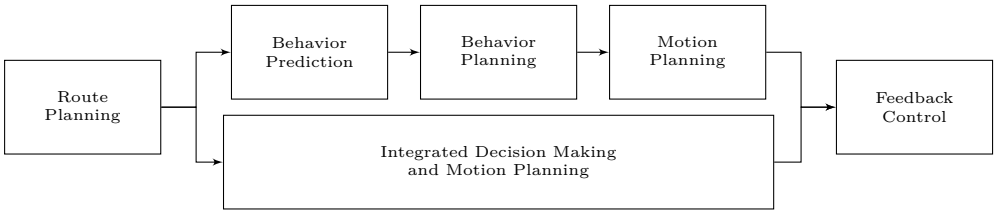

Figure 3.3: *Schema of the planning and decision-making components for automated driving. Upper branch represents hierarchical planning approach and lower branch represents integrated planning approach.*

locity. This approach leads to a not-so-precise estimation and a loss of admissibility if velocities are lower than the tuned value. Another use of A* search was presented in [131], but authors did not reveal the computation of their heuristics. From the results, it is clear that some of the heuristics are giving a suboptimal solution.

Conclusion on Energy-efficient Driving

Based on reviewed work we can conclude that research in energy-efficient driving so far considers only single lane driving. In [148] an overview of the existing approaches and current state of the art can be found. Furthermore, the presented solutions are partial and consider only specific scenarios and not general solution exist. This might be partially caused by limited actions, as most of the time solutions are presented to a driver as an advice and therefore have to stay simple. However, this should change with vehicle automation so concepts from energy-efficient driving research should be integrated into automated driving.

3.1.2 Automated Driving

Driving automation is based on classic robotics Sensing-Planning-Acting cycle, where Motion Planning (MP) is the crucial step. Task of MP is to provide a collision-free motion plan from the given starting pose to the given goal region, taking into account system dynamics, obstacles and possibly desired criteria (cost function). MP has been researched since 1970s [149], mostly in robotics. However, vehicle automation application brings new challenges as the environment is cluttered, dynamic, complex, uncertain and the vehicle is often operating on the limits of its dynamics. Several works give a comprehensive overview of current motion planning approaches in vehicle automation domain [3][150][2][151][152].

Hierarchical planning

Usually, planning for automated vehicles is structured hierarchically, with *route planning* at the top, and is operating with the smallest frequency (e.g. once a trip), followed by the behavioral layer responsible for *decision-making* on maneuvers to be executed. When a decision is known, the local *motion planning* layer generates a trajectory or a

waypoint that satisfies safety and traffic rules, and is further executed by a stabilizing controller [3].

The behavioral layer was initially implemented using finite state machines and most of the participants in DARPA Urban challenge used it [74][153][75]. To deal with uncertainty, solutions based on Partially observable Markov Decision Process (POMDP) [154] were also proposed. In general, decision making requires a sample trajectory to estimate whether a certain maneuver is possible. This natural coupling of trajectory planning and decision making calls for the integration of behavioral and local MP [2]. However, integration of the behavioral layer with local MP introduces combinatorial aspects and a local minimum problem [155].

Integrated decision-making and motion planning

Coupling of behavioral and local motion planning has been tackled by several approaches with limited success. In [156], where authors proposed spatiotemporal state lattices used with a dynamic programming search to plan collision-free motion in the presence of dynamic obstacles. The proposed search was rather fast (less than 20 ms), yet only a limited number of velocity variants (7 different) were used and lattice construction is such that full stop is not possible. Similar approach with spatiotemporal lattices is presented in [157]. Another approach is presented in [158], authors formalized the generation of all possible combinations and used local planning [155] for each one of them. The best one of them is then chosen as the global optimal result. This approach is not applicable to environments where many combinations are possible, especially where traffic lights are present as they introduce infinitely many combinations. Additionally, several authors [159][160] used mixed integer programming approaches to treat multiple variants with the assumption that the desired velocity is defined and the deviation from this velocity is used within a cost function for optimization. This simplification leads to the local optimal solution as the gap for the lane change can be influenced by the velocity history of the ego vehicle.

Interactive driving

Vehicles in traffic do not operate in isolation, actions of the ego vehicle will have effects on other drivers' actions and vice versa [99]. Interaction can be modelled by Game Theory [161], i.e. as cooperative [162], non-cooperative [163] game, etc. But Game Theory restricts interaction to the simple set of rules and therefore does not represent full specter of interactions. Several works [164] [165] recently provided solutions for some scenarios, but there still does not exist a general solution dealing with all traffic situation [166].

Search-based motion planning for automated driving

As this PhD work relies on search-based methods (i.e. A* search), in this section, more focus is on these methods applied to automated driving. One of the first uses of A* search for autonomous vehicle motion planning was presented in [167], the result

of a Prometheus project. The author used A* search to find the shortest time motion in the presence of dynamic obstacles and introduced state-time space for dealing with dynamic obstacles. A* search was also used in DARPA Urban challenge by many teams, mainly for planning in unstructured environment (i.e. parking lot) and route (mission) planning. Worth mentioning here is planning in unstructured environment from Stanford team where the authors introduced Hybrid A* search to deal with the problem that kinodynamic motion primitives do not generally end exactly on grid points [168]. Another approach worth mentioning is from Carnegie Mellon University team which also used A* search, but applied multi-resolution lattice state-space to deal with kinodynamic problem [169]. Stanford team used search-based methods for planning on structured roads as well. They used Dynamic Programming applied to road network and generated cost-to-go map instead of single route and used it with a MPC-like forward trajectory planner to achieve the shortest travel time [75].

Spatiotemporal lattice approaches mentioned earlier ([157] [156]) rely on using Dynamic Programming search over the lattice graph. In [157] DP was using GPUs, so authors managed to find solution in real time. However, these methods were not so practical as they explore the whole search-space and problem explodes when reasonable precision is required. These solutions rely on 6 or 7 dimensional lattices, consisting of two-dimensional position, orientation, time, velocity, etc.

Several works in past few years rely on heuristic search-based approaches (like A* search), which focuses search on promising parts of the search-space. An interesting approach for kinodynamic motion planning is presented in [170] where the authors used exploration guided heuristic search to plan the ego vehicle motion in the dynamic environment. Another recent application of A* search for planning safe trajectories on structured roads while driving in dynamic traffic is shown in [171]. Authors used flexible steps to deal with kinodynamic problem. However, they consider only 3D search-space (two-dimensional position and time), and only 9 branches, which makes it not complete.

Very recently, some hybrid approaches appeared, which heavily rely on search algorithms. Authors of [172] apply hierarchical decomposition of the problem to path and velocity planning problems to solve the problem. Also, recent work similar to our and referencing our work is presented in [173]. Authors used Dynamic Programming for rough planning with spline based quadratic programming

Conclusion on motion planning for automated driving

As it was shown, there are many different approaches to planning for automated driving. Search was established recently as very popular solutions, due to combinatorial nature of the problem. Usually, it is used together with some optimization based smoothing to improve final solution quality. However, the problem is far from being fully solved. Many of the solutions are shown to work only in specific scenarios and it is hard to generalize them to consider all different traffic situations that may occur (i.e. multilane driving with traffic lights, merging, intersections, unstructured environment etc.). Having a single planner for considering all scenarios is crucial for proving safe operations. Additionally, none of the above mentioned motion planners considers

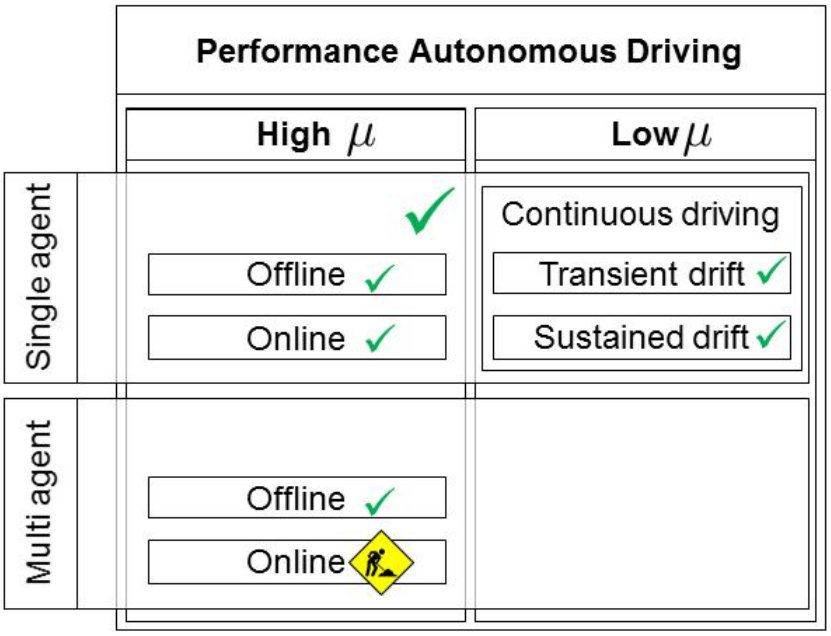

Figure 3.4: *Performance Driving overview.*

optimal passing through multiple intersections with traffic lights. Although some approaches show potential to be extended, many would probably fail. Regarding solution quality, almost all solutions either decompose the problem or make some assumptions, which makes them not complete and therefore not optimal. For those which are complete, it is challenging to achieve real-time performance. Besides extending to different scenarios, most of the approaches are not well suited for the extension to multi-agent situations where interaction aspects are important, robust planning and execution or perception limits-aware planning.

3.1.3 Performance autonomous driving

Similarly to other use-cases, autonomous vehicle control for Performance Driving is based on a Sensing-Planning-Acting cycle. Due to different dominant effects in vehicle dynamics, we can distinguish different approaches suitable for road surfaces with high and low friction coefficients. High friction surfaces enable to achieve better controllability of the vehicle, while on lower friction surfaces the control action often enters the saturated region.

High friction surfaces (high μ)

For driving on high μ roads, many different solutions were presented so far, and some were even tested on real vehicles [174][175]. Predictive planning of future vehicle states can enable real-time control of driving while avoiding static obstacles [176]. Recently the approach has been extended to racing scenarios with multiple agents (although

not real-time) [163]. Although these approaches use nonlinear bicycle model with Pacejka's tire model, road surface had a high μ, which can be observed as vehicle is not performing drifting or trail-braking maneuvers. This Motion Planning (MP) approach is based on an exhaustive search and works well only for short horizons (due to exponential complexity). Furthermore, it is well suited only for high friction conditions where fast transitions between simple, constant velocity, primitives can be achieved. Approaches like this are not well suited for controlling a vehicle in lower friction conditions, which require larger horizons and a more detailed vehicle model as the control action often enters the saturated region and where road surface is not consistent. As in minimum-time driving on high μ surfaces optimal trajectory minimizes curvature of the driving path, the path is on the edge of the road, so it is not robust for a gravel like roads which are not consistent.

Slippery surfaces (low μ)

Another line of work considers driving on gravel-like roads, i.e. driving with high side-slip angles like drifting, trail-braking, etc. to improve robustness of the trajectory.

Most of the current works in this direction consider two specific scenarios: sustained drift or transient drift. One example of a transient drift scenario is drift parking, as shown in [177], where the vehicle enters temporarily a drift state. On the other hand, in sustained drift scenarios, the goal is to maintain steady-state drifting. Velenis et al. modelled high side-slip angle driving and showed that for certain boundary conditions it can be a solution for the minimum-time cornering problem [178], [179]. Tavernini et al. showed that aggressive drifting maneuvers provide minimum time cornering in low-friction conditions [180]. Because of computational complexity, most of these works cannot achieve online performance.

Based on results generated offline, by using [179], You and Tsiotras proposed a solution for learning the primitive trail-braking behavior, enabling online generation of trail-brake maneuvers [181]. This approach decomposes trail-braking into three stages: entry corner guiding, steady-state sliding and straight line exiting. A similar decomposition of the problem was also presented in [182]. This one divides the horizon into three regions, finds a path for each region (using Rapidly-exploring Random Trees (RRT), rule-based sampling and Proportional Integral Control) and then concatenates them. As it appears from the results, though, in the drifting region the rule-based solution produces non optimal solutions. Impressive demonstration of model-based reinforcement learning approach on scaled vehicle is shown in [183]. However, as for the aforementioned sustained drift approaches, considered scenario is relatively simple with only one curve. It is hard to expect that these approaches generalize well to more complicated scenarios.

Conclusion on Performance autonomous Driving

Automated performance driving of the vehicle on the empty road (single agent) with high road friction coefficient μ can be considered as a solved problem. Although recently several solutions were proposed also for multi-agent driving case, appropriate

solution for real-time control still does not exist. For low μ surfaces, development still lags. Several solutions are available for specific scenarios such as drift parking and trail-braking in a single curve, but driving the full track generally requires solving more curves, with variable curvature radius and a mix of right and left curves. For this scenario decoupling approach is not well suited, as finding out how to split the horizon and assigning the segments arises as a problem, which can be viewed as a combinatorial optimization problem. This confirms that the problem of continuous driving is a different one from sustained drift or transient drift.

3.2 Theoretical foundations

Regarding the approaches for solving these problems, we can make rough distinctions between three orthogonal approaches: *Control*, *Planning* and *Learning*.

Control approaches try to devise a general mechanisms or a rule which can be applied to bring the system from current to desired state. Rules and mechanisms should be as general as possible and provide strong theoretical guarantees on performance such as stability, optimality, etc. They generally assume that the desired behavior is available (i.e. reference).

Planning approaches rely on deliberative reasoning about current state and sequence of future reachable states to solve the problem. In many cases planning approaches reason about what desired state should be.

Learning approaches are focused on systems which improve automatically based on experience or available data, in many cases starting with no a priori knowledge ('tabula rasa'). Practical solutions are commonly a mix of these approaches.

All of these approaches represent dedicated research fields and many textbooks cover these topics. Some of the most notable textbooks focused on methods are [184][185][186] for Control, [105][187] for Planning and [188] for Learning.

In the following sections an unjust overview of several methods will be presented, to help better understanding the related work and placing of contributions of this work.

3.2.1 Control and Planning

Control and Planning are complementary approaches, with complementary strengths and weaknesses. Control approaches itself are generally computationally light as they use simple mechanism, while Planning algorithms generally require intensive computations. Control approaches are generally robust to disturbances as they devise general rules applicable for wide regions of state-space. On the other hand, planning approaches usually devise single plan from a current state and are invalid if plan is not perfectly executed. Additionally, Control approaches deal better with continuous system dynamics (or discretized) while Planning approaches deal better with discrete problems (i.e combinatorial optimization problems). However, devising control mechanisms is possible only for a quite restricted class of problems. They are hard to devise for the more complicated systems such as hybrid systems or systems with multiple in-

puts and outputs (MIMO). On the other hand, planning approaches are not so much influenced by system complexity as they use computational methods for simulating system evolutions. And in fact, many approaches are based on combining the benefits from both of these approaches.

The *Model Predictive Control (MPC)* is an example of effective combination of Control and Planning. It combines the concept of Moving Horizon and Optimal Control (predictive planning). MPC is solving optimal control problem, starting from the current state for certain horizon in future, executing a single step, and then repeating the procedure with new state. While concept of Moving Horizon can be considered a Control concept, which provides the feedback by taking the current state in each step, underlaying solver for finding optimal control is usually optimization or planning solver. However, most of the solvers still rely on numerical optimization methods (i.e. convex programming), and full potential of the synergy with advanced planning approaches is still not capitalized. In fact, many of the planning solutions are used with tracking controller and not in Moving Horizon fashion. The one of the reasons might be that planning methods are generally much slower than convex optimization methods and that they deal with different horizons. Related MPC work generally deals with limited horizon while Planning generally deals with infinite horizon (until the goal is reached). Because of limited horizon, MPC cannot guarantee globally optimal solutions and suffers from terminating set problem (how to value different terminating states), but is generally dealing well with changing environment as replanning is executed often. On the other hand, optimal planning approaches generally guarantee globally optimal solutions for the initial problem, but not in the presence of disturbances. Effective balance between limited horizon and infinite horizon is an open problem, and decaying horizon is one of the approaches, but this is suboptimal as well.

Planning for dynamical systems requires special treatment of *kinodynamic* constraints. Usually, full motion trajectories are constructed by concatenating so called "motion primitives" [189]. Motion primitives represent dynamically feasible trajectories which evolve according to the system dynamics. They can be generated by simulating system evolutions and assuming piecewise constant inputs (by gridding the inputs), starting from the state defined by end state of previous motion primitive. Motion primitives can be also precomputed by gridding the compact subset of state and input sets in some cases, when system exhibits invariance to some operations (i.e. most ground vehicles exhibit invariance to translation in the horizontal plane and to rotation about a vertical axis). In this case, it is important to take care about the feasibility of concatenation of different motion primitives. For unstable systems it might also be useful to use closed-loop prediction approach (i.e. CL-RRT [190]). Closed-loop prediction approaches use motion primitives of closed-loop system for the expansion when open-loop dynamics are unstable, as the exploration by variations in the open-loop dynamics becomes inefficient.

Another example of combining Control and Planning is feedback motion planning [105]. Feedback motion planning approaches provide not a single planing solution but a solution tree which then can be used even when system deviates from optimal trajectory. For example LQR-Tree [191] provides a set of controllers and their operational regions, covering effectively the whole controllable state-space.

Controlling *hybrid discrete-continuous systems* is a challenging problem. This problem is present is different fields such as Control of hybrid automata [192] in Control Theory, Combined Task and Motion Planing Problems [193][194][195] in Robotics, Mixed-Integer Programing [196] in Optimization, Mixed Discrete-Continuous Domains [197][198][199] in AI Planning. Research in all of these fields can benefit from cross-fertilization of the ideas.

3.2.2 Planning and Learning

As mentioned earlier, Planning and Learning also have complementary strengths and weaknesses. It is an issue for learning approaches to provide any guarantees on performance, have safe exploration or to learn long-term rewards. In these aspects, planning algorithms can provide valuable support. On the other hand, planning algorithms are rather slow in high dimensional spaces. This can be improved if planning is properly guided.

Having fast planning algorithms is crucial for practical use of robots in changing environments and safety critical tasks. Efficiency of Heuristic Search-based planning (A* Search [48]) largely depends on the quality of the heuristic function for estimation of the cost-to-go [200]. Ideally, if we knew the exact cost-to-go (oracle), we could find the optimal solution with minimum effort (practically traversing greedy Value function). If the robot operates in similar environments, previous search experience might be useful for the learning of the Value function. Effective synergy of Planning and Learning provided exceptional results so far including seminal achievement of SuperHuman performance in the game of Go [201].

Interaction of Planning and Learning has a long history [43][202][203], with several modern directions, including End-to-end learning approximations of planning algorithms (i.e inspired by Value Iteration algorithm [204], MCTS [205], MPC [206]), planning to guide exploration in Reinforcement Learning [207], [208] and learning to guide planning [209][210][211][212], as well as model learning and Model Based Reinforcement Learning in general.

Planning and Learning work related to the work in this thesis is in direction of learning the value function in order to guide heuristic search-based planning. The most similar approaches to one presented here appeared concurrently in [213] and in [214]. However, in [213] authors used only nodes from the shortest path for learning and in [214] authors used backward Dijkstra's algorithm which explores the whole search space. Our proposed approach has certain advantages when compared to both of these.

3.3 Thesis contributions

The main goal of this work is to develop a general decision making and control solution for autonomous vehicles, applicable to all use-cases, while achieving SuperHuman performance. Because of the diversity of communities interested in this topic, contributions of this work are mainly grouped in three use-cases of autonomous vehicles

and minor contributions related to basic research in fields of Control, Planning and Learning.

3.3.1 Energy-efficient Driving (eco-driving) use-case

Regarding Energy-efficient Driving use-case contributions of this work can be summarized as:

- A novel concept *optimal cruising velocity* (OCV) and analytical formula for calculating it.
- Efficient, custom implementation of Dynamic Programming-based solution for planning long trips (longer than 40 km) in reasonable time (less than 5 seconds).
- Comprehensible and concise cost function based on energy-efficiency including economic aspect such as hourly rate.
- Arguably, the first solution for multilane eco-driving, considering overtaking possibility.
- Arguably, the first solution for optimal motion planning for automated driving in multilane traffic with traffic lights.
- Extensive study with more than 20 human drivers in urban traffic.

Through this work, I developed two solutions for classic eco-driving problem, longitudinal velocity planning. The first solution is custom developed without using any libraries and is based on Dynamic Programming (DP) [5]. It was approximately 100 times faster than state-of-the-art solution based on Dynamic Programming [215].

The other solution is based on A* search [48], utilizing heuristic function based on a novel theoretical concept, *optimal cruising velocity* (OCV). Optimal cruising velocity is the velocity which minimizes energy consumption for driving on a flat, horizontal, empty road. It finds optimal balance between air drag influence and time-proportional cost influence. I derived analytical formula for calculating optimal cruising velocity as:

$$v^* = \sqrt[3]{\frac{P_{\text{aux}} + P_{\text{hr}}}{\rho_{\text{a}} c_{\text{d}} A_{\text{f}}}}. \tag{3.1}$$

As it can be seen from the formula, the optimal cruising velocity is unique for the vehicle (and it's hourly rate of operational costs k_{hr}). It depends on the vehicle aerodynamic shape ($c_{\text{d}}, A_{\text{f}}$), air density ($\rho_{\text{a}}$) and power consumption (auxiliary power P_{aux} and power equivalent of operational costs P_{hr}). It was shown that proposed heuristics based on OCV significantly improves the precision of the energy consumption estimate, which in turn then improves planning efficiency (decreases the number of examined nodes) compared to the state-of-the-art heuristics. Moreover, the proposed solution explores significantly less trajectories compared to DP approach.

Classic eco-driving work generally assumes single lane driving and often empty roads as well. If leading vehicles are neglected in the optimization, the unconstrained planning will not be fully achievable in real driving conditions, and may in some situations lead to drawbacks in energy consumption and very likely to bad driver acceptance. The

work which considers leading vehicles or other traffic constraints such as Traffic Lights still considers single lane driving and therefore longitudinal vehicle control only (i.e. only vehicle following).

This work focuses on the integration of constraints imposed by leading vehicles and other traffic constraints (i.e. traffic lights) in a global approach to optimize energy consumption. In [4] we presented the first solution (based on my knowledge) for multilane eco-driving, considering overtaking possibility in eco-driving. This was further extended to full Level 5 automated driving with energy-efficiency considerations. In [7] we presented the first solution (based on my knowledge) for optimal motion planning for automated driving in multilane traffic with traffic lights.

To benchmark benefits of the proposed solution compared to human drivers, I organized a study in driving simulator with more than 20 participants. The results confirm significant improvements in safety and efficiency compared to human drivers.

3.3.2 Unified motion planning solution for automated driving

As previously presented, many different approaches for motion planning are available, but still, finding a collision-free motion plan, while taking into account system dynamics, dynamic obstacles and possibly desired criteria (cost function) in a real time is an unsolved challenge.

I developed Search-Based Optimal Motion Planning framework (SBOMP) presented in [7] that is scalable to complex driving situation (combinations of multi-lane driving, traffic lights, forbidden lane change, lane termination etc.) and can also enable energy optimal driving (eco-driving), among other criteria. SBOMP framework relies on MPC-like scheme and reusing the cost-to-go map to increase the search efficiency. Cost-to-go map can be computed numerically by applying dynamic programming (DP) to a relaxed problem, [5][4] or model-based (MB) [6].

SBOMP is based on the following features:

- a convenient search space definition, enabling intuitive formulation of a wide variety of semantic constraints and their catalog including: traffic lights, other vehicles, forbidden lane change, speed limits, etc.,
- the possibility to reuse backward planning results from a relaxed problem for shorter planning times,
- integrated reference lane decision making and velocity trajectory planning (longitudinal and lateral motion),
- hybrid time/distance horizon and discretization steps that enable both slow and fast trajectories,
- search in continuous time, distance and lane space provided by hybrid A* search,
- linear lateral motion model for efficient and effective lane-change planning.

In [7] we provided the first demonstration on the complex use-case with multi-lane driving in a presence of traffic lights.

SBOMP framework is well suited for future extensions such as:

- Smooth transition between different planning problems such as structured and unstructured environment.
- Any kind on behavior prediction algorithms as well as interactive driving.
- Integration of Machine Learning approaches.
- Consider arbitrary type of constraint.
- Partial Observability.
- Non-deterministic behaviors.

3.3.3 Performance Driving use-case

The proposed A* search-based motion planning framework was adapted for performance driving use-case and used to generate dynamically feasible trajectories on a slippery surface, which is a novel approach in this field. The proposed method extends drift-like driving from a steady state drifting on single curve to a continuous driving on the road effectively entering and exiting drifting maneuvers and switching between right and left turns. It assumes that the vehicle parameters and the road surface properties are known to a certain degree, which allows it to define a set of steady-state cornering maneuvers. The method is evaluated on a mixed circuit characterized by slippery conditions (gravel), which contains several road sections of varying curvature radii R_c.

As generating references in a continuous driving problem can be considered as a combinatorial optimization problem, heuristic search methods like A* can be effectively used for automated optimal trajectory generation. The space of the possible trajectories is explored by expanding different combinations of motion primitives in a systematic way, guided by a heuristic function. Motion primitives are generated using two different vehicle models. A bicycle model is used for small side-slip angle operations (i.e. entry and exit maneuvers and close-to-straight driving) and a full nonlinear vehicle model for steady state cornering maneuvers. Approach of generating motion primitives for steady state cornering maneuvers is similar to the closed-loop prediction approach (CL-RRT) [190], which uses closed-loop motion primitives for the expansion when open-loop dynamics are unstable, so that the exploration by variations in the open-loop dynamics becomes inefficient. Such automated motion primitives generation enables it to generate arbitrary trajectories, not limited to just single curve as in previous approaches.

3.3.4 Basic research-related contributions

Besides direct contributions to the autonomous vehicle application, research work presented here contributes to several notable directions of fundamental research in the fields Control, Planning and Learning as well. Contributions mainly lay in overlap between these fields.

Planning and Control

In the context of Planning and Control, contributions of this work can be summarized as:

- extending solver methods used in MPC framework from generally used numerical optimization (i.e convex optimization) to motion planning methods.
- enabling the use of results from infinite horizon in limited horizon MPC avoiding problems of terminating set;
- using of optimal control results from simplified problem (i.e. optimal cruising velocity, driving on the empty road) to improve search efficiency in complex problem;
- using of equilibrium state manifold to generate motion primitives by sampling from latent space, extension of closed-loop predictions (ideal closed-loop);
- efficient planning in some hybrid continuous-discrete problems.

The proposed solution is based on motion planning used in Moving Horizon fashion. This is possible as motion planning can provide solution practically in real-time. Moving Horizon fashion with limited horizon enables fast adaptation to the changing environment. Furthermore, solution effectively combines infinite and finite horizon planning. It uses solutions from infinite horizon planning for a simplified problem (in a form of cost-to-go function) to guide search-based forward planning for limited horizon. This avoids the need for a terminating set and reference, as forward planning gets long term benefits of the final state trough cost-to-go function. Use of cost-to-go function also improves computational performances and informs planner more. We also provided theoretical solutions for simplified problem *Optimal Cruising Velocity*, which can be computed from analytical solution.

For performance driving use-case, the presented approach uses closed-loop predictions to generate motion primitives by sampling from latent space - equilibrium state manifold. The presented approach is similar to the closed-loop prediction approach (CL-RRT) [190], which uses closed-loop motion primitives for the expansion when open-loop dynamics are unstable, so exploration by variations in the open-loop dynamics becomes inefficient. This work is further improved as only solution from equilibrium state are used effectively reducing sampling from 6-dimensional space to sampling in the 2-dimensional space.

The proposed solution demonstrates no challenges when dealing with nonlinear, hybrid continuous-discrete systems, combinatorial optimization problems and switching between models used for generation of motion primitives. It solves multiple combinatorial optimization problems behavior and motion planning for urban automated driving and performance automated driving with different modes.

Planning and Learning

To improve computational performance of search-based planning, it is critical to improve precision of heuristic function. We proposed to use Machine Learning (ML) to

approximate Value function and use it as a heuristic function and used it in deterministic planning framework such as SBOMP. The proposed ML-based heuristic function takes into account dynamic obstacles, thus adding to the performance consistency for achieving real-time implementation of the motion planning [8]. The large drawbacks of pure ML-based solution, such as hard-to-guarantee safety, could be improved if used in deterministic frameworks, such as SBOMP [7]. Proposed ML heuristic is bounded by admissible heuristic (acquired by solving relaxed problem using DP), so guarantees on sub-optimality can be provided as well. Heuristic is bounded to be ε-admissible so the solution is always maximum ε times greater than the optimal solution [216]. Values of ε closer to 1 guarantee smaller deviation from optimal solution but reduce computational performance.

The main contribution of this work can be summarized as:

- systematic dataset generation from exact optimal solutions for supervised learning of optimal behavior,
- convenient representation of driving situation as input for machine learning algorithm,
- use of machine learning for a heuristic in a deterministic planning framework with guaranteed maximum deviation from optimal solution,
- use of receding horizon approach instead of greedy policy search on value function.

Further on, that work was significantly improved and generalized to Shortest Path Problems in general [9]. The main contribution is a novel search-inspired approach for efficient and systematic exploration of the models for Value Function (cost-to-go) learning, based on *backward* and *prolonged* heuristic search. While other concurrently introduced approaches in this direction use only states from the solution path (which wastes most of the explored states) or expand all states with Dijkstra's like algorithms (which explores unnecessary states as well), our approach offers balance and ensures that only interesting nodes are explored and all explored nodes are used for value function learning.

Our work is based on a premise: *For learning of the Value function it is more beneficial to explore states in the neighborhood of the optimal path (policy) than elsewhere, as the agent will spend most of the time in the neighborhood of optimal path.*

Having explored neighboring region around optimal path helps to get back to the optimal path if the planner deviates and provides a better coverage for A* algorithm, which always looks for neighboring states. This, in return, improves both the efficiency and robustness of successive planning.

Motivated by this idea, we can prolong the search even after the optimal path was found. Search does not stop when the first path connecting goal and start is found (optimal path), but rather prolongs search such that k-times more states are explored. This ensures that wider region around the shortest path is explored. Additionally, the direction of the search is flipped, such that the search starts from the goal node. In this way, goal state is reachable from all explored nodes and therefore the exact cost-to-go value is available.

Presented approach is not restricted only to problems like this, but can also be used in reinforcement learning frameworks.

4

Problem Definition

A problem well stated is a problem half-solved.

Charles Kettering

So far in introduction chapters, we have seen anoverview of autonomous vehicle technology and related work in planning for autonomous vehicles, as well as concrete contributions of this work. This chapter aims to provide the definition of concrete problems tackled in the thesis. These problems are then further investigated in Modelling and Motion Planning chapters that follow.

This chapter is organized as follows. In section 4.1, the ultimate goal of the research is defined. In section 4.2, the general driving problem is introduced as well as three concrete use-cases covered in this thesis. In sections 4.3 and 4.4, preliminaries about vehicle dynamics and environment modeling are presented. In section 4.5, evaluation of SuperHuman driving performance is discussed. In section 4.6, a concise analysis of the problem complexity is presented. This is followed by a summary of the challenges and specific problems.

4.1 SuperHuman Autonomous Vehicle

The ultimate goal of my research is to develop real-time decision-making and control systems that would enable *autonomous vehicles* to achieve *SuperHuman driving performance*, while operating autonomously in full real world complexity. The vehicle equipped with such systems is referred to as ***SuperHuman Autonomous Vehicle (SHAVe)***. SHAVe would be able to take over repetitive driving tasks from humans when needed, operating with increased safety, efficiency and comfort, while providing humans with the time for more creative tasks and an increased quality of life. Besides performing the regular driving task better than human drivers, SHAVe would be capable to do even some tasks that regular human drivers are not capable of (i.e. performance autonomous driving).

Here, I define **SuperHuman** driving performance as a performance better or equal to the performance achievable by human drivers in terms of: safety, efficiency and comfort. In the section 4.5 evaluation of each of these is defined in more detail. Achieving SuperHuman performance might be possible by intrinsic properties of the digital sys-

tem such as robustness to the disadvantages of human drivers due to fatigue, lack off attention, driving under influence of drugs or alcohol etc. Additionally, autonomous vehicles could have faster response time and make more informed decisions, as they can rely on more sources of information, i.e. use additional sensors that work even in conditions where visibility is low and get information from other vehicles or infrastructure using connectivity. To operate in all conditions a human can operate, with improved performance, SuperHuman autonomous vehicles should make use of all available information, but should however not critically rely on them, as the vehicle will most probably have to co-exist with human drivers in the traffic.

In Artificial Intelligence, Control Theory and related fields, the term **autonomous** is widely used for a long time. In Control Theory, autonomous system is the system that has no input, so dynamics of the system cannot be manipulated [184]. In Artificial Intelligence, autonomy is a property of intelligent agents. The term agent is used to represent an entity existing in an environment, capable of perceiving it's environment through sensors and acting upon it through actuators. If an agent is behaving in a such way that it maximizes some expected performance measure, the agent is called to be a rational agent and can exhibit intelligent behaviors [93, p. 34]. A rational agent should be autonomous, meaning it should be able to perform its intended functions without being directly operated by a person [187]. An agent can be equipped with computational deliberation capabilities (planning) that allow an agent to "reason about its actions, choose them, organize them purposefully, and act deliberately to achieve an objective" [187]. For low diversity tasks and fully specified environments, the agent may not need deliberation (i.e. manufacturing robot), as it can be pre-programmed with a library of reactive behaviors. However, an agent will require some degree of deliberation if it has to operate in diverse environments, tasks and interactions that is definitely the case for Autonomous Vehicle (Level 5 Automated Driving).

In the SAE J3016 report [89], it is recommended to avoid using the term "autonomous vehicle" as the meaning is often inconsistent and can sometime refers to Autonomous Driving Systems with Level 3 that still relies on contribution of human operator. In this work, the term **autonomous vehicle** is used for Automated Driving Systems (ADS) with full driving automation (Level 5). Additionally, an autonomous vehicle transcends Level 5 ADs, as it can achieve some tasks that regular driver cannot (i.e. performance autonomous driving).

Developing SuperHuman autonomous vehicles is a tremendous feat far beyond a single PhD thesis. Therefore, this work is focused on three use-cases, sampled from a width of autonomous vehicle operation. Energy-efficient Driving, i.e. applicable to heavy duty vehicles operating with increased energy-efficiency; Automated Driving in complex, urban environment; Performance Autonomous Driving, exploiting the full vehicle dynamics. Moreover, autonomous vehicles should perform all these tasks achieving a performance that significantly exceeds average human driving performance, even professionals.

Many other use-cases remain out of the scope of this thesis, including long-term autonomy vehicles (delivery robots, exploration rovers etc.), multi-vehicle fleets (cooperative driving, platoons, Mobility-on-demand, etc.), etc.

4.2 Driving task

Driving is a complex task consisting of continuous perception, planning and execution in order to achieve desired goals while avoiding collisions with other participants, obeying traffic rules, complying with vehicle dynamics and factors like comfort, safety and efficiency. To fully automate driving, the vehicle has to be able to autonomously make decisions and plan its motion, while considering all mentioned requirements. The environment is usually highly dynamic, with speeds that may reach $50m/s$ or above (highway driving). Moreover, the environment may be complex, including many different participants, traffic rules, traffic control devices, etc. The mentioned conditions impose many different constraints on the driving. To achieve safety levels higher than a human driver in full AV, errors introduced by technology should be minimized, which is still a challenging goal (210 million km driven between two fatal accidents) [21].

For developing a functional system architecture for automated driving, Ulbrich et al. [92] provides a very insightful overview of research about human driving behavior. They mention that [217] distinguishes three levels of performance of skilled human operators for performing general tasks. These behaviors are: "skill-based behavior", "rule-based behavior", and "knowledge-based behavior". Skill-based behaviors are the lowest level, sub-conscious, sensory-motor activities. Rule-based behaviors are, as the name indicates, based on rules defined for a certain situation. If the operator does not have defined rule for a certain situation (i.e. did not have a similar experience), existing knowledge can be used for reasoning about how to achieve the goal, that is knowledge-based behavior. Furthermore, they mention that Donges [218] distinguishes three hierarchical levels of driving tasks. These are: "navigation", "guidance", and "stabilization". Hale et al. [219] combines these two classifications and suggests that the Donges' three levels of driving tasks and the three levels of Rasmussen are orthogonal to each other. Depending on the setup, different levels of human behavior might be used for doing the same task as shown in the Table 4.1.

Table 4.1: *Examples of driving tasks with related processing levels based on Hale. Ilustration is based on Ulbrich's et al. [92] translation of [220].*

		Processing level		
		Skill-based	Rule-based	Knowledge-based
Driving task	Navigation	Daily commute	Choice between familiar routes	Navigating in foreign town
	Guidance	Negotiating familiar junctions	Passing other car	Controlling a skid on icy roads
	Stabilization	Road following around corners	Driving an unfamiliar car	Learner on first lesson

Based on Donges' hierarchical levels of driving tasks we can define, a rather usual, architecture of vehicle control, consisting of "mission (route) planning", "behavior and

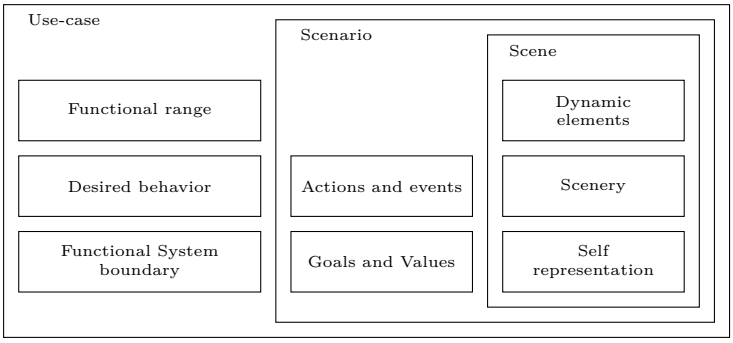

Figure 4.1: *Scene, Scenario and Use-Case definitions based on [221].*

motion planning" and "stabilizing control". Focus of this research is on behavior and motion planning.

Formally, Society of Automotive Engineers (SAE) defines Dynamic Driving Task (DDT) in SAE J3016 [89] as:

"Dynamic driving tasks comprise all the real-time operational and tactical functions required to operate a vehicle in on-road traffic, excluding the strategic functions such as trip scheduling and selecting destinations and waypoints, and including without limitation:

- Lateral vehicle motion control via steering (operational)
- Longitudinal vehicle motion control via acceleration and deceleration (operational)
- Monitoring the driving environment via object and event detection, recognition, classification, and response preparation (operational and tactical)
- Object and event response execution (operational and tactical)
- Maneuver planning (tactical)
- Enhancing conspicuity via lighting, signaling and gesturing, etc. (tactical)"

4.2.1 Tackled use-cases

Across the literature many terms related to automated driving are not clearly defined (i.e. scene, scenario, use-case). Ulbrich et al. [221] provides extensive definitions for describing autonomous vehicle operations. These definitions are adopted in this work. Ulbrich et al. [221] defines scene as:

"A scene describes a snapshot of the environment including the scenery and dynamic elements, as well as all actors' and observers' self-representations, and the relationships among those entities. Only a scene representation in a simulated world can be all-encompassing (objective scene, ground truth). In the real world it is incomplete, incorrect, uncertain, and from one or several observers' points of view (subjective scene)." Further scenario builds upon scenes and includes as well "actions and events" and "goals and values". Ulbrich et al. [221] defined it as:

"A scenario describes the temporal development between several scenes in a sequence of

scenes. Every scenario starts with an initial scene. Actions & events as well as goals & values may be specified to characterize this temporal development in a scenario. Other than a scene, a scenario spans a certain amount of time." Temporal development of scenario is generally conditioned by actions of ego-vehicle as well. Finally, the use-case builds upon scenarios and includes additionally "functional range", "desired behavior" and "functional system boundary". Ulbrich et al. [221] defined it as:

"A use-case entails a description of the functional range and the desired behavior, the specification of system boundaries, and the definition of one or several usage scenarios. While these scenario descriptions might be rough and incomplete in the first phase, they may be detailed to achieve fully testable test-cases in the development process."

These relations are depicted in Figure 4.1.

As mentioned in introductory chapters, there are three major use-cases treated in this work: Energy-efficient Driving, Automated Driving and Performance Autonomous Driving. For each of the use-cases all aspects will be defined.

4.2.2 Problem 1: Energy-efficient Driving (eco-driving)

The goal of this use-case is to execute transportation task from point A to point B using the least energy, while complying with constraints (i.e. speed limits), based on available map data. It is assumed that the vehicle drives on a single-lane road without other traffic participants. The route is already available trough navigation system. The vehicle can be represented by a mass-point longitudinal dynamic model considering resistive forces like air-drag, roll and road slope. A typical scenario of automated driving use-case is driving over hilly, empty road while considering long-term benefits of velocity adaptations to minimize energy consumption. An extension to multi-lane driving in traffic is tackled in Automated Driving use-case.

4.2.3 Problem 2: Automated Driving

The other concern is to fully free human drivers and improve traffic safety, efficiency and comfort by automating driving task. In this use-case we assume that vehicle can operate in full velocity range, fully independently, and do all dynamic driving tasks (DDT) without any input from a human (Level 5 AD). The goal is to achieve acceptable behavior while driving among other traffic in everyday driving situations and satisfying traffic rules (i.e. driving on multilane roads, highway, intersection, etc). While executing driving task, vehicle should aim to minimize energy consumption. It is assumed that the vehicle is equipped with a robust perception system providing information about the environment (i.e. position and velocity of all vehicles around). Also, it is assumed that this kind of driving does not require extreme maneuvers that would push vehicle dynamics to the limits. Additionally, it is assumed that motion of the other vehicles can be predicted and all eventual deviations (including deviations due to ego-vehicle actions) can be compensated by frequent replanning. An typical scenario of automated driving use-case is multi-lane urban driving in the presence of traffic lights. In this scenario, vehicle is driving on multi-lane road, multiple vehicles

are driving around ego vehicle and traffic lights are present. Autonomous Vehicle drives in this situation while minimizing energy consumption as it can use information about traffic light timings.

4.2.4 Problem 3: Performance autonomous driving

The tackled problem in this use-case is to achieve a minimum lap time driving on an empty track in low friction conditions, i.e. gravel road. It is assumed that the vehicle is equipped with a map of the road and a localization system. Therefore, the vehicle has the information about the road ahead, as well as left/right boundaries and exact position and orientation. Moreover, the full vehicle state feedback information is available. In particular, besides dynamic states, the low-level controller (that is not covered in this work) for state tracking requires measurements and estimates of several quantities, including wheel forces and wheel slips, both longitudinal and lateral. Finally, the combined longitudinal/lateral tire-road contact forces characteristics are assumed to be known and constant. The road, on the other hand, is assumed to be empty, flat, with static road-tire characteristic and can have arbitrary shape with constant width. In this use-case, the typical scenario could be driving in sharp curves (i.e. with radius $15m$) and entering high side slip angle states (drifting).

4.3 Vehicle dynamics

Vehicle dynamics can be challenging to model, especially in highly dynamical conditions such as performance driving. Although some aspects of vehicle dynamics can be analytically expressed, some can not (i.e. lookup tables for energy-efficiency). However, even analytical aspects represent a challenge for some methods as they are nonlinear (i.e. air drag resistance is proportional to the square of the velocity, road slope can have arbitrary shape, etc.). Including the motor efficiency introduces discontinuity, as there is a sign function in motor torque calculation. Vehicle kinematics introduces non-holonomic constraints that can also influence feasibility of some trajectories, requiring so-called **kinodynamic** motion planning. Additionally, adequate modeling of tire-road interaction forces is also challenging. Upon everything, the system is controlled by only two inputs (steering and acceleration/braking), making it **underactuated** system which is additional challenge.

4.4 Environment

The vehicle does not operate in isolation. Rather, it is operating in an environment which is complex and dynamic, with many other traffic participants. The vehicle has to avoid collisions with surrounding entities, therefore, the environment imposes dynamic constraints on the motion of ego vehicle (that are not known in advance). The incorporation of the dynamic constraints increases the problem complexity and causes some method that do not build trajectory from start to fail. Further, to do predictive planning, future trajectories of other entities have to be predicted. In some

cases, future trajectories of other traffic participants are affected by the actions of ego-vehicle, making it also a **multi-agent problem**.

Beside pure collision avoidance, the vehicle should posses a certain level of **situation-awareness** and behave appropriately for a given situation, and tend to comply with **traffic rules** at all times. Driving behaviors acceptable in one situation might not be acceptable in another. The problem is that driving situations vary a lot, so a general solution is hard to formulate. As shown in Figure 4.1 and Section 4.2.1, scenarios are mainly represented by at least 5 factors, each of them having multiple variants, causing combinatorial explosion. Also, representing traffic rules as such that they can be processed by an algorithm to validate motion plan might be challenging, and integrating it with other tasks of planning (i.e. avoiding collision) might be extremely challenging.

4.4.1 External constraints

External constraints can be classified according to the dependence on two variables relevant for the optimization problem. These are space and time. This means that external constraints can be grouped into four different groups summarized in Table 4.2.

Table 4.2: *External constraints classification.*

	Time variant	Time invariant
Space variant	other traffic participants	resting time (e.g. every 2h)
Space invariant	traffic lights	traffic signs (e.g. speed limits), road curvature, lane markings, etc.

Generally speaking, invariant constraints are easier to integrate into the optimization problem than varying constraints. In some cases integration of time/space variant constraints can lead to tremendous efforts.

Time-and-space-invariant constraints are straight forward to integrate as they do not change neither in time nor in space. Examples are traffic signs that limit maximum speed on some road segments or road curvature that limits maximum speed due to the risk of loosing the track due to under-steering. On curved segments, a minimum longitudinal acceleration limit can also be imposed. The resulting acceleration would generate an inertial force pushing back the driver and improving the drivability feeling since it may partly compensate the uncomfortable centrifugal forces. Constraints of this type are also lane markings. An example of a *time-invariant-space-variant constraint* could be resting time. Usually the driver has to make resting stops after continuously driving for longer periods. This generally does not explicitly depend on space as there are more resting spots along the road. With some approximation, charging (or fueling) of the vehicle could be considered as a constraint of this type.

Time-variant-space-invariant constraints do not change in space but change with time. A typical example of this is traffic light, which sets the maximum speed limit to zero when the red light is active. This happens in that time intervals at a fixed location in space.

Time-and-space-variant constraints are usually hardest to consider, sometimes even impossible with reasonable effort. Typical examples of a constraint of this type are other vehicles moving in the surrounding traffic. Constraints are on the speed and position of the controlled vehicle. Time and space of such constraints are not fixed, since the velocity of the controlled vehicle itself influences the constraint. For example, if the controlled vehicle is moving faster, it will reach a leading vehicle sooner in time and space. Things get even more complicated when considering the possibility of overtaking. In this case, the speed of the controlled vehicle has to be significantly higher than the speed of the leading vehicle, providing a speed difference to safely overtake.

4.4.2 Observability

Generally, AV does not have the full information about the environment and self-state. There are two major ways how AV can get information about the environment, by communicating with environment or on-board sensors.

Vehicle to Infrastructure communication (V2X)

Vehicle communication is a wide topic. In this work, it is of interest to utilize communication with infrastructure and to utilize Traffic Light Signal Phase and Timing (SPaT) to better predict future states of traffic lights. Systems like this are already available is some vehicles, i.e. Audi announced the first vehicle-to-infrastructure (V2I) service, where state of the traffic light and duration until the next change is available to the driver in some cities [222].

Besides traffic light SPaT signal, it is useful to a acquire map of the road with elevation profile of the road ahead for predictive velocity planning with aim to reduce energy consumption. Imminent road structure (i.e. lane reduction) can be provided by the map as well.

Environment perception

Reliable prerception of the environment is a challenging task as it was addressed in section 2.3.2. In this work, it is assumed that AV is equipped with appropriate on-board perception system capable of providing adequate information about the environment, mainly position and velocity of all surrounding vehicles. Even though, real-time environment perception systems with high degree of reliability are still being developed, it is reasonable to assume that they will be available soon.

Future behavior of other agents

Even if the perfect information about other vehicles operating in the environment was available to the AV, the future movement is generally unknown, and some assumptions must be made. Furthermore, future trajectories of other vehicles can also be affected by actions of ego vehicle. Behavior prediction is a complex topic. In this work, it is assumed that a valid behavior prediction of other vehicles is available and discrepancies can be compensated with frequent replanning.

Vehicle state information

As it was mentioned earlier, vehicle dynamics is complex to express. Additionally, not all vehicle states can be directly measured (i.e. side slip angle). However, it is reasonable to assume that appropriate state estimators, that complete missing measurements and provide information about all vehicle states, are available.

4.5 Performance Measure

SuperHuman driving performance is manifested in improved safety, efficiency and comfort, compared to human drivers. Safety improvements are manifested in reduced number of collisions with vulnerable road users and other traffic participants, while satisfying traffic rules (e.g. speed limits, traffic lights, lane availability). Efficiency improvements are manifested in energy consumption reduction. Finally, improved comfort should manifest in better human acceptance. A cost function can be used to measure quality of a given solution trajectory. It should be possible to evaluate multiple goals, such as short travel time, comfort, safety, energy-efficiency, or some combination. Generally, it is a multi-objective optimization problem. The design of the cost function is particularly important as it influences vehicle behavior and defines the optimal solution. Human preferred driving behavior might be hard to express using cost function.

4.6 Problem complexity

Driving in traffic is generally a combinatorial optimization problem. For example, if there are two lanes and ego vehicle is in the right lane, with a slow vehicle in front and other vehicle approaching in other direction, there are two possibilities to overtake the slow vehicle. The first option is to slow down and wait until vehicle in the left lane passes and then overtake the slow vehicle. The second option is to speed up and overtake slow vehicle before the vehicle on the left lane approaches. In more complex environments, number of variants grows exponentially. Also, for performance driving problem, driving along the full track generally requires driving trough multiple curves, with variable curvature radius and a mix of right and left curves. When different modes are possible (i.e. drifting, straight exiting/entering, etc.), deciding the mode for a segment affects subsequent segments, as transitions between modes are not trivial.

Assigning the modes to the segments arises as a problem, which can be viewed as a combinatorial optimization problem. Due to the combinatorial nature, many methods cannot be used to solve this problem (i.e. convex optimization).

If the optimization is solved numerically, dynamical system state discretization is necessary. Increasing the order of dynamical system (problem dimensions) considered in optimization problem, the number of possible discrete state combinations increases exponentially. For example, if each dynamical state has 100 distinct values, a 2 dimensional system has 10.000 possible state combinations and introducing the third dimension it becomes 1.000.000. This requires also the number of samples to raise exponentially to avoid sparse sampling. Bellman called this problem the "**curse of dimensionality**" [43]. This problem becomes even more challenging when considering multi-stage optimization problems, i.e. optimal control, where the state trajectory evolves over the time. For example, if a branching factor, representing the number of possible state transitions at each stage, is 100, on the second stage it is already 10.000 different trajectories and on the 12th stage it will reach the 10^{24}, that is an order of magnitude as the number of stars in the Universe. In practice, it is very challenging to apply Dynamic Programming for problems with more than 4 dimensions, and some of these problems (i.e performance driving) require at least 6 dimensions to describe all the states. "Curse of dimensionality" directly affects computational complexity. As the focus is on the real-time applications within a vehicle, the general trade-off between computation time and precision (sampling factor) has to be taken into account, which will have an impact on the choice of the methods. In general, long planning horizons are necessary in order to achieve long-term benefits, such as energy-efficiency.

4.7 Challenges

Based on all mentioned aspects of this problem, major challenges can be summarized as follows.

- Human-level safety driving performance is high.
- Vehicle dynamics is complicated.
- Environment is complex.
- Performance measure that express human preferred driving is hard to formulate.
- Real-time performance is challenging as the problem is Combinatorial Optimization Problem with high dimensionality and long horizons.

4.8 Problem summary

In this section, the goal is to summarize the general problem and three use-cases by defining clear assumptions and requirements based on previous discussion. Firstly, general assumptions and requirements are stated, then additional assumptions and requirements for each of the use-cases are defined, and in the end, some requirements that should support future extensions are stated.

4.8.1 General problem

Assumptions and requirements common for all use-cases are defined as follows.

Assumptions:

A0.1 The AV is equipped with the system that provides estimate of the full vehicle state.

A0.2 The AV is equipped with the system that provides information about the road ahead including elevation, speed limits and road/lane boundaries.

A0.3 The AV is equipped with the system that provides appropriate route for the whole trip.

A0.4 The AV is equipped with the low-level controller that assures the plan is executed.

A0.5 The AV is equipped with the appropriate vehicle model, including vehicle consumption model.

Requirements:

R0.1 The AV planning system shall provide dynamically feasible trajectories.

R0.2 The AV planning system shall have close-to real-time performance.

4.8.2 Problem 1: Energy-efficient Driving

Assumptions and requirements for Energy-efficient Driving use-case are defined as follows.

Assumptions:

A1.1 The AV drives on an empty road with a single lane.

Requirements:

R1.1 The AV planning system shall provide a solution that minimizes energy consumption of the vehicle.

R1.2 The AV planning system shall provide a solution for the full trip.

R1.3 The AV shall obey existing traffic rules (i.e. speed limits).

4.8.3 Problem 2: Automated Driving (Level 5)

Assumptions and requirements for Automated Driving use-case are defined as follows.

Assumptions:

A2.1 The AV is equipped with perception system that provides positions and velocity of surrounding vehicles.

A2.2 The AV is equipped with prediction system that provides prediction about behavior of other participants including their estimated state trajectory.

A2.3 The AV can drive in all available lanes.

A2.4 The AV does not operate in extreme maneuvers, so some vehicle dynamics simplifications are possible.

A2.5 The AV drives in the traffic (other vehicles are present).

A2.6 The AV drives in the presence of traffic lights.

Requirements:

R2.1 The AV shall be capable to drive in all scenarios on structured roads (urban, suburban and highway).

R2.2 The AV shall obey existing traffic rules.

R2.3 The AV shall drive along required route using the least energy.

R2.4 The AV shall be capable to drive in full velocity range, including the full stop.

R2.5 The AV shall use all available information from communication, however do not critically rely on them.

4.8.4 Problem 3: Performance Autonomous Driving

Assumptions and requirements for Performance Autonomous Driving use-case are defined as follows.

Assumptions:

A3.1 The AV drives on the slippery road (i.e. gravel road) with known, constant road-tire characteristic.

A3.2 The AV drives on the empty track.

Requirements:

R3.1 The AV should drive safely on the road while aiming for the minimum lap time.

R3.2 The AV should be capable to drive in arbitrary planar road geometry (i.e. varying curvature radius, mixed right and left curves) without need for adjustments.

4.8.5 Future extensions

There are several requirement that should enable future extension of the planning system:

R4.1 The AV planning system shall be expendable to consider use-cases beside structured road (i.e. parking).

R4.2 The AV planning system shall be expendable to consider different behavior prediction algorithms.

R4.3 The AV planning system shall be expendable to consider non-deterministic effects.

R4.4 The AV planning system shall be expendable to consider limitations of perception system (i.e. blind spots).

R4.5 The AV planning system shall be expendable to consider interaction with other participants (i.e. multi-agent problem).

5

Modelling

All models are wrong, but some are useful.

George E. P. Box

Since Isaac Newton's invention of differential and integral calculus, his discovery of laws of motion and explanation of planetary motion, it is clear that in many cases motion of physical systems can be well described using analytical equations. Selecting a model that represents physical system realistically enough yet simple enough is sometimes challenging. Simplistic models enable better computational performances and wider range of methods that can be applied. However, they might not represent the real system precisely enough and thus cause poor operational performance when used in predictive control setup. They can even provide solution trajectories infeasible for the real system.

For autonomous vehicles, there are several aspects which should be modelled properly. These include vehicle dynamics, environment (including road, traffic rules, other traffic participants, etc.) and driving objective. These can serve to model different scenarios. Each of these aspects is covered in more details in the following sections.

5.1 Vehicle model

Depending on the specific use-case problem, different levels of details are needed to properly model the vehicle and use models for a predictive control. For example, Energy-efficient Driving requires a vehicle consumption model, but the vehicle dynamics model can be much simpler than in the case of Performance Driving. On the other hand, Performance Driving requires detailed tire-road friction model and simple model will not suffice. For Automated Driving and Performance Driving use-case, Motion Planning algorithm constructs vehicle trajectories by concatenating smaller segments of trajectories, the so-called "motion primitives", that are generated based on the vehicle model. Therefore, the feasibility of these trajectory depends on the quality of vehicle model approximations.

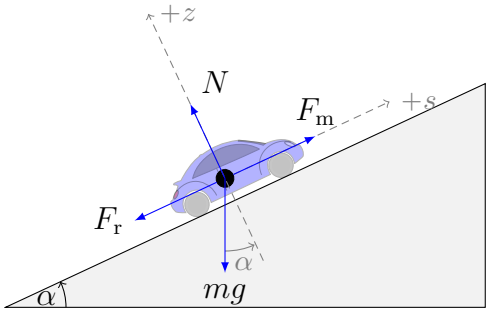

Figure 5.1: *Vehicle on a slope, the force balance.*

5.1.1 Longitudinal motion model

Assuming that the vehicle orientation does not deviate much from the road direction and the road is close to straight, vehicle motion can be modelled using longitudinal motion model [223]. This is generally the case for energy efficient driving scenarios. Several resistive forces can be modelled including air drag resistance, roll resistance and gravity component. Figure 5.1 illustrates the force balance for vehicle on a slope. The resulting force is the force accelerating the vehicle. Gravity has two projections, one on s-axis along the road representing the direct resistive force and other on z-axis contributing to roll resistance. The longitudinal motion is given by:

$$v(t) = \frac{\partial s(t)}{\partial t}, \tag{5.1}$$

$$a(t) = \frac{\partial v(t)}{\partial t} = \frac{F_\mathrm{m}(t) - F_\mathrm{r}(t)}{m}, \tag{5.2}$$

$$F_\mathrm{r}(t) = \underbrace{\frac{1}{2}\rho_\mathrm{a} c_\mathrm{d} A_\mathrm{f} v(t)^2}_{\text{air-drag}} + \underbrace{c_\mathrm{r} mg\cos\left(\alpha\left(s(t)\right)\right)}_{\text{roll resistance}} + \underbrace{mg\sin\left(\alpha\left(s(t)\right)\right)}_{\text{gravity resistance}}, \tag{5.3}$$

where $v(t)$ and $a(t)$ are velocity and acceleration along s, and $F_\mathrm{m}(t)$ and $F_\mathrm{r}(t)$ are forces generated by the motor and total resistive force respectively. Other parameters represent vehicle aerodynamic shape (c_d - drag coefficient, A_f - front surface area), air density (ρ_a) and roll coefficient (c_r).

5.1.2 Planar motion model

Modelling planar motion of the vehicle is very challenging. As mentioned in Chapter 4 (Problem Definition), there are multiple aspects to be considered, including longitudinal, lateral and yaw dynamics, tire-road forces, load transfer, etc.

For vehicle motions that do not stress vehicle dynamics, several simplifications can be used to effectively reduce the problem complexity. For example, for small side slip angle motions, linearized model can be used. Moreover, for driving in structured environments with defined lanes, lane change model can be used to avoid full kinematic motion in a plane.

For vehicle motions that push vehicle dynamics to the limits, more detailed vehicle models are required. Deep insight in vehicle dynamics modelling is provided in [180],

that is used in Performance Driving use-case. In this work, only Equilibrium State Manifold concept is presented, as it is required to understand further generation of motion primitives in the Chapter 6 (Motion Planning).

Kinematic vehicle model

A kinematic model defines the state evolution based on pure geometry, without considering forces and evolution of higher order dynamics, which are treated as inputs. These states are considered in the full dynamics model. Such a model is comprised of six states $[x, y, \psi, v, \beta, \dot{\psi}]^T$, where x, y and ψ represent kinematic states (position and yaw angle), and v, β are vehicle velocity and side-slip angle respectively. The evolution of x and y is given by:

$$\dot{x} = v \cdot \cos(\psi + \beta), \tag{5.4a}$$
$$\dot{y} = v \cdot \sin(\psi + \beta), \tag{5.4b}$$

where v, β, $\dot{\psi}$ are given by the selected vehicle model.

In regular driving situations, i.e. for small values of β, the trajectory identified by (5.4) is mostly determined by v and $\dot{\psi}$, and therefore can be planned by means of linearized vehicle models, valid for small variations of β and $\dot{\psi}$. When driving on slippery surfaces, such solution is not suitable anymore, due to the effect of β in (5.4) and to the complexity of the model which describes the vehicle motion for larger values of β. A major limitation which stems from using a full nonlinear vehicle model is the so-called "curse of dimensionality", and related computational explosion when introducing new states. In fact, if motion primitives were generated with the full nonlinear vehicle model, the computational burden would increase excessively, thus making it a non-viable option for an online implementation.

To overcome unnecessary increase in computational burden, we generate motion primitives by using following simplifications:

- for Automated Driving use-case:
  - we neglect side slip and use vehicle longitudinal motion model and **lane change** approximation.
- for Performance Driving use-case we use:
  - the so-called "bicycle model" [224] for straight-driving/mild-turning maneuvers, and
  - a convenient approximation of the full nonlinear model, based on the theoretical vehicle **equilibrium-states** for steady-state cornering maneuvers.

Lane change model

Since urban driving includes lane changes as well, modelling the lateral motion is of particular importance. This is not straightforward, because of the vehicle kinematics and dynamics. For planning purposes, it is important that the model is conservative,

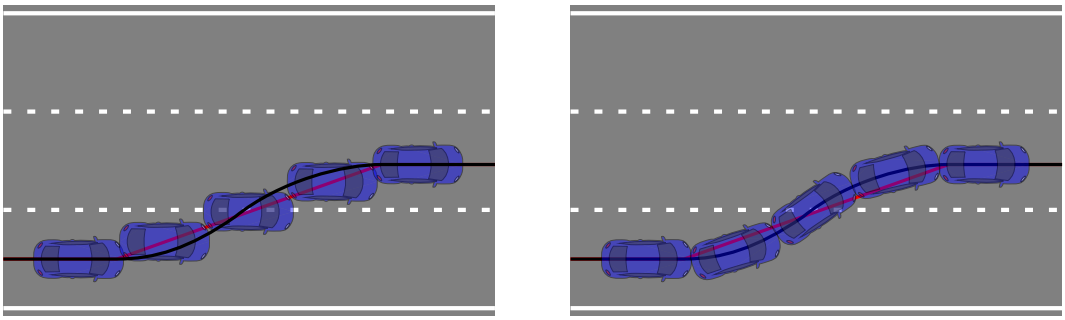

Figure 5.2: *Lateral motion model.*

so that the resulting trajectory is feasible for the lower level controller, yet not too conservative to prohibit many, otherwise feasible, solutions. For planning in structured environment, such as highway driving or urban driving on designated lanes, lateral motion can be simplified to three modes: driving in a lane and lane changing (right and left) as introduced in [7]. To provide feasibility to lower level controller, it is modelled with maximum time T_{LC} needed for the vehicle to execute the lane change. This is consistent with [225], where the authors stated that most of the lane-changes are executed in $3-8$ seconds. This simplification works well for larger velocities (approx. greater than $2m/s$). For smaller velocities, time for lane change is generally larger, but this can be considered by the planner dealing with unstructured environment (i.e. parking maneuver). Alternatively, the clearance for the lane change on smaller velocities can be provided with a greater safety buffer around the obstacles.

Equation (5.5) describes the lateral motion for these three modes. Variable l represents the lane in which vehicle is driving as it is describe in more details in section 5.2.1. As it can be seen in Figure 5.2, we can effectively reduce one degree of freedom (yaw angle ψ), and still provide kinematic feasibility of the solution trajectories (as the rate of lateral change is conservative). Additionally, this approximation is not too conservative as it is still possible to provide solution in very tight maneuvers.

$$\frac{\partial l(t)}{\partial t} = \begin{cases} -\frac{1}{T_{\mathrm{LC}}}, & \text{lane change right,} \\ 0, & \text{stay in lane,} \\ \frac{1}{T_{\mathrm{LC}}}, & \text{lane change left.} \end{cases} \tag{5.5}$$

Equilibrium States Manifold

For cornering maneuvers in performance driving, the motion primitives are generated based on previously computed Equilibrium State Manifold. When generating target reference states, one of the main issues is ensuring that the reference values can be actually reached in a sufficiently short time. This problem is made harder by the slow dynamics of the vehicle longitudinal and lateral accelerations, due to the slippery surface considered. Two actions are taken to counter this problem:

- "slow varying" reference set-points are used, in order to allow the low level actuator to bring the actual state in proximity of the desired one,

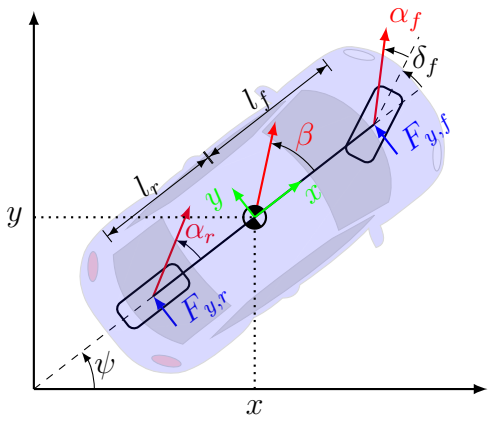

Figure 5.3: *Bicycle model.*

- only reference states belonging to a specific "Equilibrium States Manifold" (ESM) are considered.

To obtain such manifold, that requires a reliable model for vehicle and tire-road contact forces, an off-line computation is performed, that provides the steady-state solutions of the vehicle cornering at different curvature radii [226]. These solutions include the vehicle control inputs (steering wheel angle and rear wheels slip) as well as the vehicle states v, β, $\dot{\psi}$. For the offline computation of the equilibrium points, the full-vehicle nonlinear model is exploited, where the tire-road forces F_x, F_y for each axle ($F_{x,f} = 0$ due to the Rear Wheel Drive (RWD) configuration) are obtained from the normal forces F_z and the combined longitudinal/lateral friction model (see [227]):

$$F_{x,i} = F_{z,i}\mu_x(\lambda_i, \alpha_i), \quad F_{y,i} = F_{z,i}\mu_y(\lambda_i, \alpha_i), \tag{5.6}$$

where μ is the friction coefficient, λ and α the longitudinal and lateral slips respectively.

The nonlinear friction functions take the form of the *Magic Formula* (MF) tire friction model [227], with an isotropic friction model being used for simplicity. This requires the computation of the theoretical slip quantities (σ_j, $j \in \{x, y\}$), which can be obtained from α, λ as follows:

$$\sigma_x = \frac{\lambda}{1+\lambda}, \quad \sigma_y = \frac{\tan\alpha}{1+\lambda}, \quad \sigma = \sqrt{\sigma_x^2 + \sigma_y^2}. \tag{5.7}$$

Then, the one-directional friction coefficients are given by:

$$\mu_i = \frac{\sigma_i}{\sigma} D \sin\Big(C_\lambda \arctan\big(\sigma B - E(\sigma B - \arctan \sigma B)\big)\Big), \tag{5.8}$$

for $i \in \{x, y\}$, with $B = 1.5289$, $C = 1.0901$, $D = 0.6$, $E = -0.95084$ being the Pacejka parameters corresponding to gravel.

The vehicle responses can be obtained from the following system of nonlinear equations, which considers the lateral, longitudinal and rotating balance equilibrium equa-

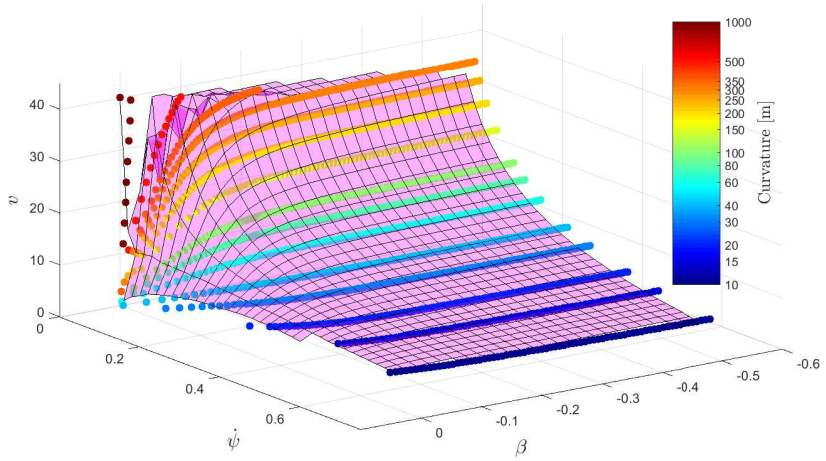

Figure 5.4: *Equilibrium points sets S_{ss} (and linear interpolation) in the $v, \beta, \dot{\psi}$ space for counter-clockwise cornering maneuvers with different curvature radii R_c.*

tions around the vehicle center of gravity:

$$\dot{v} = \dot{\psi} v \beta + \frac{F_{x,r}}{m}, \tag{5.9a}$$

$$mv(\dot{\beta} + \dot{\psi}) = F_{y,f} + F_{y,r}, \tag{5.9b}$$

$$J_z \ddot{\psi} = l_f F_{y,f} - l_r F_{y,r}, \tag{5.9c}$$

where J_z is the vehicle inertia around the z-axis, m the vehicle mass and l_f, l_r are the distances of the vehicle COG from the front and rear axles respectively. Moreover, the longitudinal weight transfer is also considered. Assuming a rear-wheel drive (RWD) drivetrain configuration, and given different sets of values of the constant control inputs (steering wheel angle δ, driving wheels slip λ), the equilibrium points $S_{ss} = [v_{ss}, \beta_{ss}, \dot{\psi}_{ss}]^T$ can be computed, in order to achieve different constant curvature radii R_c, by considering the uniform circular-motion relation $\dot{\psi} = \frac{v}{R_c}$, and imposing in (5.9) the steady-state condition:

$$\dot{v} = \dot{\beta} = \ddot{\psi} = 0. \tag{5.10}$$

In Figure 5.4, these sets are graphically displayed in the 3-dimensional latent state-space, for different values of R_c.

A race track is composed of different sections, with varying curvature radii, as well as straight segments. Therefore, in order to determine the sequence of vehicle states to be tracked by the vehicle, a continuous transition between different equilibrium points would be required (by deviating from condition (5.10)). For this reason, the sets $S_{ss}(R_{c_i})$, $i = 1, .., r$ in Figure 5.4 have been interpolated into a map $v = f(\beta, \dot{\psi})$, that represents the ESM.

Let us assume that the tire-road contact model and the vehicle model are correct, and that a trajectory of any curvature radius R_c is given, for which at least one reference state $S_{ss}(R_c)$ exists. Then, if a locally stable feedback controller for the tracking of the state is designed, such trajectory can be tracked, given an initial condition close enough to the target state.

Semi-linearized bicycle model approximation

When driving conditions are close enough to the origin of the $\beta - \dot{\psi}$ plane, i.e. close-to-straight driving, a nonlinear bicycle model can be simplified. Forces in (5.9) can be replaced with their linearized approximations as:

$$F_{y,f} = -C_f\Big(\beta + l_f\frac{\dot{\psi}}{v} - \delta\Big), \quad F_{y,r} = -C_r\Big(\beta - l_r\frac{\dot{\psi}}{v}\Big), \quad F_{x,r} = -C_x\lambda. \tag{5.11}$$

In (5.11), the longitudinal and lateral stiffness coefficients C_x, C_f, C_r are consistent with the full characteristics given by (5.8). This model is valid for range defined by $|\beta| < \beta_{lin}$ and $|\dot{\psi}| < \dot{\psi}_{lin}$.

5.1.3 Powertrain model

Internal constraints

Internal constraints in the optimization problem originate from constrained system dynamics, constraints on states, initial and final conditions. As internal system constraints, vehicle speed and acceleration limits are:

$$v_{\min} \leqslant v \leqslant v_{\max}, \tag{5.12}$$
$$a_{\min} \leqslant a \leqslant a_{\max}. \tag{5.13}$$

The complete behavior of the electric motor such as maximum available torque, rotational speed and efficiency is modelled within the efficiency map.

Energy consumption model

The propulsion element, an electric motor, with inner torque T_m can be modelled statically by:

$$F_\mathrm{m}(t) = \frac{kT_\mathrm{m}(t)\eta^{\mathrm{sgn}(T_\mathrm{m}(t))}}{r_\mathrm{w}}, \qquad k = \frac{2r_\mathrm{w}\pi\omega_\mathrm{m}(t)}{v(t)}, \tag{5.14}$$

where η is an efficiency coefficient scheduled by a map, k is a combined transmission ratio of the powertrain, and r_w is the radius of the wheels. The ratio between rotational speed ω_m of the motor and vehicle speed v is defined by k.

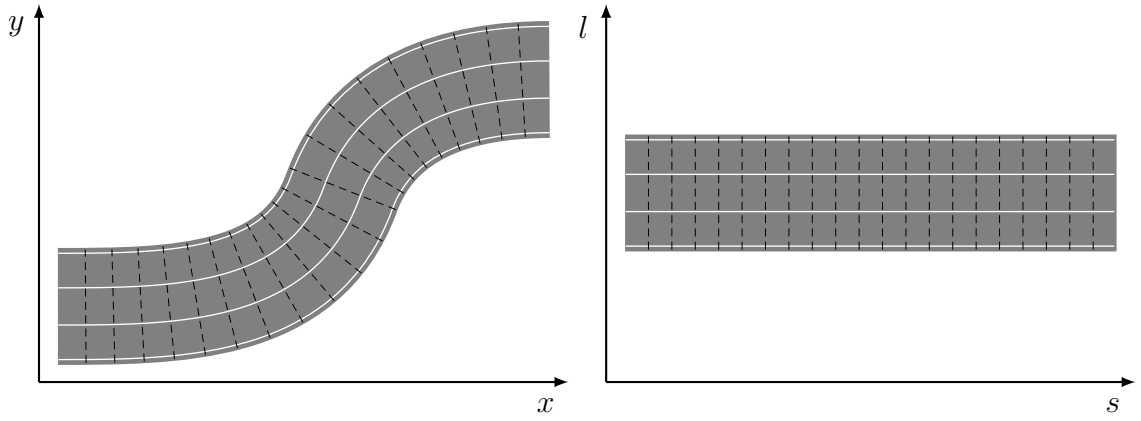

Figure 5.5: *Curvilinear road representation (Cartesian and Frenet-frame).*

5.2 Environment

Autonomous vehicles operate in complex environments. In this section, different aspects of vehicle environment are modelled in detail, so they can be used for motion planning.

5.2.1 Road modelling

As road geometries vary a lot, they can introduce unnecessary complications for motion planing to generate trajectory that keeps the vehicle on the street. To simplify planning, the drivable road is modelled using a Frenet frame [228]. Instead of using x and y coordinates, in Frenet frame, one dimension represents the distance traveled along the road s, and other represents orthogonal motion. Two different variants are used for orthogonal motion: the deviation from the road centerline d in Performance Driving use-case and lane l in Automated Driving use-case. By using Frenet frame, some operations become trivial. For example, to determine whether the vehicle is on the road, it is sufficient to check if the lateral deviation d in Frenet frame is exceeding half of the road width w_{road}. The Frenet frame also ensures that the planning procedure remains the same for each segment of the road. Operations in Frenet frame are used for trajectory evaluation during planning (i.e. distance traveled, collision checking is trajectory on the road, etc.) and grid forming for underlining data structure in planning. For motion primitive generation, vehicle dynamic model in Cartesian coordinate system is used. Efficient transformations between frames are necessary as they are used in each step. Figure 5.5 illustrates the procedure of this transformation. Road geometry from Cartesian coordinate system (left) is represented as straight road in the Frenet frame (right). For Automated Driving use-case, we can additionally abstract it even further, so dimension l is defined as such that the middle of the rightmost lane has value 1, while the middle of the left-most lane has value N_l (number of lanes). The value 1.5 means that the vehicle is halfway between lane 1 and 2.

5.2.2 Dynamic Environment

To plan in dynamic environments, there are two major approaches, path-velocity decomposition (PDV) [229] and space-time augmentation [167]. Path represents connected sequence of positions in the space. If it is parametrized by time (has associated velocity) it is a trajectory. Path-velocity decomposition first solves one problem (plans either path or velocity trajectory along the path) and then finds solution for other one. It is generally computationally less complex method than space-time augmentation, but it is suboptimal and might get stuck by finding paths for which collision-free trajectory can not be provided, and vice-versa. This is especially critical in cluttered environments (i.e. urban driving) and long planing horizons. To avoid the risk of losing quality solutions, in this work, the space-time augmentation method is used. The 3D space-time configuration space Ω (SLT space) is constructed via Cartesian product of 2D configuration space ($s \times l$) and 1D time dimension t. By introducing time augmentation, dynamic environment becomes static configuration space (with increased dimensionality) defined as:

$$\Omega = \left\{ \mathbf{q} \equiv \left[t, s, l\right]^T \mid t \in \mathbb{R}^+, s \in \mathbb{R}^+, l \in \left[1, N_l\right] \right\}. \tag{5.15}$$

Here, t is time, s is the longitudinal position along the road and l is the lateral position on a road. It is important to note that trajectories are restricted to evolve in positive time direction.

5.2.3 Traffic rules

Expressing traffic rules as such that they can be processed by an algorithm is challenging. In this work, traffic rules are expressed geometrically, in the defined configuration space. They can depend directly on configuration variables t, s and l or on their derivatives. Several types of constraints/obstacles are considered, such as speed limits, forbidden lane-change, lane-availability, traffic lights and other traffic participants.

Speed limits

Speed limits may originate from speed limit signs, road curvature or some other factors. They are defined on certain segment of the road, or in the following region of Ω:

$$\mathcal{O}_k^{SL} = \left\{ \mathbf{q} \in \Omega \mid s \in \left[s_k, s_k + \Delta s_k\right] \right\}. \tag{5.16}$$

Speed limit violation can be evaluated by collision check mechanism:

$$\tau \left(\mathbf{q} \in \mathcal{O}_k^{SL} \mid \frac{\partial s(t)}{\partial t} \geqslant v_{k,\max} \right), \tag{5.17}$$

where $\tau\left(S\right)$ denotes the logical value (1 or 0) of the statement S. On this segment, the vehicle velocity represented by a gradient in direction of t, must not exceed the defined value $v_{k,\max}$. Similar limits can be defined for minimum velocity limit (i.e. for left lanes on the highway).

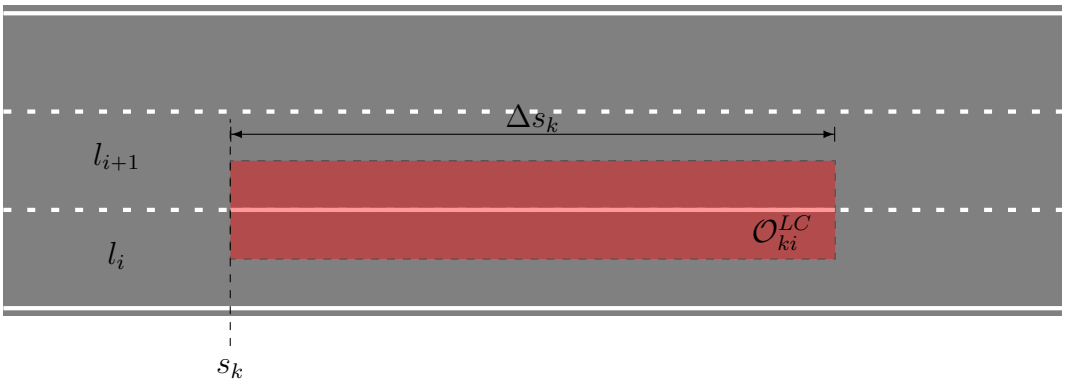

Figure 5.6: *Forbidden lane-change.*

Forbidden lane-change (solid lane line)

Lane-change prohibition can also be defined on certain segments of the road, for specific lane and lane-change direction. It is usually marked with the solid lane line as shown in Figure 5.6. The obstacle representation is given as:

$$\mathcal{O}_k^{LC} = \{\mathbf{q} \in \Omega \mid s \in [s_k, s_k + \Delta s_k], l(t) \in (l_k, l_k + 1)\}\,. \tag{5.18}$$

Prohibition may be applicable to both directions, where the collision check is performed by:

$$\tau\left(\mathbf{q} \cap \mathcal{O}_k^{LC} \neq \varnothing\right). \tag{5.19}$$

Alternatively, the prohibition can hold for single direction. Left-wise lane change prohibition is defined via collision test (5.20), while the right-wise is defined via negative partial derivative.

$$\tau\left(\mathbf{q} \in \mathcal{O}_k^{LC} \mid \frac{\partial l(t)}{\partial t} > 0\right). \tag{5.20}$$

Traffic light obstacle

The traffic light is a traffic control device which prohibits passing the defined line, during specific periods in time. Figure 5.7 shows obstacles created by two traffic lights. The obstacle can be active on one or several lanes. The obstacle is defined as:

$$\mathcal{O}_{ki}^{TL} = \{\mathbf{q} \in \Omega \mid s = s_k, t \in [t_{ki}, t_{ki} + \Delta t_{ki}]\}\,. \tag{5.21}$$

Each traffic light represents infinitely many obstacles, periodic in t, with the constant or variable period depending on the traffic light control system. The collision check is performed by evaluating:

$$\tau\left(\mathbf{q} \cap \mathcal{O}_{ki}^{TL} \neq \varnothing\right). \tag{5.22}$$

The vehicle trajectory should not pass trough the region of the traffic light at the time when the red light is on.

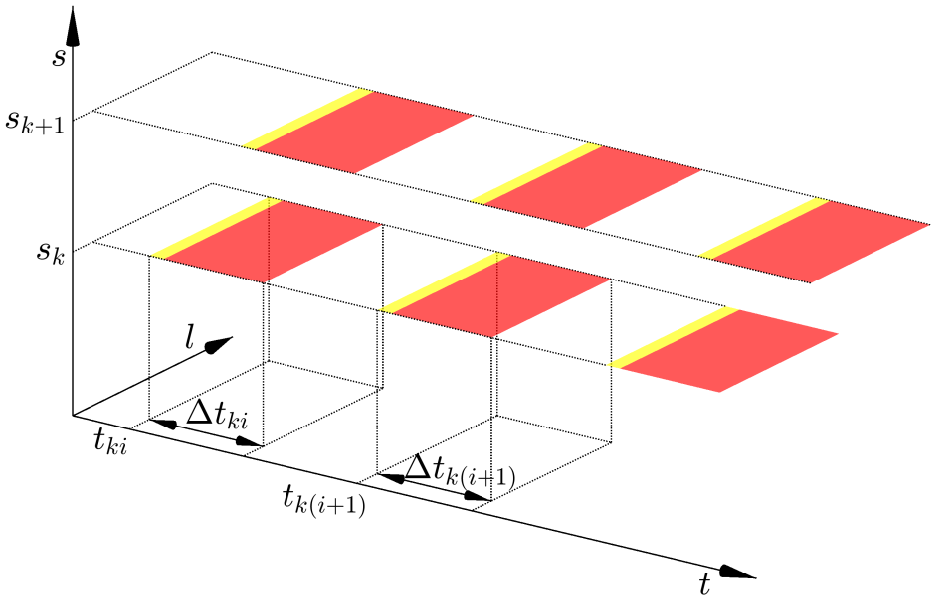

Figure 5.7: *Obstacles created by two traffic lights.*

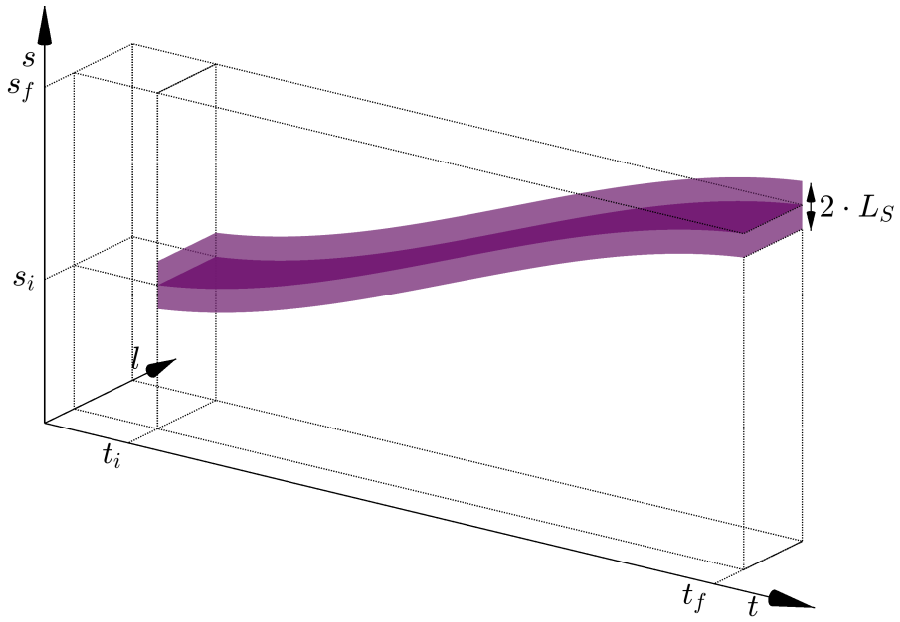

Figure 5.8: *Obstacle created by vehicle which is speeding up and slowing down.*

Other vehicles

One of the most crucial rules is that the vehicle should avoid collision with other traffic participants. Other vehicles on the road represent obstacles for ego vehicle and clearly constrain its motion. The violation of these constraints can be manifested not only as a direct collision with other vehicles, but also as a violation of driving rules, such as overtaking from right or slow overtaking from the left side. For a certain vehicle $\mathcal{V}_k$, the trajectory of its center is described with $s_k(t)$ and $l_k(t)$, while suitable lower and

upper bounds can be defined as:

$$L_S = L_k/2 + L_{ego}/2, \tag{5.23}$$

$$\underline{s}_k(t) = s_k(t) - L_S, \tag{5.24}$$

$$\overline{s}_k(t) = s_k(t) + L_S, \tag{5.25}$$

where L_k is the length of vehicle $\mathcal{V}_k$ and L_{ego} is the length of ego vehicle. Thus, the length L_{ego} is practically incorporated within the obstacle. Based on this, the corresponding obstacle can be defined as:

$$\mathcal{O}_k^V = \{\mathbf{q} \in \Omega \mid s(t) \in [\underline{s}_k(t), \overline{s}_k(t)]\}. \tag{5.26}$$

A collision check for a given $\mathbf{q}$, or the condition for which collision occurs can be validated by:

$$\tau\left(\mathbf{q} \in \mathcal{O}_k^V \mid l(t) \in (l_k(t) - 1, l_k(t) + 1)\right). \tag{5.27}$$

The assumption made here is that each vehicle occupies the whole lane, so if the ego vehicle center deviates from the middle of the lane, it is colliding with a vehicle in the adjacent lane. It is important to note that this is different from driving in a lane when executing the plan, as the control is not ideal and the vehicle can deviate from the middle of the lane, as long as it is within the lane. Figure 5.8 shows an example of geometric representation of the vehicle obstacle within a defined search space Ω. The presented vehicle speeds up and then slows down.

Apart from collision, it is sometimes also forbidden to overtake the vehicle from the right side. This can be expressed by a collision-test given in (5.28). This is modelled by prohibiting velocities greater than the velocity of a vehicle on the left. Beside using the velocity limit, acceleration is used so that in the case when a vehicle tries to overtake the ego vehicle, and ceases overtaking for some reason, the ego vehicle does not slow down as well. The corresponding collision test is formulated via:

$$\tau\left(\mathbf{q} \in \mathcal{O}_k^V \mid l(t) \in [1, l_k(t) - 1], \frac{\partial s(t)}{\partial t} \geqslant \frac{\partial s_k(t)}{\partial t}, \frac{\partial^2 s_k(t)}{\partial t^2} \geqslant 0\right). \tag{5.28}$$

Practically, overtaking a vehicle requires only a velocity greater than the velocity of a vehicle. However, to limit the time of overtaking maneuver, in several countries (e.g. Austria), there is also a limit on the minimum velocity difference Δv_{OV}, when overtaking other vehicles. The corresponding collision test can be formulated as:

$$\tau\left(\mathbf{q} \in \mathcal{O}_k^V \mid l(t) \in [l_k(t) + 1, N_l], \frac{\partial s(t)}{\partial t} \leqslant \frac{\partial s_k(t)}{\partial t} + \Delta v_{\mathrm{OV}}\right). \tag{5.29}$$

In multilane urban driving scenarios, rules for overtaking are not applicable.

The formulated obstacles represent the most common constraints in everyday driving, and almost all situations can be described by the combination of these. Clearly, multiple obstacles can be active at the same time.

It is worth pointing out that the collision checking with respect to such defined obstacles appears to be rather trivial, since it is usually reduced to a closed-form analysis whether some elementary, analytically defined curves intersect or not, or if the gradients of these curves attain certain values.

5.3 Objective

As mentioned several times, SuperHuman driving performance is defined as improved safety, efficiency and comfort compared to human drivers. This is natively multi-objective optimization problem. Objective can be formulated as a cost function, so it can be used to evaluate the quality of a given trajectory. It can reflect different goals such as: short travel time, comfort [230], safety [231], energy-efficiency [5], traffic rules (driving on the rightmost lane [232]) or a combination of some of these [135]. The design of the cost function is particularly important as it influences vehicle behavior and defines the optimal solution. Here, I provide two cost functions that are used throughout this work; one for energy-optimal driving, and the other for minimum lap-time driving .

5.3.1 Energy-optimal driving

For Energy-efficient Driving, the cost function has to reflect the initial requirement of minimal energy consumption. When considering only the propulsion power in the cost function, energy-efficient behavior results in a zero velocity trajectory. To avoid this, some authors introduced a term to the cost function to weight the traveling time [1]. The weighting coefficient is then tuned as such that the travel time is comparable to times achieved by human drivers. However, energy consumption includes energy used for auxiliaries P_{aux} (e.g. infotainment, air conditioning, etc.), which can be approximated with a constant load. It is important to include auxiliary power as during some periods large heating and air conditioning loads can significantly influence the vehicle energy consumption and driving range of electric vehicles. This load corresponds to a weighting coefficient presented in other works, but it is experimentally determined and not tuned. P_{aux} can include also some economic costs such as hourly rate (k_{hr}) of the operator, vehicle renting, etc. These are then divided by the current electricity price (k_{ep}) to get the power equivalent:

$$P_{\mathrm{tot}} = P_{\mathrm{aux}} + \frac{k_{\mathrm{hr}}}{k_{\mathrm{ep}}}. \tag{5.30}$$

The total power consumption of the vehicle is then the sum of auxiliary power consumption P_{tot} and power consumption of the motor, which is calculated from the product of the rotational speed of the motor ω_{m} and motor torque T_{m}. Consequently, the energy used starting from initial time t_{I} to final time t_{G} is equal to the integral of the power over time, represented as:

$$E = \int_{t_{\mathrm{I}}}^{t_{\mathrm{G}}} \left(\omega_{\mathrm{m}}(t) T_{\mathrm{m}}(t) + P_{\mathrm{tot}}\right) dt. \tag{5.31}$$

Instead of using time as a variable for integration, the distance can be used as well ([116]). This offers some advantages when solving the problem where the final time t_{G} is not known, while the final distance s_{G} is, and road slope appears as a function of

distance. Therefore:

$$E = \int_{s_\mathrm{I}}^{s_\mathrm{G}} \left(\frac{kT_\mathrm{m}(s)}{2r_\mathrm{w}\pi} + \frac{P_\mathrm{tot}}{v(s)} \right) ds. \tag{5.32}$$

Same cost function can also be used for Automated Driving use-case.

5.3.2 Minimum lap-time driving

In the Performance Driving use-case, the goal is to minimize time T necessary to drive the full lap. As path and velocity are solved together, this differs from pure maximizing velocity along some path. The criteria can be formulated as in (5.33), where $\psi(s) + \beta(s) - \psi_\mathrm{road}(s)$ represents the angle between vehicle velocity and the road tangent. The whole determinant represents the velocity component along the road.

$$T = \int_{s_\mathrm{I}}^{s_\mathrm{G}} \frac{ds}{v(s) \cdot \cos\big(\psi(s) + \beta(s) - \psi_\mathrm{road}(s)\big)}. \tag{5.33}$$

As it can be seen, this equation also uses the distance as bound variable as it is easier to relate it to the lap start and lap end. The cost function formulated like this can be used to find the global optimal solution. However, in MPC approach with fixed horizon, a different formulation will be used with the same goal to achieve minimum-lap time.

5.4 Scenario

As described in Section 4.2.1, scenario is defined by 5 factors: “Dynamic elements”, “Scenery”, “Self representation” (which build up a “Scene”), “Actions and events” and “Goals and values”. Figure 5.9 presents one scene in defined configuration space, using models presented in this chapter, to demonstrate versatility of the proposed solution. All mentioned aspects for this scenario are defined as follows.

- **Actions and events:** vehicle in front driving in the same lane, traffic light is about to turn into red.
- **Goals and values:** maintain safety, drive energy-efficiently (catch the green light).
- **Scene:**
  - **Dynamic elements:** 1 vehicle (besides ego vehicle) in the same lane approx one vehicle length in front and 1 traffic light, approx. 5 vehicle length in front.
  - **Scenery:** 3 lanes, dashed lane marking, speed limit at 60 km/h etc.
  - **Self representation:** Ego vehicle is a level 5 AV (full velocity range, capable of changing lanes), equipped with a complete perception and capable of communicating SPaT with TL.

5.5 Summary of Motion Planning problems

After all aspects of the driving problem are modelled, it is possible to define the motion planning problem. As mentioned earlier, SuperHuman driving performance is defined as improved safety, efficiency and comfort compared to human drivers, which is natively multi-objective optimization problem. To overcome that, in this work, safety and comfort are introduced as eliminatory requirements and formulated as constraints, so trajectories which do not satisfy minimum requirements for safety and comfort are removed and the rest of trajectories is optimized based on energy-efficiency (in Automated Driving and Energy-efficient Driving use-cases) or minimum lap-time (in Performance Driving use-case). Safety is provided by enforcing constraints from environment definitions (section 5.2) and comfort is provided by enforcing constraints by vehicle model (i.e. defining maximum acceleration and time of lane-change). Lower-level controller is assumed to assure that trajectories also have limited jerk. Each of the use-cases has specific setup of a problem. The problems are complementary and lead toward achieving SuperHuman Autonomous Vehicle. The problems of finding appropriate vehicle trajectory can be considered as an optimal motion planning problem. This problem has differential constraints defined by a vehicle dynamic model, state-dependent constraints introduced by environment (which have to be satisfied) and a cost function (which has to be minimized).

5.5.1 Problem 1: Energy-efficient Driving

For Energy-efficient Driving use-case, the task is planning longitudinal velocity trajectory of the vehicle, driving on the single lane road with hils while satisfying static constraints, as illustrated on the Figure 5.10. To provide completeness of the solu-

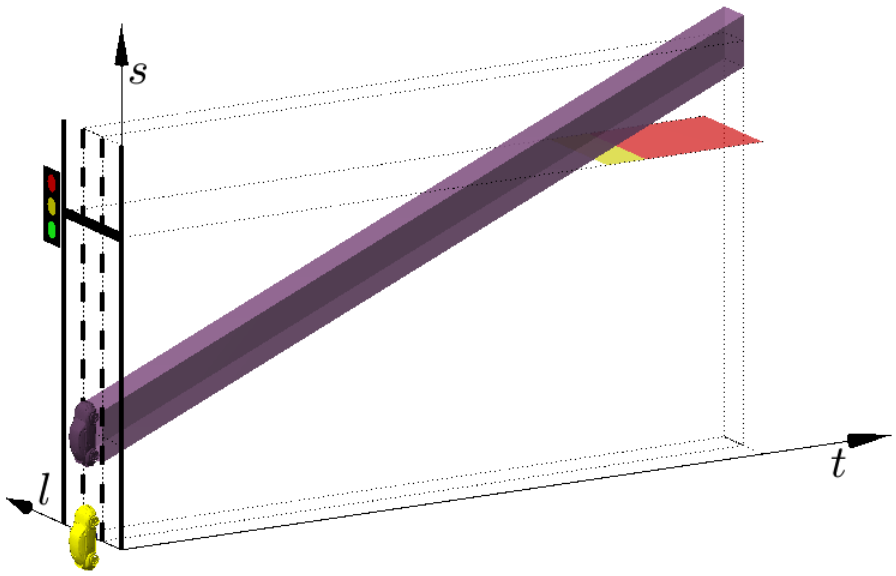

Figure 5.9: *Example urban driving scenario with a vehicle in front and traffic light about to turn red.*

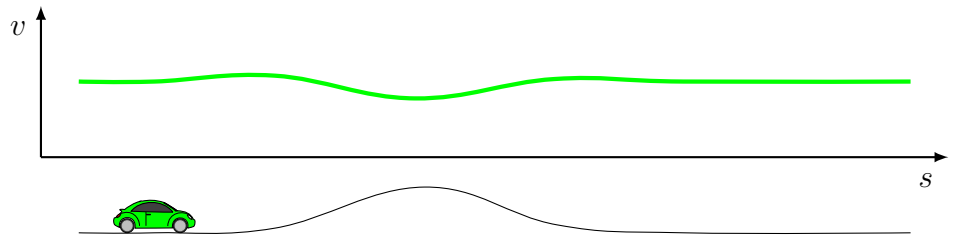

Figure 5.10: *Energy-efficient Driving Problem.*

tion, configuration space defined by environment Ω has to be augmented with velocity state v due to kinodynamic constraints on acceleration. Additionally, the lane is fixed, which reduces configuration space. The final search-space is defined as:

$$\mathcal{X}^{ED} = \left\{ \mathbf{x} \equiv \left[\mathbf{q}, v\right]^T \mid \mathbf{q} \in \Omega, v \in (v_{\min}, v_{\max}), l = 1 \right\}. \tag{5.34}$$

Motion planning problem for Energy-efficient Driving can be formulated as follows.

Given:

- **search space:** $\mathcal{X}^{ED}$, $(s \times v)$,
- **vehicle model:** longitudinal (section 5.1.1) + powertrain (section 5.1.3),
- **constraints:**
  - **internal:** acceleration and velocity (section 5.1.3),
  - **external:** speed limits (section 5.2.3),
- **objective:** energy-efficiency (section 5.3.1),
- **a query:** initial state $\mathbf{x}_0$ and the final state $\mathbf{x}_{\mathrm{G}}$.

Compute continuous path $\gamma(\cdot)$ that moves the vehicle from initial to final state while satisfying all the constraints ($\gamma : [0,1] \mapsto \mathcal{X}^{ED}_{\text{free}}$ such that $\tau(0) = \mathbf{x}_0$, $\tau(1) = \mathbf{x}_{\mathrm{G}}$) and objective is minimized.

5.5.2 Problem 2: Automated Driving

For Automated Driving use-case, the task is to plan vehicle trajectory in the complex environment, while satisfying all the constraints and minimizing energy consumption. Motion planning problem for Automated driving is illustrated at Figure 5.11.

To provide completeness, configuration space Ω has to be augmented with velocity state v and lane-change direction $\dot{l}$ due to kinodynamic constraints on acceleration and lane-change direction. The final search-space is defined as:

$$\mathcal{X}^{AD} = \left\{ \mathbf{x} \equiv \left[\mathbf{q}, v, \dot{l}\right]^T \mid \mathbf{q} \in \Omega, v \in (v_{\min}, v_{\max}), \dot{l} \in \{-1, 0, 1\} \right\}. \tag{5.35}$$

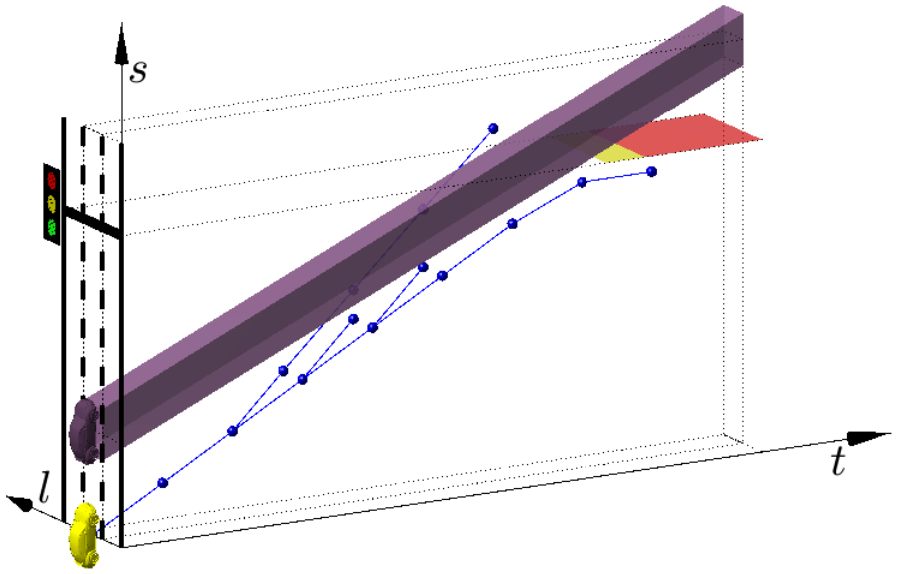

Figure 5.11: *Automated Driving behavior and motion planning problem.*

As planning is executed in moving horizon fashion with time and distance horizons, the goal region is defined as:

$$\mathcal{X}_G^{AD} = \{\mathbf{x} \mid \mathbf{x} \in \mathcal{X}^{AD}, s \geqslant S_{\text{hor}} \vee t \geqslant T_{\text{hor}}\}. \tag{5.36}$$

Automated Driving motion planning problem can be formulated as follows.
Given:

- **search space:** $\mathcal{X}^{AD}$,$(s \times l \times t \times v \times \dot{l})$,
- **vehicle model:** longitudinal (section 5.1.1) + lane-change (section 5.1.2) + powertrain (section 5.1.3),
- **constraints:**
  - **internal:** acceleration and velocity (section 5.1.3), lane-change time (section 5.1.2),
  - **external:** driving environment (section 5.2),
- **objective:** energy-efficiency (section 5.3.1),
- **a query:** initial state $\mathbf{x}_0$ and the final state region $\mathcal{X}_G^{AD}$.

Compute continuous path $\gamma(\cdot)$ that moves the vehicle from initial to final state while satisfying all the constraints ($\gamma : [0,1] \mapsto \mathcal{X}_{\text{free}}^{AD}$ such that $\tau(0) = \mathbf{x}_0$, $\tau(1) \in \mathcal{X}_G^{AD}$) and minimizing the objective.

5.5.3 Problem 3: Performance Autonomous Driving

For Performance Driving use-case, the task is to plan vehicle trajectory that keeps vehicle on the road and achieves minimum lap-time. Motion planning problem for Performance driving is illustrated at Figure 5.12.

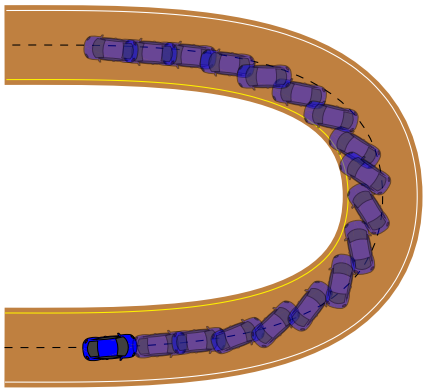

Figure 5.12: *Performance Driving Problem.*

To provide completeness, configuration space Ω has to be augmented with higher dynamical states, due to kinodynamic constraints by vehicle dynamics. The final search-space is defined as:

$$\mathcal{X}^{PD} = \left\{ \mathbf{x} \equiv \left[\mathbf{q}, \psi, v, \beta, \dot{\psi} \right]^T \mid \mathbf{q} \in \Omega, \psi \in (0, 2\pi), \ldots, \dot{\psi} \in (\dot{\psi}_{\min}, \dot{\psi}_{\max}) \right\}. \quad (5.37)$$

As planning is executed in a moving horizon fashion with time horizon, the goal region is defined as:

$$\mathcal{X}_G^{PD} = \{ \mathbf{x} \mid \mathbf{x} \in \mathcal{X}^{PD}, t \geqslant T_{\text{hor}} \}. \quad (5.38)$$

Performance Driving motion planning problem can be formulated as follows.

Given:

- **search space:** $\mathcal{X}^{PD}$, $(s \times l \times \psi \times v \times \beta \times \dot{\psi} \times t)$,
- **vehicle model:** Equilibrium State Manifold (section 5.1.2), semi-linearized bicycle model (section 5.1.2),
- **constraints:**
  - **internal:** acceleration and velocity (section 5.1.3), surface (section 5.1.2), steering and slip (section 5.1.2),
  - **external:** vehicle is on the road $|l| \leqslant \frac{w_{\text{road}}}{2}$ with is slightly more complex extension [233] for the full vehicle geometry,
- **objective:** minimum lap-time (section 5.3.2),
- **a query:** initial state $\mathbf{x}_0$ and the final state region $\mathcal{X}_G^{PD}$.

Compute continuous path $\gamma(\cdot)$ that moves the vehicle from initial to final state while satisfying all the constraints ($\gamma : [0, 1] \mapsto \mathcal{X}_{\text{free}}^{PD}$ such that $\tau(0) = \mathbf{x}_0$, $\tau(1) \in \mathcal{X}_G^{PD}$) and minimizing the objective.

5.6 Chapter Conclusion

This chapter provided models for different foreseeable aspects that should be modelled properly for autonomous vehicles. These include vehicle dynamics, environment

(including road, traffic rules, other traffic participants, etc.) and objective. Regarding vehicle dynamic different models are presented, suitable for different problem variants. For Energy-efficient Driving, longitudinal motion model is sufficient together with powertrain consumption model. However, for Automated Driving and Performance Driving, lateral motion has to be considered as well. For driving on structured roads lateral motion can be conservatively approximated by lane-change model, but for Performance Driving, even more detailed model is necessary. Environment is modelled geometrically, in augmented state-time space to deal with dynamic obstacles. In such space, constraints imposed by environment can be elegantly expressed. Objective is modelled as a scalar-valued functional representing energy-efficient and minimum lap-time driving.

These aspects can serve to model different scenarios and to formalize general planning problem for automated driving. The formalization approach is scalable to wide range of scenarios and facilitates efficient solving of the problem by motion planning approaches. Three problems are summarized for three tackled use-cases, Energy-efficient Driving, Automated Driving and Performance Driving.

6

Motion Planning

He who fails to plan is planning to fail.

Winston Churchill

This chapter aims to provide insight into specific Motion Planning solutions developed for three tackled use-cases. Motion planning solutions in this work are mainly based on Graph search methods, Dynamic Programming and A* search. Efficient custom implementations, frameworks based on effective combinations of different variants, as well as several custom developed heuristics help to solve the problem more efficiently. Planning trajectories that satisfy continuous kinodynamic motions is achieved by employing motion primitives and hybrid A* search-based solution. As it was shown when discussed about problem complexity (Section 4.6), it easily explodes, therefore a good trade-off is crucial for achieving real-time performance.

There are four main sections in this chapter, three sections dedicated for each of the use-cases and one section dedicated to combining planning and learning. Figure 6.1 illustrates the work presented in the chapter. As mentioned in the introduction, motivating use-case was Energy-efficient Driving (section 6.1). The first solution for this use-case is based on Dynamic Programming (section 6.1.1), which was approx. 100 times faster than the other state-of-the-art solution. Additionally, the same problem was solved using A* search and model-based heuristic based on a novel concept called *Optimal Cruising Velocity (OCV)* (Section 6.1.2). However, planning an energy-

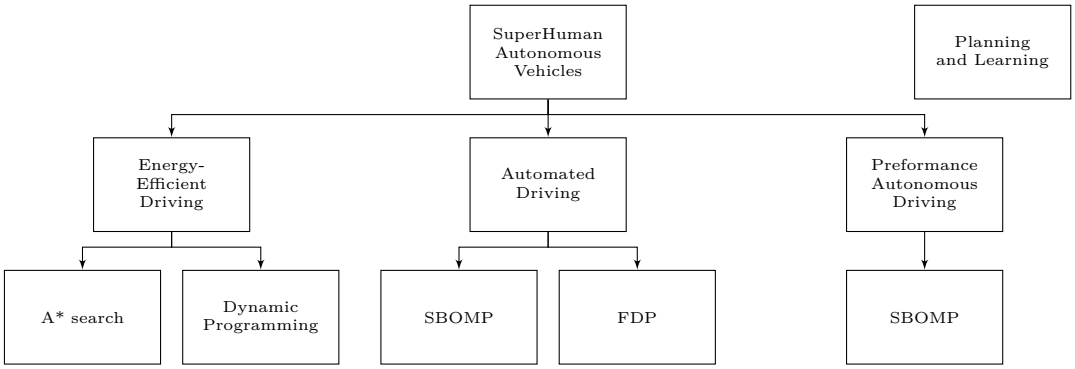

Figure 6.1: *Motion planning work overview.*

efficient velocity trajectory without considering other traffic participants is impractical in real traffic. It was noted that even in the automated driving community, a general solution for planning in different situations was missing. Therefore, further work was focused on the practical solution for automated driving in full complexity of driving situations, while keeping energy-efficiency as objective when choosing among collision-free trajectories (safety objective). Considering time-varying constraints (i.e. other vehicles, traffic lights, etc.) requires forward planning with frequent replanning. Additionally, time-varying constraints and multilane driving (considering overtaking possibility) require to increase search space from $s \times v$, and include lane l and time t dimensions. An even larger issue is that as space is multidimensional, it is not straightforward to form a regular grid for a graph search. The first solution for this problem was the Forward Dynamic programming (Section 6.2.3). However, this was abandoned as it showed to be very impractical due to the fact that Dynamic Programming explores the whole search-space. Therefore, SBOMP (Section 6.2.2) framework was developed, using the solution from the relaxed problem, solved with discrete Dynamic Programming, as a heuristic to guide the search. The SBOMP showed to be very efficient as it does not search the whole space and deals well with continuous dynamics due to motion primitives and hybrid A* approach. The SBOMP framework was further extended to consider Performance Driving use-case, where even higher dimensional search-space (7D) was explored, employing two different models for a combined generation of motion primitives (section 6.3). To improve computational performance, an approach using Machine Learning to learn heuristic function from previous searches, was introduced. This approach was further extended and generalized to a novel prolonged heuristic search (PHS)-based method for model (or simulator) exploration, based on prolonged backward heuristic search.

6.1 Energy-efficient Driving (classic eco-driving) problem

The problem of finding an energy efficient vehicle velocity trajectory can be considered as an optimal motion planning problem. This problem has differential constraints defined by a vehicle dynamic model, state-dependent constraints (which have to be satisfied) and a cost function (which has to be minimized). As the planning task for Energy-efficient Driving use-case is solved by discrete graph search methods, state discretization is necessary and an appropriate graph has to be constructed. There are two usual approaches to represent this problem, we can use time or distance as a basis for discretization. In this work, the graph is constructed by discretizing distance (s) and velocity (v) states by discretization steps Δs and Δv respectively. Thus, nodes n represent unique states from $s \times v$ space as it is illustrated in Figure 6.2. Choosing distance is useful when the final time is not fixed and the system is space-dependent (i.e. speed limits, road slope). This choice brings several disadvantages e.g. if a trajectory passes zero velocity, it is impossible to calculate the time spent in that state (we always assume 0 seconds). An additional disadvantage is that on high velocities times of transitions shorten (as distance step Δs is fixed), so with fixed velocity discretization step Δv the number of possible transitions (which satisfy the maximum acceleration constraints) decreases. Nevertheless, an appropriate selection

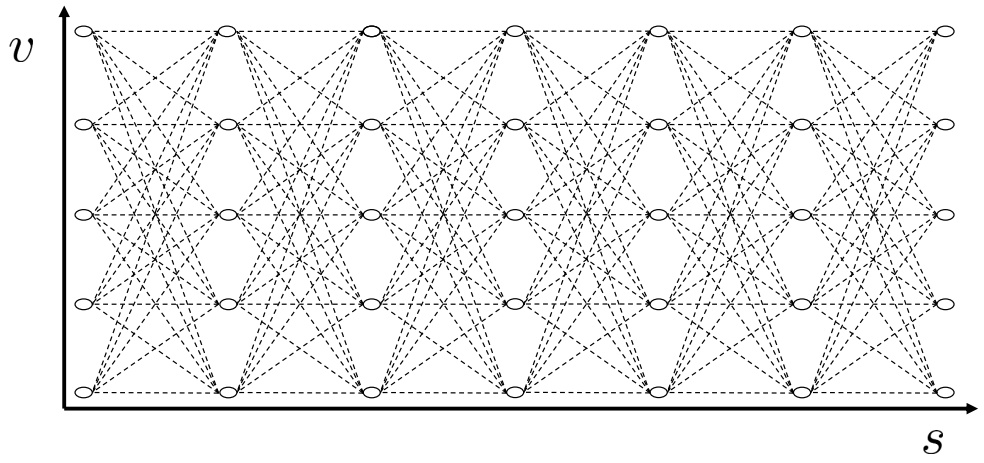

Figure 6.2: *Search space for Energy-efficient Driving problem constructed as a graph.*

of discretization steps leads to a valid solution. By assuming positive velocities only, the graph is directed. The vehicle model provides energy consumption of transitions between nodes. Transition from a node with initial velocity v_i at s_i to a node with final velocity v_f at s_{i+1}, using current road slope $\alpha(s_i)$ and other vehicle model parameters is computed as $\mathcal{M}_E(v_i, v_f, \alpha(s_i), \Delta s)$. Solving the eco-driving problem is equivalent to finding the shortest path in this graph and minimizing (6.1).

$$E_{\min} = \min_{T_{\mathrm{m}}(\cdot)} \sum_{i}^{N_s} \mathcal{M}_E(v_i, v_{i+1}, \alpha(s_i), \Delta s). \tag{6.1}$$

Several methods can be applied to solve this problem. In this work, Dynamic Programming-based and A* search-based solutions are provided.

6.1.1 Dynamic Programming

Dynamic programming is a very common method used for solving the optimal control problem discussed in this work. The main advantages are its flexibility and possibility to incorporate different models and constraints. It is based on the Principle of Optimality, and it was introduced by R. Bellman [43].

Principle of Optimality: *"An optimal policy has the property that whatever the initial state and initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decisions."*

This principle enables an iterative search for the optimal solution starting from the goal state and building an optimal trajectory towards the start, solving only one transition each step. In each step, transitions from all possible current states to all states

from the previous step are computed and transitions with minimum total costs (costs of transition and accumulated cost-to-go values) are chosen. Thus, going backwards from the final step, the space is systematically searched by computing each time only one step transition and reusing computation results from the previous steps. For further details about Dynamic Programming, the interested reader is referred to [186]. Dynamic Programming represents also a problem solving paradigm in Computer Science.

For eco-driving problem, in each stage of Dynamic Programming backward transitions are made from position s_{i+1} to position s_i with the step Δs. It is assumed that the acceleration is constant during this transition. Vehicle model provides energy consumption of transitions from initial velocity v_i at s_i to final velocity v_f at s_{i+1}, using current road slope $\alpha(s_i)$ and other vehicle model parameters ($\mathcal{M}(v_i, v_f, \alpha)$). Dynamic Programming solution can be improved if some computations are already prepared for reuse and transitions are computed in a matrix form (for 2D case). For example, all transitions can be organized as matrices where rows represent initial velocities and columns represent final velocities. Several matrices can be precomputed i.e. $\mathbf{V}_\mathrm{i}$ - matrix of initial velocities, $\mathbf{V}_\mathrm{f}$ - matrix of final velocities, $\mathbf{V}_\mathrm{av}$ - matrix of average velocities, $\mathbf{T}_\Delta$ - matrix of time for transition, $\mathbf{A}_\mathrm{acc}$ - matrix of acceleration during transition. These matrices can be used in model $\mathcal{M}$ for faster computations. Equations 6.2 and 6.3 are used forthe mentioned precomputation. Operation "$\oslash$" and "$\circ$;; denote element-wise division and multiplication. Vehicle model $\mathcal{M}$ contains also internal constraints, cost function, etc. For reconstruction of the optimal solution, child node matrix - N^* contains optimal transitions for each node. Besides that, optimal cost-to-go matrix (V^*) contains the optimal cost-to-go values for each node. At each stage, based on Bellman's equation, optimal transitions are chosen as the lowest of all transitions based on cost of transition and cost-to-go value of the new state.

$$\mathbf{V}_\mathrm{i} = \begin{bmatrix} v_\mathrm{min} & v_\mathrm{min} & \dots & v_\mathrm{min} \\ \vdots & \vdots & \ddots & \vdots \\ v_\mathrm{max} & v_\mathrm{max} & \dots & v_\mathrm{max} \end{bmatrix}, \mathbf{V}_\mathrm{f} = \begin{bmatrix} v_\mathrm{min} & \dots & v_\mathrm{max} \\ v_\mathrm{min} & \dots & v_\mathrm{max} \\ \vdots & \ddots & \vdots \\ v_\mathrm{min} & \dots & v_\mathrm{max} \end{bmatrix}, \mathbf{V}_\mathrm{av} = \frac{\mathbf{V}_\mathrm{i} + \mathbf{V}_\mathrm{f}}{2}, \quad (6.2)$$

$$\mathbf{T}_\Delta = \Delta s \oslash \mathbf{V}_\mathrm{av}, \qquad \mathbf{A}_\mathrm{acc} = \frac{2}{\Delta s} \cdot \mathbf{V}_\mathrm{av} \circ \mathbf{V}_\mathrm{av} - \frac{2}{\Delta s} \cdot \mathbf{V}_\mathrm{i} \circ \mathbf{V}_\mathrm{av}. \quad (6.3)$$

For this work, as presented in [5], a tailored and computationally optimized solution for optimal velocity trajectory planning based on Dynamic Programming. It was developed in MATLAB, making intensive use of matrix calculus, as mentioned before. To validate the implementation, both forward and backward dynamic programming schemes were implemented. The achieved results are identical, as it was expected. The advantage of the backward computation is that the computed results can be reused during the trip by taking the branch of the tree based on current position and velocity. This is possible as Backward Dynamic Programming problem only depends

Algorithm 1: BDP for eco-driving

input : v_{G}, s_{G}, $\alpha(s)$, $\mathcal{M}(v_i, v_f, \alpha)$
output: $N*$, $V*$

begin
  $\mathbf{v} \leftarrow v_{\min} : \Delta v : v_{\max}$ // possible velocities
  $\mathbf{V}_{\mathrm{i}}, \mathbf{V}_{\mathrm{f}}, \mathbf{V}_{\mathrm{av}}, \mathbf{A}_{\mathrm{acc}}, \mathbf{T}_{\Delta} \leftarrow$ `Precompute()` // precomputed matrices for matrix computation
  $V*(\mathrm{end}, :) \leftarrow \infty$,
  $V*(\mathrm{end}, v_G) \leftarrow 0$; // define goal
  for $i \leftarrow N_s - 1$ **to** 0 **do**
    $C \leftarrow \mathrm{meshgrid}(V*(i+1, :)')$ // reshape cost-to-go
    $C(\mathbf{v} > v_{\mathrm{lim}}(i), :) \leftarrow \infty$ // maximum velocity limits
    $C \leftarrow C + \mathcal{M}(\mathbf{v}, \mathbf{v}, \alpha(s))$ // cost with transition
    $V*(i, :) \leftarrow \min(C, [], 2)$ // minimum from each row
    $N*(i, :) \leftarrow \arg\min(C, [], 2)$ // minimum argument from each row
  return $N*$, $V*$

on the final state. This is not the case with the Forward Dynamic Programming computation, where results are related to the specific initial state. On the other hand, the advantage of the forward computation is that other states such as the position of other vehicles can be computed as the initial time is always known.

In order to deeply understand the optimal control discussed in this work we introduce two helpful concepts: the optimal trajectory tree and the cost-to-go map.

Optimal velocity trajectory tree

The optimal velocity trajectory tree is a tree-like structure formed by connecting all optimal transitions. Together with a cost-to-go map, it gives insight into the optimal driving behavior when only static constraints are present. It can be noted generally, that if two different trajectories have a common node they will continue on the same trajectory towards the goal state n_{G}. This implies that when planning a trajectory with the forward approach, if constraints introduced by other traffic participants are not active any more, a trajectory from a backward planning starting from that state towards the goal can be reused. This is a consequence of the Bellman's principle of optimality. This property of the optimal trajectory tree will be used to reduce the computational effort needed in Automated Driving use-case. An optimal velocity trajectory tree for a problem considered in this work with discretization steps of 5 m for distance and 0.5 m/s for velocity is shown in Figure 6.3. This map is generated from the goal state towards the start using backward DP. Besides the optimal trajectory for the given initial condition, multiple other trajectories (branches) for different initial conditions are available.

Cost-to-go map

The cost-to-go map provides complementary information to the optimal trajectory tree. It represents the minimum energy needed to finish a trip from that state. It can be achieved by following the optimal trajectory, represented as a branch on the optimal trajectory tree, starting from that state. In Figure 6.4 the cost-to-go map for the same problem as in Figure 6.3 is shown.

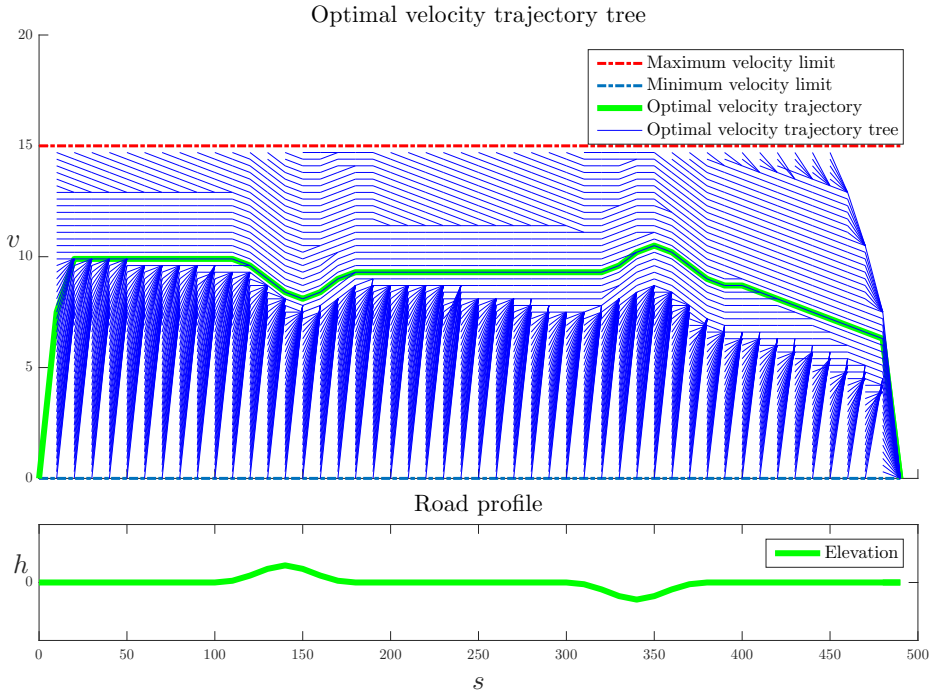

Figure 6.3: *Optimal velocity trajectory tree from Backward Dynamic Programming.*

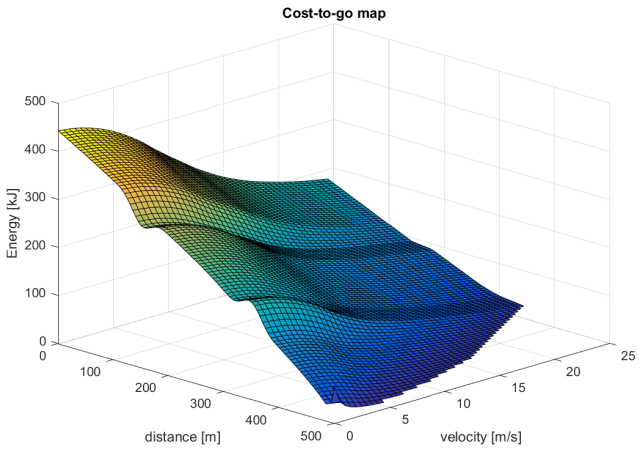

Figure 6.4: *Cost-to-go map.*

The shape of the cost-to-go map is closely related to the cost function and problem instance. It can be noted that cost value increases as distance is further from the goal. Additionally, cost value is smaller on higher velocities because of the larger kinetic energy stored in a moving vehicle. Cost value is also smaller on a hill and higher in the valley, because of the potential energy.

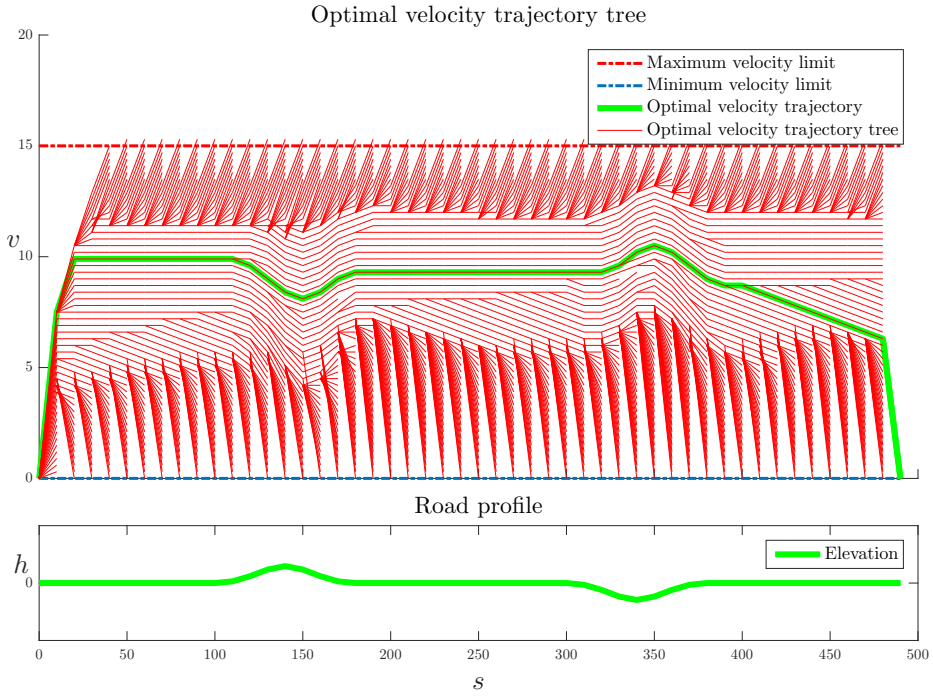

Figure 6.5: *Optimal velocity trajectory tree from Forward Dynamic Programming.*

Forward Dynamic Programming

Similar to Backward Dynamic Programming, Forward Dynamic Programming Solution can be developed. The difference here is that in each stage of Dynamic Programming, backward transition is made from position s_i to position s_{i+1} with the step Δs. The Figure 6.5 shows the optimal trajectory tree obtained by forward DP.

6.1.2 A* search

Although significantly less popular, A* search can also be used for solving eco-driving problem. A* search is one of the earliest, yet one of the most used methods for path planning problem ([48]). It is based on the well-known Dijkstra's algorithm, but instead of exploring the whole space it uses heuristics to guide exploration to the nodes, that lead to the solution more quickly. Therefore, it generally has a better performance compared to Dijkstra's algorithm. A* is an optimal method for finding the optimal path for some admissible heuristic function.

Starting from the *initial node* n_{I}, which is chosen as the initial *current node* n, all neighbors n' are determined and added to the OPEN list. From the OPEN list, the node with the lowest cost is chosen to be the next *current node*. This is repeated until the *goal node* is reached or the whole graph is explored. The cost $f(n)$ is computed using (6.4). Function $g(n)$ represents the cost to travel from *initial node* n_{I} to *current node* n (*cost-to-come*), and function $h(n)$ represents the cost to travel from *current node* n to *goal node* n_{G} (*cost-to-go*), estimated using some heuristic function based on

prior knowledge about the problem.

$$f(n) = g(n) + h(n). \tag{6.4}$$

There can be several issues when implementing A*. To accelerate accessing the minimum of all open nodes, the OPEN list is usually organized as some kind of a priority queue. In this work, the OPEN list is implemented as a binary heap with a hash table. A binary heap is a structure which keeps the minimum element at the top. Therefore, it is easy to access the next *current node*. The hash table keeps a position of every node on the heap, for an easier accessing when comparing with newly expanded nodes. Besides, the hash table keeps information if a node is already in the CLOSED list.

Cost-to-come

Cost-to-come $g(n)$ is the minimum cost necessary to come from the *initial node* n_I to the *current node* n. It is exact as it is computed cumulatively as nodes are explored. One step transition cost is added to the *parent node*'s *cost-to-come*. To compute transition costs, the vehicle model (5.1) and (5.14) is used along with the cost function (5.32).

Heuristic function for cost-to-go estimation

As the path to the goal is not known, the cost of travel has to be estimated. The heuristic function $h(n)$ is used to estimate the cost needed to travel from any node n based on some prior knowledge about the problem. As it is shown in [48], if the heuristic function is admissible (underestimates the *cost-to-go*), the result of A* search is the optimal trajectory. For the Shortest Path planning problem, the usual heuristic function is the Euclidean distance. The distance between any two points is always greater or equal to the Euclidean distance. However, it is also important that heuristic function is precise, as the precision influences the size of the explored space, and therefore the efficiency of the search. In general, as the heuristic function tends to the exact *cost-to-go*, the explored space shrinks and the search time shortens. In the ideal case, if the heuristic function gives the exact cost-to-go, only nodes which belong to the optimal trajectory would be explored.

To find the energy-optimal velocity trajectory, the heuristic function must always underestimate the energy needed to drive from any node n (v_i and s_i) to the goal node n_G, defined by the initial velocity and position (v_f and s_f). The energy needed to drive based on model (5.1),(5.14) and the cost function (5.32), can be presented as the sum of the work needed for acceleration, overcoming air drag, rolling resistance, road slope and the energy used by the auxiliaries (or operational costs).

$$W = W_\mathrm{acc} + W_\alpha + W_\mathrm{roll} + W_\mathrm{air} + E_\mathrm{aux}. \tag{6.5}$$

Assuming no energy is needed to drive to the final state would provide admissible heuristic if recuperation would not be allowed, as the work is always positive. Since recuperation is allowed, under certain conditions, the work can be negative as well. So, assuming a value of zero could lead to a suboptimal solution. Air drag, rolling resistance, and auxiliary-related works are always positive and therefore can be underestimated by zero. Nevertheless, this is an imprecise estimation, far from the real value, leading to the loss of A* efficiency as informed search.

Heuristic function for acceleration-related work If no efficiency losses are considered, the work needed to accelerate the vehicle moving in the horizontal plane results in a change of kinetic energy. Therefore, the work needed for acceleration can be described only by the initial and the final velocity states, v_i and v_f as in (6.6). If losses are present, the needed energy will only be higher.

$$W_{\text{acc}} \geqslant \Delta E_k = m\frac{v_f^2 - v_i^2}{2}. \tag{6.6}$$

If the vehicle is driving on the horizontal plane, without slopes, this work alone expresses admissible heuristic. If this is not the case, the influence of road slope has to be included additionally.

Including slope related work Overcoming road slope results in climbing up or down a hill and therefore increases or decreases the potential energy of the vehicle. In this case, the work can be described as:

$$W_\alpha + W_{\text{acc}} \geqslant \Delta E_k + \Delta E_p = \Delta E_k + mg(h(s_f) - h(s_i)), \tag{6.7}$$

where $h(s)$ is the road elevation profile dependent on the distance traveled (s).

Rolling resistance related work As the rolling resistance force is modeled as a constant, it does not depend on vehicle driving trajectory. It can be exactly computed for the specific road segment and included in the heuristics. Assuming the rolling resistance work as a constant provides a more precise, and still admissible heuristic, but cannot be used alone, apart from $\Delta E_k + \Delta E_p$. The expression for rolling resistance related work is given as:

$$W_{\text{roll}} = c_{\text{r}} mg \int_{s_i}^{s_f} \cos\left(\alpha\left(s\right)\right) ds. \tag{6.8}$$

Heuristic function with air-drag and auxiliary power Above mentioned components are used for constructing the state-of-the-art admissible heuristic function. So far, based on the authors best knowledge, none of the heuristics successfully incorporated air-drag resistance and auxiliary power to provide admissible heuristic.

As opposed to W_{acc}, W_α and W_{roll}, which can be expressed by the initial and the final states only, W_{air} and E_{aux} depend on velocity trajectory between the initial and the

final state. We have that:

$$W_{\text{air}} + E_{\text{aux}} = \int_{s_i}^{s_f} \underbrace{\left(\underbrace{\frac{1}{2}\rho_{\text{a}} c_{\text{d}} A_{\text{f}} v(s)^2}_{F_{\text{air}}} + \frac{P_{\text{aux}}}{v(s)} \right)}_{F_{\text{v}}} ds, \tag{6.9}$$

$$W_{\text{air}} + E_{\text{aux}} = \int_{s_i}^{s_f} F_{\text{v}}\left(v(s)\right) ds, \tag{6.10}$$

$$F_{\text{v}} = \frac{1}{2}\rho_{\text{a}} c_{\text{d}} A_{\text{f}} v(s)^2 + \frac{P_{\text{aux}}}{v(s)}. \tag{6.11}$$

F_{v} can be considered as a virtual force consisting of air drag force and virtual force by P_{aux}. As it can be seen in Figure 6.6 based on the convex shape of F_{v}, we can conclude that there exists one optimal velocity for which F_{v} is minimized, noted by v^* and named *Optimal Cruising Velocity (OCV)*.

Optimal Cruising Velocity (OCV) Optimal Cruising Velocity v^* is the velocity which minimizes energy consumption for driving on a flat, horizontal, empty road. It finds optimal balance between air drag influence and time-proportional cost influence. Optimal Cruising Velocity v^* achieves minimum energy consumption by minimizing the virtual force F_{v}:

$$F_{\text{v,min}} = F_{\text{v}}(v^*). \tag{6.12}$$

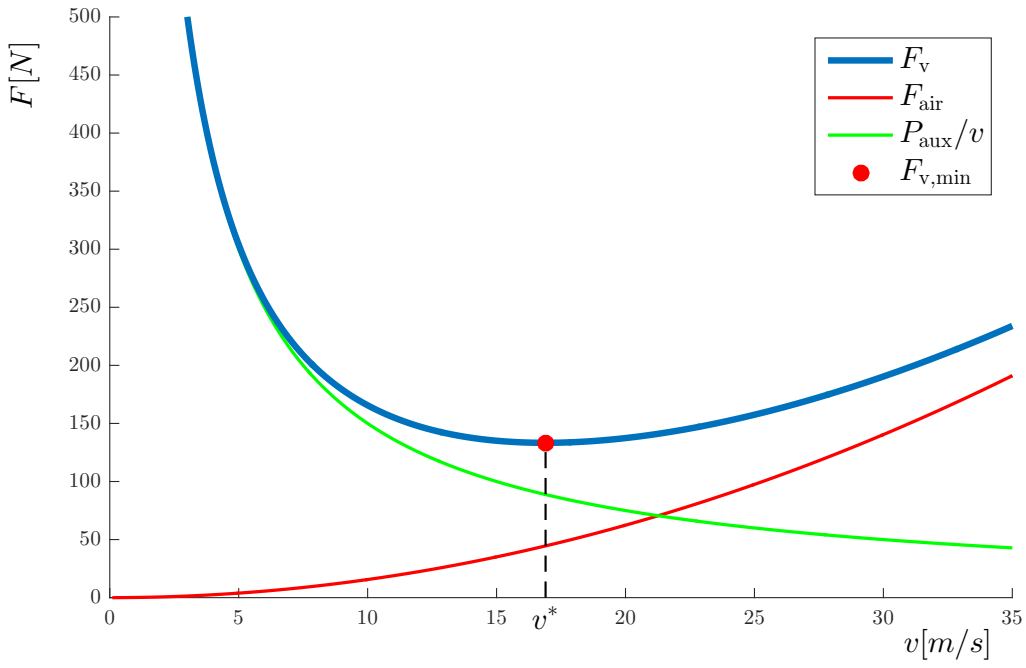

Figure 6.6: *Velocity-dependent virtual force F_{v}, generated by combined air drag resistance and auxiliary power.*

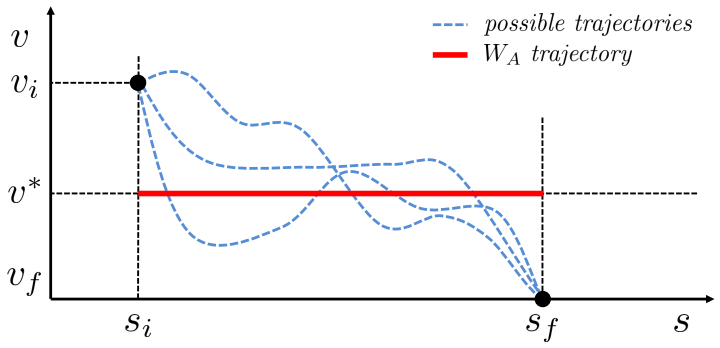

Figure 6.7: *Some of possible trajectories for driving from initial to final state, and lower bound trajectory for W_A computation.*

To calculate v^*, the derivative of F_v at v^* should be zero:

$$\frac{\partial F_\mathrm{v}(v^*)}{\partial v} = 0 \quad \Rightarrow \quad v^* = \sqrt[3]{\frac{P_\mathrm{aux}}{\rho_\mathrm{a} c_\mathrm{d} A_\mathrm{f}}}. \tag{6.13}$$

The optimal cruising velocity is unique for the vehicle (and it's hourly rate k_hr). It depends on vehicle aerodynamic shape, air density and P_aux (eventually, motor efficiency can be included). The minimum virtual force $F_\mathrm{v,min}$, can be used to estimate the lower bound of $W_\mathrm{air} + E_\mathrm{aux}$, defined only with the initial and the final states.

$$W_\mathrm{air} + E_\mathrm{aux} \geqslant W_A = \int_{s_i}^{s_f} F_\mathrm{v,min} ds = F_\mathrm{v,min}(s_f - s_i). \tag{6.14}$$

As it was shown in (6.14), air drag and auxiliary-related work, $W_\mathrm{air} + E_\mathrm{aux}$ for any trajectory, is always greater or equal to the work when driving with constant velocity v^* (Figure 6.7). This work is computed as a product of the minimum virtual force $F_\mathrm{v,min}$ and the distance traveled.

A tendency to drive with optimal cruising velocity can be noticed in [129], where authors analyzed a conventional vehicle using approximated engine fuel injection rate map. Since in a conventional vehicle auxiliary power comes from the alternator, which represents an additional constant load on the engine, the results are similar. In this work, a theoretical explanation of this phenomenon (optimal cruising velocity), reasoning and influencing factors and relations are provided.

Improved heuristic function with air-drag and auxiliary power For situations where the initial or the final velocity is not equal to the optimal cruising velocity v^*, the air drag influence can be more precisely estimated by including acceleration periods to reach v^*. As acceleration is limited, this transition is not instantaneous. It is important to note that work needed for acceleration is not computed in this trajectory, as it is included in kinetic and potential energy. Only energy related to air-drag and auxiliary power ($W_\mathrm{air} + E_\mathrm{aux}$), as if the vehicle drives this trajectory is considered.

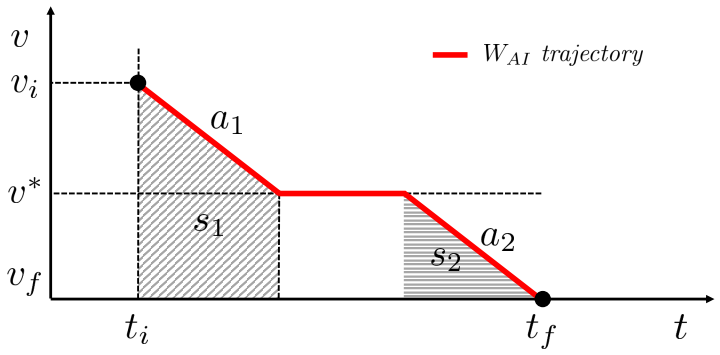

Figure 6.8: *Velocity trajectory for W_{AI} computation.*

A general trajectory with acceleration and deceleration phases is depicted on picture Figure 6.8. For a simpler visualization, time t is used as the axis instead of distance s. While driving with uniform acceleration, from some velocity v_1 to some velocity v_2, the distance traveled can be computed as:

$$s = \frac{v_2^2 - v_1^2}{2 \cdot a}. \tag{6.15}$$

For computing s_1 from Figure 6.8, general formula (6.15) is used, with $v_1 = v_i$, $v_2 = v^*$ and $a = a_1$. For computing s_2, the same formula is used, with $v_1 = v^*$, $v_2 = v_f$, and $a = a_2$. Depending on the initial and the final velocity, accelerations a_1 and a_2 are determined as:

$$a_1 = \begin{cases} a_{\min}, v_i > v^*, \\ a_{\max}, v_i < v^*, \end{cases} \quad \text{and} \quad a_2 = \begin{cases} a_{\min}, v_f < v^*, \\ a_{\max}, v_f > v^*. \end{cases} \tag{6.16}$$

The work $W_{\mathrm{AT}} = W_{\mathrm{air}} + E_{\mathrm{aux}}$, when driving with uniform acceleration, for time T, from velocity v_1 to v_2 is:

$$W_{\mathrm{AT}} = \int_0^T \left(\frac{1}{2} \rho_{\mathrm{a}} c_{\mathrm{d}} A_{\mathrm{f}} v(t)^3 + P_{\mathrm{aux}} \right) dt, \tag{6.17}$$

with

$$v(t) = v_1 + at, \text{ and } T = \frac{v_2 - v_1}{a}. \tag{6.18}$$

Therefore:

$$W_{\mathrm{AT}} = \frac{1}{2} \rho_{\mathrm{a}} c_{\mathrm{d}} A_{\mathrm{f}} \left(v_1^3 T + \frac{3 v_1^2 a T^2}{2} + \frac{a^3 T^4}{4} \right) + P_{\mathrm{aux}} T. \tag{6.19}$$

Using relations (6.18) and (6.19), the cost of driving with uniform acceleration between two velocities can be computed. The total work W_{AI}, in this case, can be computed as

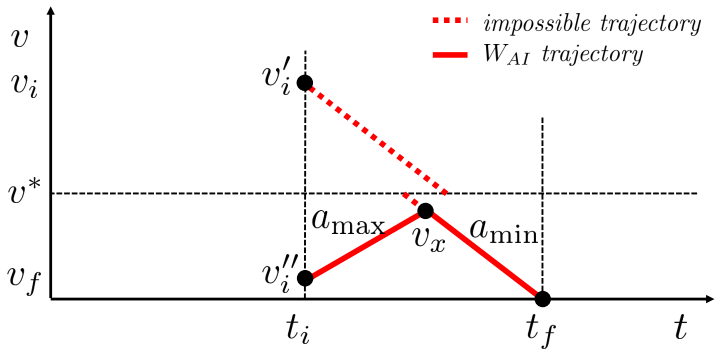

Figure 6.9: *Special cases of velocity trajectories for W_{AI} computation.*

the sum of two works (corresponding to acceleration and deceleration) and the driving with constant velocity v^*:

$$W_{\mathrm{AI}} = W_{\mathrm{AT1}} + F_{\mathrm{v,min}} \cdot (s_f - s_i - s_1 - s_2) + W_{\mathrm{AT2}}. \tag{6.20}$$

If $s_1 + s_2 > s_f - s_i$, the optimal velocity cannot be reached, as shown in Figure 6.9. There are two sub-cases in this case.

- When v_i and v_f are on opposite sides of v^*. This is a trivial case, as the final state is not reachable for limited acceleration, and therefore the cost is infinite. This case is marked with dashed red line in Figure 6.9.
- When both v_i and v_f are either smaller or greater than v^*. There will be no constant velocity driving part as v^* will not be reached, it will accelerate until it reaches v_x and then decelerate, or vice-versa. This is depicted with a solid red line in Figure 6.9.

By using the $s_1 + s_2 = s_f - s_i$, the velocity at which acceleration changes sign can be determined as:

$$v_x = \sqrt{\frac{2a_1a_2s + a_2v_i^2 - a_1v_f^2}{a_2 - a_1}}. \tag{6.21}$$

Velocity v_x is then used instead of v^* for computing W_{AT1} and W_{AT2}. The work W_{AI}, in this case, can be computed as the sum of two works (acceleration and deceleration):

$$W_{\mathrm{AI}} = W_{\mathrm{AT1}} + W_{\mathrm{AT2}}. \tag{6.22}$$

Motor efficiency The motor efficiency certainly influences the energy used. It does not have to be considered to have admissible heuristics, but it improves precision and therefore search efficiency. Applying the maximum motor efficiency $\eta_{\max}$ as a constant efficiency for any operating point will result in underestimating heuristic, that is more precise than when omitting efficiency. The motor efficiency can be applied only to power which flows through the motor. Therefore, it should not be applied to P_{aux} as this energy does not flow through the motor.

Remarks on model-based heuristics Combining all relevant work elaborated in the introduction, the best state-of-the-art, admissible heuristic, can be constructed as a sum of kinetic and potential energy, rolling resistance loss with applied motor efficiency as in (6.23) - (6.24).

$$W_{\text{tot}} = \Delta E_k + \Delta E_p + W_{\text{roll}}, \tag{6.23}$$

$$h_{\text{SoA}} = W_{\text{tot}} \cdot \eta_{\text{max}}^{-\text{sgn}(W_{\text{tot}})}. \tag{6.24}$$

The proposed heuristic, however improves this state-of-the-art heuristic by additional work for air drag resistance and auxiliary energy (and operation costs) as shown in the sections 6.1.2 and 6.1.2. The efficiency cannot be applied to the proposed heuristic as auxiliary power does not flow trough motor. The proposed heuristic is computed as:

$$h_{\text{pro}} = h_{\text{SoA}} + W_{\text{AI}}. \tag{6.25}$$

6.1.3 Remarks on Energy-efficient Driving

Two different solutions for planning motion in Energy-efficient Driving use-case were presented. Solutions are efficient for providing long-term motion plan for tens of kilometers. However, they assume empty roads without considering other traffic participants. That is impractical in real traffic as there are rare occasions where other participants can be neglected (i.e. highway driving with very low traffic). Generally, in eco-driving community, some works consider other traffic participant but only partially (i.e. only vehicle following possibility) and no work was done on general multilane energy-efficient driving in different situations. It was noticed that even in automated driving community, general solution for planning in different situations was missing. Therefore, further work was focused on practical solution for automated driving in full complexity of driving situations, while keeping energy-efficiency as driving objective for choosing among collision-free trajectories (safety objective).

6.2 Automated Driving

The second use-case focuses on automating the driving task. In this use-case, we assume that the vehicle can operate in full velocity range, fully independently, and perform all dynamic driving tasks (DDT) without any input from a human (level 5 AD). The goal is to develop general Motion Planning solution for autonomous vehicles, applicable to all driving scenarios, achieving SuperHuman performance.

Planning methods used in this work (Dynamic Programming and A* search) can be executed starting from the goal state n_{G} towards the initial state n_{I} (backward planning) or vice-versa (forward planning). The advantage of the backward planning is that the computed results can be reused during the trip as long as the final state does not change, as they only depend on the final state. This is not the case with the forward planning, where results are related to specific initial states and the results is not much useful if the situation deviates from planned. On the other hand, forward

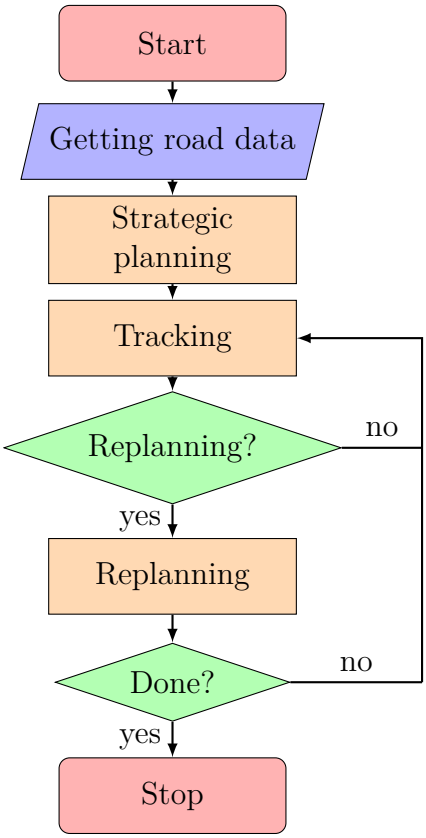

Figure 6.10: *Optimal motion planner flowchart.*

planning can tackle constraints which backward cannot (time-varying constraints, interactive driving, etc.). Furthermore, due to curse of dimensionality, planning in higher dimensional spaces in infeasible for longer horizons.

Considering time-varying constraints (i.e. other vehicles, traffic lights, etc.) requires to employ forward planning with frequent replanning. Moreover, time-varying constraints and multilane driving (overtaking possibility) requires to increase search space from $s \times v$, and include lane l and time t. Even larger issue is that, as the space is multidimensional, it is not straightforward to form a regular grid for a graph search.

6.2.1 Framework

The main approach adopted here is to use benefits of both backward and forward planning. Mainly, using backward planning for long-term planning with relaxed problem and using frequent forward replanning with shorter horizon (10s) for correcting plans with higher dimensional space. The flowchart in Figure 6.10 illustrates this approach. First, route data is acquired including road slope, street geometry, speed limits, etc. Then, backward planning is employed to compute optimal velocity trajectory tree on relaxed problem assuming empty streets. Optimal trajectory is tracked by lower level controller until replanning is initiated and forward planning in MPC-manner provides adapted trajectory. Tracking continues until the goal is reached.

Strategic Planning

The strategic planning phase is executed only once at the beginning of a trip, or if the target location changes. It is based on backward dynamic programming starting from the goal state, backward in space along the route provided by route planner. In this phase, only time-invariant constraints (e.g. velocity limits), topological road profile and vehicle model are considered. The results of this phase are the optimal trajectory tree and the cost-to-go map. The initial optimal velocity trajectory can be constructed as well, by choosing a branch of the optimal trajectory tree which starts from the initial state.

Replanning triggering

The replanning procedure is repeated during the entire trip. It can be triggered by either time period, spatial length traveled, an event (e.g. detecting of other traffic participants, detecting a significant deviation from predicted situation) or some combination of these. Generally MPC approach considers defined replanning frequency (time period).

Replanning

During the replanning phase, the optimal trajectory is adjusted by considering dynamic environment, traffic rules and current driving situation. Two different methods were developed for replanning. The first method was based on Forward Dynamic Programming (section 6.2.3), but it turned out not to be very practical. The second method, which appeared to be remarkably practical, is based on hybrid A* search and motion primitives (SBOMP in section 6.2.2).

6.2.2 Search-Based Optimal Motion Planning (SBOMP)

In this section, we elaborate on the proposed Search Based Optimal Motion Planing (SBOMP) framework (presented in [7]) for fast and robust motion planning designed to facilitate automated driving. The framework allows for real-time computation even for horizons of several hundred meters and thus enabling automated driving in urban conditions. This is achieved through several features. Firstly, a convenient geometrical representation of both the search space and driving constraints from Section 5.2 enables the use of classical path planning approach. Thus, a wide variety of constraints can be tackled simultaneously (other vehicles, traffic lights, etc.). Secondly, an exact cost-to-go map, obtained by solving a relaxed problem (as shown in Section 6.1.1), is then used by A*-based algorithm with model predictive flavour (moving horizon) in order to compute the optimal motion trajectory. The algorithm takes into account both distance and time horizons. Non-holonomic motion constraints are handled by conservatively approximating lateral motion with effective hybrid discrete-linear lateral motion model of lane-change behavior (as in Section 5.1.2), so that planned lane-changes are always feasible.

Algorithm 2: A* search for horizon

input : n_{start}, Obstacles data ($\mathcal{O}$), $h(s, v)$
output: $v_{ref}, l_{ref}, t_{ref}$ *trajectory for horizon*

begin
  $n, n_r \leftarrow n_{\mathrm{I}}$ // Start pose
  CLOSED $\leftarrow \varnothing$ // list of closed nodes
  OPEN $\leftarrow n$ // list of opened nodes
  while $n \in [0, S_{\mathrm{hor}}] \times [0, T_{\mathrm{hor}}]$ ***and*** OPEN $\neq \varnothing$ ***and*** *!timeout* **do**
    CLOSED $\leftarrow$ CLOSED $\cup\, n$
    OPEN $\leftarrow$ OPEN$\backslash n$
    foreach $n' \in$ `Expand`$(n, \mathcal{O},\, h(s, v))$ **do**
      if $n' \in$ CLOSED **then**
        continue
      else if $n' \in$ OPEN **then**
        if *new* n' *is better* **then**
          $n'.parent \leftarrow n$ // update parent
        else
          continue
      else
        OPEN $\leftarrow$ OPEN $\cup\, n'$ // add to list
        if n' *closer to horizons than* n_r **then**
          $n_r \leftarrow n'$
    $n \leftarrow \arg\min n.f \in$ OPEN
  reconstruct trajectory starting from n_r backwards
  return $n_{\mathrm{I}}, \ldots, n_r\}$ // trajectory

This section is organized as follows. First, some general aspects of the framework are described, followed by the clarification of individual features like motion primitives generation, heuristic function, search horizons and other vehicle behavior prediction.

Framework

The proposed framework is based on A* search method [48], guided by an exact cost-to-go map from a relaxed problem in an MPC-like replanning scheme. Each T_{rep} seconds, replanning is triggered with the current measurement of positions and velocities of other vehicles, traffic lights timing data together with the map data. Based on measurements, the motion of other vehicles is predicted and collision-free trajectory for a defined horizon is generated.

In each replanning step, the ego vehicle trajectory is generated by concatenating motion primitives guided by a grid-like search using A*. The grid is constructed by the discretization of s, l and t from the original continuous search space definition (section 5.2.2). The search space is augmented by velocity v to provide completeness because of the longitudinal dynamics. Starting from the initial configuration, defined as the *initial node*, chosen as the first *current node*, all neighbors are determined by expanding the *current node*. The resulting *child nodes* are added to the OPEN list. If the *child node* is already in the OPEN list, and new *child node* has a lower cost, the parent of that node is updated, otherwise it is ignored. From the OPEN list, the node with the lowest cost is chosen to be the next *current node*. The procedure is repeated until one of the limits of horizons is reached, the whole graph is explored or the time limit for planning is reached. Finally, the node closest to the horizons is used to reconstruct the trajectory. The pseudocode for this procedure is presented in Algorithm 2.

To avoid rounding errors, as the expansion of node creates multiple transitions which in general do not end at grid points, the hybrid A*-inspired approach [75] is used for planning. Hybrid A* also uses the grid, but it does not round the values to the grid as it saves continuous value for the next expansion. Thus preventing the accumulation of rounding errors.

As v belongs to the discrete set of values as defined in the expansion (Algorithm 3), the hybrid A* approach is used only for t, s, l. Therefore, each *node* n contains 14 values: four indices for v, s, l, t ($n.v_k$, $n.s_k$, $n.l_k$, $n.t_k$), four indices for the *parent node* (to reconstruct trajectory), three remainders from the discretization of s, l and t ($n.s_r$, $n.l_r$, $n.t_r$), the direction of the lane-change $n.l_{dir}$, the exact cost-to-come to the node ($n.g$), and the estimated total cost of traveling from the initial node to the goal region ($n.f$). The value $n.f$ is computed as $n.g + h(n)$, where $h(n)$ is the heuristic function.

The planning clearly requires processing time. The compensation of the planning time can be achieved by introducing T_{plan}, a guaranteed upper bound on planning time. The planning is then initiated from a position where the vehicle would be after the T_{plan}. The old trajectory is executed while the new one is being prepared. Thus, the new trajectory is already planed when T_{plan} arrives. This approach has been widely used in MP for automated vehicles [155].

Motion Primitives

Algorithm 3: Expand function

input : n, Obstacles data ($\mathcal{O}$), $h(s, v)$
output: n' *array*

```
1  begin
       /* generate array n_lon of longit. variants                          */
2      v_i ← n.v_k · Δv
3      v_f ← [0 : Δv : v_max]
4      n_lon ←transitions from v_i to v_f
5      n_lon.g ← n.g + costtrans(vi, vf, t_t)
6      n_lon.f ← n_lon.g + h(s, v_f)
7      n' ← n_lon
       /* generate lateral variants and add to n'                           */
8      if mod(n.l_k, 1) ≠ 1 then
9          progress n'.l_k                          // lane change in progress
10         increase n'.f and n'.g
11     else
12         if n.l_k > 1 then
13             n_r ← n_lon                                // lane change right
14             increase n_r.l_k
15             increase n_r.f and n_r.g
16             n' ← n' ∪ n_r
17         if n.l_k < N_l then
18             n_l ← n_lon                                 // lane change left
19             decrease n_l.l_k
20             increase n_l.f and n_l.g
21             n' ← n' ∪ n_l
22     n_o ← {x ∈ n' | τ(x, O) = 1}                          // collision check
23     n' ← n'\n_o
24     return n'
```

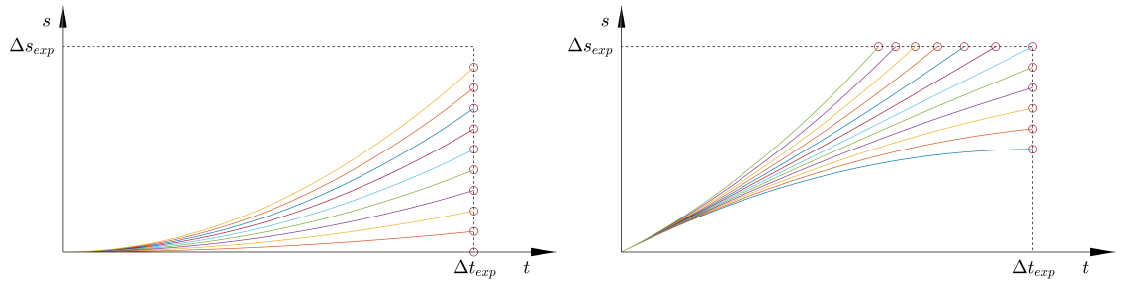

Figure 6.11: *Expanding parent node n to different child nodes n' by piecewise constant acceleration, from initial velocity $v = 0$ (left), $0 < v < v_{\max}$ (right).*

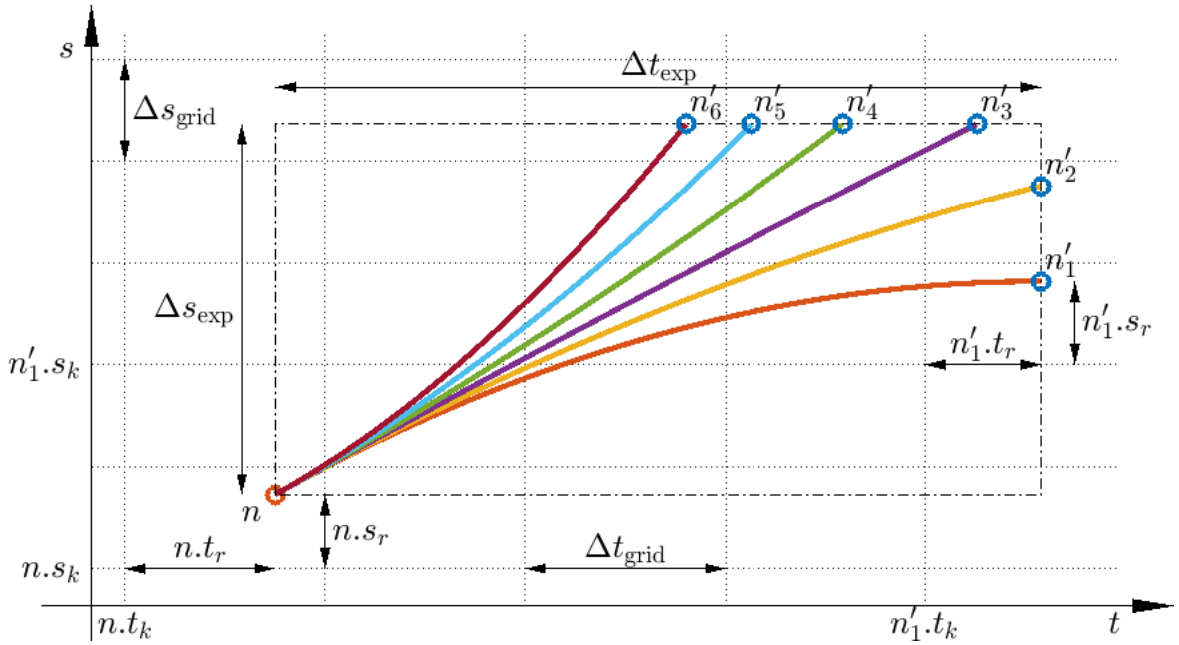

Figure 6.12: *Expanding parent node n to different child nodes n' by piecewise constant acceleration.*

To build trajectories iteratively, *nodes* are expanded and *child nodes* are generated, progressing toward the goal. From each node n, only dynamically feasible and collision-free *child nodes* n' should be generated. A single *child node* is generated for each possible combination of longitudinal and lateral motion variants. The procedure is presented in Algorithm 3. First, longitudinal motion variant are generated and then based on the state of the node n combination with lateral variants generate all possible **motion primitives**. On the end of each feasible motion primitive one *child node* n' is generated.

Longitudinal motion variants The longitudinal motion variants are generated by assuming constant accelerations from the inherited parent velocity, so that the discrete final velocities (represented by the array $v_{\min} : \Delta v : v_{\max}$) are reached at the boundaries

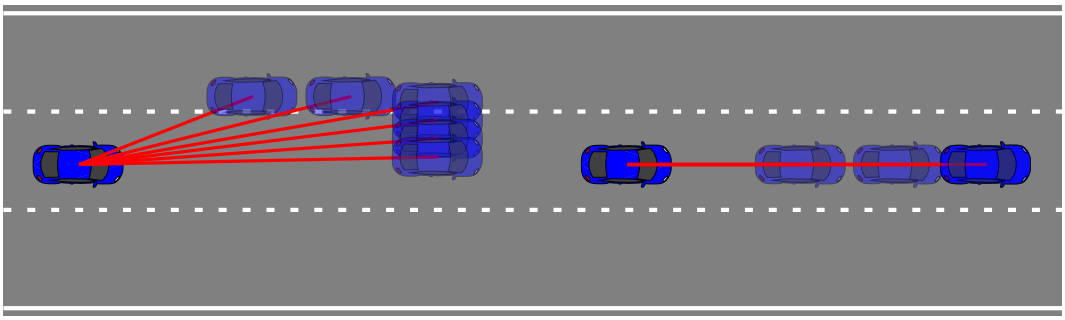

Figure 6.13: *Motion primitives for lane-change left (left) and stay in the lane (right).*

of expansion horizons. Expansion boundaries are defined by Δs_{exp} for distance s and Δt_{exp} for time t (Figure 6.11). Expansion like this enables better coverage of state space by sampled trajectories. Figure 6.11 shows expansions from different initial velocities. Since trajectories reflect the motion with the uniform acceleration, the average velocity of a specific motion variant equals $\overline{v} = (v_i + v_f)/2$. If $\overline{v} < \Delta s_{\text{exp}}/\Delta t_{\text{exp}}$, the trajectory will end on time expansion horizon $t = \Delta t_{\text{exp}}$. Otherwise, it will end on the distance expansion horizon $s = \Delta s_{\text{exp}}$. Position and time values for each variant are added to the parent node remainders $n.s_r$ and $n.t_r$. Indices $n.s_k$ and $n.t_k$ are increased by the quotient of division of resulting values with discretization steps of the grid Δs_{grid} and Δt_{grid}, and the new remainders are computed (Figure 6.12). For each *child node* from the array n', costs are computed as well. Cost-to-come is inherited from the parent and then increased by the cost of transition. Cost-to-go is provided by the heuristic function explained in the following subsection. The compliance with the vehicle's internal constraints (e.g. maximum acceleration) is checked and the nodes that violate these constraints are removed.

Lateral motion variants Generated longitudinal motion variants are then used for lateral motion expansion. If the parent node is in the middle of the lane (l is the integer), variants for possible lane change right and left, beside staying in the lane are generated. Generated variants are tripled, one set of longitudinal variants for each. The values $n'.l_k$ and $n'.l_r$ are increased or decreased for the lane changes left or right respectively. They are modified by the value t_{exp}/T_{LC} based on the travel time of the particular variant, and the desired lane change time T_{LC}. If the parent node is already in the process of lane change, only the progress is increased without generating other lateral motion variants (ln 9, Alg. 3). Figure 6.13 illustrates final motion primitives generated based on longitudinal and lateral motion variants. On the left side motion primitives for lane-change left are shown,a and on the right for keeping the lane. As it can be seen, final states have different longitudinal and lateral positions. Longitudinal position is determined based on distance traveled in longitudinal motion variant and lateral position based on travel time of longitudinal motion variant and T_{LC}.

Collision check Finally, compliance with obstacles such as other vehicles, traffic lights, etc. (section 5.2) is checked and all child nodes and motion variants that are

not collision-free are removed (ln 22, 23, Alg. 2).

Heuristic function

As it was mentioned earlier, the heuristic function $h(n)$ (cost-to-go) is used to estimate the cost needed to travel from some node n to the goal state n_{G}. As it is shown in [48], if the heuristic function is underestimating the real cost-to-go, A* search provides the optimal trajectory. In this framework, the exact cost-to-go map resulting from the backward search solving relaxed problem is used as a heuristic function. The backward cost-to-go computation phase is executed only once at the beginning of the trip, or if the goal is changed. The computation is performed by using backward DP, starting from the goal state (s and v), backward in s, as it was shown in section 6.1.1. In this phase, only time invariant constraints are considered (e.g. velocity limits) with topological road profile and the vehicle model. Time-varying constraints are not considered. The resulting cost-to-go map is an admissible heuristic, as other vehicles can prohibit certain regions of the state space, which may only increase the cost to travel from the initial state to the goal region. This is valid if platooning effects are neglected, as platooning can potentially reduce the airdrag effect (which is considered in the initial cost-to-go computation), and decrease the cost of travel. For compact vehicles, this effect is usually negligible. Cost-to-go map $h(s, v)$ depends only on s and v. Using a similar approach as in Forward Dynamic Programming (section 6.2.3), the resulting cost-to-go map is applied based on child node's s and v values, while neglecting t and l values.

Search horizons

As it was noted in Algorithm 2, the planing is performed until any of the trajectories reaches time (T_{hor}) or distance (S_{hor}) search horizon. Slower driving trajectories will reach the time, while faster trajectories will reach the distance horizon. Thus, the unnecessary planning can be avoided. If only one is chosen (e.g., T_{hor}) other one could adopt a large value (s for fast trajectories and vice versa). The search horizons should not be confused with local expansion horizons, which uses a similar principle, but represents atomic motion segments when building the whole trajectory.

Other vehicle motion prediction

Though it is required for prediction of potential collisions, the perfect knowledge of the future motion of other vehicles is not available, in principle. A naive way to predict the motion is to assume that the vehicles will continue to drive with their current velocity and stay in the current lane. On the other hand, a motion planning framework should provide collision-free plans even if trajectories deviate from the predicted one and the environment perception system introduces estimation errors. Therefore, safety buffers are used to increase obstacle regions, and frequent replanning is executed to adapt to changes. The approach introduced in this framework is an additional safety mechanism that ensures a collision-free plan even for the worst case error regarding the relative

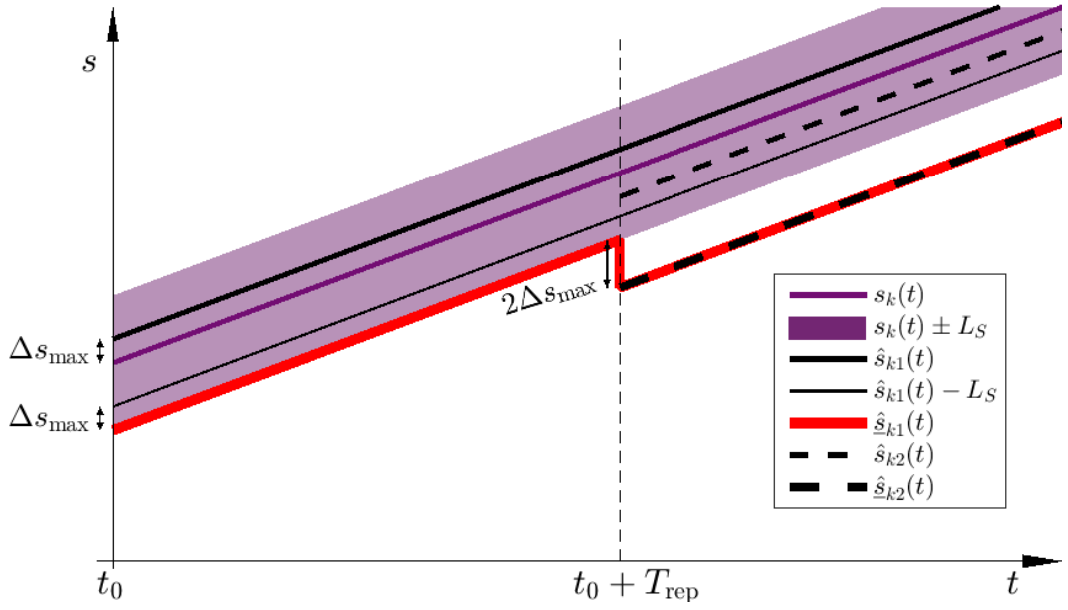

Figure 6.14: *Predicting movement of other vehicle - linearization.*

distance estimation. This is provided by adding a step-like safety buffer to the obstacle prediction. The lower and upper bound of the vehicle obstacle can be defined as:

$$\underline{\hat{s}}_k(t) = \hat{s}_k(t_0) + \hat{v}_k(t_0)(t - t_0) - L_S - s_b(t), \tag{6.26a}$$

$$\overline{\hat{s}}_k(t) = \hat{s}_k(t_0) + \hat{v}_k(t_0)(t - t_0) + L_S + s_b(t), \tag{6.26b}$$

$$s_b(t) = \begin{cases} \Delta s_{\max}, & t_0 \leqslant t < t_0 + T_{\text{rep}}, \\ 3 \cdot \Delta s_{\max}, & t_0 + T_{\text{rep}} \leqslant t \leqslant t_0 + T_{\text{hor}}, \end{cases} \tag{6.26c}$$

where $\Delta s_{\max}$ is the maximum error of the vehicle relative position estimation. The safety buffer s_b is increased after T_{rep} (the next replanning instance) to maintain robustness, so that in the next re-planning instance, the vehicle always starts from the position that is collision-free according to a new safety buffer. This is visualized in the Figure 6.14, showing the worst case scenario. The estimation error is such that in the first planning instance, $\Delta s_{\max}$ is positive, while in the second instance, it is negative. It can be seen that the safety buffer from the first planning instance ensures that the trajectory is outside of obstacle area for the time interval $[t_0, t_0 + T_{\text{rep}})$ and the trajectory is outside of the safety buffer from the second planning instance for the time interval $[t_0 + T_{\text{rep}}, t_0 + 2 \cdot T_{\text{rep}})$. This safety buffer provides a partial robustness for deviations from the predicted trajectory as well, but no guarantees can be provided.

6.2.3 Forward Dynamic Programming

The other approach for replanning is based on Forward Dynamic Programming. The main idea here, as introduced in [4], is based on the combination of the advantages of

forward and backward Dynamic Programming. The planning problem is addressed by splitting into strategic planning and situation dependent replanning. The results once computed by backward programming, in strategic planning phase, are continuously reused for the ongoing replanning during driving. Replanning is done using forward planning from the current state, for a certain prediction horizon into the future, and merged with previously obtained results from backward planning. During replanning, dynamic constraints and additional states (e.g. lanes, travel time) are considered. Thus, the whole trip is taken into consideration along with dynamic constraints, but only a planning for a defined horizon is performed. This promises benefits of both, forward planning the entire trip (globally optimal solution) and adaptability of MPC with significant reduction in computational effort.

Situation-dependent replanning

During the replanning phase, the optimal trajectory is adjusted by taking into consideration dynamic constraints. The adjustment is done by replanning the optimal trajectory in an efficient way by reusing the cost-to-go map and optimal velocity trajectory tree. The replanning is performed with forward Dynamic Programming starting from the current state for a defined prediction horizon in the future. Several safety factors such as maximum time of overtaking execution (constraint on minimum velocity difference), minimum distance from the leading vehicle and clearance needed for lane changing are considered as constraints in this phase, as defined in section 5.2. The future movement of other vehicles is estimated using a simple prediction model, that assumes that the leading vehicle will continue moving with constant velocity and that it will slow down (to avoid collision) if it is behind ego vehicle (after being overtaken). More sophisticated models of the leading vehicle velocity that may depend on space, time and the controlled vehicle can be also included.

An illustrative example is shown in Figure 6.15. Blue lines represent the optimal trajectory tree, constructed in the strategic planning phase. The green line is the initial optimal trajectory, which also results from the strategic planning phase. The vehicle tracks the optimal trajectory until situation-dependent replanning is triggered. By using forward planning, a forward optimal trajectory tree starting from the current state is constructed (red lines) considering dynamic constraints such as other traffic participants, traffic lights, etc.

The merging of two trajectory trees (forward and backward) is performed at the end of the horizon. Cost-to-come values at the possible joining nodes are summed with cost-to-go values at these nodes from backward planning. Thus, combined costs of trajectories, partially planned forward and partially planned backward, are computed. The minimum among these costs is chosen, defining an optimal joining node, and the new optimal trajectory (solid black node) passing trough that node. Starting from this node backwards, towards the actual state, a new optimal trajectory can be constructed iteratively. The new optimal trajectory does not have to be constructed forward to the final state, if replanning is frequent. Therefore, unnecessary computations can be avoided. In Figure 6.15, additional state dimensions (e.g. time t, lane l) are not shown for forward planning, although they are considered. This is only to keep figure

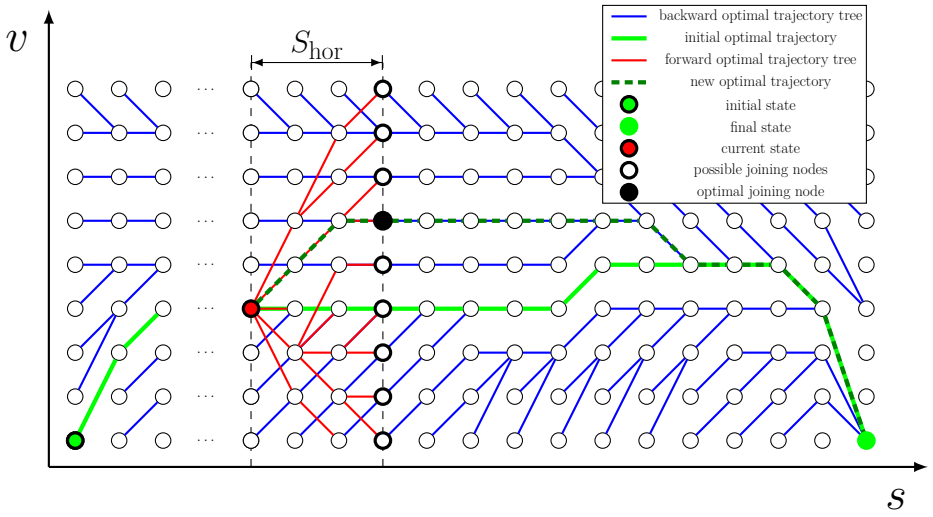

Figure 6.15: *Forward Dynamic Programming-based replanning.*

simple and focus on the working principle. During merging, for choosing the optimal joining node, an appropriate node from cost-to-go is chosen based on the velocity v and position s values only. So it is shared for all time t and lane l variants.

Replanning triggering

In this Forward Dynamic Programming implementation, the replanning triggering is determined by the distance traveled. This means that the replanning frequency is not constant, as it depends on driving velocity. For applications such as driving on highway, this might not be a problem as there are no large fluctuations of velocity and velocity is generally high. However, it is problematic for low velocity driving. Frequent replanning is important if the environment is highly dynamic and the prediction of other traffic participant motion is not precise.

Prediction horizon

As spatial discretization is used as a basis for motion planning, the prediction horizon is defined by the length at which a forward planning is executed. It is important to choose a proper horizon length as a so-called short sightedness can appear otherwise. Several constraints such as clearance needed for changing a lane can block lane changing if horizon length is too short. In general, the horizon should be as long as computational resources enable it, but keeping in mind that accuracy of motion prediction of other traffic participants decreases with time.

6.2.4 Remarks on Automated Driving

To solve Automated Driving use-case and enable driving in all driving scenarios, achieving SuperHuman performance, two different solutions were developed. The first solution was based on Forward Dynamic programming (Section 6.2.3). However, this

was abandoned as it showed to be very impractical. The other solution , SBOMP (Section 6.2.2), was based on hybrid A* equipped with motion primitives. An efficient framework was developed using solution from the relaxed problem solved with discrete Dynamic Programming as a heuristic to guide the search. Search like this showed to be very efficient as it does not search the whole space and deals well with continuous dynamics due to motion primitives and hybrid A* approach. The SBOMP framework showed to be scalable to consider more detailed vehicle model and applicable to Performance Driving use-case.

6.3 Performance Autonomous Driving

The tackled problem in this use-case is minimum lap-time driving on empty tracks in low friction conditions, i.e. gravel road. The road, on the other hand, is assumed to be flat, with static road-tire characteristic and can have arbitrary shape (varying curvature radius, mixed right and left curves, etc.) with constant width. As planning motion trajectories for continuous driving in this kind of conditions can be considered as a combinatorial optimization problem, heuristic search methods like A* can be effectively used for automated optimal trajectory generation. In this work, an A* search-based approach was used, that is a novel approach in this field. The presented A* search-based planner is a modified version of the one illustrated in Section 6.2.2, where a rather simple vehicle model was used to generate vehicle trajectories in complex urban driving scenarios. Here, more complex vehicle model was used. The space of the possible trajectories is explored by expanding different combinations of motion primitives in a systematic way, guided by a heuristic function. Motion primitives are generated using two different vehicle models. A bicycle model is used for small side-slip angle operations (i.e. entry and exit maneuvers and close-to-straight driving) and a full nonlinear vehicle model for steady state cornering maneuvers. Such automated motion primitives generation enables to generate arbitrary trajectories, not limited to just one curve as in other approaches in this field.

6.3.1 Framework

The proposed Motion planning framework , as presented in [10], is based on the A* search method [48], guided by an heuristic function in an MPC-like replanning scheme. After each time interval T_{rep}, replanning is triggered from the current vehicle state, together with the information about the drivable road ahead.

The trajectory is generated by a grid-like search using A* search (see Algorithm 4), in the same way as in SBOMP from section 6.2.2. Each *node* n contains 20 values: 6 indexes for each state in $\mathbf{x}$ ($n.x_k$, $n.y_k$, $n.\psi_k$, etc.), 6 indexes for the *parent node* (to reconstruct the trajectory), six remainders from the discretization of states ($n.x_r$, $n.y_r$, $n.\psi_r$, etc.), the exact *cost-to-come* to the node ($n.g$), and the estimated total cost of traveling from the initial node to the goal region ($n.f$). The value $n.f$ is computed as $n.g + h(n)$, where $h(n)$ is the heuristic function.

Algorithm 4: A* Heuristic Search for a horizon

input : n_{I}, Obstacles data ($\mathcal{O}$), $h(n)$

```
1 begin
2     n ← n_I, CLOSED ← ∅, OPEN ← n                                  // initialization
3     while n.k ⩽ k_hor and OPEN ≠ ∅ and OPEN.size() ⩽ N_timeout do
4         n ← Select(OPEN)
5         (n', n_C) ← Expand(n, Children, ColCheck)
6         CLOSED ← CLOSED ∪ n ∪ n_C, OPEN ← OPEN\n ∪ n'
7     return Path ()
```

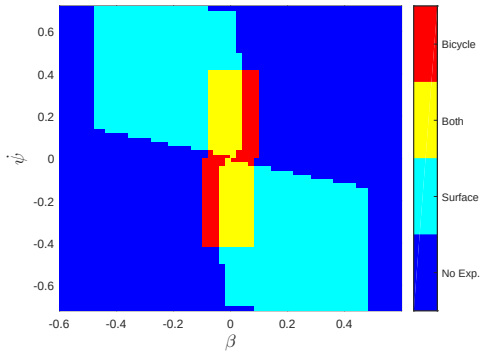

Figure 6.16: *$\beta - \dot{\psi}$ map representing the expansion modes depending on initial states.*

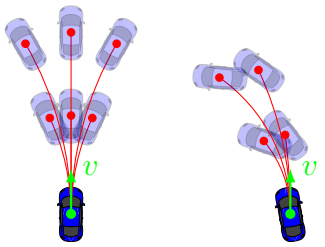

Figure 6.17: *Motion primitives for performance driving using semi-linearized bicycle model (left) and ESM (right).*

6.3.2 Motion Primitives

To build trajectories iteratively, *nodes* are expanded and *child nodes* (final states on the motion primitives) are generated, progressing towards the goal. From each node n, only dynamically feasible and collision-free *child nodes* n' should be generated. As mentioned before, child nodes (motion primitives) are generated based on two models, semi-linearized bicycle model for a close-to-straight driving and vehicle-equilibrium-states during cornering as illustrated in Figure 6.17 (Section 5.1.2).

During cornering, the child nodes (motion primitives) are generated based on the steady state dynamic states S_{ss}, as explained in Section 5.1.2. Based on the current steady state S_{ss}, several reachable steady states are sampled, and model (5.4) is used to simulate the evolution of other states (x, y, ψ) during the linear transition to sampled new steady states. New steady states are obtained by sampling the $\beta - \dot{\psi}$ space around the current value $(\beta_0, \dot{\psi}_0)$, with the density of the samples decreasing as

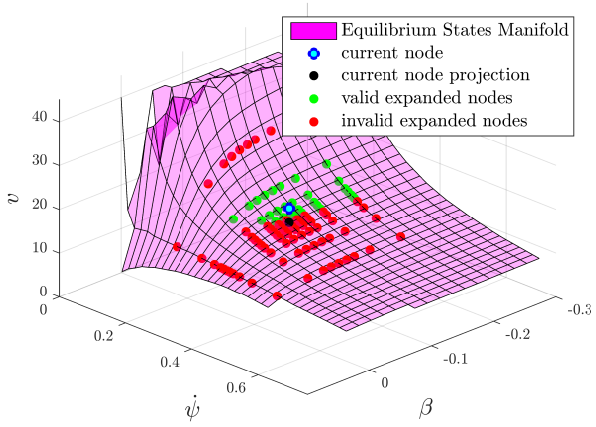

Figure 6.18: *Expanding parent node n to different child nodes n' on the ESM.*

the distance from the current position increases (see Figure 6.18). Different sampling approaches can be also employed. The number of evaluated samples is a tuning parameter, that impacts considerably the performance of the search, as a trade-off between computation time and sufficient space exploration is needed. Since (5.10) assumes that the rate of change of states is equal to zero, in order to generate trajectories which keep the vehicle on the road by moving to other steady states, this constraint must be "softened".

In order to avoid generating and propagating an excessive amount of branches, before the nodes which are in collision are removed, the following rules are considered:

- only equilibrium points defined within the surface in Figure 5.4 are considered, so that the minimum reachable curvature radius is $R_{c,min} = 10m$;
- the (small) portion of curve such that $\beta \cdot \dot{\psi} > 0$ is neglected, since in general equilibrium points in which β and $\dot{\psi}$ share the same sign are associated with low velocity conditions;
- a maximum velocity deviation Δv between two successive nodes is defined, such that $\frac{\Delta v}{Ts} < a_{\max}$, where $a_{\max}$ is the estimated maximum deceleration allowed on the given road surface.

Nodes are always expanded by "exploring" the neighboring states of the ESM. However, as illustrated in Figure 6.16, when the current conditions are close enough to the origin of the $\beta - \dot{\psi}$ plane, i.e. for $|\beta| < \beta_{lin}$ and $|\dot{\psi}| < \dot{\psi}_{lin}$, also dynamic states and trajectories generated according to a semi-linearized bicycle model are considered (Section 5.1.2). In order to generate motion primitives, the steering wheel angle δ and the rear wheels slip λ are varied within the defined ranges.

Figure 6.17 illustrated motion primitives for two models. On the left, motion primitives are generated using semi-linearized bicycle model with 2 variations in the rear wheels slip λ and 3 variations in steering wheel angle δ. In total 6 motion primitives are generated. On the left, 4 motion primitives generated by sampling in ESM are illustrated. In practice, many more motion primitives are generated (about 100).

6.3.3 Heuristic function

The heuristic function $h(n)$ is used to estimate the cost needed to travel from some node n to the goal state (*cost-to-go*). As it is shown in [48], if the heuristic function is underestimating the exact cost to go, the A* search provides the optimal trajectory. For the shortest path search, the usual heuristic function is the Euclidean distance. On the other hand, to find the minimum lap time, the heuristic should estimate the distance which the vehicle can travel from the current node during the defined time horizon. It is optimistic to assume that the vehicle accelerates (with maximum acceleration) in the direction of the road central line until it reaches the maximum velocity, and then maintains it for the rest of time horizon. Based on this velocity trajectory, the maximum travel distance can be computed and used as heuristic.

Sacrificing optimality, in order to bias expansions towards the preferred motion and a better robustness, the heuristic function is augmented considering, among others:

- a "dynamic states evolution" cost, which helps limiting the rate of change of the references v, β, $\dot{\psi}$, in order to obtain smooth trajectories and facilitate state tracking;
- penalization for trajectories approaching the road boundary;
- penalization of the nodes with less siblings, thus biasing the search to avoid regions where only few trajectories are feasible.

6.3.4 Remarks on Performance Driving

In this section, a novel application of SBOMP for performance driving is presented. The proposed method extends drift-like driving from a steady state drifting on a single curve to a continuous driving on the road, effectively entering and exiting drifting maneuvers and switching between right and left turns. The proposed method assumes that the vehicle parameters and the road surface properties are known to a certain degree, which allows to define a set of steady-state cornering maneuvers.

6.4 Planning and Learning

Planning and Learning are complementary approaches. Planning relies on deliberative reasoning about the current state and sequence of future reachable states to solve the problem. Learning, on the other hand, is focused on improving system performance based on experience or available data. Learning to improve the performance of planning based on experience in similar, previously solved problems, is ongoing research. One approach is to learn Value function (cost-to-go), that can be used as heuristics for speeding up search-based planning. Existing approaches in this direction use the results of the previous search for learning the heuristics.

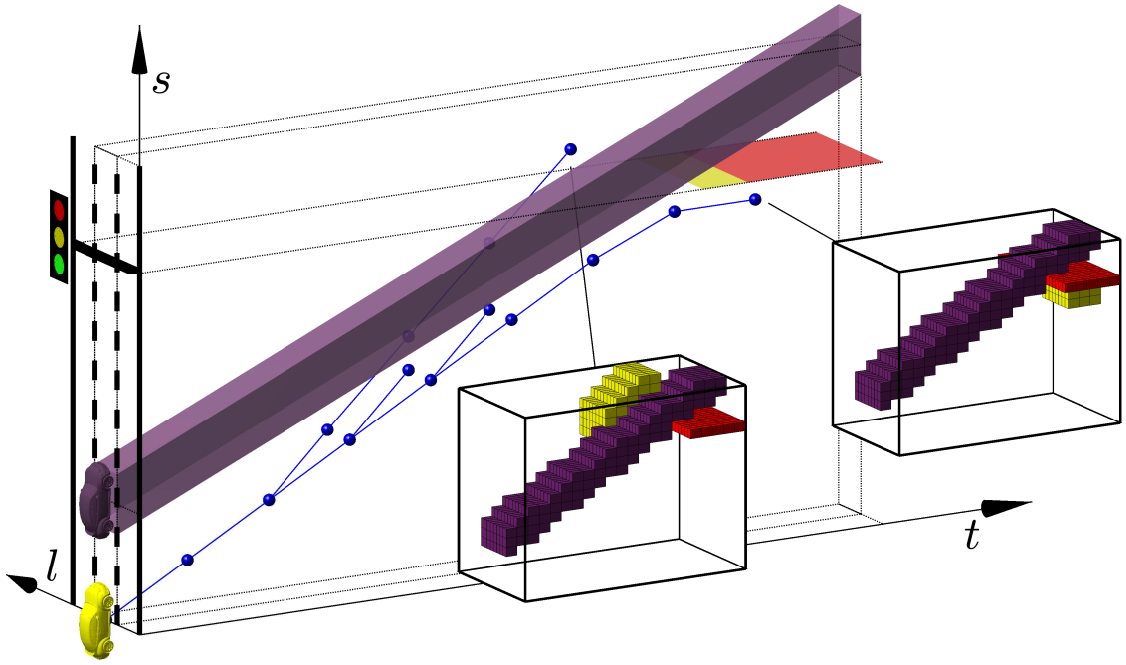

Figure 6.19: *MP for ego vehicle (yellow) in scenario with receding vehicle (purple) and traffic light. Situation representation for 2 nodes.*

6.4.1 ML-based heuristic function for Automated Driving

Previous work, presented in Section 6.2.2, used Search-Based Optimal Motion Planning framework (SBOMP) with numerical or model-based heuristics. Heuristics did not consider dynamic obstacles. Optimal solution was still guaranteed since dynamic obstacles can only increase the cost. However, significant variations in the search efficiency are observed depending whether dynamic obstacles are present or not. To improve the performance consistency for achieving real-time implementation, machine learning (ML)-based heuristic was introduced [8], which takes into account full driving situation including dynamic obstacles.

The original SBOMP for optimal motion planning is enhanced by ML heuristic h_{ML}, bounded by an admissible heuristic h_{adm} for providing guarantees on sub-optimality. Proposed h_{ML} takes as input discretized SLT 3D structure (dSLT) representing node n and a driving situation (obstacles $\mathcal{O}$) and returns as a result a scalar value representing an estimated cost to reach horizon limits from that node. Using modified SBOMP and admissible heuristic (h_{adm}), dataset ($\mathcal{D}$) of exact input-output data points is generated. Dataset is then used for training of the ML heuristic. In principle, this approach differs from reinforcement learning since it is supervised and from imitation learning since the exact optimal solution is used instead of expert demonstrations.

Situation representation

Node n and driving situation are represented by a discretized SLT (dSLT) 3D structure with longitudinal distance s, lane l and time t dimensions (Figure 6.19). 3D structure has $N_{ks\text{hor}} \times N_{kt\text{hor}} \times N_{kdl}$ cells. Each cell represents a part of the search space and can be free or occupied by some obstacle or ego vehicle. Obstacles are uniquely marked for several types: other vehicles, traffic lights, forbidden lane change, etc. (as shown in Section 5.2). Parent node n is represented by virtual obstacle as if the vehicle is continuing to move with velocity $n.v$ from it's initial position. The advantage of this formulation is that the representation keeps a consistent 3D size for each driving situation regardless of the number of obstacles and it is computed only once per replanning instance. This structure can be useful for other tasks that require as input full driving situation with the temporal evolution (so-called time-series in ML). Such tasks might be behavior prediction, scenario classification, etc. In those cases, it is more useful to keep trajectory history instead of prediction.

Dataset generation

For generation of dataset $\mathcal{D}$, the SBOMP framework was modified to enable theoretically inexhaustible generation from different scenarios and initial conditions as it is shown in Algorithm 5. Contrary to the original SBOMP, where the goal was to find only one collision-free trajectory, for dataset generation the goal is to generate as many different data points as possible. Therefore, the search is executed not only until some trajectory reaches horizon, but until all nodes in OPEN list are explored. Trajectory branches are constructed backward from each node n_h at the end of horizons, starting from nodes which reached horizon first. For each node n on the branch, corresponding 3D structure and cost are stored in dataset $\mathcal{D}$. Cost is computed by subtracting $g(n)$ value from the cost of the final node $g(n_h)$. It can be noted that computed cost represents *cost-to-horizon* and not *cost-to-go* anymore, but this should not affect the results of the search. The remaining nodes, not belonging to the branches that reach horizons, lead to the collision, and thus are assigned with infinite cost. These nodes partially resemble inevitable collision state [234].

Using ML heuristic function

Proposed ML heuristic h_{ML} is bounded by an admissible heuristic h_{adm} as shown in (6.27), so that guarantees on sub-optimality can be provided. In this way, heuristic is ε-admissible so the solution is always maximum ε times greater than the optimal solution [216]. Values of ε closer to 1 guarantee smaller deviation from optimal solution but reduce computational performance. Alternative approach would be to use Multi-Heuristic A* Search (MHA*) [235].

$$h_{\text{adm}} \leqslant h_{ML} \leqslant \varepsilon \cdot h_{\text{adm}}, \quad \varepsilon \geqslant 1. \tag{6.27}$$

Algorithm 5: modified SBOMP for dataset generation

```
input : k_scen                                              // Number of initial poses
output: D                                                   // Dataset
1  begin
2      D ← ∅                                                // Dataset
3      foreach k ∈ [1, k_scen] do
4          n, O ← ScenarioGen()                             // generated driving scenario
5          CLOSED ← ∅                                       // list of closed nodes
6          OPEN ← n                                         // list of opened nodes
7          while OPEN ≠ ∅ do
8              CLOSED ← CLOSED ∪ n                          // Exploring
9              OPEN ← OPEN\n
10             foreach n' ∈ Expand(n, O, h(s, v)) do
11                 if n' ∈ CLOSED then
12                     continue
13                 else if n' ∈ OPEN then
14                     if new n' is better then
15                         n'.parent ← n                    // update parent
16                     else
17                         continue
18                 else
19                     OPEN ← OPEN ∪ n'                     // add to list
20             n ← arg min n.f ∈ OPEN
21         foreach n_h ∈ CLOSED | n_h.s = S_hor ∨ n_h.t = T_hor do
22             n ← n_h                                      // end of one traj. branch
23             while n ∈ CLOSED do
24                 D ← D ∪ {n, O(n), (n_h.g − n.g)}
25                 CLOSED ← CLOSED\n
26                 n ← n.parent
27         foreach n ∈ CLOSED do
28             D ← D ∪ {n, O(n), inf}                       // dead-end trajectoriess
29     return D
```

Appropriate ML architecture for this problem have yet to be developed and learnability validated. However, a more general approach for model exploration and value function learning was proposed and successfully trained on Grid World domain and Shortest Path Planning problem.

6.4.2 Prolonged Heuristic Search-based model exploration

This section presents a search-inspired approach to systematic model exploration for value function learning, introduced in [9]. It is based on prolonged and backward heuristic search that does not stop when a plan is available, but rather continues such that wider region around the optimal path is explored (not only resulting optimal path). The search is flipped backward so that all explored nodes can be used as they all are connected to the goal. The presented approach uses an existing admissible heuristic function (h_{adm}) and a known model (or simulator) $\mathcal{M}$ of the system to a generate dataset $\mathcal{D}$ of exact state-cost data points. The dataset can be used for supervised learning of the value function. The learned value function is then used as heuristic function h_{ML} in the search, bounded by the admissible heuristic to provide guarantees on sub-optimality.

This approach to exploration can be also used within Model Based Reinforcement

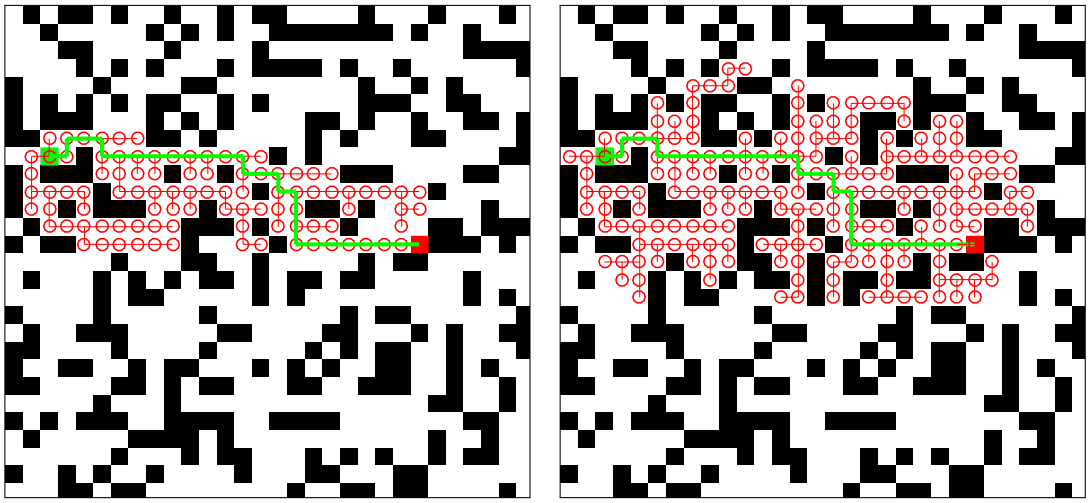

Figure 6.20: *Nodes explored for Vanilla Shortest Path problem (left) and Prolonged Heuristic Search (right).*

Learning framework. When model or simulator is available, it enables systematic, theoretically inexhaustible data generation from different scenarios and initial conditions, therefore using computational resources offline to have faster planning online (when it is more critical). In Reinforcement Learning setup, this approach can be also used in cases when it is important to focus the exploration to certain parts of the state space to reduce uncertainty in value function approximation.

Dataset generation

The dataset $\mathcal{D}$ is similar to dataset in Section 6.4.1, but it is extended also with the goal state. It consists of data points which carry information about the scenario (obstacles, initial and goal state) and current state together with the corresponding cost-to-go (i.e. from current state to the goal state). For the generation of dataset $\mathcal{D}$, planning algorithms can be used to generate data points with the exact cost-to-go. In the vanilla Shortest Path Planning problem (SP), the goal is to find only one collision-free path (i.e. from initial to the goal state), so planning is stopped when the goal state is reached. As the goal state is reached only from one node (and each node has only one parent), the exact cost-to-go can be computed only for nodes on the optimal path. Contrary to SP, in the dataset generation, the objective is to generate as many different paths toward the goal (data points) as it is possible. One approach is to use Backward Dynamic Programming (or Dijkstra's algorithm). However, this would explore the whole search space which is not practical in higher dimensional problems.

We proposed a novel exploration approach based on Prolonged Heuristic Search (PHS), to generate the dataset $\mathcal{D}$, as it is shown in Algorithm 6. In this approach, search is done *backwards* from n_{G} so that all explored nodes can be used in dataset as they contain exact cost to n_{G}. Additionally, as the region of higher interest is in the neighborhood of optimal path, the search is not stopped when the initial node n_{I} is reached by some path (as in the SP problem), but *prolonged* until CLOSED list gets k_{expl}

Algorithm 6: Prolonged Heuristic Search (PHS) for dataset generation

input : $k_{\text{scen}}, k_{\text{expl}}, \mathcal{M}, h_{\text{adm}}(\cdot)$
output: $\mathcal{D}$ // Dataset

```
begin
    D ← ∅                                                        // Dataset
    foreach k ∈ [1, k_scen] do
        ⟨n_I, n_G, O⟩ ← NewScenario()                            // New scenario
        // Search-based Exploration
        ⟨OPEN, CLOSED⟩ ← PHS(⟨n_I, n_G, O⟩, k_expl, M, h_adm(·))
        // Extracting data from the search
        foreach n ∈ CLOSED ∪ OPEN do
            D ← D ∪ (n, O, g(n))                                 // Data points
            CLOSED ∪ OPEN ← CLOSED ∪ OPEN\n
    return D
```

times more nodes. This prolongation assures that more nodes in the neighborhood of the optimal path are explored. Data points are constructed such that, for each node n in the OPEN and CLOSED lists, the corresponding scenario structure (grid) and cost-to-go are stored in dataset $\mathcal{D}$. Cost-to-go from node n to goal node n_{G} is actually cost-to-come $g(n)$ in backward search. In this way, paths do not have to be reconstructed and all expanded nodes are used in the dataset.

Value Function Learning

Learning of the Value Function h_{ML}, in this approach, is a supervised learning problem (regression). The proposed h_{ML} takes as input an image representing the current and the goal nodes (n_{I}, n_{G}) and a situation (obstacles $\mathcal{O}$), as can be seen in Figure 6.20. (grayscale part), and returns as a result a scalar value representing an estimated cost to reach the goal from that node.

As it is preferred that the heuristic function underestimates the exact cost (admissibility), a non-symmetric loss function can be used. Asymmetry can be introduced by augmenting Mean Square Error Loss function as:

$$e_i = y_i - \hat{y}_i, \tag{6.28}$$

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} e_i^2 \cdot (\text{sign}(e_i) + a)^2, \tag{6.29}$$

with parameter $a < 0$ to emphasize the penalty for positive errors e.

6.4.3 Remarks on Planning and Learning

The presented approach offers the possibility to effectively include Machine Learning into a deterministic planning framework, promising significant performance improvements manifested in a reduced number of explored nodes compared to those obtained using admissible heuristic (h_{adm}) while keeping guarantees on sub-optimality of the solution. The proposed Prolonged Heuristic Search (PHS) approach to exploration uses the maximum of invested computational resources in planning as all expanded nodes in planning are used for learning.

6.5 Chapter Conclusion

It this chapter, Motion Planning solutions developed in the framework of this PhD work are presented. As it is shown, solutions are based on graph search methods (Dynamic Programming and A* search). First, different custom solutions for specific problems (i.e. energy efficient driving) were presented. Later, forward and backward planning solutions were combined into effective framework SBOMP, that can deal with complex problems (i.e. Automated Driving, Performance Driving). As a highlight, Search-based Optimal Motion Planning (SBOMP) was presented. It is an resolution complete optimal kinodynamic motion planning algorithm based on heuristic search. It relies on motion primitives and hybrid A* search-based solution.

Developed methods were applied to three targeted use-cases. In Energy-efficient Driving use-case, Dynamic Programming and A* search-based custom solutions were provided to facilitate planning of trajectories for the full trip spanning even to tens of kilometers. To improve search efficiency, some theoretical results were developed, such as the concept of Optimal Cruising Velocity together with analytical expression for its computation. It was then used in heuristic function to improve the search. The developed SBOMP framework was used for Automated Driving use-case together with scalable environment representation (as shown in section 5.2) and effective vehicle model (section 5.1.2). This approach can be applied all foreseeable scenarios on structured roads (urban, suburban and highway). It can incorporate traffic rules, forbidden lanes, other vehicles and complex dynamic environment in general. Trajectory is planned such that both fast and slow trajectories are generated making it capable to drive in full velocity range, including the full stop. If there is information about the traffic light timing, it can also be incorporated. The driving aims at using the least energy, effectively extending eco-driving approaches to multi-lane automated driving in complex environments. SBOMP was further extended with more complex model and used for performance driving. The developed solution is capable of driving on roads with arbitrary planar geometry (i.e. varying curvature radius, mixed right and left curves).

Finally, an approach of integrating ML into heuristic search was presented and method for using previous heuristic search results to improve future planning. That approach was further extended and generalized to a novel approach to model exploration (PHS) for value function learning, that can be also used in Model-based Reinforcement Learning setups.

7

Simulation

Imagination will often carry us to worlds that never were. But without it we go nowhere.

Carl Sagan

This chapter presents the results of the simulation study performed to verify the applicability of developed methods to targeted use-cases and benchmark benefits of developed planners compared to the other state-of-the-art approaches. For each use-case, appropriate scenarios are created and solved using proposed methods to verify important aspects of the method. For benchmarking, several comparisons to related state-of-the-art approaches are performed, mainly based on energy-efficiency and travel time or computational efficiency. The main benefits come from extending Energy-efficient Driving (eco-driving) to Automated Driving, that considers energy-efficiency and utilizes connectivity. This extends usability of eco-driving to a wide range of scenarios, including multi-lane driving in the presence of traffic lights.

The first section presents the applicability of proposed method to three use-cases. The second section presents the benchmarks with different approaches. The third section presents the results of experimental analysis of computational complexity. Finally, the fourth section presents the results on the planning and learning approach, including dataset generation, value function learning and verifying example.

7.1 Use cases

In this section, proposed planners are verified in several scenarios, including driving on the open road with speed limits for Energy-efficient Driving use-case; several urban, rural and highway scenarios for Automated Driving use-case; and driving on the circuit in slippery road conditions for Performance Driving use-case.

7.1.1 Energy-efficient Driving (eco-driving)

The proposed planners for Energy-efficient Driving use-case, as described in section 6.1, are based on Dynamic Programming and A* search. They are implemented in Matlab and used in Simulink as a Matlab compiled function. This planner enables the

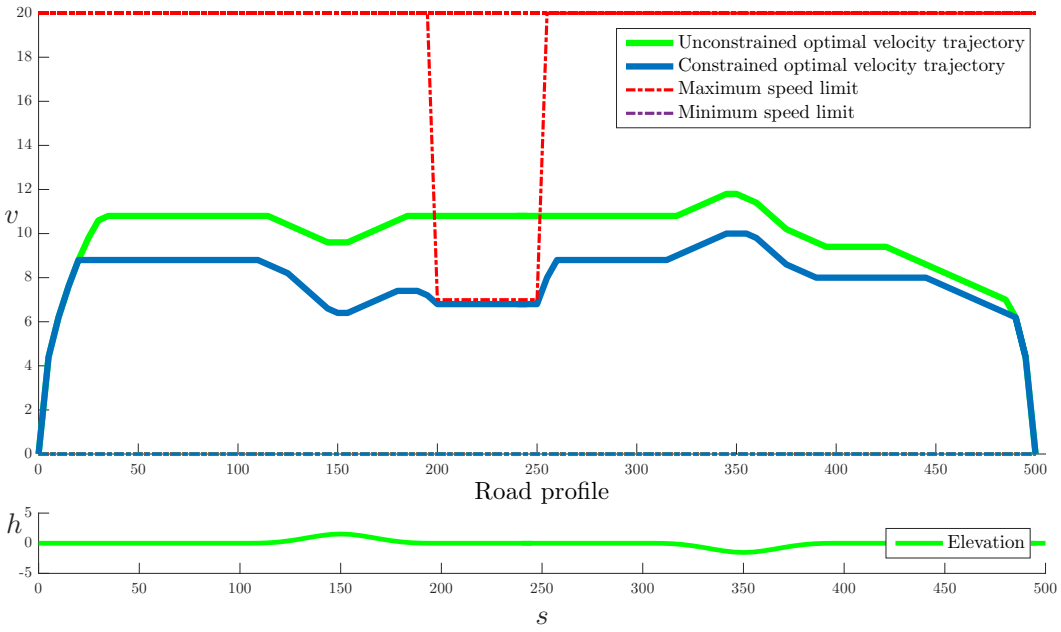

Figure 7.1: *Energy efficient driving with speed limits.*

ecoCC function, meaning it considers energy-efficiency while controlling velocity, but does not consider other traffic. The velocity trajectory is planned for the whole trip.

For the purpose of verifying the ecoCC planner, driving on an empty, hilly road with speed limits is considered. An artificial road segment, 500 m long, with a hill and a valley, is used. The hill is 1.6 m high, with a maximum slope of 6% and the valley has a symmetrical shape. There is one speed limit on the segment (from 200 m to 250 m), limiting maximum speed to 7 m/s. No other vehicles are present on the segment. Initial and final velocity for ego-vehicle is set as standstill ($v_i = 0$ m/s, $v_f = 0$ m/s).

Planners consider road slope, air resistance, roll coefficient, auxiliary power, power-train efficiency and speed limits to find an optimal velocity trajectory (in the sense of energy consumption) for the given trip. The grid is formed by discretizing the space with discretization step size for distance $\Delta s = 1$ m and for velocity $\Delta v = 0.03$ m/s.

Table 7.1: *Driving and planning performance for Energy-efficient Driving.*

method	energy used [kJ]	travel time[s]	nodes expl.	comp. time [s]
DP	210.75	66.66	15251	0.067
A*	210.75	66.66	7683	10.21

Figure 7.1 visualizes the optimal velocity trajectory for this trip without considering speed limit (green) and when considering speed limit (blue). The same trajectories

are obtained by both approaches, Dynamic Programming and A* search. As it can be observed, when the speed limit is not considered, resulting trajectory consists of a constant speed driving at about 15 m/s (Optimal Cruising Velocity (OCV) for this vehicle and air drag resistance), on a flat road, slowing down uphill, and speeding up downhill. The trip starts with maximum acceleration to reach OCV and finishes with coasting down and maximum braking at the end. When the speed limit is considered, trajectory is modified to satisfy the limit. Table 7.1 presents the results of the experiment. As it can be observed, the number of nodes explored is smaller when A* search is used, compared to Dynamic Programming. While Dynamic Programing explores the whole search-space, A* search is guided by heuristic to a more promising region. The benefit is not drastic in this problem (2D problem with limited state-space), but it becomes drastically larger in higher dimensional problems that are not limited, i.e. Performance Driving use-case. Although the number of explored nodes is smaller, the computation time is higher. This is due to efficient exploitation of parallel matrix computations in Dynamic Programming (possible in 2D problem) and interpreted language use for this A* search implementation. More efficient A* search implementation is used in SBOMP. In further work, for Optimal Velocity trajectory tree, Dynamic Programing is used.

7.1.2 Automated Driving

The proposed planner for Automated Driving use-case, as described in section 6.2.2, is based on SBOMP framework. The SBOMP framework was implemented in Matlab and used in Simulink as a Matlab compiled function together with PreScan, where a detailed vehicle model is simulated. Even though this implementation is not optimal, the results meet real-time requirements. For example, the proposed framework computes the plan with a horizons of $S_{\text{hor}} = 100$ m and $T_{\text{hor}} = 10$ s, discretization of $\Delta v = 1$ m/s and $\Delta s = 10$ m, in **116 ms** in average. The computation time limits the maximum frequency of replanning to approx. 10 Hz. For verification purposes, several scenarios are presented, ranging from urban, rural and highway driving. The SBOMP enables ecoAD function, indicating that energy-efficiency is used as an objective and it performs fully Automated Driving (Level 5).

The planner uses same planning parameters setup for all scenarios. Only the parameter for P_{tot} is changed to facilitate appropriate cruising velocity, depending on the environment (highway, rural, urban). Presented computation time is based on Matlab implementation without compilation, as it allows a better handling of different scenarios and runtime analysis.

Full stop

To verify full velocity range operation and ability of ecoAD to provide slow trajectories, the full stop scenario is simulated. Ego-vehicle has to fully stop, due to the road blockage by other stopped vehicle. In the scenario, single lane, 500 m long road, is used. The speed limit of 60 km/h $\approx$ 16.67 m/s is active on the road. There is a leading vehicle $\mathcal{V}_{\text{LV}}$ stopped at the distance of 300 m. The ego-vehicle drives with an

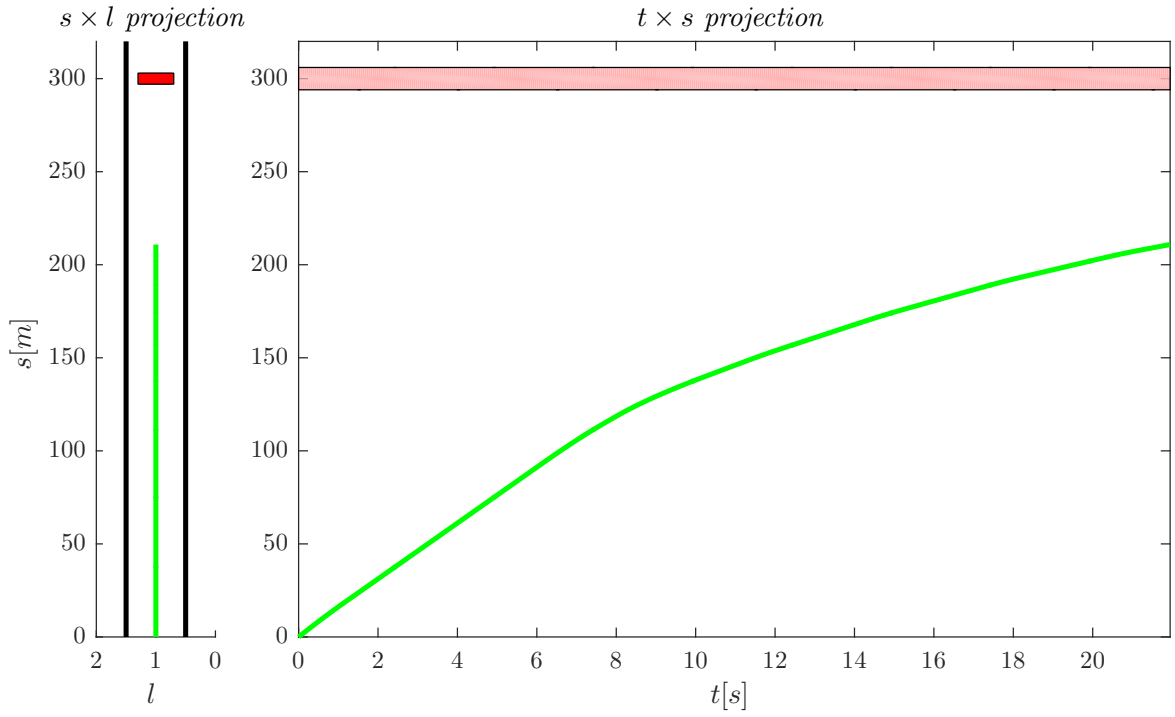

Figure 7.2: *Results from full stop scenario.*

initial velocity of 15 m/s, approaching the vehicle which is at standstill. Planner uses ecoAD's setting for urban driving, where optimal cruising velocity is 16.87 m/s.

Table 7.2: *Driving and planning performance for full stop scenario.*

energy used [kJ]	travel time[s]	nodes expl.	comp. time [s]
-38.41	21.95	438.54 ± 238.51	0.143 ± 0.076

Figure 7.2 visualizes the resulting vehicle trajectory. On the right $t \times s$ projection, red area represents the region occupied by the other vehicle $\mathcal{V}_{\text{LV}}$, green line represents ego-vehicle driving trajectory. The ego-vehicle manages to fully stop without colliding with the vehicle or violating traffic rules. Table 7.2 presents the planner performance in terms of planning time and explored nodes, and driving performance in terms of energy consumption and travel time. As it can be observed in the table, energy used is negative, since the vehicle is braking from some initial velocity and kinetic energy can be recuperated.

Vehicle following

This scenario considers driving behind the other vehicle. As energy-efficiency is considered, this provides so-called ecoACC function, indicating that the function is per-

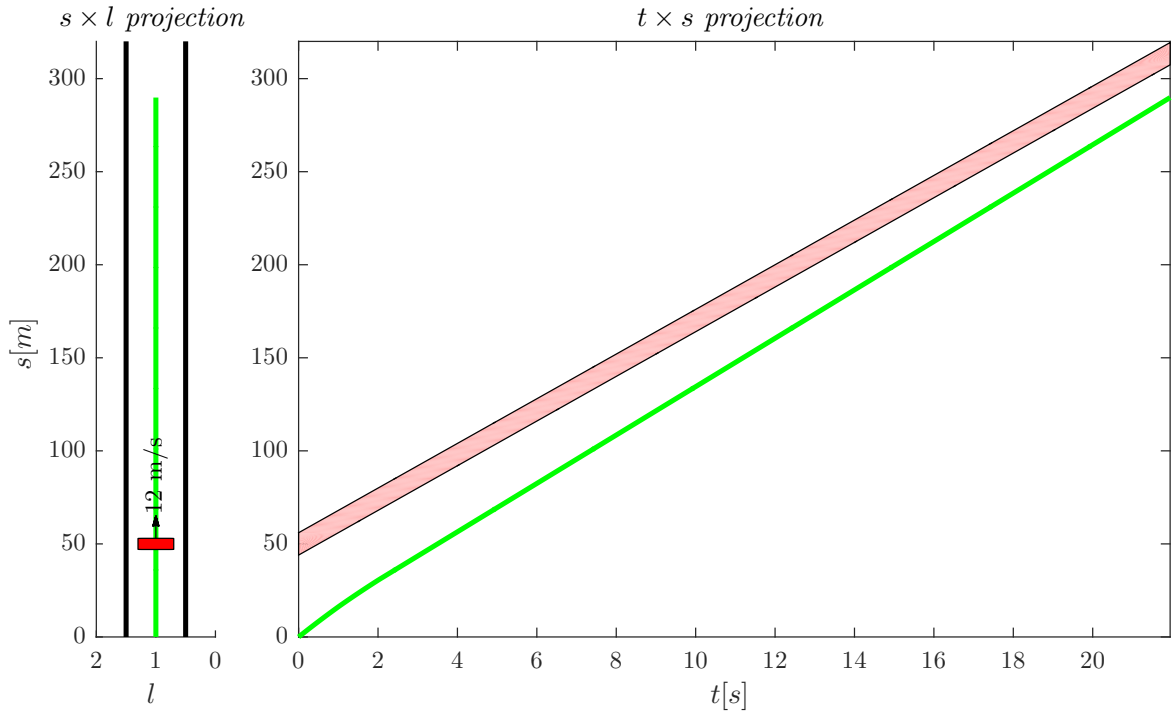

Figure 7.3: *Results from vehicle following scenario.*

forming Adaptive Cruise Control (ACC). In the scenario, the same single lane road, 500 m long, is used. There is a leading vehicle $\mathcal{V}_{\text{LV}}$ driving in front, at the initial distance of 50 m, with velocity of 12 m/s. The ego-vehicle drives with an initial velocity of 16.87 m/s approaching the vehicle and continuing to drive behind it.

Table 7.3: *Driving and planning performance for vehicle following scenario.*

energy used [kJ]	travel time[s]	nodes expl.	comp. time [s]
27.29	21.95	141.54 ± 83.96	0.049 ± 0.028

Figure 7.3 visualizes the driving trajectory from the scenario execution. On the right $t \times s$ projection, red area represents the region occupied by the other vehicle, green line represents ego-vehicle driving trajectory. The ego-vehicle manages to smoothly follow the leading vehicle $\mathcal{V}_{\text{LV}}$ without colliding or violating traffic rules. Table 7.3 presents the planner performance in terms of planning time and explored nodes, and driving performance in terms of energy consumption and travel time. It can be observed that the computational performance is better as this is generally easier problem than the full stop.

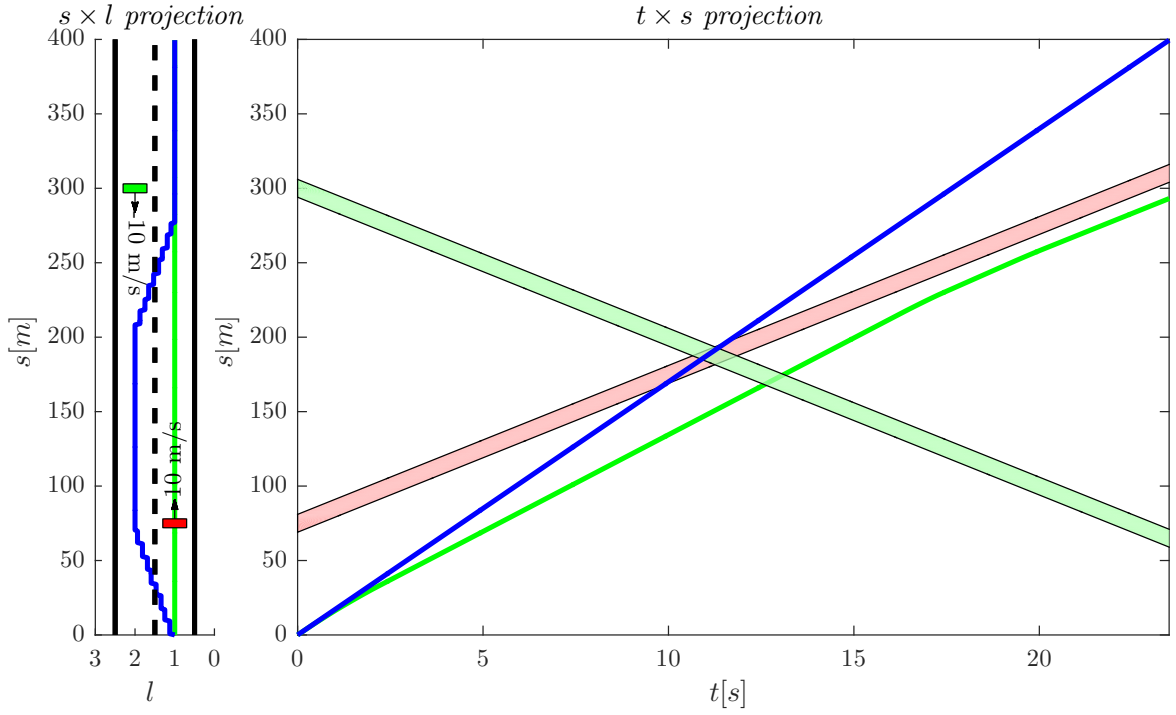

Figure 7.4: *Results from rural overtaking scenario.*

Rural Overtaking with oncoming traffic

This scenario deals with rural driving on roads with traffic in both directions. The scenario is focused on overtaking while there is an oncoming traffic from the other direction, which is considered to be a challenging task. The ego-vehicle is driving on a two-lane road with traffic in both directions, with one leading vehicle $\mathcal{V}_{\text{LV}}$ driving in front, and other vehicle $\mathcal{V}_{\text{OV}}$ approaching in other lane, from the opposite direction. Speed limit of the road is 80 km/h $\approx$ 22.22 m/s. The leading vehicle $\mathcal{V}_{\text{LV}}$ is initially located 75 m ahead of the ego-vehicle and it is driving with constant velocity of 10 m/s. On the other lane, other vehicle $\mathcal{V}_{\text{OV}}$ is driving in opposite direction with a constant velocity of 10 m/s. There are two scenario variants, variant 1 with the initial position of vehicle $\mathcal{V}_{\text{OV}}$ at 500 m and variant 2 with the initial position at 300 m. The ego-vehicle is driving with an initial velocity of 16.87 m/s, using SBOMP, to achieve ecoAD. Same planner settings are used as in previous scenarios, with ecoAD setup for rural road (Optimal Cruising Velocity is 16.87 m/s).

Figure 7.4 visualizes the results from the scenario execution. On the right $t \times s$ projection, red area represents the region occupied by the leading vehicle $\mathcal{V}_{\text{LV}}$, green region starting from 300 m represents $\mathcal{V}_{\text{OV}}$ in variant 2 and green region showing after 10 s represents $\mathcal{V}_{\text{OV}}$ in variant 1. It can be observed that in variant 2, two other vehicles meet around 200 m at about 11 s. The ego-vehicle manages to safely drive without colliding with the vehicle or violating traffic rules in both variants. In the variant 1, ego-vehicle decides to overtake $\mathcal{V}_{\text{LV}}$ before $\mathcal{V}_{\text{OV}}$ approaches (blue line). In the second variant, ego-vehicle decides to wait until $\mathcal{V}_{\text{OV}}$ passes, as there is not enough

Table 7.4: *Driving and planning performance for rural overtaking scenario.*

	energy used [kJ]	travel time[s]	nodes expl.	comp. time [s]
Var 1	125.6	23.5	249.67±286.85	0.56 ± 0.55
Var 2	36.87	34.15	1133.08 ± 1305.55	2.50 ± 2.35

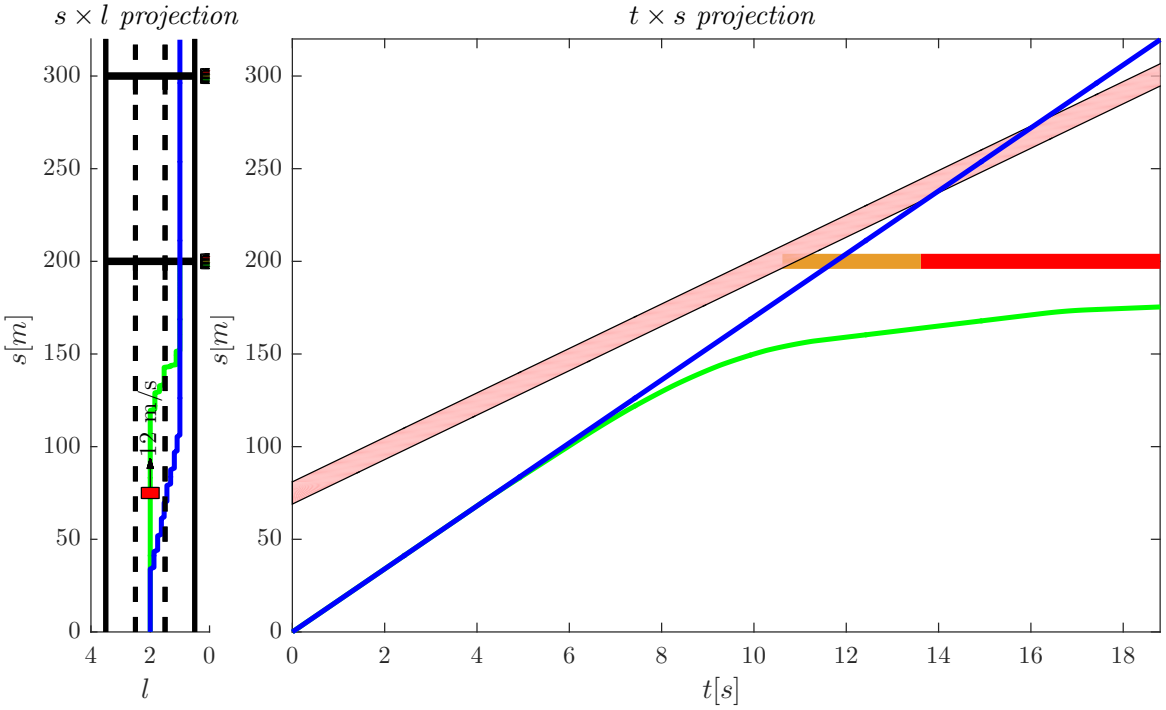

Figure 7.5: *Results from urban scenario with traffic light.*

room or the energy lost for accelerating to overtake is too large, so there is no benefit compared to waiting (green line). Table 7.5 presents the planner performance in terms of planning time and explored nodes, and the driving performance in terms of energy consumption and travel time. Direct comparison based on energy consumption cannot be made, as the final velocity is different. However, it can be observed that in variant 2 travel time is higher, as the vehicle is stuck behind the other vehicle. Additionally, large variations in computation time can be observed in variant 2, as slowly driving leading vehicle $\mathcal{V}_{\text{LV}}$ motivates planner to look for a possible overtaking, that eventually shows to be not feasible, and represents an dead-end in the search space.

Urban multi-lane driving with Traffic Lights

This scenario deals with the urban driving on multi-lane roads with the traffic in same direction. Scenario is focused on utilizing the communication with traffic lights,

and knowledge about traffic light signal phase and timing (SPaT). The ego-vehicle is driving on the road, 1000 m long, with three lanes. There is a leading vehicle $\mathcal{V}_{\text{LV}}$ diving in the middle lane and a traffic light in front of the ego-vehicle. There are two variants of the scenario. At the beginning of the scenario, the traffic light is 200 m (variant 1) and 300 m (variant 2) in front of the ego-vehicle. The leading vehicle $\mathcal{V}_{\text{LV}}$ drives with velocity of 12 m/s and traffic light turns red just after $\mathcal{V}_{\text{LV}}$ passes it. The same scenario is visualized earlier in Figure 5.9. Speed limit of 60 km/h is active on the track. The ego-vehicle starts the scenario in the middle lane behind $\mathcal{V}_{\text{LV}}$, with an initial velocity 16.87 m/s. Same planner settings are used as in the previous scenarios, with ecoAD setup for urban road (Optimal Cruising Velocity is 16.87 m/s).

Table 7.5: *Driving and planning performance for urban scenario with traffic light.*

	energy used [kJ]	travel time[s]	nodes expl.	comp. time [s]
Var 1	-78.28	19.95	2462 ± 169.05	9.42 ± 0.52
Var 2	94.07	17.6	597.56 ± 594.58	2.63 ± 2.55

Figure 7.5 visualizes vehicle driving trajectory from the scenario execution. The ego-vehicle manages to safely drive without colliding with the vehicle or violating traffic rules. In the variant 2, the ego-vehicle decides to overtake $\mathcal{V}_{\text{LV}}$ to catch the green light (blue line). In the variant 1, ego-vehicle decides to stop at the red light, as there is not enough room or the energy lost for accelerating to overtake is too large, so there is no benefit compared to waiting, (green line). In the figure blue line seems as if the vehicle passes trough the yellow light at 200 m, but blue line actually represents variant 2 where the traffic light is located at 300 m and this is the traffic light from variant 1. Table 7.5 presents the planner performance in terms of planning time and explored nodes, and driving performance in terms of energy consumption and travel time. As it can be observed in the table, energy used is negative for variant 1, as the vehicle is braking from some initial velocity and energy can be recuperated. Additionally, large variations in computation time can be observed as traffic light is blocking the optimal driving. The planner has to find a trajectory that ends at time horizon, which requires to explore a large part of search space.

Conclusion on Automated Driving use-case

Presented scenarios in Automated Driving use-case cover wide situations, from urban, rural and highway driving. Planner based on SBOMP managed to provide appropriate vehicle trajectories, and to control the vehicle achieving the ecoAD function. Even several challenging scenarios (i.e. overtaking with oncoming traffic) are successfully executed. Planner successfully considers speed limits, other vehicles, forbidden lane-change, blocked lanes and traffic lights.

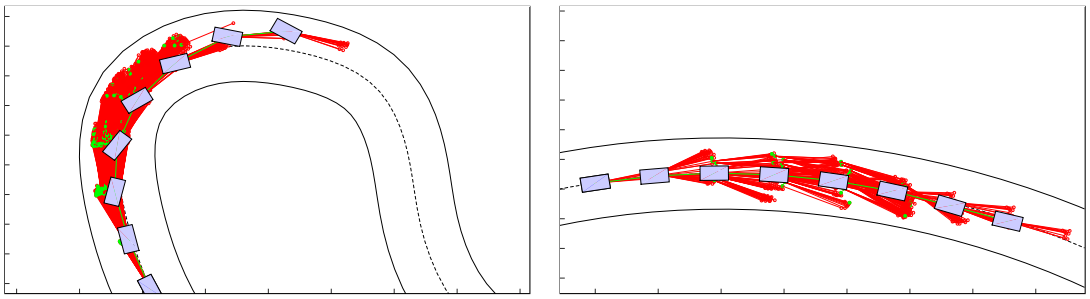

Figure 7.6: *Graphical representation of the trajectories exploration in U-turn (left) and wide turn (right).*

7.1.3 Performance Autonomous Driving

Presented SBOMP framework was adapted for Performance Driving use-case and also implemented in the Matlab/SIMULINK environment. As the focus is on efficient trajectory generation, the planner is verified assuming perfect actuation, i.e. the actual vehicle dynamical states/positions match the ones planned at the previous iteration.

For verification purpose, an artificial mixed circuit was used, characterized by slippery conditions (gravel), which contains several road sections of varying curvature radii, as it can be seen in Figure 7.9.

Proposed planner manages to find the appropriate vehicle trajectory for driving on the track. Figure 7.6 visualizes state exploration for finding the optimal trajectory in case of a U-turn and of a wider curve. The explored branches are represented by the red links, and the closed nodes are marked as green. The light-blue car frames represent the optimal vehicle states.

In Figure 7.7, several frames of the same maneuver are shown (the top left turn in the track illustrated in Figure 7.9). From these, it is possible to grasp how at each iteration the optimal trajectory is re-computed based on the current position. Given the nature of the receding horizon approach, it is not guaranteed (nor preferred) that all or part of the previously computed trajectory are kept in the next iteration. In fact, while in the first step the trajectory "dangerously" approaches the side of the road, in the next two steps, the trajectory is incrementally improved, thanks to the fact that the exploration of such portion of the track is now being evaluated.

The dynamical states, which represent the output of the trajectory generation, are visualized in Figure 7.8. One can see how the generated references are varied smoothly, in particular in terms of v and β, which are the quantities characterized by slower actuation dynamics. Moreover, it is possible to clearly distinguish 4 intervals in which the optimal generated maneuver is a "drift" one with $\beta > 0.4$ rad. These same intervals can be recognized in Figure 7.9, where the overall trajectory on the considered 10 m-wide track can be evaluated.

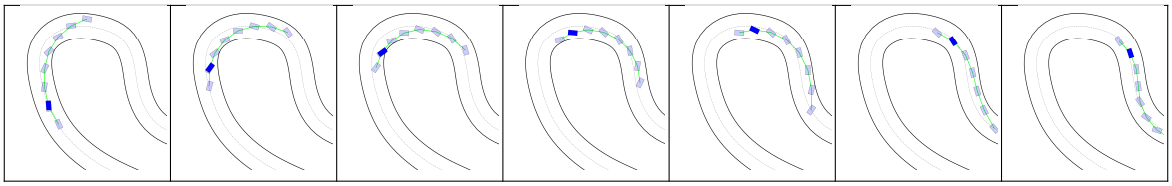

Figure 7.7: *U-turn maneuver: consecutive frames.*

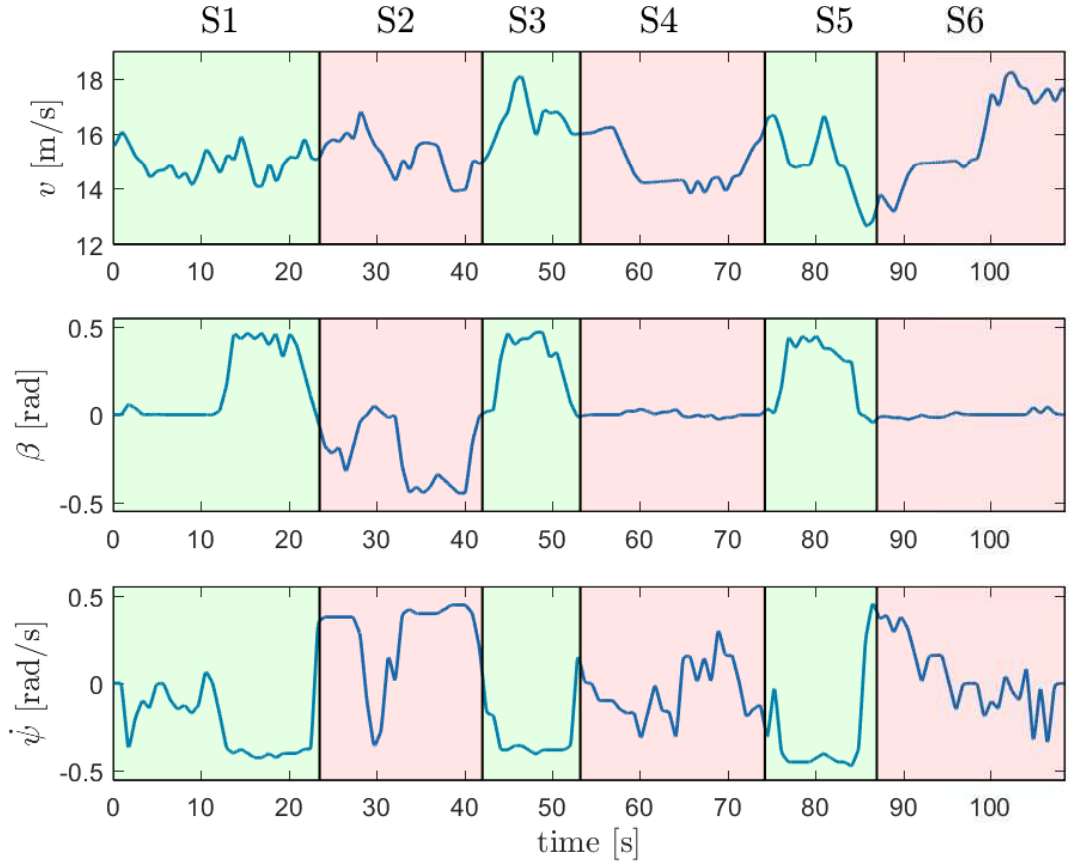

Figure 7.8: *Reference dynamical states over the full test-circuit.*

Conclusion on Performance Driving

In this section, SBOMP is used to generate dynamically feasible trajectories for performance driving on a slippery surface. The proposed method extends drift-like driving from a steady state drifting on single curve to a continuous driving on the road effectively entering and exiting drifting maneuvers and switching between right and left turns. The proposed method assumes that the vehicle parameters and the road surface properties are known to a certain degree, which allows to define a set of steady-state cornering maneuvers. The method is evaluated on a mixed circuit characterized by slippery conditions (gravel), which contains several road sections of varying curvature radii R_c. In several instances, due to the particular road surface considered, the optimal selected trajectory involves drifting, which in certain conditions ensures the maximum lateral acceleration. Such a result demonstrates the capability of the

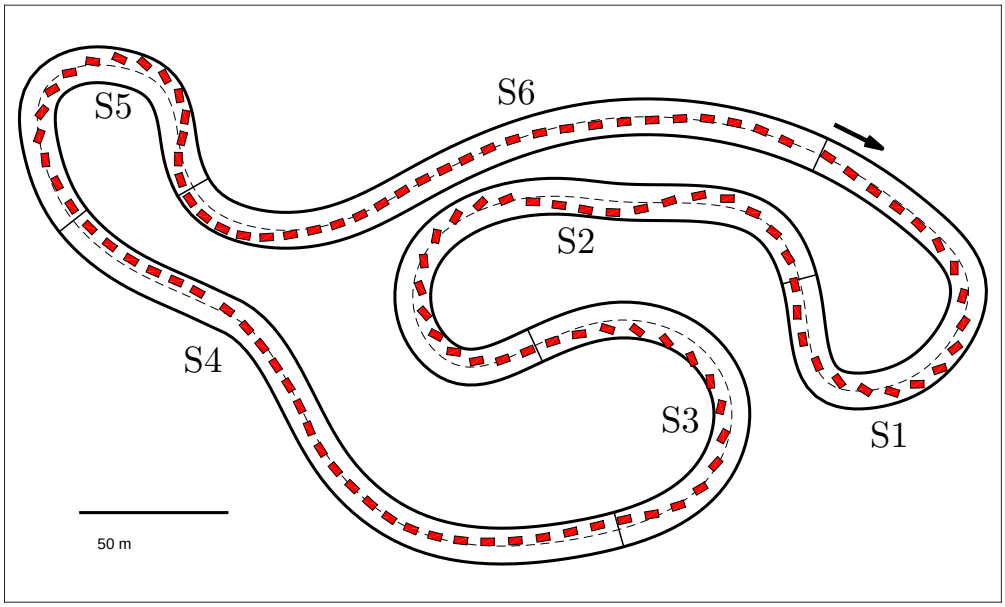

Figure 7.9: *Obtained driving trajectory over the full test-circuit.*

proposed SBOMP to generate feasible sub-optimal trajectories on slippery conditions, while considering a limited prediction horizon. Moreover, when considering U-turns with curvature radius as tight as 15 m, such trajectories are comparable in shape to the ones obtained when the full segment is optimized in order to find the minimum time optimal maneuver, as in e.g. [180].

7.2 Benchmarking

In this section, comparisons to other state-of-the-art approaches are presented in order to highlight the fundamental contributions and advantages of the proposed approach. Discussion of the resulting driving behavior is based on the vehicle trajectories presented in distance-velocity $s - v$, time-distance $t - s$ and distance-lane $s - l$ plots. Appropriate tables show the differences in energy usage and travel time (for driving performance), and computation time and numbers of node explored (for computational performance).

7.2.1 Energy-efficient Driving

Benchmarks in Energy-efficient Driving are focused on benchmarking driving performance benefits of using ecoCC compared to constant velocity driving and computational benefits of using introduced heuristic function based on Optimal Cruising Velocity.

Energy consumption: Constant velocity (CC) vs. optimized velocity trajectory (ecoCC)

To highlight the benefits of eco-driving, driving with optimal velocity trajectory using ecoCC is compared with constant velocity driving (i.e. using Cruise Control (CC)). The example simulation is setup as follows. The road as in section 7.1.1 is used. Three different velocity set points (7 m/s, 9 m/s, 11 m/s) are used. Vehicle starts and finishes with zero velocity, standstill. In contrast to the constant velocity driving, optimal driving using ecoCC, as described in section 6.1, considers road slope, air resistance, roll coefficient, auxiliary power consumption, and powertrain efficiency. An (in the sense of energy usage) optimal velocity trajectory is generated for the given travel.

Table 7.6: *Comparison of ecoCC with CC regarding energy consumption and travel time.*

Ego velocity [m/s]	energy used [kJ]	diff. [%]	travel time [s]	diff. [%]
ecoCC	213.87	0	59.25	0
const 9 m/s	231.68	+8.3	58.49	-1.28
const 11 m/s	237.98	+11.3	49.06	-17.20
const 7 m/s	237.97	+11.3	73.75	+24.47

As it can be observed from Figure 7.10, ecoCC adapts velocity (shown in green) to the oncoming road geometry. Other trajectories keep constant velocity. Table 7.6 summarizes the results from Figure 7.10. It reveals that ecoCC planner had better performance in regards to the energy consumption, compared to the constant velocity approaches approach. Concluding this example, an improvement in energy consumption above 8% can be observed, while prolongation of the trip is only about 1.3%. Interestingly, for lower velocity driving, improvements in both energy and travel time can be observed. This is due to the considered auxiliary power, which is usually omitted or included in fuel-map in combustion engines.

Search efficiency: Novel MB heuristic vs. State-of-the-art heuristic function

For a computational performance benchmark of A* search for Energy-efficient Driving use-case, three methods are compared: DP, A* search with state-of-the-art heuristic and A* search with the proposed heuristic. Methods are used to plan the optimal velocity trajectory for driving on a 1 km long segment of the A9 highway, in the vicinity of Graz, Austria. All methods result in the exact same optimal velocity trajectory (Figure 7.11), which verifies implementations of both algorithms (DP and A*), and the admissibility of both heuristics. Total energy needed for this trip was computed as $422,8$ kJ. A comparison between state-of-the-art heuristic and proposed heuristic is made based on the error of the estimation and search efficiency represented

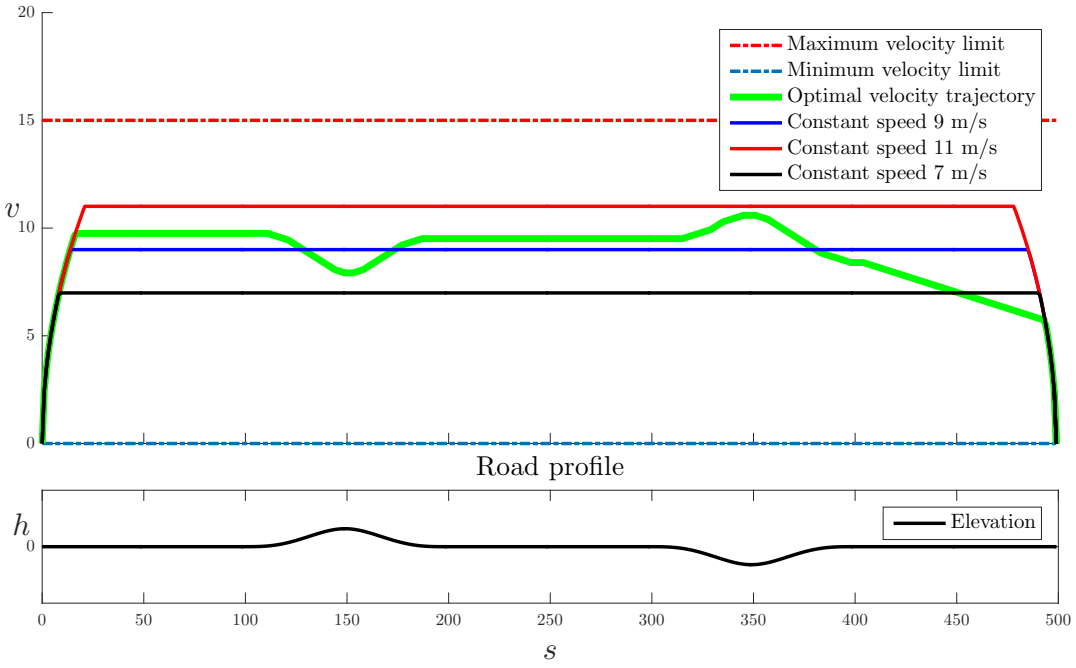

Figure 7.10: *Benchmarking ecoCC and constant speed driving.*

by the number of explored nodes. DP provides the exact *cost-to-go* computation, that is used as a reference for computation of the estimation error.

Table 7.7: *Comparison of different heuristic functions regarding computational performance.*

	DP	A* with h_{SoA}	A* with h_{pro}
nodes expl.	50200	41125	25052
average error [kJ]	0	-84,2	-15,2
minimum error [kJ]	0	-173,5	-37,6
maximum error [kJ]	0	0	0

The results are shown in Table 7.7 along with the results from DP. The first column contains the results of using DP. As DP provides the exact *cost-to-go* errors are zero. The second and the third column contain results of A* search with state-of-the-art heuristics (6.24) and the proposed heuristics (6.25) respectively. As it is shown in Table 7.7, the average error improved almost **5 times** and number of explored nodes almost **2 times** when the proposed heuristic (h_{pro}) is used, compared to the state-of-the-art heuristic (h_{SoA}). As it can be seen, the error is always negative, which means that the heuristics always underestimates the *cost-to-go*. A*, in general, examines a smaller number of nodes than DP, and it further decreases as the precision of the heuristics increases. Figure 7.11 presents the optimal trajectory and space searched

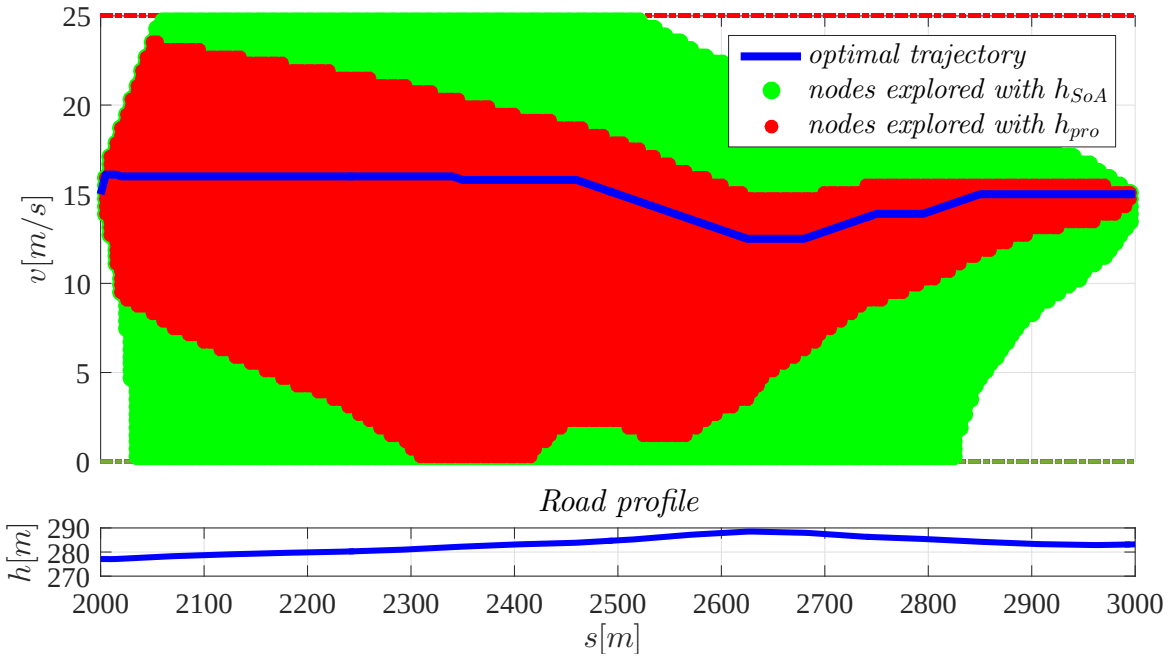

Figure 7.11: *Space searched by using different heuristics.*

by state-of-the-art and proposed heuristics from the same simulation. It is important to note that, the number of explored nodes for DP depends on the maximum velocity used; and for A* it does not.

7.2.2 Automated Driving

Benchmarks in Automated Driving mainly focus on quantifying the benefit of extending eco-driving to automated driving and considering multilane driving and traffic light in optimal motion planning. The planner used for automated driving enables ecoAD functionality but can also be reduced to other functionalities such as ecoCC, ecoACC and Traffic Light Assistant (TLA). In this section, ecoAD is compared to other variant.

Proposed ecoAD vs. state-of-the-art overtaking approach

In this subsection, the proposed approach (ecoAD) is compared to the existing optimal overtaking approach, that exists in the literature. As mentioned in the introduction, optimal overtaking is usually incorporated as a least quadratic (LQ) deviation from the desired vehicle trajectory. This means that an unconstrained optimal velocity trajectory is generated (or i.e. desired velocity is set). Then overtaking is executed so that it satisfies safety requirements and avoids collisions, with the least quadratic deviation from desired trajectory. The LQ approach does not have proper decision making, so it cannot decide should the vehicle overtake the other vehicle or not. One approach shown in literature is to set a threshold velocity so when the ego-vehicle

desired velocity is higher than other vehicle's velocity by some threshold, the overtaking is executed. In this work this threshold is set to 3 m/s.

In the following simulated scenario, it is assumed that a leading vehicle ($\mathcal{V}_{\mathrm{LV}}$) is moving with a constant velocity of 12 m/s, located initially 75 m ahead of the ego-vehicle. The ego-vehicle starts from the initial velocity $v = 16.87$ m/s. In the case of ecoAD, it uses SBOMP for planning optimal driving in a moving horizon fashion, that also considers long-term benefits from strategic planning.

Table 7.8: *Benefits of ecoAD compared to least quadratic (LQ) optimal overtaking planner.*

LV velocity	energy used			travel time		
	ecoAD [kJ]	LQ [kJ]	diff. [%]	ecoAD [s]	LQ [s]	diff. [%]
empty road	157.13	157.13	0	29.4	29.4	0
12	157.13	165.36	-4.97	29.4	28.55	+2.97

In the Figure 7.12, variant with leading vehicle driving at 12 m/s is shown. Optimal Cruising Velocity is used as a reference for generating a collision free trajectory with the least quadratic (LQ) deviation (shown in green color), as proposed in literature. Contrary to this, the blue trajectory is the optimal velocity trajectory resulting from directly incorporating the constraint into the optimization problem and jointly planning velocity and overtaking by SBOMP. As it can be observed from Figure 7.13, SoA approach (LQ) speeds up to overtake the vehicle and then introduces unnecessary braking after other vehicle has passed. This is due to formulated optimization problem, to minimize the difference between desired and actual velocity. This is not the case in SBOMP, as there is no energy-efficiency related benefit in braking. This behavior of SBOMP can also be interpreted from the optimal trajectory tree. As the vehicle accelerated, due to the need to overtake, it is now in other state and it should follow the branch from the optimal trajectory tree from that state, and not the initial optimal trajectory anymore. This is based on Bellman's principle of optimality.

Table 7.8 reveals that the proposed ecoAD planner has better performance in regards to the energy consumption, compared to LQ approach. The ecoAD constrained optimization does not increase the energy consumption compared to the unconstrained case as vehicle does not accelerate to overtake the other vehicle. The LQ approach, on the other hand, leads to a larger difference (+4.97%). This implies that integration of the leading vehicle as a constraint into the optimization problem may play a considerable role in reducing the energy consumption. There is even a potential to additionally save the traveling time.

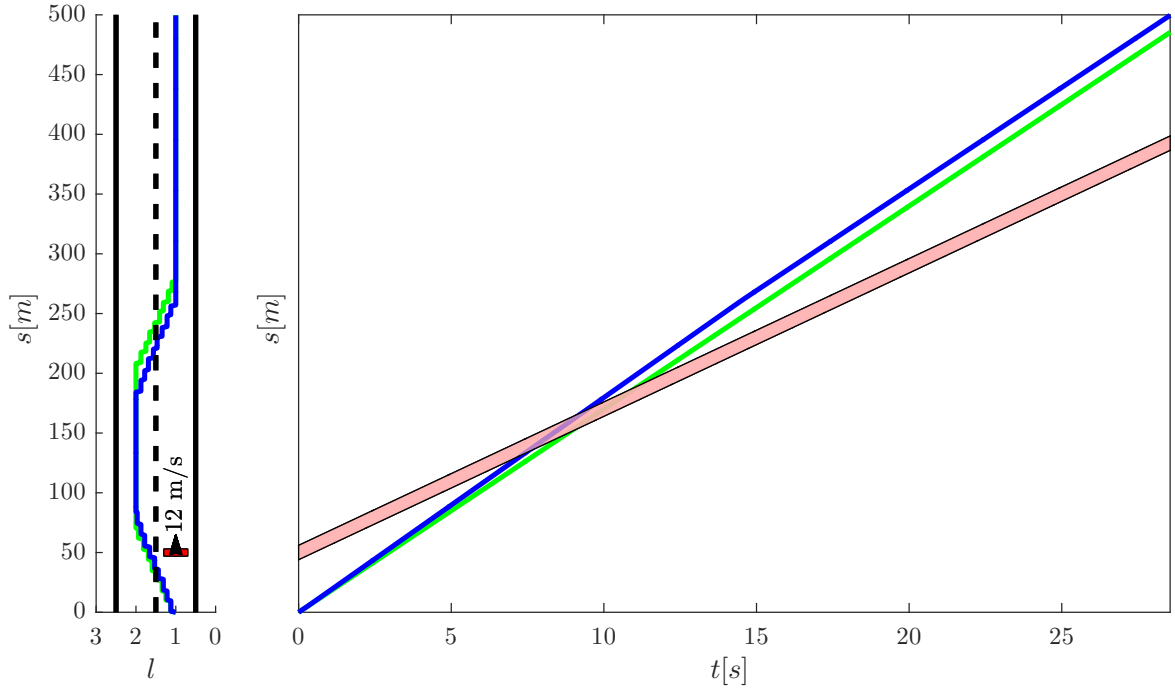

Figure 7.12: *Comparison of ecoAD overtaking and state-of-the-art optimal overtaking (LQ deviation).*

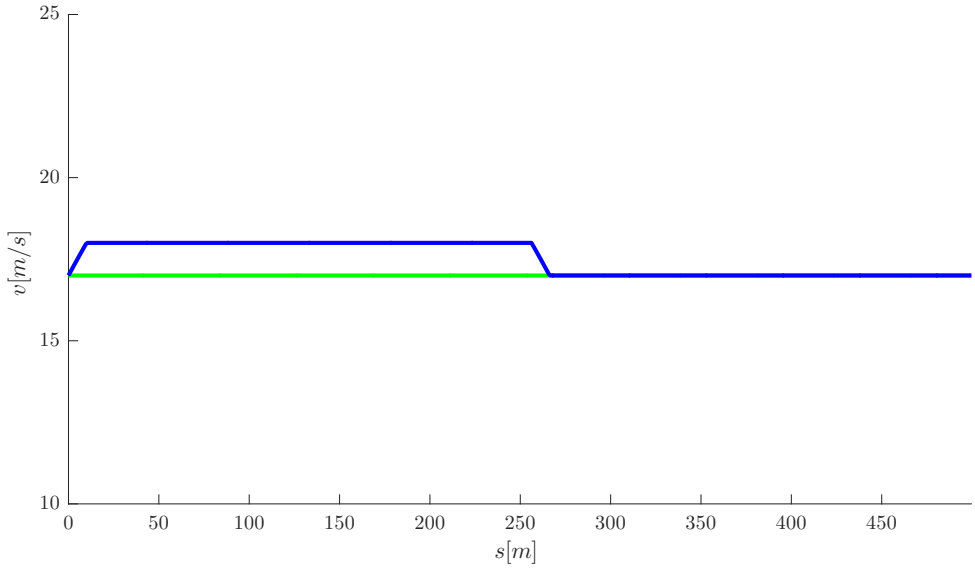

Figure 7.13: *Comparison of ecoAD overtaking and state-of-the-art optimal overtaking (LQ deviation).*

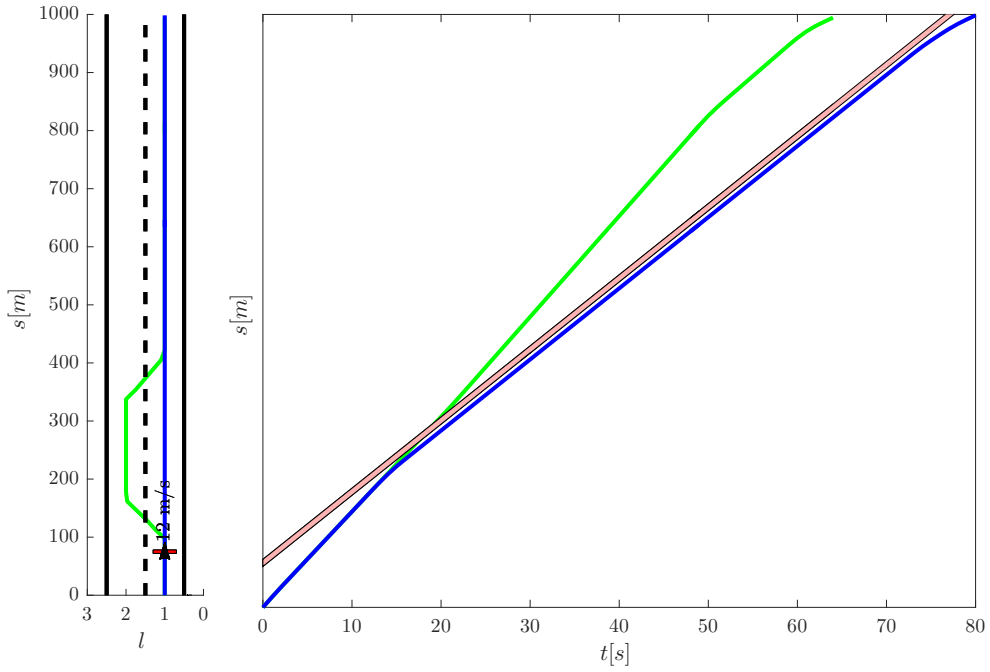

Figure 7.14: *Comparison of ecoAD and state-of-the-art single-lane ecoACC.*

Single lane ecoACC vs. multilane ecoAD

This scenario aims to benchmark the benefits of the proposed ecoAD system (that considers energy-efficiency and multilane driving) compared to state-of-the-art ecoACC approach (which consider only energy-efficient vehicle following). The same scenario variants as in previous experiment are repeated. To achieve ecoACC, SBOMP is used with prohibited lane-changing .

Table 7.9: *Benefits of ecoAD compared to ecoACC.*

	energy used			travel time		
LV velocity	ecoAD [kJ]	ecoACC [kJ]	diff. [%]	ecoAD [s]	ecoACC [s]	diff. [%]
Empty road	156.54	156.54	0	62	62	0
8	156.54	241.99	-35.31	62	119	-47.89
10	156.54	178.79	-12.45	62	96	-35.42
12	164.15	163.55	+0.36	64	80	-20

Figure 7.14 presents driving trajectories for both variants. Blue line represents trajectory driven by ecoACC and green trajectory represent trajectory driven by ecoAD. To

make a valid comparison in regards to energy consumption, final velocity limit is set at 8 m/s. It can be observed that in the case of ecoAD, vehicle overtakes the slower moving vehicle. Table 7.9 reveals that proposed ecoAD planner had better performance in regards to the energy consumption, as well as the traveling time, compared to ecoACC approach. For the leading vehicle velocity of 8 m/s, it can be observed that energy saving exceed 35% and time saving is almost 50%, when overtaking is considered. This is highly affected by the velocity of the leading vehicle and ecoACC can fully fail if the other vehicle fully stops, making the ecoAD dominant approach.

Single lane Traffic Light Assistant (TLA) vs. multilane ecoAD

In this scenario, the performance of proposed ecoAD approach is compared to the performance of the Traffic Light Assistant (TLA) that considers single lane driving for optimal passing through traffic lights. As mentioned in the related work, this is the state-of-the-art approach and it considers only single lane driving and vehicle following. The tackled scenario is the same as in the previous experiment except that there is now a traffic light located at 250 m, that turns red after 20 s and stays red for 40 s.

Table 7.10: *Benefits of ecoAD compared to TLA.*

LV velocity	energy used			travel time		
	ecoAD [kJ]	TLA [kJ]	diff. [%]	ecoAD [s]	TLA [s]	diff. [%]
7	8.0169	118.4584	-93.23	25.75	98.05	-73.73
9	23.4897	92.25	-74.54	24	78.8	-69.54
12	9.8052	9.9784	-1.73	25	28	-10.71

Figure 7.15 presents the driving trajectories for both variants. Blue line represents the trajectory driven by TLA and green line represent the trajectory driven by ecoAD. To make a valid comparison in regards to the energy consumption, final velocity limit is set at 10 m/s. Figure shows the variant with $\mathcal{V}_{\mathrm{OV}}$ driving at 7 m/s so it cannot catch the green light and stops at the red light. It can be observed that in the case of ecoAD, vehicle overtakes slower moving vehicle and catches the green light, while TLA follows the other vehicle and stops at the red light. Table 7.10 reveals that proposed ecoAD planner had better performance regarding the energy consumption, as well as the traveling time, compared to TLA approach. This scenario shows that the benefit of TLA can be totally lost in the traffic, if the vehicle in front does not catch the green light. Using ecoAD is robust to this, in scenarios where overtaking is possible, achieving the savings that can range up to 93% less energy used and 73.73% less time spent, when looking at the segment with one traffic light.

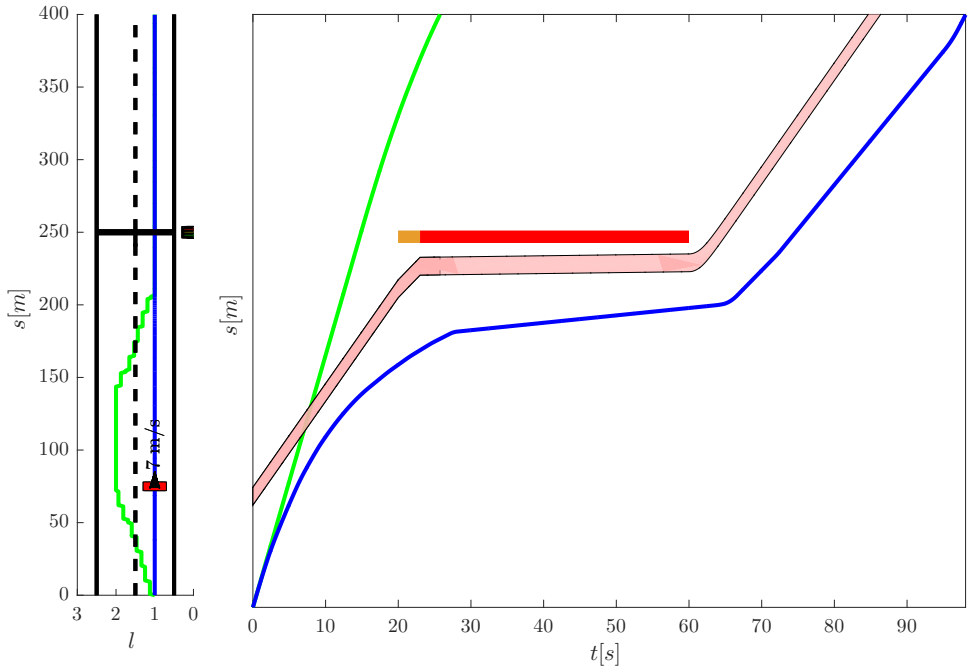

Figure 7.15: *Comparison of ecoAD and state-of-the-art single-lane TLA.*

7.3 Computational Analysis

In this section, the results of experimental computational complexity analysis are presented. Analysis is performed by varying several planner parameters to get an insight into the computational complexity of the problem. Two planners, Discrete Dynamic Programming and SBOMP, are analyzed in this section. The performance is measured based on the solution quality (optimality) and computation time. Optimality is measured as a difference compared to the best planner parameters. Computations are executed on the PC with Intel Core i7 9700 processor, with frequency up to 4.7GHz and 32GB of DDR4 memory.

7.3.1 Discrete Dynamic Programming

For analyzing the performance of Discrete Dynamic Programming solution for ecoCC, sensitivity analysis to parameters Δv and Δs is performed by varying each parameter individually. Prior to the analysis, empirically, an acceptable parameterization is used as $\Delta v = 0.1$ m/s and $\Delta s = 5$ m. Therefore, during the analysis the other parameter is set to the default value. It is important to note that there is a cross-influence of these parameters. For example, if Δs is too small on higher velocities, many transitions can be unfeasible, due to maximum acceleration $a_{\max}$, requiring very small values of Δv.

In the first experiment, Δv is varied in the range from 0.5 to 0.01 m/s. As it can be observed from table 7.11, computational time increases as discretization step Δv

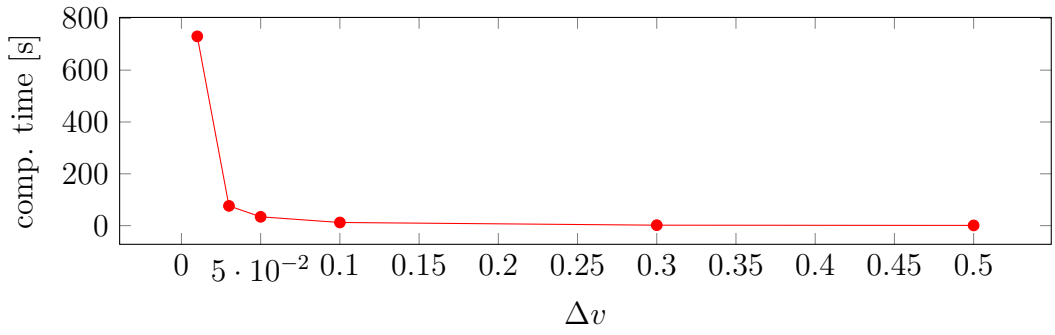

Figure 7.16: *Influence of the Δv step ($\Delta s = 5$ m, $S_{\text{hor}} = 18000$ m) on computation time Dynamic Programming for Energy-efficient Driving.*

decreases. On the other hand, energy consumption converges to the optimal value. However, as it can be seen in Figure 7.16, after $\Delta v < 0.1$ m/s optimality improves only about 2%, while the computation time increases significantly.

Table 7.11: *Influence of the Δv step ($\Delta s = 5$ m, $S_{\text{hor}} = 18000$ m) on computation time and optimality in Dynamic Programming for Energy-efficient Driving.*

Δv [m/s]	**energy used** [kJ]	**travel time** [s]	**comp. time** [s]
0.5	8167.50	1903.32	0.8837
0.3	7344.90	1694.58	1.7647
0.1	6833.45	1573.22	12.18
0.05	6660.47	1556.70	34.04
0.03	6670.59	1561.09	76.25
0.01	6639.87	1559.68	730.40

Table 7.12: *Influence of the horizon length S_{hor} (with $\Delta s = 5$ m $\Delta v = 0.03$ m/s) on computation time in Dynamic Programming for Energy-efficient Driving.*

trip length	computation time [s]
1000	4.96
2000	10.21
4000	19.76
8000	37.24
16000	76.25

As the final experiment from Energy-efficient Driving, length of the trip S_{hor} is varied from 1000 m to 18000 m. As it can be observed from table 7.12, computational time

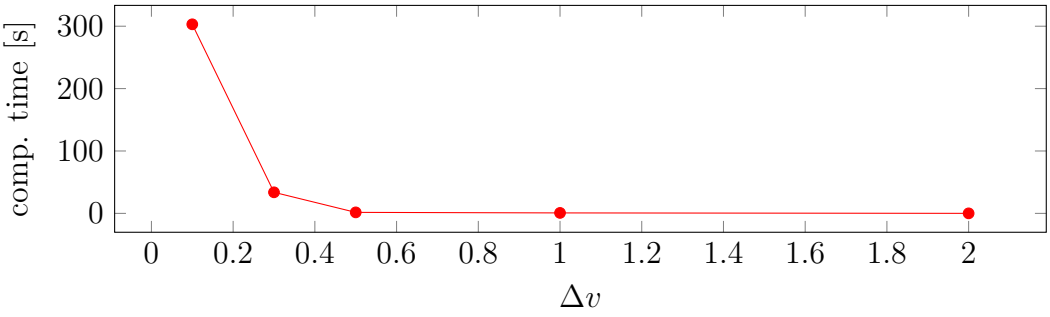

Figure 7.17: *Influence of the Δv step ($\Delta s = 10$ m, $\Delta t = 1$ s, $T_{\text{hor}} = 20$ s) on computation time of SBOMP in Automated Driving.*

increases as travel time increases, approximately linearly with the horizon.

7.3.2 SBOMP for Automated Driving

For automated driving use-case, SBOMP needs several parameters. The most critical parameters are horizon lengths (T_{hor} and S_{hor}) and velocity discretization step Δv. The ratio between horizons is kept constant, so if T_{hor} is defined, S_{hor} can be computed as $S_{\text{hor}} = k \cdot T_{\text{hor}}$, $k = 1.5$.

Table 7.13: *Influence of the Δv step ($\Delta s = 10$ m, $\Delta t = 1$ s, $T_{\text{hor}} = 20$ s) on computation time and optimality of SBOMP in Automated Driving.*

Δv [m/s]	energy used [kJ]	travel time [s]	nodes expl	comp. time [s]
2	collision	∞	32.29 ± 26.36	0.026 ± 0.021
1	154.6	18.2	507.66 ± 388.85	0.74 ± 0.51
0.5	107.71	18.1	554.6 ± 758.61	1.59 ± 1.82
0.3	12.07	24.5	4855.3 ± 1416.2	33.76 ± 8.687
0.1	-6.6871	24.55	12080.8 ± 5616	303.057 ± 159.4839

Table 7.13 presents planning results when different values of Δv are used for planning. As it can be observed, the computational time increases as Δv decreases, while energy used and travel time decrease. There are some variations but this is due to the rounding error and specific scenario. It can be observed that after $\Delta v = 0.5$ m/s the improvement in travel time is smaller than 2% the computation time increases excessively. It can be also observed that for too large values of Δv (i.e. $\Delta v = 2$ m/s) many solutions are lost due to limited acceleration so the solution lead to collision as slowing down is not possible. Figure 7.17 visualizes the dependency of computation time and energy used on Δv.

Table 7.14: *Influence of the horizon length T_{hor} (Δs = 10 m, Δv = 1 m/s, and Δt = 1 s) on computation time of SBOMP in Automated Driving.*

T_{hor} [s]	energy used [kJ]	travel time [s]	nodes expl	comp. time [s]
5	collision	∞	10.4 ± 7.56	0.0065 ± 0.0034
10	25.94	24.2	105.7 ± 101.9	0.044 ± 0.037
20	154.6	18.2	507.66 ± 388.85	0.74 ± 0.51
30	24.55	24.15	6514.58 ± 4479.75	28.96 ± 18.66

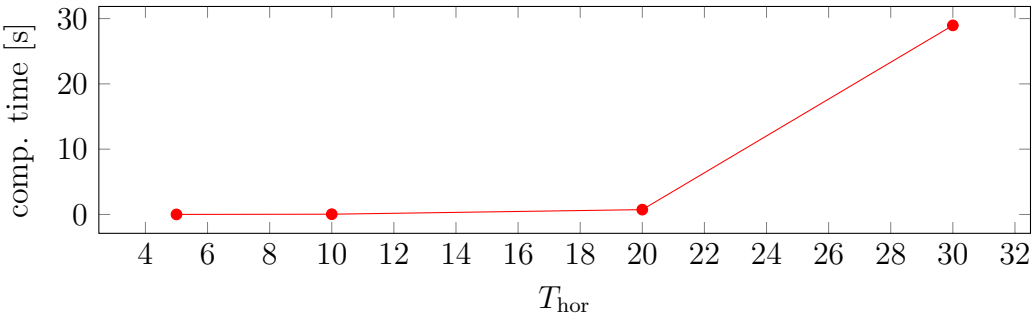

Figure 7.18: *Influence of the T_{hor} step (Δs = 10 m, Δt = 1 s, Δv = 1 m/s) on computation time of SBOMP in Automated Driving.*

Table 7.14 presents the results from different values of the horizons T_{hor}. As it can be observed, energy improvement increases as the horizon is longer. For the case when T_{hor} = 20 s, AV decided to overtake, therefore energy used is higher but travel time is shorter. As it can be observed on the Figure 7.18, computation time increases excessively (resembling exponential increase) as horizon increases.

7.3.3 SBOMP for Performance Driving

As SBOMP for Performance Driving has many parameters and carefully adjusted set of parameters, only horizon length is varied here. Horizon is defined as maximum number of stages of Δt_{exp} expansions. It is important to note here that the maximum number of expanded nodes is $N_{\text{timeout}} = 600$.

Table 7.15 presents the results of planning with different horizon values. As it can be seen, lap-time improves as horizon is longer. However, after more than 8 stages, the improvement is around 6%, while the computational time increases 4 times.

Table 7.15: *Influence of the horizon length T_{hor} ($\Delta t = 0.8$ s) on computation time and lap-time in Performance Driving.*

T_{hor} [s]	lap-time [s]	diff. [%]	nodes expl	diff. [%]	comp. time [s]	diff. [%]
4	crashed	N/A	30.78 ± 40.45	0	2.98 ± 3.48	0
6.4	58.8	+6.23	107.06 ± 124.51	+256.6	10.28 ± 10.79	+244.63
9.6	55.35	0	404.85 ± 208.88	+1215.3	36.3 ± 17.72	+1118.12

7.3.4 Conclusion on Computational Analysis

Several experiments were performed to provide an insight into the computational complexity of developed planners. As usually, there is a trade-off between solution quality and computational performance, so this analysis helps to justify choice of the appropriate parameters. It is important to note, that the implementation of proposed planners is still not the optimal, since it uses Matlab. A lot of benefit is already shown by only compiling the function and using it in Simulink. However, the runtime analysis in Simulink is hard to achieve.

7.4 Planning and Learning

For Planning and Learning experiment, we use grid world domain with 4-connected neighbors and 33% of cells in average are covered with obstacles. In total, 531 different random scenarios were used for dataset generation. From each scenario, multiple data points are generated. For comparison, two datasets were created. One dataset (representing the approach from [213]) is using only nodes from the optimal path ($\mathcal{D}_{\text{VAN}}$, 12.007 data points) and the other (as proposed in this work) uses all nodes explored in Backward Prolonged Heuristic Search ($\mathcal{D}_{\text{PHS}}$, 122.449 data points). Advantage of Backward Prolonged Heuristic Search is already visible, as from the same number of scenarios about 10 times more data points are generated, even in simple 2D problem. This is expected to be even larger in higher dimensional problems.

Value Function approximation using Deep Learning

In this experiment, a fully Convolutional Neural Network (CNN) was used for the value function approximation. The complete model architecture can be seen in Table 7.16. Each layer uses the *SELU* nonlinear activation [236]. The networks were trained for 4096 steps using a batch size of 1024 images and a learning rate of 0.001. The networks were initialized using the variance-scaling initializer [237] and optimized with

the *ADAM* [238] optimizer. In the asymmetric loss function, the parameter a is set to value -2.5.

Using ML Value Function as Heuristic function

Three different heuristic functions are used in the experiment and compared based on the solution quality (i.e. path length) and planning efficiency (i.e. number of explored nodes). The first heuristic function is admissible heuristic function h_{adm} based on the Manhattan distance. The second heuristic function is value function h_{VAN} trained on dataset $\mathcal{D}_{\text{VAN}}$ generated from the solution path only. The third function h_{PHS} is trained on $\mathcal{D}_{\text{PHS}}$ from the proposed Prolonged Heuristic Search.

7.4.1 Results

Experimental results support the initial premise that generating dataset using Prolonged Heuristic Search improves the quality and learnability of the value function. Figure 7.19 shows the training and test loss for both the $\mathcal{D}_{\text{PHS}}$ and $\mathcal{D}_{\text{VAN}}$ datasets. It can be observed that the training on $\mathcal{D}_{\text{VAN}}$ performs poorer than the training on $\mathcal{D}_{\text{PHS}}$, with slower convergence and a larger difference between the training and test set, indicating over-fitting. One reason might be the fact that $\mathcal{D}_{\text{VAN}}$ is smaller than $\mathcal{D}_{\text{PHS}}$, as the proposed approach manages to extract more data from the same scenarios. The other reason might be that the $\mathcal{D}_{\text{VAN}}$ was not diverse enough (i.e. many variations of the same scenario) and the network was not able to learn to generalize well. Contrary to that, the $\mathcal{D}_{\text{PHS}}$ offers more diverse training data (with many similar scenarios as all explored nodes are used), and the network is able to reduce the loss much further.

Additionally, the learned value functions were used as heuristic functions in the search. In total, 100 random scenarios were created, and both h_{PHS} and h_{VAN} were used. Results were compared based on the number of explored nodes (for search efficiency) and path length (for solution quality). Both were compared with the admissible heuristic h_{adm} while changing ε, which bounds the influence of ML heuristics h_{ML}. Figure 7.20 (left) shows that the path length does not increase significantly even for $\varepsilon = 3.5$, which means that both ML heuristics still provide the solutions close to the optimal. While

Table 7.16: *ML-based model architecture.*

layer	kernel	stride	dilation	avg pool	output size
conv1	3×3	1×1	1×1	none	$30 \times 30 \times 4$
conv2	3×3	1×1	2×2	none	$30 \times 30 \times 8$
conv3	3×3	1×1	4×4	none	$30 \times 30 \times 16$
conv4	3×3	1×1	8×8	4×4	$7 \times 7 \times 32$
conv5	3×3	1×1	1×1	2×2	$3 \times 3 \times 64$
conv6	3×3	1×1	1×1	2×2	$1 \times 1 \times 1$

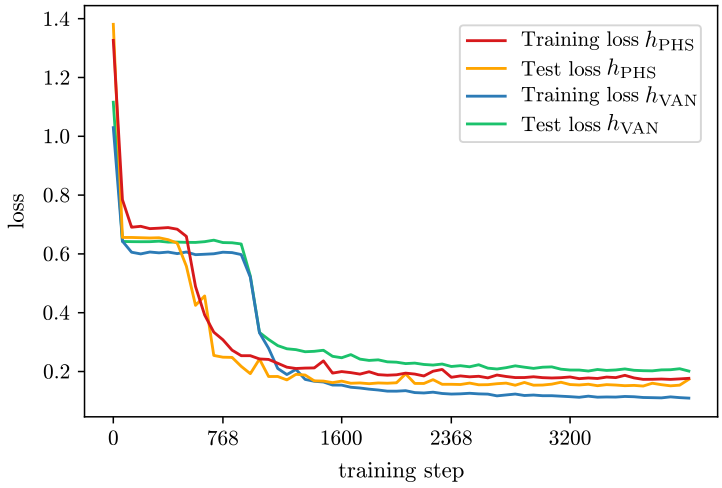

Figure 7.19: *Training and test loss for both the $\mathcal{D}_{\mathrm{PHS}}$ and $\mathcal{D}_{\mathrm{VAN}}$.*

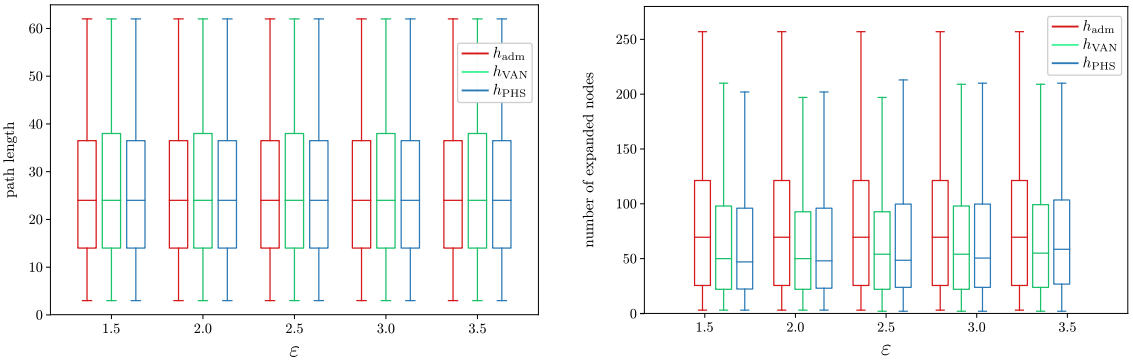

Figure 7.20: *Path length (left) and number of explored nodes (right) for h_{adm}, h_{VAN} and h_{PHS} heuristic based on ε value.*

h_{VAN} has slightly worse performance than h_{adm}, h_{PHS} has equal performance to h_{adm} in this aspect. On the other hand, Figure 7.20 (right) shows that both ML heuristics h_{PHS} expands less nodes than h_{adm}. In this example, performance of h_{VAN} and h_{PHS} varies slightly on ε, and further study on different domains and scenarios is necessary. Figure 7.21 shows an example of h_{PHS} use. It is clear from the figure that h_{PHS} expands less nodes.

7.4.2 Conclusion on Planning and Learning

The presented approach offers the possibility to effectively include Machine Learning into a deterministic planning framework, promising a significant performance improvements manifested in a reduced number of explored nodes compared to those obtained using admissible heuristic (h_{adm}), while keeping guarantees on sub-optimality of the solution. The proposed approach uses the maximum of invested computational resources in planning, as all expanded nodes in planning are used for learning. Experimental results showed a significant improvement in search performance, while keeping bounds

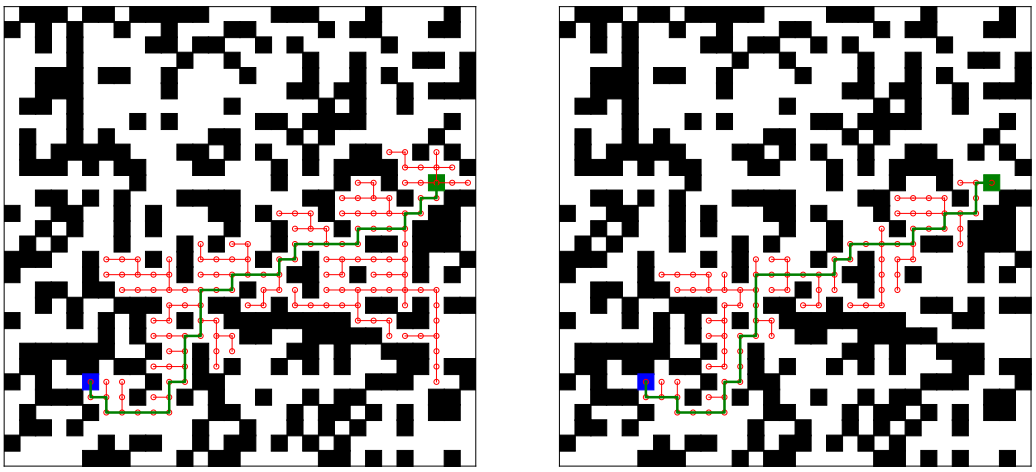

Figure 7.21: *Nodes explored while planning using admissible heuristic h_{adm} (left) and trained ML heuristic h_{PHS} (right).*

on the sub-optimality of the solution.

Future steps would include studies of behavior in different scenarios (i.e. bug-traps and tight passages), other domains (i.e. higher dimensional problems), kinodynamic motion and extension of the approach to the end-to-end Model Based Reinforcement Learning framework.

7.5 Chapter Conclusion

Proposed motion planning solutions based on the Dynamic Programming and A* search are verified on multiple scenarios from all three use-cases. It was successfully demonstrated that DP and A* search-based planners for Energy-efficient Driving consider speed limits, road slope, etc. In multiple challenging scenarios, it was demonstrated that the SBOMP planner enables Automated Driving in urban, rural and highway environments. On an mixed-curvature circuit, it was demonstrated that the SBOMP modified for performance driving extends drift-like driving from a steady state drifting on a single curve to a continuous driving on the road, effectively entering and exiting drifting maneuvers, and switching between right and left turns.

Benchmarking with the other approaches demonstrated benefits of the approaches presented in this thesis. For Energy-efficient Driving, when ecoCC is compared to the constant velocity driving, improvement in energy consumption above 8% can be observed, while prolongation of the trip is only about 1.3%. Using novel heuristic based on Optimal Cruising Velocity improved the average error of heuristic estimation almost 5 times and reduced the number of explored nodes almost 2 times, when compared to the state-of-the-art heuristic. Extending Energy-efficient Driving to Automated Driving use-case (ecoAD) provided exceptional advantages compared to the state-of-the-art approaches. Comparing with the state-of-the-art optimal overtaking approach based on the least quadratic deviation from desired velocity (LQ) showed almost 5%

reduction in energy used with only about 3% extension of travel time. Comparing to the state-of-the-art eco-driving solutions that consider only vehicle following (ecoACC) showed exceptional 35% energy saving and almost 50% time saving when overtaking is considered. This is highly affected by the velocity of the leading vehicle and ecoACC can fully fail if other vehicle fully stops, making ecoAD the dominant approach. The comparison with state-of-the-art traffic light assistant (TLA) solutions, which control the velocity in order to catch the green light, showed that the benefit of TLA can be totally lost in the traffic, if the vehicle in front does not catch the green light. Using ecoAD showed to be robust to this, in scenarios where overtaking is possible, achieving the savings that can range up to 93% less energy used and 73.73% less time spent, when looking the segment with one traffic light.

Computational complexity analysis showed that the computation performance rapidly drops as discretization step is decreased. This is expected due to curse of dimensionality and combinatorial optimization problem. However, many challenging scenarios can be solved in a reasonable time, making a good basis for real-time implementation. The chosen discretization step still has a very good resolution for behavior planning, which usually has very reduced action space (i.e. 3 actions longitudinal and 3 actions lateral).

Experimental results from learning of heuristics showed significant improvements in search performance, while keeping bounds on the sub-optimality of the solution. These results suggest that this approach is a good candidate for future improvement of computational performance.

8

Validation

> *It doesn't matter how beautiful your theory is, it doesn't matter how smart you are. If it doesn't agree with experiment, it's wrong.*
>
> *Richard P. Feynman*

Autonomous Vehicles (AV) are promising to enable Automated Driving (AD) functions with SuperHuman performance in terms of safety, efficiency, and comfort. However, beside challenges related to development involving multi-objective goals, validation and benchmarking of various AD on real vehicles, in realistic traffic situations, is one of the main critical issues.

The mainstream approach is to provide the statistical validation by comparing performance of Autonomous Vehicle fleet with regular human driving statistics. However, this requires a large fleet of automated vehicles (Level 4-5). The fleet is not yet available on the road, and requires some kind of validation in advance, making it chicken-and-egg kind of a problem.

Besides the statistical validation, it is possible to validate benefits in a particular scenario. The interest from society regarding the validation in particular scenarios may be reflected via an infamous question, which is always asked when autonomous vehicles are the topic: "Should an autonomous vehicle car kill the baby or the grandma?" (trolley problem). In this problem, the vehicle has to choose which person should it kill, if the killing is inevitable. This is an extreme scenario, yet, it should be possible to compare human performance and AV performance in defined scenarios. To quantify benefits of AD in a particular scenario, an approach could be to reconstruct a particular driving scenario and measure the performance of human drivers and AVs. The main challenge (which is practically unsolvable) in such approach is to accurately reconstruct the real driving situations involving multiple traffic participants.

This chapter presents a novel approach to reproducible, scenario-based validation by decoupling the problem into several sub-problems, without breaking the crucial couplings.

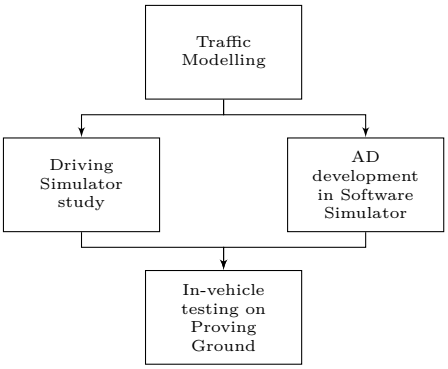

Figure 8.1: *Validation work overview.*

8.1 Autonomous Vehicle Validation Problem

Autonomous Vehicle validation is still an active research topic and many research projects are dealing with it. In [239], Beglerovic et. al provide an overview of the challenges for Autonomous Vehicle validation. That work focuses on the complete Autonomous Vehicle validation including sensors, perception, decision making, etc. For validation of the decision making and control system in particular, we can state the following major challenges:

- It is practically impossible to realistically recreate the same traffic scenario with many participants, for different experiment runs.
- The traffic is a multi-agent problem with closed loops, as actions of ego driver influence other traffic. Therefore, replaying previously recorded real traffic scenarios is not realistic for validation of the decision making and control systems (as compared to perception systems).
- Testing in-vehicle might be expensive, complicated for legal approval or dangerous.

Therefore, a decoupled methodology is presented here, that uses advantages of multiple validation modalities.

8.2 Methodology

In this work, a novel approach to reproducible, scenario-based validation is presented. Decoupling of the problem is performed by delicately choosing decoupling points, such that the crucial couplings (closed-loops) are not broken, as illustrated in Figure 8.1. First, a realistic scenario is generated from the real urban traffic. Second, human participants, in a driving simulator, drive in a virtual scenario based on the real traffic. Third, human and automated driving trajectories are reproduced and compared in the real vehicle on an empty track with simulated traffic. Thus, benefits of automation with respect to safety, efficiency and comfort can be clearly benchmarked in a reproducible manner. Decoupling makes the approach technically and economically feasible. In this work, decoupling enabled experiments to be performed at several

Table 8.1: *Comparison of validation modalities.*

	Public roads	Vehicle & Traffic simulator	Driving simulator	Proving ground
Human driver	Real	N/A	Real	Real
AI driver	N/A	Real	N/A (Real)	Real
Vehicle dynamics	Real	High fidelity	High fidelity	Real
Infrastructure	Real	High fidelity	High fidelity	Real/High fidelity
Other traffic participants	Real	High fidelity	High fidelity	Not realistic
Reproducible	No	Yes	Yes	Yes

different locations across Europe, including Graz (Austria), Delft (The Netherlands), Sarajevo (Bosnia and Herzegovina) and Gothenburg/Sandhult (Sweden), effectively, utilizing available resources. In fact, in [239], Beglerovic et. al state that "A clever combination of methods and validation environments (SiL, HiL, test-track, public road etc.) is necessary". This concurs with the approach presented here.

The assumptions made here are as follows.

- Traffic can be realistically simulated. This holds if scenario is not highly interactive, as it is the case.
- Simulated vehicle dynamics represents real vehicle dynamics.
- Driving simulator can be used to gather realistic human driving behavior.
- Vehicle dynamics and passenger comfort can be tested without other vehicles on an empty proving ground.

Table 8.1 summarized advantages provided by the each validation modality. Decoupling of the problem in performed in a way that each modality provides validation for some aspects of driving. Each aspect is real at some stage of the validation and reproducibility is provided.

The real traffic scenario is generated from real traffic. The focus is on the road geometry, environment (traffic lights and timing) and traffic density. As the situation is highly dynamic but not highly interactive, simple driver models can be used. Therefore, from public roads infrastructure and other traffic participants were used. Vehicle dynamics and traffic simulator were used for the development process of Automated Driving functions, as well as for the initial validation. It provided high fidelity model for vehicle dynamics, other traffic participants and perfect reproducibility.

To acquire referent human driving behavior, multiple human participants were tested

Figure 8.2: *Real traffic scenario.*

in driving simulator, in the same traffic scenario. This modality satisfies coupling of human driver and dynamic driving environment, with an appropriate vehicle model. It serves overall validation with realistic human driving behavior.

Finally, driving trajectories from driving simulator and Automated Driving system were executed using the real vehicle on an empty track on the proving ground. This modality satisfies the coupling of vehicle dynamics and the passenger comfort. It supports the overall validation by testing both vehicle dynamics and the comfort level.

8.3 The real traffic

For validation scenario, the traffic from a segment of the street "Zmaja od Bosne" in Sarajevo, Bosnia and Herzegovina, was chosen. In particular, the 750 m long segment from "Trg Bosne i Hercegovine" to the Campus of the University of Sarajevo was used. Figure 8.2 shows the environment from the considered street. The segment contains three traffic lights on a short distance and is relatively straight. Therefore, interesting scenarios can be reproduced on the straight test track with limited length. The timings of the traffic lights were experimentally obtained by observing a recorded video. Traffic lights are located at 188 m, 361 m and 682 m after the segment starting point. They turn yellow 12.7 s, 25.7 s and 47.7 s after the scenario start, respectively. They turn red 3 s after yellow. They have a phase of red light of 45 s, 45 s and 24 s, respectively.

Artificial traffic was created with the density of 30 veh/km/lane and the average velocity of 12 m/s. All traffic participants have implemented driver model to satisfy traffic light signals, keep the current lane and keep appropriate spacing to others.

Figure 8.3: *Prescan Simulator.*

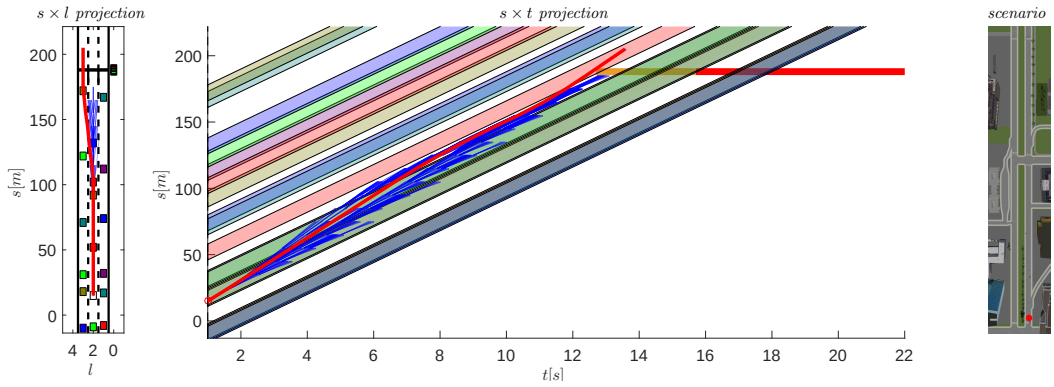

Figure 8.4: *Urban driving in the presence of traffic light.*

8.4 Validation in software simulator

The lane change feasibility within a sufficiently large time interval is validated using a higher fidelity model, Figure 8.3. The used vehicle model has 10 degrees of freedom (DoF) covering 6 DoF of the vehicle body and 4 DoF of vertical motion of unsprung masses. The vehicle body motion in space has longitudinal, lateral, vertical, roll, pitch and yaw motions. Assuming a smooth driving in high friction conditions results in small wheel slip, the wheel rotational dynamics can be neglected. We assume the linear lateral characteristic of the tire, with the relaxation behavior included. The longitudinal motion of the vehicle body is modelled taking into account the applied wheel torques (both traction and brake torques), air drag, road resistance and slope forces.

Motion planning algorithm was also validated in custom developed traffic simulator.

Test scenario was created based on a real traffic scenario as presented in section 8.3. Figure 8.4 presents one situation from the scenario. In this situation, the ego vehicle plans a lane change in order to pass the red vehicle in front and catch the green light. To make the clearance for lane change, the ego vehicle speeds up to get close to the preceding red vehicle (where the gap is), slows down during the lane change (to provide enough time for lane change) and again speeds up (to pass the red vehicle) to catch the green light. This situation truly demonstrates the importance of integrated planning for longitudinal and lateral motion. The blue tree represents searched trajectories, and the red trajectory represents the final solution. Projections of trajectories on the $s \times l$ and $s \times t$ plane are shown on the left and middle plot respectively. Vehicles shown on the left plot represent polygonal obstacles in the middle plot. The rightmost image is the screenshot from PreScan software showing the part of the real street used in the study. The resulting red trajectory shows the vehicle reaching just behind the red vehicle, slowing down to provide enough time for changing to the left lane and speeding up to passing the red vehicle, while catching the green light.

To demonstrate the robustness, stochastic variations of the scenario are created by introducing randomized perturbations of initial positions and velocities. An uncompiled script version of the algorithm was used to facilitate the randomization of the scenarios and subsequent processing of the results. This results in somewhat larger runtimes than the previously shown. Nevertheless, this does not represent the culprit for validation, since the cost and time of travel are the mayor indicators of the robustness in perturbed situations, and computation time for compiled version is previously stated. For comparison, two different heuristics were used. The first one is the result of a DP approach from the relaxed problem [5], while the second one is a model-based one (MB) from [6]. Some numerical results of the simulation are shown in Table 8.2.

Table 8.2: *Variation of urban driving scenario.*

$h(n)$	comp. time [s]	nodes exp.	energy used [kJ]	travel time [s]
DP	1.39 ± 0.75	230 ± 134	405.8 ± 20.5	56.7 ± 0.7
MB	5.71 ± 3.93	1089 ± 688	416.3 ± 11.7	57.5 ± 0.9

The results indicate that the proposed approach is robust to variations in the scenario and without significant deviations from the initial solution regarding the cost and time of travel. Moreover, using the DP heuristic is more effective compared to the MB heuristic, which is reflected in approx. **4 times** shorter **computation time** and the **number of nodes explored**. The variance of the computation time and the number of nodes explored are caused by variations in the complexity of driving situations.

8.5 Driving Simulator

Driving simulators are cost-effective tools for objective evaluation of the human driving behavior in a controlled environment, that enable reproducibility of scenarios. In this work, driving simulator was used to collect data for a comparison of AD system with actual human driving performance in the same scenario. A study of human driving behavior was performed with a support of colleagues from Human Factors group at Virtual Vehicle Research GmbH.

Human participants were asked to drive along the road as they would normally do, in a virtual environment created within the driving simulator to emulate traffic scenario as presented in section 8.3. The scenario consisted of a straight road with multiple lanes, passing through a city with relatively congested traffic. Two sessions of repetition were performed for each participant, to acquire both "normal driving" and "energy-efficient driving". The complete study, for each participant, lasted approximately 30 minutes, including small breaks after the familiarization and between the 2 driving sessions.

8.5.1 DriveLab driving simulator

For human driver study, DriveLab driving simulator located at Virtual Vehicle Research, Graz (Austria) is used. DriveLab driving simulator is state-of-the-art 3-DOF Driving Simulator. It has a mock-up vehicle cockpit that is fabricated from a small sedan car body, resembling the real vehicle interior with some added features. Software system is based on VI-DriveSim[1] (VI-grade, Italy) and SCANeR Studio[2] (AV Simulation, France) running on Concurrent Real-time Workbench[3]. It enables vehicle dynamics and environment simulations essential for this study. With the use of three integrated visual projectors with circular screen, a horizontal Field of View of 215° is achieved. Together with three rear mirror screens, it gives a fully immersive feeling to the driver. For acoustic support, a 7.1 surround sound system is used to replicate the acoustics of the real vehicle. A digital dashboard provides a feedback to the driver about the vehicle speeds and other attributes, including engine rpms and engaged gear. Vehicle with automatic transmission is considered in the study, so only two pedals (throttle and brake) are used, without gear shifting.

8.5.2 Virtual Environment

The virtual environment resembling an urban street was created within driving simulator software, based on the real traffic scenario. A three-lane road with traffic lights and lane-markings as described in section 8.3 is generated. To provide full urban environment feeling, architecture and urban design is also modelled. The traffic was created by placing 17 vehicles on predefined positions. Vehicles are programmed to drive with the target velocity of 12 m/s, but they react to the environment and slow down for the traffic light and other vehicles. They follow their initial lane and do not change

[1] https://www.vi-grade.com/en/products/static-simulator/
[2] https://www.avsimulation.fr/solutions/
[3] https://www.concurrent-rt.com/products/redhawk-linux/

Figure 8.5: *Driving simulator.*

the lane. The length of the road of interest is 1000 *meter*, ending with the finish line. To provide the same scenario for all participants and enable further benchmarking of the results, scenario trigger is used when participant reaches the segment of interest. Additional 1000 m road was added in front of the segment of interest for a driver to adjust velocity and approach the traffic. Participants start driving 1000 m before the segment of interest (the trigger point) from standstill. They drive on an empty road and approach traffic that is frozen. When they reach the trigger point, the scenario starts, including other vehicles and traffic lights timing. Participants are instructed to approach other traffic in the middle lane with approximately 50 km/h, without slowing down. The real-time data was collected, including the driven trajectories with a specific time stamp for later analysis.

8.5.3 Procedure

Each participant was assigned dedicated time-slot for the study. Participants were welcomed and introduced to the driving simulator. They were then informed about the driving task. First, they were instructed verbally, and then they were provided with a written description of the scope of the experiment and instructions for their task.

After the introduction, participants were escorted to the vehicle mock-up. Once they adjusted the position of the seat and felt comfortable, familiarization phase of the driving could start. In this phase, they were getting familiarized with the simulator, vehicle dynamics and the road environment. Familiarization environment was different then the environment used in the study. It consisted of more turns, intersections and interactions with other vehicles. It was intentionally set as such to expose participants

in order to gain more experience and develop the feeling of how the virtual vehicle was responding to the input command and how the visual scene in driving simulator behaves. Three participants showed discomfort with driving simulator at this stage. After about 5 minutes of driving in the familiarization environment, participants were asked about their confidence with the simulator, as well as their comfort inside the cabin. If necessary, the familiarization was extended until the participant indicated they felt sufficiently familiar with the vehicle and the environment.

After the familiarization phase, the actual experiment started, consisting of two sessions, with each session having three repetitions. First, a driving task was briefly rehearsed. Participants were then instructed to drive naturally as they would drive in reality, and keep an appropriate driving behavior during the whole experiment, respecting the following safety and legal considerations:

- respect the speed limit of 60 km/h,
- avoid collision with other vehicles,
- respect the traffic lights,
- it is possible to to use both sides to overtake other vehicles.

They were particularly advised to approach the trigger point with 50 km/h, so that all participants have the same scenario start. When scenario is triggered they could drive as they would normally drive in the traffic. Participant would start the scenario from standstill, drive into the traffic and trigger the traffic an drive the full scenario until the finish line. This was repeated three times for the first session. For the second session and three more repetitions, they were advised to drive more energy efficiently, to try to catch the green light but still keep smooth driving.

As a conclusion, a talk with the participants was used to inform them about future steps of the study and to get feedback about the realism of the driving simulator study. There were no objections of the participants. The complete procedure for each participant lasted about 30 minutes.

8.5.4 Results

In total 28, volunteers (23 males, 5 females) were recruited for the study in the months of May and August 2019. All of them had normal or corrected-to-normal vision. They all had driving experience and were in a possession of valid driving license. The average participant age was 28,5 years (std 4,3 years).

From 28 participants, 25 participant completed the study. Each participant provided maximum 6 driving trajectories. In total, 113 driving trajectories were collected. From 113 trajectories, 103 trajectories were within 50 ± 10 km/h initial velocity and were used for comparison. From 103 trajectories, 55 trajectories fall into category of normal driving while 48 trajectories cosrrespond to energy-efficient driving.

Different driving trajectories are presented on the Figure 8.6 and 8.7. From $t \times s$ projection, it can be observed that there are four main clusters of trajectories. The three clusters of vehicles that stop on some traffic light, and one cluster of trajectories that manage to get the green wave. It can be observed that many drivers pass during

Table 8.3: *Instances of braking traffic rules by human drivers.*

variant	number of trajectories	speed limit	speed limit +5 km/h	speed limit +10 km/h	yellow TL	red TL	collision
normal	55	28	15	7	38	0	2
normal [%]	100	50.91	27.27	12.73	69.09	0	3.64
energy-efficient	48	27	9	5	35	0	0
energy-efficient[%]	100	56.25	18.75	10.42	72.92	0	0

yellow and some even during red light. From $l \times s$ projection, it can be observed that the lateral motion is not so distinct among drivers. What can be observed is that vehicles start in the middle lane and start lane-changing relatively late. The stopping on three traffic lights can also be seen on $s \times v$ diagram from figure 8.7. From the same diagram, it can be seen that many drivers violated the speed limit of 60 km/h.

Table 8.3 presents the results of the analysis of traffic rule violation in this scenario. Clearly, drivers violated the speed limit in more than 50% of driving runs and more than 10% of driving runs with a margin of 10 km/h. In more than 69% of driving runs, drivers passed during the yellow light. Moreover, drivers caused 2 critical situations which can be categorized as collisions. Table 8.4 presents the results of the driving behavior based on energy consumption and travel time. First, results from "normal driving" session are presented, followed by "energy-efficient driving session". Additionally, the single results of the best and the worst run based on energy-consumption and travel time are presented. It can be observed that drivers generally improve performance in terms of energy consumption and travel time in the second session (energy-efficient driving). This might be caused by the advice, but might also be due to gained experience about the scenario. Still, the average performance of human drivers is more than 60% worse than AD. In fact, AD is about 20% more energy-efficient than the best energy-efficient human driver, and faster than the fastest human driver (without violating traffic rules). Presented results here include human-driven trajectories which violate traffic rules, therefore the results would be even more in advantage of AD.

8.6 Proving Ground

Once all other stages are successfully completed, tests on the proving ground with the real vehicle can be executed, as the final stage. Testing on the proving ground provides

Table 8.4: *Driving performance of human drivers based on energy and travel time.*

driving	energy used [kJ]	diff. [%]	travel time [s]	diff. [%]
normal driving	815.42 ± 128.41	+95.87	106.11 ± 24.31	+84.53
energy-efficient driving	670.94 ± 104.09	+62.44	82.14 ± 20.47	+42.85
the most energy-efficient	510.79	+22.69	68.99	+19.98
the fastest	571.59	+37.3	60.0	+4.34
Automated Driving	416.3	0	57.5	0

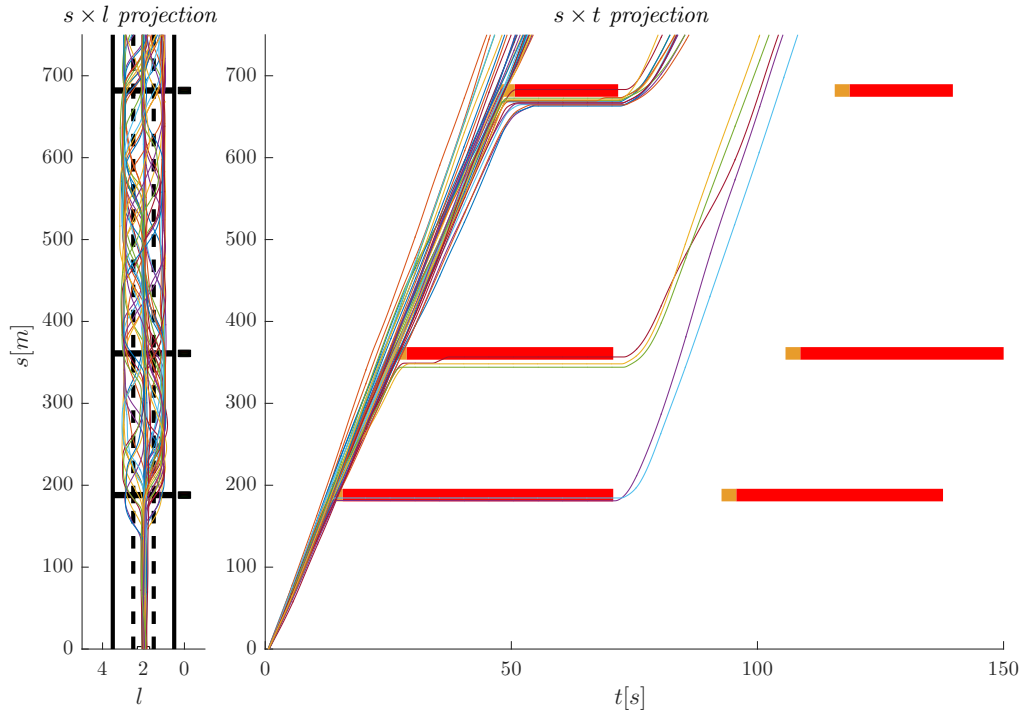

Figure 8.6: *Results from human driver study in driving simulator.*

reproducibility by the use of driving robots. Testing is performed with a goal to verify simulation results and to validate passenger's riding comfort. Verification of the simulation should assess how well the models represent the real world. The expected outcome is that the feasibility and human acceptance of the planned trajectories are confirmed. Tests in a real vehicle are also important for passenger riding comfort validation, as it cannot be validated otherwise. Driving simulators generally do not provide the full driving dynamics of the real vehicle.

In this work, testing on the proving ground is performed in cooperation with Volvo

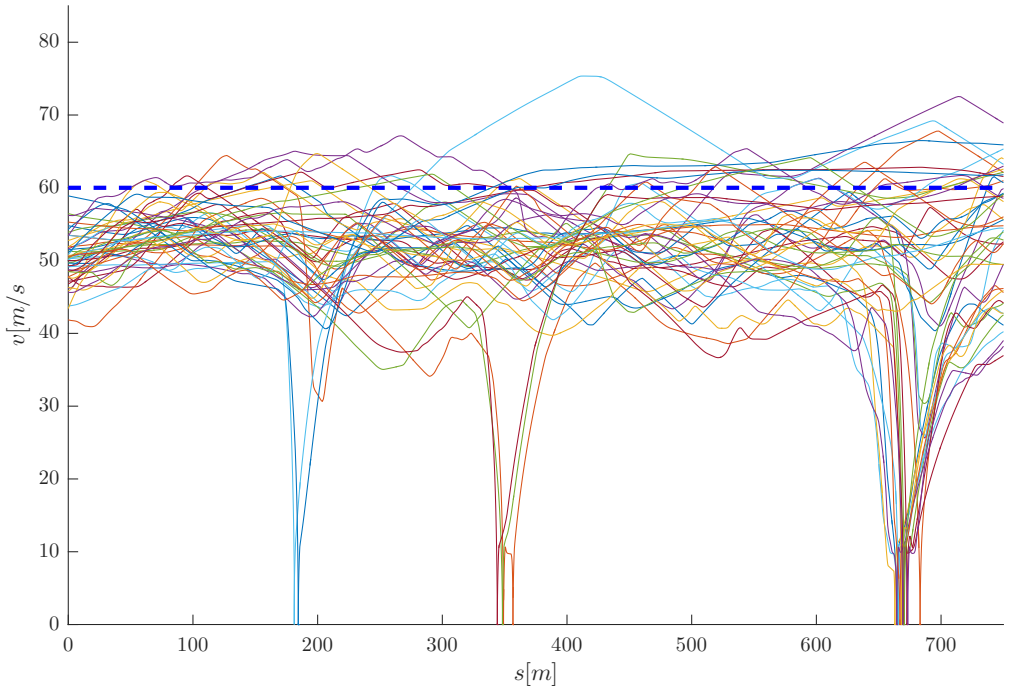

Figure 8.7: *Results from human driver study in driving simulator.*

Cars and AstaZero[4]. Human-driven trajectories from driving simulator and automated driving trajectories are executed on the empty test track by the vehicle instrumented with a steering robot. Similar approach was presented in [228]. There, the authors presented the test of an autonomous vehicle on an empty track with simulated dynamic environment. However, the results were not benchmarked with human drivers and in realistic urban traffic situation with traffic lights. On the other hand, this work provides such an extension.

8.6.1 Demo Vehicle

Volvo S90 T5 was used as a demo vehicle. Figure 8.8 presents the demo vehicle used for this work on the test-track. Demo vehicle is instrumented with the steering robot ABDynamics. Figure 8.9 shows the steering robot setup in the vehicle. Steering robot can actuate the steering wheel, gas and brake pedals. It has differential gps positioning with RTK. To execute maneuvers, steering robot uses dedicated file format, so-called PMC files, which can be recorded by driving to reproduce some tests. For this work, a simple software tool was created that interprets human-driven trajectories from driving simulator and generates appropriate PMC files, that can be further executed on the real vehicle using steering robot. During experiments, data from the steering robot and vehicle CAN signals were collected.

[4]http://www.astazero.com/

Figure 8.8: *Test vehicle Volvo S90.*

8.6.2 Proving ground

As mentioned earlier, experiments were executed at AstaZero Hällered, Sandhult, (Sweden) proving ground. In particular, multilane test track was used, as shown in Figure 8.10. Some experiments were also performed on a rural road, but they are not reported here. Multilane road is 700 m long. As some room is needed to accelerate the vehicle in order to start scenario with 50 km/h, the segment of 550 m was used for reproducing the driving trajectories. That was enough to reproduce the trajectories, including first two traffic lights. Multilane road has four lanes, from which the left three lanes are used in experiments. The cones were placed to mark the position of traffic lights

8.6.3 Experiments

Experiments were performed for a duration of several days in the months of June and August 2019. Human driven trajectories and automated driving trajectories were executed for several times each, while the robot and vehicle data were collected. The developed tools for interpreting trajectories for a robot showed to be very robust, requiring minimal additional manual work. In total, 67 trials were executed and recorded.

Figure 8.9: *Steering robot used for executing driving trajectories.*

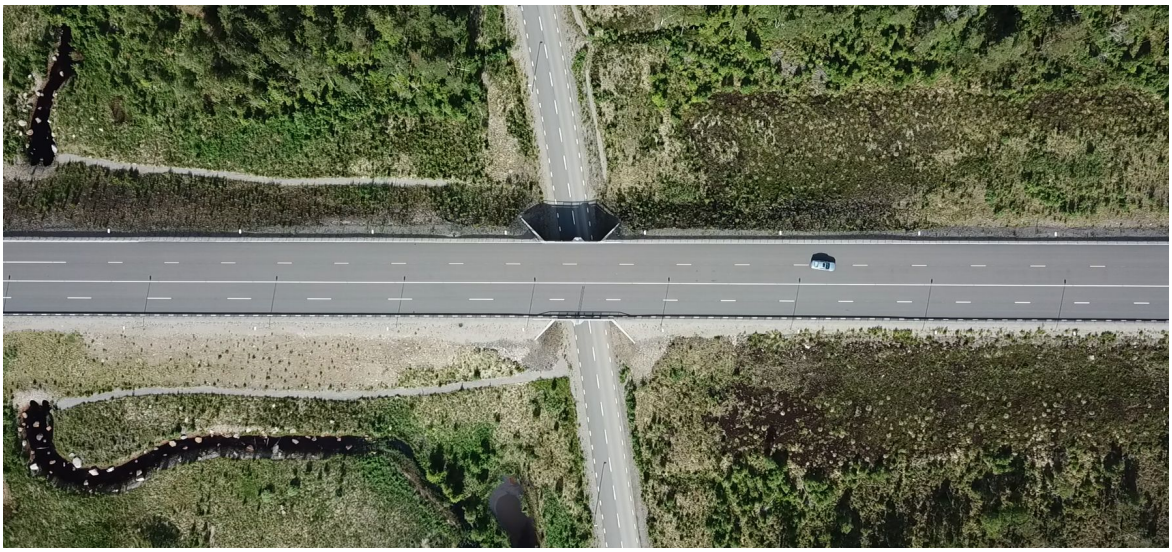

Figure 8.10: *AstaZero proving ground, multilane road.*

The trajectories were successfully reproduced with a tracking error less than 1 cm and 2 m/s, as shown in Figure 8.11. It is worth pointing out that some trajectories were

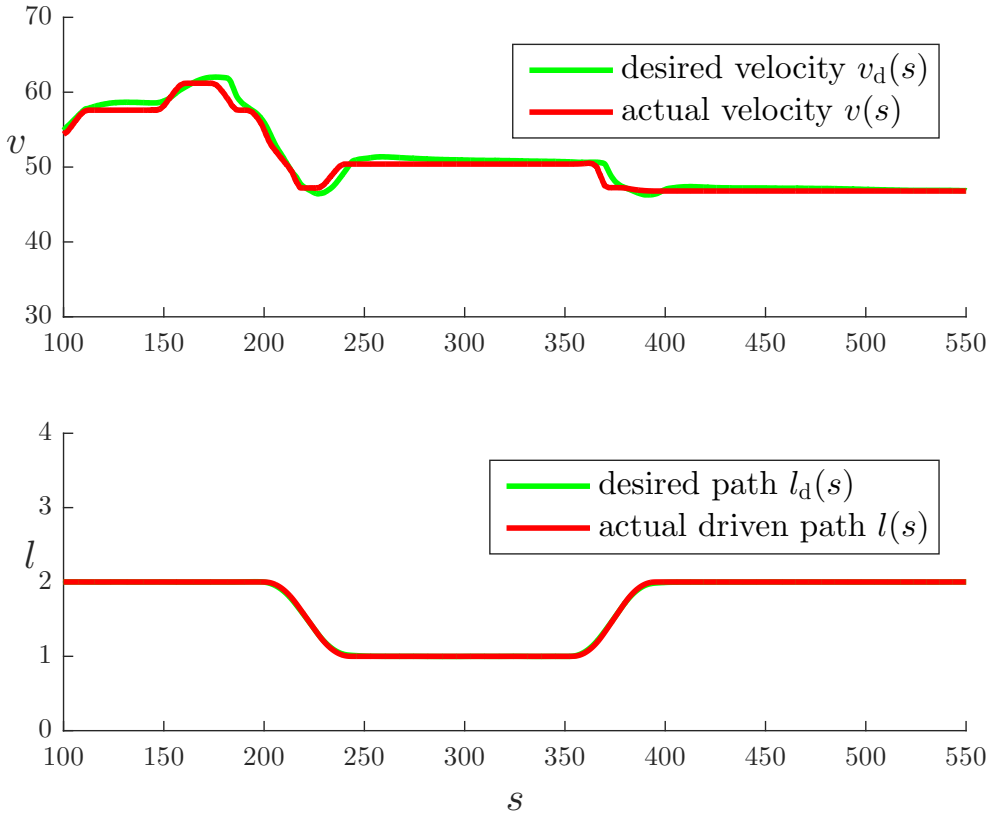

Figure 8.11: *Results from executing Automated Driving trajectory by test vehicle.*

very extreme, with accelerations exceeding 11 m/s^2. Trajectory execution was robust, so it was possible to reproduce same trajectories several months after the initial tests.

Unfortunately, vehicle consumption could not be extracted from recorded CAN data. Therefore, vehicle consumption model from simulation cannot be verified. However, such verification has already been reported in the literature [240]. The main contribution of this work is to demonstrate the feasibility of the planned trajectories and human acceptance.

To validate the perceived human safety and comfort, 8 participants were driven as passengers in the vehicle. Vehicle executed three driving trajectories. Two driving trajectories were human-driven from the driving simulator, and one corresponded to AD. Participants were asked to rate the driving on the scale of 1 to 5 based on their perceived safety and comfort. The ratings from 1 to 5 indicate driving as: “very bad”, “bad”, “ok”, “good” and “very good” respectively. Table 8.5 presents the results of the study. Average rating is presented along with the standard deviation.

Although the sample is not sufficiently large to make a general conclusion, the presented results show that automated driving was rated higher in average than two human-driven trajectories. One human trajectory in particular was rated bad, as it included a sudden stop on the second traffic light.

Table 8.5: *Passenger ratings of driving trajectories (2 human and one automated driving).*

	Comfort			Safety		
	1st run	2nd run	AD run	1st run	2nd run	AD run
Rating	1.75 ± 0.66	3.125 ± 1.05	3.625 ± 0.86	2.25 ± 1.20	3.25 ± 0.66	3.875 ± 0.60

Successful execution of human-driven and AD trajectories on the proving ground provided the confirmation of the simulation study. The reproducibility of the testing on the proving ground is demonstrated also by the fact that scenarios are reproduced even a few months after their initial setup.

8.7 Chapter Conclusion

the presented approach to scenario-based validation of AD functions shows that the problem of validation can be effectively decoupled. Modelling the realistic traffic helped to get very interesting and useful traffic scenarios. Simulation tools helped significantly during the development and the initial validation. Driving simulator was an effective solution to get a wide variety of human responses to the same driving scenario. The study confirmed that the scenario is challenging, as even the drivers with experience from the scenario had some difficulties to again achieve the same performance. Experiments on the proving ground confirmed simulation results, and enabled to acquire useful feedback from participants about perceived safety and comfort.

Validation effort showed that developed AD function achieves SuperHuman driving performance in terms of safety, efficiency and comfort. While many drivers violate traffic rules (56% of drivers violate speed limit, 10% even with margin of 10 km/h) and cause accidents, AD system does not. AD system has shown to have better energy consumption, in particular 22% better than the best human driven trajectory, from more than 100 trials and almost 30 participants. Finally, passengers rated AD better than other two human-driven trajectories in terms of perceived safety and comfort. As for now, this validation represents a proof of concept and is based on a single scenario. For more general conclusions a deeper study is necessary, including more participants and more scenarios.

9

Conclusion

Part of the journey is the end.

Tony Stark

Autonomous Vehicles are one of the biggest technical challenges of our time, which could disrupt our way of living by making our mobility safer, more efficient and comfortable. However, our state-of-the-art technology is still having difficulties even with perceiving driving environments and effectively driving in the wide complexity of the driving situations is yet to raise as an issue.

This thesis focused on three use-cases, that were sampled from a wide breadth of autonomous vehicle applications. These use-cases are Energy-efficient Driving, Automated Driving and Performance Driving, each of which brings unique challenges, contributing to the scalability and robustness of the overall solution. Developed planner is scalable for all three use-cases and wide scenario variations in Automated Driving use-case. The goal of reaching human-level performance is surpassed by providing SuperHuman performance in terms of safety, efficiency and comfort.

As introduction, thesis provided extensive overview of the autonomous vehicle technology and work related. Further on, the problem is defined, along with it's requirements and assumptions. As the first step in developing model-based solution, different aspects of driving problem are modelled, including the appropriate vehicle dynamics, driving environment (including road, traffic rules, other traffic participants, etc.) and driving objective. These models serve to formalize general planning problem for automated driving, scalable to wide range of scenarios. Appropriate modelling facilitated efficient solving using motion planning approaches. It was noticed that even in automated driving community, a general solution for planning in various scenarios was missing. Therefore, the work was focused on practical solution for automated driving in full complexity of driving situations, while keeping energy-efficiency as driving objective when choosing among collision-free trajectories (safety objective). As it was shown, solutions are based on the graph search methods (Dynamic Programming and A* search), that are combined into effective framework Search-Based Optimal Motion Planning (SBOMP). It relies on motion primitives and hybrid A* search-based solution, enabling it to deal well with complex problems (i.e. Energy-efficient Driving, Automated Driving and Performance Driving).

Energy-efficiency showed to be a very practical objective, as it provided naturally

looking trajectories much better than if otherwise desired velocity was used, as it is the case in most of other work. The proposed MP framework showed to be an efficient and robust planning solution for automated driving, even in very complex scenarios, such as multilane driving with traffic lights. The framework is capable of finding the velocity trajectory, as such that enough clearance for lane-change is provided in tight situations, due to integrated longitudinal and lateral motion planning. SBOMP demonstrated the capability to provide slow and fast trajectories, which is particularly important for treating different constraints in urban driving. Moreover, using the exact DP cost-to-go as a heuristic significantly improved efficiency of the search, compared to model-based heuristics. SBOMP was then further extended for performance driving on a slippery surface. It extends drift-like driving from a steady state drifting on a single curve (as it is done in state-of-the-art work) to a continuous driving on the road, effectively entering and exiting drifting maneuvers and switching between right and left turns.

Still, computational performance is not satisfactory for use on embedded hardware in vehicle. Computational requirements raise even more when non-deterministic and interaction-aware problem extensions are considered. To improve computational performance, previous planning experience can be used to learn better planning in future. Experimental results showed a significant improvement in search performance, while keeping bounds on the sub-optimality of the solution for a simple grid-world problem.

Benchmarking with other approaches demonstrated benefits of the approaches presented in this thesis. Most notably, In Energy-efficient Driving, using a novel heuristic based on Optimal Cruising Velocity improved the average error of heuristic estimation almost 5 times and reduced the number of explored nodes almost 2 times, compared to the state-of-the-art heuristic. Extending Energy-efficient Driving to Automated Driving use-case (ecoAD) provided exceptional advantages compared to the state-of-the-art approaches. Comparing with the state-of-the-art optimal overtaking approach based on the least quadratic deviation from desired velocity (LQ) showed almost 5% reduction in energy used with only about 3% extension of travel time. Comparing to the state-of-the-art eco-driving solutions that consider only vehicle following (ecoACC) showed exceptional 35% energy saving and almost 50% time saving when overtaking is considered. This is highly affected by the leading vehicle driving. The ecoACC can totally fail if other vehicle fully stops, making ecoAD superior. The comparison with state-of-the-art traffic light assistant (TLA) solutions, that control velocity in order to catch the green light, showed that the benefit of TLA can be totally lost in the traffic, if the vehicle in front does not catch the green light. Using ecoAD showed to be robust to this, in scenarios if overtaking is possible, achieving the savings that can range up to 93% less energy used and 73.73% less time spent, when looking the segment with one traffic light.

Finally, the developed Automated Driving function was successfully validated and *SuperHuman driving performance* in terms of safety, efficiency and comfort was successfully demonstrated. While many drivers violate traffic rules (56% of drivers violate speed limit, 10% even with margin of 10km/h) and cause accidents, AD system does not. AD system showed to have better energy consumption, in particular, 22% better than the best human driven trajectory, from more than 100 trials and almost 30

participants. As the last stage in validation, passengers rated Automated Driving trajectory better than other two human driven trajectories in terms of perceived safety and comfort. This validation is a proof of concept and is based on a single scenario. For more general conclusions a bigger study is necessary, including more participants and more scenarios.

Presented work is only one part of the complete solution. To achieve fully functional planning for level 5 Automated Driving, it is necessary to extend this solution as such that it is considering:

- uncertainties in the environment and vehicle model,
- interactions with other participant directly in planning,
- limits of the perception system,
- following up on the execution of the plans and integration with tracking control.

To improve the computational performance that would enable these advanced features, it is possible to pursue several different directions such as:

- combining planning and learning,
- better search algorithms and heuristics, i.e. multi-heuristic A* search [235],
- parallel computation.

Future steps for planning and learning could include studies of planning behavior in different scenarios (i.e. bug-traps and tight passages), other domains (i.e. higher dimensional problems), kinodynamic motion and extension of the approach to a end-to-end Model Based Reinforcement Learning framework.

Acronyms

ACC	Adaptive Cruise Control
AD	Automated Driving
ADAS	Advanced Driver Assistance System
AEB	Advanced Emergency Braking System
AI	Artificial Intelligence
AMoD	Automated Mobility on Demand
AV	Autonomous Vehicle
BFS	Breadth-first search
CAN	Controller Area Network
CAV	Connected and Autonomous Vehicle
CC	Cruise Control
CL-RRT	Closed-loop Rapidly-exploring random tree
CNN	Convolutional Neural Network
DARPA	Defense Advanced Research Projects Agency
DL	Deep Learning
DP	Dynamic Programming
ecoACC	Energy-efficient Adaptive Cruise Control
ecoAD	Energy-efficient Automated Driving
ecoCC	Energy-efficient Cruise Control
ecoTLA	Energy-efficient Traffic Light Assist
ESC	Electronic Stability Control
EU	European Union
GCDC	Grand Cooperative Driving Challenge
GDP	Gross Domestic Product
GNSS	Global Navigation Satellite System
HD	High Definition
HMI	Human-Machine Interface
HPC	High Performance Computing

IMU	Inertial Measurement Unit
ITS	Intelligent Transportation System
LQ	Least Quadratic
LQR	Linear-quadratic regulator
LV	Leading Vehicle
MaaS	Mobility as a Service
MCTS	Monte Carlo Tree Search
MDP	Markov Decision Process
MHA*	Multi-Heuristic A*
MIMO	Multiple-Input Multiple-Output
ML	Machine Learning
MP	Motion Planning
MPC	Model Predictive Control
NN	Neural Network
OCV	Optimal Cruising Velocity
OEM	Original Equipment Manufacturer (i.e. vehicle producers)
OV	Other Vehicle
PHS	Prolonged Heuristic Search
PID	Proportional–Integral–Derivative Controller
POMDP	Partially Observable Markov Decision Process
RL	Reinforcement Learning
RRT	Rapidly-exploring random tree
RWD	Rear-Wheel Drive
SAE	the Society of Automotive Engineers
SBOMP	Search-Based Optimal Motion Planning
SHAVe	SuperHuman Autonomous Vehicle
SLT	distance-lane-time space
SoA	State-of-the-Art
SPaT	Signal Phase and Timing
TLA	Traffic Light Assist
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Anything
VAN	Vanilla

Symbols

P_{tot}	Total power
P_{aux}	Auxiliary power
T_{m}	Motor torque
r_{w}	Eheel radius
ω_{m}	Rotational velocity of the motor
ρ_{a}	Air density
c_{d}	Vehicle drag coefficient
c_{r}	Vehicle roll coefficient
A_{f}	Vehicle frontal area
F_{r}	Resistive force
F_{m}	Vehicle traction force
x	Vehicle position in Euclidean space
y	Vehicle position in Euclidean space
ψ	Vehicle yaw angle
v	Vehicle velocity
β	Vehicle side-slip angle
s	Distance travelled along the road in curvilinear frame
l	Vehicle position on a lane
t	Time
T_{LC}	Maximum time for lane change
k_{hr}	Hourly rate
k_{ep}	electricity price
$\mathbf{x}$	State vector
$\mathcal{X}$	Set including all possible states
$\mathcal{X}_{\text{free}}$	Collision-free subset of the states
$\mathbf{q}$	Configuration
$\mathcal{O}$	Obstacle
n	Node
Open	Open list for A* search
Closed	Closed list for A* search
$h(\cdot)$	Heuristic function for A* search

$\mathbf{V}_\mathrm{i}$	Matrix of initial velocities for Dynamic Programming
$\mathbf{V}_\mathrm{f}$	Matrix of final velocities for Dynamic Programming
$\mathbf{V}_\mathrm{av}$	Matrix of average velocities for Dynamic Programming
$\mathbf{T}_\Delta$	Matrix of time for transition for Dynamic Programming
$\mathbf{A}_\mathrm{acc}$	Matrix of acceleration during transition for Dynamic Programming
T_rep	Period of replanning
T_hor	Planning time horizon
S_hor	Planning distance horizon
Δt_grid	Time step of the grid
Δs_grid	Distance step of the grid
Δt_exp	Time limit for motion primitives
Δs_exp	Distance limit for motion primitives
t_exp	Time travelled of motion primitive
s_exp	Distance of motion primitive
T_plan	Time required for planning
$\mathcal{M}$	Vehicle model
$\mathcal{D}$	Dataset

List of Figures

List of Tables

Bibliography

[1] A. Sciarretta, G. de Nunzio, L. L. Ojeda, Optimal ecodriving control: Energy-efficient driving of road vehicles as an optimal control problem, IEEE Control Systems Magazine 35 (5) (2015) 71–90. (cited on pages 3, 11 and 67.)

[2] W. Schwarting, J. Alonso-Mora, D. Rus, Planning and decision-making for autonomous vehicles, Annual Review of Control, Robotics, and Autonomous Systems 1 (2018) 187–210. (cited on pages 3, 29 and 30.)

[3] B. Paden, M. Cap, S. Z. Yong, D. Yershov, E. Frazzoli, A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles, IEEE Transactions on Intelligent Vehicles 1 (1) (2016) 33–55. doi:10.1109/TIV.2016.2578706. (cited on pages 3, 29 and 30.)

[4] Z. Ajanović, M. Stolz, M. Horn, Energy-Efficient Driving in Dynamic Environment: Globally Optimal MPC-like Motion Planning Framework, in: C. Zachäus, B. Müller, G. Meyer (Eds.), Advanced microsystems for automotive applications 2017, Vol. 12 of Lecture Notes in Mobility, Springer, Cham, Switzerland, 2018, pp. 111–122. doi:10.1007/978-3-319-66972-4{\textunderscore}10. (cited on pages 5, 38 and 96.)

[5] Z. Ajanović, M. Stolz, M. Horn, Energy Efficient Driving in Dynamic Environment: Considering Other Traffic Participants and Overtaking Possibility, in: D. Watzenig, B. Brandstätter (Eds.), Comprehensive energy management, Vol. 6 of SpringerBriefs in applied sciences and technology, Automotive engineering : simulation and validation methods, 2191-530X, Springer, Cham, Switzerland, 2017, pp. 61–80. doi:10.1007/978-3-319-53165-6{\textunderscore}4. (cited on pages 5, 37, 38, 67, 78 and 142.)

[6] Z. Ajanović, M. Stolz, M. Horn, A novel model-based heuristic for energy-optimal motion planning for automated driving, IFAC-PapersOnLine 51 (9) (2018) 255–260. doi:10.1016/j.ifacol.2018.07.042. (cited on pages 5, 38 and 142.)

[7] Z. Ajanović, B. Lacevic, B. Shyrokau, M. Stolz, M. Horn, Search-Based Optimal Motion Planning for Automated Driving, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 10/1/2018 - 10/5/2018, pp. 4523–4530. doi:10.1109/IROS.2018.8593813. (cited on pages 6, 38, 41, 58 and 90.)

[8] Z. Ajanović, B. Lacevic, G. Stettinger, D. Watzenig, M. Horn, Safe learning-based optimal motion planning for automated driving.
URL http://arxiv.org/pdf/1805.09994v2 (cited on pages 6, 41 and 103.)

[9] Z. Ajanović, H. Beglerovic, B. Lacevic, A novel approach to model exploration for value function learning, arXiv preprint arXiv:1906.02789 [Add to Citavi project by ArXiv ID]. (cited on pages 6, 41 and 105.)

[10] Z. Ajanović, E. Regolin, G. Stettinger, M. Horn, A. Ferrara, Search-Based Motion Planning for Performance Autonomous Driving, in: IAVSD Vehicles on Road and Tracks 2019, 2019. (cited on pages 6 and 99.)

[11] Z. Ajanović, M. Stolz, Predictive control system for autonomous driving vehicle (2016). (cited on page 6.)

[12] Z. Ajanović, M. Stolz, M. Horn, Energy-efficient driving in a dynamic environment, Virtual vehicle magazine. (cited on page 6.)

[13] 5 Trends Emerge in the Gartner Hype Cycle for Emerging Technologies, 2018.
URL \https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging-technologies-2018/ (cited on page 9.)

[14] Frankfurter Allgemeine Zeitung GmbH, Plattform von BMW und Daimler: Vollautonome Autos zu entwickeln, gleicht einer Mars-Mission.
URL https://www.faz.net/aktuell/wirtschaft/gleicht-der-weg-zu-vollautonomen-autos-einer-mars-mission-16075349.html?_lrsc=a92e582c-8584-4914-aae2-65bd2bbe417e (cited on page 9.)

[15] D. Watzenig, M. Horn, Introduction to Automated Driving, in: D. Watzenig, M. Horn (Eds.), Automated driving, Springer, Cham, 2016, pp. 3–16. doi:10.1007/978-3-319-31895-0{\textunderscore}1. (cited on page 9.)

[16] D. Milakis, B. van Arem, B. van Wee, Policy and society related implications of automated driving: A review of literature and directions for future research, Journal of Intelligent Transportation Systems 21 (4) (2017) 324–348. doi:10.1080/15472450.2017.1291351. (cited on pages 9, 11 and 12.)

[17] World Health Organization, et al., Global status report on road safety 2018. (cited on page 10.)

[18] M. A. Belin, R. Johansson, J. Lindberg, C. Tingvall, The Vision Zero and its consequences, in: Proceedings of the 4 th International Conference on Safety and the Environment in the 21 st Century, 1997, pp. 23–27. (cited on page 10.)

[19] S. Kockum, R. Örtlund, A. Ekfjorden, P. Wells, Volvo Trucks Safety Report 2017, Volvo Truck Corporation. (cited on page 10.)

[20] B. Fildes, M. Keall, N. Bos, A. Lie, Y. Page, C. Pastor, L. Pennisi, M. Rizzi, P. Thomas, C. Tingvall, Effectiveness of low speed autonomous emergency braking in real-world rear-end crashes, Accident; analysis and prevention 81 (2015) 24–29. doi:10.1016/j.aap.2015.03.029. (cited on page 10.)

[21] W. Wachenfeld, H. Winner, The Release of Autonomous Vehicles, in: M. Maurer, J. C. Gerdes, B. Lenz, H. Winner (Eds.), Autonomous driving, Springer Berlin Heidelberg, New York NY, 2016, pp. 425–449. doi:10.1007/978-3-662-48847-8{\textunderscore}21. (cited on pages 10 and 45.)

[22] European Environment Agency, Agencia Europea de Medio Ambiente, Towards a Resource Efficient Transport System: TERM 2009; Indicators Tracking Transport and Environment in the European Union, Office for Official Publications of the European Union, 2010. (cited on page 10.)

[23] US Environmental Protection Agency, Inventory of US greenhouse gas emissions and sinks: 1990–2010 (2012). (cited on page 10.)

[24] D. Schrank, B. Eisele, T. Lomax, 2014 Urban mobility report: powered by Inrix Traffic Data. (cited on page 10.)

[25] European Commission. Directorate-General for Energy, Keep Europe moving: sustainable mobility for our continent: mid-term review of the European Commission's 2001 Transport White Paper, Office for Official Publications of the European Communities, 2006. (cited on page 10.)

[26] C. Bingham, C. Walsh, S. Carroll, Impact of driving characteristics on electric vehicle energy consumption and range, IET Intelligent Transport Systems 6 (1) (2012) 29. doi:10.1049/iet-its.2010.0137. (cited on pages 11 and 25.)

[27] A. Alam, B. Besselink, V. Turri, J. Martensson, K. H. Johansson, Heavy-duty vehicle platooning for sustainable freight transportation: A cooperative method to enhance safety and efficiency, IEEE Control Systems Magazine 35 (6) (2015) 34–56. (cited on page 11.)

[28] Where the Energy Goes: Gasoline Vehicles.
URL https://www.fueleconomy.gov/feg/atv.shtml (cited on page 11.)

[29] B. Asadi, A. Vahidi, Predictive Cruise Control: Utilizing Upcoming Traffic Signal Information for Improving Fuel Economy and Reducing Trip Time, IEEE Transactions on Control Systems Technology 19 (3) (2011) 707–714. doi:10.1109/TCST.2010.2047860. (cited on pages 11 and 28.)

[30] A. Ferrara, S. Sacone, S. Siri, Freeway traffic modelling and control, Advances in industrial control, 1430-9491, Springer, Cham, Switzerland, 2018. (cited on page 11.)

[31] K. Dresner, P. Stone, A Multiagent Approach to Autonomous Intersection Management, Journal of Artificial Intelligence Research 31 (2008) 591–656. doi:10.1613/jair.2502. (cited on page 11.)

[32] D. A. Lazar, R. Pedarsani, K. Chandrasekher, D. Sadigh, Maximizing Road Capacity Using Cars that Influence People, in: 2018 IEEE Conference on Decision and Control (CDC), IEEE, [Piscataway, New Jersey], 2018?, pp. 1801–1808. doi:10.1109/CDC.2018.8619287. (cited on page 12.)

[33] B. Lenz, E. Fraedrich, New Mobility Concepts and Autonomous Driving: The Potential for Change, in: M. Maurer, J. C. Gerdes, B. Lenz, H. Winner (Eds.), Autonomous driving, Springer Berlin Heidelberg, New York NY, 2016, pp. 173–191. doi:10.1007/978-3-662-48847-8{\textunderscore}9. (cited on page 12.)

[34] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, D. Rus, On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment, Proceedings of the National Academy of Sciences of the United States of America 114 (3) (2017) 462–467. doi:10.1073/pnas.1611675114. (cited on page 12.)

[35] M. Maurer, J. C. Gerdes, B. Lenz, H. Winner (Eds.), Autonomous driving, Springer Berlin Heidelberg, New York NY, 2016. doi:10.1007/978-3-662-48847-8. (cited on page 13.)

[36] H. M. Morrison, A. F. Welch, E. A. Hanysz, Automatic highway and driver aid developments, SAE Transactions 69 (1961) 31–53. (cited on page 13.)

[37] K. Gardels, Automatic car controls for electronic highways. (cited on page 13.)

[38] J. C. Maxwell, I. On governors, Proceedings of the Royal Society of London (16) (1868) 270–283. (cited on page 13.)

[39] Stuart Bennett, A brief history of automatic control, IEEE Control Systems Magazine 16 (3) (1996) 17–25. doi:10.1109/37.506394. (cited on page 13.)

[40] N. Wiener, Cybernetics or Control and Communication in the Animal and the Machine, Technology Press, 1948. (cited on page 13.)

[41] N. J. Nilsson, The quest for artificial intelligence: A history of ideas and achievements / Nils J. Nilsson, Cambridge University Press, Cambridge, 2010. (cited on page 13.)

[42] J. Moor, The Dartmouth College artificial intelligence conference: The next fifty years, AI Magazine 27 (4) (2006) 87. (cited on page 13.)

[43] R. Bellman, On the Theory of Dynamic Programming, Proceedings of the National Academy of Sciences of the United States of America 38 (8) (1952) 716–719. (cited on pages 13, 36, 52 and 77.)

[44] L. S. Pontryagin, Mathematical Theory of Optimal Processes, Routledge, 2018. doi:10.1201/9780203749319. (cited on page 13.)

[45] J. Richalet, A. Rault, J. L. Testud, J. Papon, Model predictive heuristic control, Automatica 14 (5) (1978) 413–428. doi:10.1016/0005-1098(78)90001-8. (cited on page 13.)

[46] C. E. García, D. M. Prett, M. Morari, Model predictive control: Theory and practice—A survey, Automatica 25 (3) (1989) 335–348. doi:10.1016/0005-1098(89)90002-2. (cited on page 13.)

[47] N. J. Nilsson, Shakey the Robot.
URL https://apps.dtic.mil/dtic/tr/fulltext/u2/a458918.pdf (cited on page 14.)

[48] P. Hart, N. Nilsson, B. Raphael, A Formal Basis for the Heuristic Determination of Minimum Cost Paths, IEEE Transactions on Systems Science and Cybernetics 4 (2) (1968) 100–107. doi:10.1109/TSSC.1968.300136. (cited on pages 14, 36, 37, 81, 82, 91, 95, 99 and 102.)

[49] R. O. Duda, P. E. Hart, Use of the Hough Transformation to Detect Lines and Curves in Pictures.
URL https://apps.dtic.mil/dtic/tr/fulltext/u2/a457992.pdf (cited on page 14.)

[50] B. Kuipers, E. A. Feigenbaum, P. E. Hart, N. J. Nilsson, Shakey: From Conception to History, AI Magazine 38 (1) (2017) 88–103. (cited on page 14.)

[51] H. P. Moravec, The Stanford Cart and the CMU Rover, Proceedings of the IEEE 71 (7) (1983) 872–884. doi:10.1109/PROC.1983.12684. (cited on page 14.)

[52] S. Tsugawa, T. Yatabe, T. Hirose, S. Matsumoto, An automobile with artificial intelligence, in: Proceedings of the 6th international joint conference on Artificial intelligence-Volume 2, 1979, pp. 893–895. (cited on page 14.)

[53] The man who invented the self-driving car (in 1986) (2018).
URL https://www.politico.eu/article/delf-driving-car-born-1986-ernst-dickmanns
(cited on page 14.)

[54] E. d. Dickmanns, A. Zapp, A Curvature-based Scheme for Improving Road Vehicle Guidance by Computer Vision, in: N. Marquina, W. J. Wolfe (Eds.), Mobile Robots I, SPIE Proceedings, SPIE, 1986, p. 161. doi:10.1117/12.937795. (cited on page 14.)

[55] M. Williams, PROMETHEUS-The European research programme for optimising the road transport system in Europe, in: IEE Colloquium on Driver Information, 1988, p. 1. (cited on page 14.)

[56] D. Ferguson, T. M. Howard, M. Likhachev, Motion planning in urban environments, Journal of Field Robotics 25 (11-12) (2008) 939–960. doi:10.1002/rob.20265. (cited on page 14.)

[57] M. A. Turk, D. G. Morgenthaler, K. D. Gremban, M. Marra, VITS-a vision system for autonomous land vehicle navigation, IEEE Transactions on Pattern Analysis and Machine Intelligence 10 (3) (1988) 342–361. doi:10.1109/34.3899. (cited on page 14.)

[58] T. Jochem, D. Pomerleau, B. Kumar, J. Armstrong, PANS: a portable navigation platform, in: Intelligent vehicles '95, IEEE, 1995, pp. 107–112. doi:10.1109/IVS.1995.528266. (cited on page 14.)

[59] D. E. Rumelhart, G. E. Hinton, R. J. Williams, et al., Learning representations by back-propagating errors, Cognitive modeling 5 (3) (1988) 1. (cited on page 14.)

[60] Dean A. Pomerleau, Alvinn: An autonomous land vehicle in a neural network, in: Advances in neural information processing systems, 1989, pp. 305–313. (cited on page 15.)

[61] D. Pomerleau, T. Jochem, Rapidly adapting machine vision for automated vehicle steering, IEEE Expert 11 (2) (1996) 19–27. doi:10.1109/64.491277. (cited on page 15.)

[62] California PATH.
URL https://path.berkeley.edu/ (cited on page 15.)

[63] S. E. Shladover, PATH at 20—History and Major Milestones, IEEE Transactions on Intelligent Transportation Systems 8 (4) (2007) 584–592. doi:10.1109/TITS.2007.903052. (cited on page 15.)

[64] M. Parent, P.-Y. Texier, F. Leurent, F. Dumontet, Design and Implementation of a Public Transportation System Based on Self-Service Electric Cars, IFAC Proceedings Volumes 27 (12) (1994) 483–488. doi:10.1016/S1474-6670(17)47517-3. (cited on page 16.)

[65] C. Laugier, S. Sekhavat, F. Large, J. Hermosillo, Z. Shiller, Some Steps Towards Autonomous Cars, IFAC Proceedings Volumes 34 (19) (2001) 9–17. doi:10.1016/S1474-6670(17)33106-3. (cited on page 16.)

[66] M. Parent, Automated public vehicles: A first step towards the automated highway, in: MOBILITY FOR EVERYONE. 4TH WORLD CONGRESS ON INTELLIGENT TRANSPORT SYSTEMS, 21-24 OCTOBER 1997, BERLIN.(PAPER NO. 2389), 1997. (cited on page 16.)

[67] COMM/RTD, Cybercars, the ecological future of automobile technology | Report Summary | CYBERCARS | FP5 | CORDIS | European Commission (04.03.2019).
URL https://cordis.europa.eu/project/rcn/61003/reporting/en (cited on page 16.)

[68] CORDIS, Close Communications for Cooperation between Cybercars | Projects | FP6 | CORDIS | European Commission (19.03.2019).
URL `https://cordis.europa.eu/project/rcn/80600/factsheet/en` (cited on page 16.)

[69] I. E. Paromtchik, C. Laugier, Autonomous parallel parking of a nonholonomic vehicle, in: Proceedings of the 1996 IEEE intelligent vehicles symposium, IEEE, 1996, pp. 13–18. `doi:10.1109/IVS.1996.566343`. (cited on page 16.)

[70] J. Baber, J. Kolodko, T. Noel, M. Parent, L. Vlacic, Cooperative autonomous driving - Intelligent vehicles sharing city roads cooperative autonomous driving, IEEE Robotics & Automation Magazine 12 (1) (2005) 44–49. `doi:10.1109/MRA.2005.1411418`. (cited on page 16.)

[71] M. Buehler, K. Iagnemma, S. Singh, The 2005 DARPA grand challenge: the great robot race, Vol. 36, Springer, 2007. (cited on page 16.)

[72] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, P. Mahoney, Stanley: The robot that won the DARPA Grand Challenge, Journal of Field Robotics 23 (9) (2006) 661–692. `doi:10.1002/rob.20147`. (cited on page 16.)

[73] M. Buehler, K. Iagnemma, S. Singh, The DARPA urban challenge: autonomous vehicles in city traffic, Vol. 56, Springer, 2009. (cited on page 16.)

[74] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. R. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, D. Ferguson, Autonomous driving in urban environments: Boss and the Urban Challenge, Journal of Field Robotics 25 (8) (2008) 425–466. `doi:10.1002/rob.20255`. (cited on pages 16 and 30.)

[75] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, S. Thrun, Junior: The Stanford entry in the Urban Challenge, Journal of Field Robotics 25 (9) (2008) 569–597. `doi:10.1002/rob.20258`. (cited on pages 16, 30, 31 and 92.)

[76] J. Hurdus, A. Bacha, C. Bauman, S. Cacciola, R. Faruque, P. King, C. Terwelp, P. Currier, D. Hong, A. Wicks, C. Reinholtz, VictorTango Architecture for Autonomous Navigation in the DARPA Urban Challenge, Journal of

Aerospace Computing, Information, and Communication 5 (12) (2008) 506–529. doi:10.2514/1.40547. (cited on page 16.)

[77] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, A perception–driven autonomous urban vehicle, Journal of field Robotics 25 (10) (2008) 727–774. (cited on page 16.)

[78] M. Bertozzi, A. Broggi, E. Cardarelli, R. Fedriga, L. Mazzei, P. Porta, VIAC Expedition Toward Autonomous Mobility [From the Field], IEEE Robotics & Automation Magazine 18 (3) (2011) 120–124. doi:10.1109/MRA.2011.942490. (cited on page 16.)

[79] T. Robinson, E. Chan, E. Coelingh, Operating platoons on public motorways: An introduction to the sartre platooning programme, in: 17th world congress on intelligent transport systems, Vol. 1, 2010, p. 12. (cited on page 16.)

[80] E. van Nunen, M. R. J. A. E. Kwakkernaat, J. Ploeg, B. d. Netten, Cooperative Competition for Future Mobility, IEEE Transactions on Intelligent Transportation Systems 13 (3) (2012) 1018–1025. doi:10.1109/TITS.2012.2200475. (cited on page 17.)

[81] A. Geiger, M. Lauer, F. Moosmann, B. Ranft, H. Rapp, C. Stiller, J. Ziegler, Team AnnieWAY's Entry to the 2011 Grand Cooperative Driving Challenge, IEEE Transactions on Intelligent Transportation Systems 13 (3) (2012) 1008–1017. doi:10.1109/TITS.2012.2189882. (cited on page 17.)

[82] J. Funke, P. Theodosis, R. Hindiyeh, G. Stanek, K. Kritatakirana, C. Gerdes, D. Langer, M. Hernandez, B. Muller-Bessler, B. Huhnke, Up to the limits: Autonomous Audi TTS, in: IEEE Intelligent Vehicles Symposium (IV), 2012, IEEE, Piscataway, NJ, 2012, pp. 541–547. doi:10.1109/IVS.2012.6232212. (cited on page 17.)

[83] Google gets first self-driven car license in Nevada | Reuters (2012). URL https://www.reuters.com/article/uk-usa-nevada-google/google-gets-first-self-driven-car-license-in-nevada-idUSLNE84701320120508 (cited on page 17.)

[84] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, E. Kaus, R. G. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knoppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, E. Zeeb, Making Bertha Drive—An Autonomous Journey on a Historic Route, IEEE Intelligent Transportation Systems Magazine 6 (2) (2014) 8–20. doi:10.1109/MITS.2014.2306552. (cited on page 17.)

[85] Tesla beams down 'autopilot' mode to Model S (2015). URL https://www.autonews.com/article/20151014/OEM06/151019938/tesla-beams-down-autopilot-mode-to-model-s (cited on page 17.)

[86] Benjamin Zhang, ELON MUSK: In 2 years your Tesla will be able to drive from New York to LA and find you.
URL https://finance.yahoo.com/news/elon-musk-two-years-car-202858960.html?guccounter=1&guce_referrer=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnLw&guce_referrer_sig=AQAAAFKYHuV2Hw9GIkRCIFvWDThATFA10B65TTAhk4OWRoRfMWJV8Hjtl2AAvPK1QwzuyZ1E0_SXQUr4_lUYJaSI8ZVADQ4eT5Qyqr3YxTEx7-7EhHrYu5zPKtysaf78bpeE0WMkh6V (cited on page 17.)

[87] On the Road – Waymo.
URL https://waymo.com/ontheroad/ (cited on page 17.)

[88] W. Team, Waymo One: The next step on our self-driving journey (2018).
URL https://medium.com/waymo/waymo-one-the-next-step-on-our-self-driving-journ
(cited on pages 17, 19 and 22.)

[89] On-Road Automated Driving (ORAD) committee, Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. doi:10.4271/J3016{\textunderscore}201806. (cited on pages 18, 44 and 46.)

[90] Autopilot (28.03.2019).
URL https://www.tesla.com/autopilot (cited on page 19.)

[91] What Is Mercedes-Benz DISTRONIC PLUS®? | Mercedes-Benz of San Diego (2018).
URL https://www.mbsd.com/mercedes-benz-distronic-plus/ (cited on page 19.)

[92] S. Ulbrich, A. Reschka, J. Rieken, S. Ernst, G. Bagschik, F. Dierkes, M. Nolte, M. Maurer, Towards a Functional System Architecture for Automated Vehicles.
URL http://arxiv.org/pdf/1703.08557v2 (cited on pages 19, 45 and 167.)

[93] S. J. Russell, P. Norvig, Artificial Intelligence : A Modern Approach, Malaysia and Pearson Education Limited, 2016. (cited on pages 19 and 44.)

[94] J. Ibañez-Guzmán, C. Laugier, J.-D. Yoder, S. Thrun, Autonomous Driving: Context and State-of-the-Art, in: A. Eskandarian (Ed.), Handbook of intelligent vehicles, Springer reference, Springer, London, 2012]-, pp. 1271–1310. doi:10.1007/978-0-85729-085-4{\textunderscore}50. (cited on page 20.)

[95] B. Okumura, M. R. James, Y. Kanzawa, M. Derry, K. Sakai, T. Nishi, D. Prokhorov, Challenges in perception and decision making for intelligent automotive vehicles: A case study, IEEE Transactions on Intelligent Vehicles 1 (1) (2016) 20–32. (cited on page 20.)

[96] J. van Brummelen, M. O'Brien, D. Gruyer, H. Najjaran, Autonomous vehicle perception: The technology of today and tomorrow, Transportation Research Part C: Emerging Technologies 89 (2018) 384–406. doi:10.1016/j.trc.2018.02.012. (cited on page 20.)

[97] H. Bast, D. Delling, A. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner, R. F. Werneck, Route planning in transportation networks, in: Algorithm engineering, Springer, 2016, pp. 19–80. (cited on pages 21 and 26.)

[98] S. Lefèvre, D. Vasquez, C. Laugier, A survey on motion prediction and risk assessment for intelligent vehicles, ROBOMECH journal 1 (1) (2014) 1. (cited on page 21.)

[99] D. Sadigh, S. Sastry, S. A. Seshia, A. D. Dragan, Planning for autonomous cars that leverage effects on human actions, in: Robotics: Science and Systems, Vol. 2, 2016. (cited on pages 21 and 30.)

[100] Z. Lu, B. Shyrokau, B. Boulkroune, S. van Aalst, R. Happee, Performance benchmark of state-of-the-art lateral path-following controllers, in: I. I. W. o. A. M. Control (Ed.), Proceedings 2018 IEEE 15th International Workshop on Advanced Motion Control (AMC), IEEE, Piscataway, NJ, 2018, pp. 541–546. doi:10.1109/AMC.2019.8371151. (cited on page 22.)

[101] A. Sorniotti, P. Barber, S. de Pinto, Path Tracking for Automated Driving: A Tutorial on Control System Formulations and Ongoing Research, in: D. Watzenig, M. Horn (Eds.), Automated driving, Vol. 53, Springer, Cham, 2016, pp. 71–140. doi:10.1007/978-3-319-31895-0{\textunderscore}5. (cited on page 22.)

[102] K. L. Talvala, K. Kritayakirana, J. C. Gerdes, Pushing the limits: From lane-keeping to autonomous racing, Annual Reviews in Control 35 (1) (2011) 137–148. doi:10.1016/j.arcontrol.2011.03.009. (cited on page 22.)

[103] H. Winner, S. Witte, W. Uhler, B. Lichtenberg, Adaptive Cruise Control System Aspects and Development Trends, in: SAE Technical Paper Series, SAE Technical Paper Series, SAE International400 Commonwealth Drive, Warrendale, PA, United States, 1996. doi:10.4271/961010. (cited on page 22.)

[104] R. Felton, Google's Self-Driving Cars Have Trouble With Basic Driving Tasks: Report.
URL https://jalopnik.com/googles-self-driving-cars-have-trouble-with-basic-dri
(cited on page 22.)

[105] S. M. LaValle, Planning algorithms, Cambridge University Press, Cambridge, 2006. (cited on pages 23, 34 and 35.)

[106] Car Myths, Debunked! (2010).
URL https://www.popularmechanics.com/science/g475/15-favorite-mythbusters-car-myths/?slide=1 (cited on page 24.)

[107] C. F. Minett, A. M. Salomons, W. Daamen, B. van Arem, S. Kuijpers, Eco-routing: Comparing the fuel consumption of different routes between an origin and destination using field test speed profiles and synthetic speed profiles, in: C. Herget (Ed.), 2011 IEEE Forum on Integrated and Sustainable Transportation

Systems (FISTS), IEEE, Piscataway, NJ, 2011, pp. 32–39. doi:10.1109/FISTS.2011.5973621. (cited on page 26.)

[108] L. Hu, Y. Zhong, W. Hao, B. Moghimi, J. Huang, X. Zhang, R. Du, Optimal Route Algorithm Considering Traffic Light and Energy Consumption, IEEE Access 6 (2018) 59695–59704. doi:10.1109/ACCESS.2018.2871843. (cited on page 26.)

[109] K. ICHIKAWA, Application of Optimization Theory for Bounded State Variable Problems to the Operation of Train, Bulletin of JSME 11 (47) (1968) 857–865. doi:10.1299/jsme1958.11.857. (cited on page 26.)

[110] P. Howlett, The optimal control of a train, Annals of Operations Research 98 (1-4) (2000) 65–87. (cited on page 26.)

[111] J. N. Hooker, A. B. Rose, G. F. Roberts, Optimal Control of Automobiles for Fuel Economy, Transportation Science 17 (2) (1983) 146–167. doi:10.1287/trsc.17.2.146. (cited on page 26.)

[112] J. N. Hooker, Optimal driving for single-vehicle fuel economy, Transportation Research Part A: General 22 (3) (1988) 183–201. doi:10.1016/0191-2607(88)90036-2. (cited on page 26.)

[113] E. G. Gilbert, Vehicle cruise: Improved fuel economy by periodic control, Automatica 12 (2) (1976) 159–166. doi:10.1016/0005-1098(76)90079-0. (cited on page 26.)

[114] E. Hellström, Explicit use of road topography for model predictive cruise control in heavy trucks (2005). (cited on page 26.)

[115] E. Hellström, Look-ahead control of heavy vehicles, Vol. 1315 of Linköping studies in science and technology. Dissertations, Department of Eelectrical Engineering, Linköping University, Linköping, 2010. (cited on page 26.)

[116] B. Saerens, Optimal control based eco-driving, Theoretical Approach and Practical Applications. Heverlee: Katholieke Universiteit Leuven. (cited on pages 26 and 67.)

[117] W. Dib, L. Serrao, A. Sciarretta, Optimal control to minimize trip time and energy consumption in electric vehicles, in: 2011 IEEE vehicle power and propulsion conference (VPPC 2011), IEEE, Piscataway NJ, 2011, pp. 1–8. doi:10.1109/VPPC.2011.6043133. (cited on page 26.)

[118] F. Mensing, R. Trigui, E. Bideaux, Vehicle trajectory optimization for application in ECO-driving, in: 2011 IEEE vehicle power and propulsion conference (VPPC 2011), IEEE, Piscataway NJ, 2011, pp. 1–6. doi:10.1109/VPPC.2011.6042993. (cited on page 26.)

[119] E. Ozatay, S. Onori, J. Wollaeger, U. Ozguner, G. Rizzoni, D. Filev, J. Michelini, S. Di Cairano, Cloud-Based Velocity Profile Optimization for Everyday Driving: A Dynamic-Programming-Based Solution, IEEE Transactions on Intelligent Transportation Systems 15 (6) (2014) 2491–2505. doi:10.1109/TITS.2014.2319812. (cited on page 26.)

[120] V.-D. Doan, H. Fujimoto, T. Koseki, T. Yasuda, H. Kishi, T. Fujita, Iterative Dynamic Programming for Optimal Control Problem with Isoperimetric Constraint and Its Application to Optimal Eco-driving Control of Electric Vehicle, IEEJ Journal of Industry Applications 7 (1) (2018) 80–92. doi:10.1541/ieejjia.7.80. (cited on page 26.)

[121] D. Grossoleil, D. Meizel, Practical design of minimal energy controls for an electric bicycle, in: 9th International Conference on Modeling, Optimization & SIMulation, 2012. (cited on pages 26 and 28.)

[122] Wissam Dib, Alexandre Chasse, Philippe Moulin, Antonio Sciarretta, Gilles Corde, Optimal energy management for an electric vehicle in eco-driving applications, Control Engineering Practice 29 (2014) 299–307. doi:10.1016/j.conengprac.2014.01.005.
URL http://www.sciencedirect.com/science/article/pii/S0967066114000355 (cited on page 26.)

[123] E. Hellström, M. Ivarsson, L. Nielsen, et al., Look-ahead control for heavy trucks to minimize trip time and fuel consumption, IFAC Proceedings Volumes 40 (10) (2007) 439–446. (cited on page 26.)

[124] C. Flores, V. Milanes, J. Perez, D. Gonzalez, F. Nashashibi, Optimal energy consumption algorithm based on speed reference generation for urban electric vehicles, in: 2015 IEEE Intelligent Vehicles Symposium (IV), IEEE, Piscataway, NJ, 2015, pp. 730–735. doi:10.1109/IVS.2015.7225771. (cited on page 26.)

[125] A. Gaier, A. Asteroth, Evolution of optimal control for energy-efficient transport, in: 2014 IEEE Intelligent Vehicles Symposium Proceedings, 2014, pp. 1121–1126. (cited on pages 26 and 28.)

[126] S. Xu, H. Peng, Design and Comparison of Fuel-Saving Speed Planning Algorithms for Automated Vehicles, IEEE Access 6 (2018) 9070–9080. doi:10.1109/ACCESS.2018.2805883. (cited on page 26.)

[127] M. A. S. Kamal, M. Mukai, J. Murata, T. Kawabe, Ecological Vehicle Control on Roads With Up-Down Slopes, IEEE Transactions on Intelligent Transportation Systems 12 (3) (2011) 783–794. doi:10.1109/TITS.2011.2112648. (cited on page 26.)

[128] A. Gaier, A. Asteroth, Evolving look ahead controllers for energy optimal driving and path planning, in: Proceedings, IEEE, Piscataway, NJ, 2014, pp. 138–145. doi:10.1109/INISTA.2014.6873610. (cited on page 27.)

[129] S. Xu, S. E. Li, B. Cheng, K. Li, Instantaneous Feedback Control for a Fuel-Prioritized Vehicle Cruising System on Highways With a Varying Slope, IEEE Transactions on Intelligent Transportation Systems 18 (5) (2017) 1210–1220. doi:10.1109/TITS.2016.2600641. (cited on pages 27 and 85.)

[130] F. Mensing, E. Bideaux, R. Trigui, H. Tattegrain, Trajectory optimization for eco-driving taking into account traffic constraints, Transportation Research Part D: Transport and Environment 18 (2013) 55–61.
URL https://www.sciencedirect.com/science/article/pii/S1361920912001137 (cited on page 27.)

[131] F. Flehmig, A. Sardari, U. Fischer, A. Wagner, Energy optimal Adaptive Cruise Control during following of other vehicles, in: 2015 IEEE Intelligent Vehicles Symposium (IV), IEEE, Piscataway, NJ, 2015, pp. 724–729. doi:10.1109/IVS.2015.7225770. (cited on pages 27 and 29.)

[132] M. Vajedi, N. L. Azad, Ecological Adaptive Cruise Controller for Plug-In Hybrid Electric Vehicles Using Nonlinear Model Predictive Control, IEEE Transactions on Intelligent Transportation Systems 17 (1) (2016) 113–122. doi:10.1109/TITS.2015.2462843. (cited on page 27.)

[133] M. Wang, W. Daamen, S. P. Hoogendoorn, B. van Arem, Rolling horizon control framework for driver assistance systems. Part I: Mathematical formulation and non-cooperative systems, Transportation Research Part C: Emerging Technologies 40 (2014) 271–289. (cited on page 27.)

[134] M. Wang, W. Daamen, S. P. Hoogendoorn, B. van Arem, Rolling horizon control framework for driver assistance systems. Part II: Cooperative sensing and cooperative control, Transportation Research Part C: Emerging Technologies 40 (2014) 290–311. (cited on page 27.)

[135] M. Wang, S. P. Hoogendoorn, W. Daamen, B. van Arem, R. Happee, Game theoretic approach for predictive lane-changing and car-following control, Transportation Research Part C: Emerging Technologies 58 (2015) 73–92.
URL https://www.sciencedirect.com/science/article/pii/S0968090X15002491 (cited on pages 27 and 67.)

[136] N. Murgovski, J. Sjoberg, Predictive cruise control with autonomous overtaking, in: I. C. o. D. a. Control (Ed.), 2015 54th IEEE Conference on Decision and Control (CDC), IEEE, [Piscataway, NJ], 2015, pp. 644–649. doi:10.1109/CDC.2015.7402302. (cited on page 27.)

[137] M. A. S. Kamal, S. Taguchi, T. Yoshimura, Efficient Driving on Multilane Roads Under a Connected Vehicle Environment, IEEE Transactions on Intelligent Transportation Systems 17 (9) (2016) 2541–2551. doi:10.1109/TITS.2016.2519526. (cited on page 27.)

[138] T. Shamir, How should an autonomous vehicle overtake a slower moving vehicle: Design and analysis of an optimal trajectory, IEEE Transactions on Automatic Control 49 (4) (2004) 607–610. (cited on page 27.)

[139] T. J. Li, M. Shen, H. Zheng, Overtaking or Merging? Eco-Routing Decision and Speed Trajectory with Full Terrain Information. (cited on page 27.)

[140] G. de Nunzio, C. C. de Wit, P. Moulin, D. Di Domenico, Eco-driving in urban traffic networks using traffic signals information, International Journal of Robust and Nonlinear Control 26 (6) (2016) 1307–1324. doi:10.1002/rnc.3469. (cited on page 28.)

[141] E. Kural, S. Jones, A. F. Parrilla, A. Grauers, Traffic light assistant system for optimized energy consumption in an electric vehicle, in: 2014 International Conference on Connected Vehicles and Expo (ICCVE), IEEE, Piscataway, NJ, 2014, pp. 604–611. doi:10.1109/ICCVE.2014.7297619. (cited on page 28.)

[142] G. Mahler, A. Vahidi, Reducing idling at red lights based on probabilistic prediction of traffic signal timings, in: Ieee (Ed.), 2012 American Control Conference, IEEE, [Place of publication not identified], 2012, pp. 6557–6562. doi:10.1109/ACC.2012.6314942. (cited on page 28.)

[143] H. Chen, L. Guo, H. Ding, Y. Li, B. Gao, Real-Time Predictive Cruise Control for Eco-Driving Taking into Account Traffic Constraints, IEEE Transactions on Intelligent Transportation Systems (2018) 1–11doi:10.1109/TITS.2018.2868518. (cited on page 28.)

[144] L. Zhang, W. Liang, X. Zheng, Eco-Driving for Public Transit in Cyber-Physical Systems Using V2I Communication, International Journal of Intelligent Transportation Systems Research 16 (2) (2018) 79–89. doi:10.1007/s13177-017-0139-1. (cited on page 28.)

[145] S. Bae, Y. Kim, J. Guanetti, F. Borrelli, S.J. Moura, Design and Implementation of Ecological Adaptive Cruise Control for Autonomous Driving with Communication to Traffic Lights, in: American Control Conference (ACC), 2019. (cited on page 28.)

[146] L. Eckstein, J. Bock, P. Themann, Optimisation of energy efficiency based on average driving behaviour and driver's preferences for automated driving, IET Intelligent Transport Systems 9 (1) (2015) 50–58. doi:10.1049/iet-its.2013.0121. (cited on page 28.)

[147] O. Chevrant-Breton, T. Guan, C. W. Frey, Search space reduction in dynamic programming using monotonic heuristics in the context of model predictive optimization, in: Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on, IEEE, [Piscataway, N.J.], 2014?, pp. 2113–2118. doi:10.1109/ITSC.2014.6958015. (cited on page 28.)

[148] J. Guanetti, Y. Kim, F. Borrelli, Control of connected and automated vehicles: State of the art and future challenges, Annual Reviews in Control 45 (2018) 18–40. doi:10.1016/j.arcontrol.2018.04.011. (cited on page 29.)

[149] S. La Valle, Motion Planning, IEEE Robotics & Automation Magazine 18 (2) (2011) 108–118. doi:10.1109/MRA.2011.941635. (cited on page 29.)

[150] D. Gonzalez, J. Perez, V. Milanes, F. Nashashibi, A Review of Motion Planning Techniques for Automated Vehicles, IEEE Transactions on Intelligent Transportation Systems 17 (4) (2016) 1135–1145. doi:10.1109/TITS.2015.2498841. (cited on page 29.)

[151] S. M. Veres, L. Molnar, N. K. Lincoln, C. P. Morice, Autonomous vehicle control systems — a review of decision making, Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 225 (2) (2011) 155–195. doi:10.1177/2041304110394727. (cited on page 29.)

[152] S. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. Eng, D. Rus, M. Ang, Perception, Planning, Control, and Coordination for Autonomous Vehicles, Machines 5 (1) (2017) 6. doi:10.3390/machines5010006. (cited on page 29.)

[153] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, O. Koch, Y. Kuwata, D. Moore, E. Olson, S. Peters, J. Teo, R. Truax, M. Walter, D. Barrett, A. Epstein, K. Maheloni, K. Moyer, T. Jones, R. Buckley, M. Antone, R. Galejs, S. Krishnamurthy, J. Williams, A perception-driven autonomous urban vehicle, Journal of Field Robotics 25 (10) (2008) 727–774. doi:10.1002/rob.20262. (cited on page 30.)

[154] S. Ulbrich, M. Maurer, Probabilistic online POMDP decision making for lane changes in fully automated driving, in: 16th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2013, IEEE, Piscataway, NJ, 2013, pp. 2063–2067. doi:10.1109/ITSC.2013.6728533. (cited on page 30.)

[155] J. Ziegler, P. Bender, T. Dang, C. Stiller, Trajectory planning for Bertha — A local, continuous method, in: 2014 IEEE Intelligent Vehicles Symposium Proceedings, IEEE, 6/8/2014 - 6/11/2014, pp. 450–457. doi:10.1109/IVS.2014.6856581. (cited on pages 30 and 92.)

[156] J. Ziegler, C. Stiller, Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, IEEE, Piscataway, NJ, 2009, pp. 1879–1884. doi:10.1109/IROS.2009.5354448. (cited on pages 30 and 31.)

[157] M. McNaughton, C. Urmson, J. M. Dolan, J.-W. Lee, Motion planning for autonomous driving with a conformal spatiotemporal lattice, in: ICRA 2011, IEEE, [Piscataway, N.J.], 2011, pp. 4889–4895. doi:10.1109/ICRA.2011.5980223. (cited on pages 30 and 31.)

[158] P. Bender, O. S. Tas, J. Ziegler, C. Stiller, The combinatorial aspect of motion planning: Maneuver variants in structured environments, in: 2015 IEEE Intelligent Vehicles Symposium (IV), IEEE, Piscataway, NJ, 2015, pp. 1386–1392. doi:10.1109/IVS.2015.7225909. (cited on page 30.)

[159] J. Nilsson, J. Sjoberg, Strategic decision making for automated driving on two-lane, one way roads using model predictive control, in: 2013 IEEE Intelligent Vehicles Symposium (IV), IEEE, 6/23/2013 - 6/26/2013, pp. 1253–1258. doi:10.1109/IVS.2013.6629638. (cited on page 30.)

[160] X. Qian, F. Altche, P. Bender, C. Stiller, A. de La Fortelle, Optimal trajectory planning for autonomous driving integrating logical constraints: An MIQP perspective, in: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), IEEE, [Piscataway, New Jersey], 2016, pp. 205–210. doi:10.1109/ITSC.2016.7795555. (cited on page 30.)

[161] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, A. R. Girard, Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems, IEEE Transactions on control systems technology 26 (5) (2018) 1782–1797. (cited on page 30.)

[162] M. Naumann, C. Stiller, Towards Cooperative Motion Planning for Automated Vehicles in Mixed Traffic.
URL http://arxiv.org/pdf/1708.06962v1 (cited on page 30.)

[163] A. Liniger, J. Lygeros, A Noncooperative Game Approach to Autonomous Racing, IEEE Transactions on Control Systems Technology (2019) 1–14doi:10.1109/TCST.2019.2895282. (cited on pages 30 and 33.)

[164] C. Hubmann, J. Schulz, G. Xu, D. Althoff, C. Stiller, A Belief State Planner for Interactive Merge Maneuvers in Congested Traffic, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 11/4/2018 - 11/7/2018, pp. 1617–1624. doi:10.1109/ITSC.2018.8569729. (cited on page 30.)

[165] W. Song, B. Su, G. Xiong, S. Li, Intention-aware Decision Making in Urban Lane Change Scenario for Autonomous Driving, in: 2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES), IEEE, Piscataway, NJ, 2018, pp. 1–8. doi:10.1109/ICVES.2018.8519506. (cited on page 30.)

[166] C. Stiller, W. Burgard, B. Deml, L. Eckstein, F. Flemisch, Kooperativ interagierende Automobile, at - Automatisierungstechnik 66 (2) (2018) 81–99. doi:10.1515/auto-2017-0129. (cited on page 30.)

[167] T. Fraichard, Dynamic trajectory planning with dynamic constraints: A 'state-time space' approach, in: IROS '93, Institute of Electrical and Electronics Engineers, Piscataway, N.J., 1993, pp. 1393–1400. doi:10.1109/IROS.1993.583794. (cited on pages 30 and 63.)

[168] D. Dolgov, S. Thrun, M. Montemerlo, J. Diebel, Practical search techniques in path planning for autonomous driving, Ann Arbor 1001 (48105) (2008) 18–80. (cited on page 31.)

[169] M. Likhachev, D. Ferguson, Planning Long Dynamically Feasible Maneuvers for Autonomous Vehicles, The International Journal of Robotics Research 28 (8) (2009) 933–945. doi:10.1177/0278364909340445. (cited on page 31.)

[170] C. Chen, M. Rickert, A. Knoll, Kinodynamic motion planning with Space-Time Exploration Guided Heuristic Search for car-like robots in dynamic environments, in: W. Burgard (Ed.), 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Piscataway, NJ, 2015, pp. 2666–2671. doi:10.1109/IROS.2015.7353741. (cited on page 31.)

[171] Z. Boroujeni, D. Goehring, F. Ulbrich, D. Neumann, R. Rojas, Flexible unit A-star trajectory planning for autonomous vehicles on structured road maps, in: I. I. C. o. V. E. a. Safety (Ed.), 2017 IEEE International Conference on Vehicular Electronics and Safety (ICVES), IEEE, [Piscataway, NJ], 2017, pp. 7–12. doi:10.1109/ICVES.2017.7991893. (cited on page 31.)

[172] W. Lim, S. Lee, M. Sunwoo, K. Jo, Hierarchical Trajectory Planning of an Autonomous Car Based on the Integration of a Sampling and an Optimization Method, IEEE Transactions on Intelligent Transportation Systems 19 (2) (2018) 613–626. doi:10.1109/TITS.2017.2756099. (cited on page 31.)

[173] H. Fan, F. Zhu, C. Liu, L. Zhang, L. Zhuang, D. Li, W. Zhu, J. Hu, H. Li, Q. Kong, Baidu Apollo EM Motion Planner.
URL http://arxiv.org/pdf/1807.08048v1 (cited on page 31.)

[174] M. I. Valls, H. F. Hendrikx, V. J. Reijgwart, F. V. Meier, I. Sa, R. Dube, A. Gawel, M. Burki, R. Siegwart, Design of an Autonomous Racecar: Perception, State Estimation and System Integration, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 21.05.2018 - 25.05.2018, pp. 2048–2055. doi:10.1109/ICRA.2018.8462829. (cited on page 32.)

[175] J. Betz, A. Wischnewski, A. Heilmeier, F. Nobis, T. Stahl, L. Hermansdorfer, B. Lohmann, M. Lienkamp, What can we learn from autonomous level-5 motorsport?, in: P. Pfeffer (Ed.), 9th International Munich Chassis Symposium 2018, Proceedings, Springer Vieweg, Wiesbaden, 2019, pp. 123–146. doi:10.1007/978-3-658-22050-1{\textunderscore}12. (cited on page 32.)

[176] A. Liniger, A. Domahidi, M. Morari, Optimization-based autonomous racing of 1:43 scale RC cars, Optimal Control Applications and Methods 36 (5) (2015) 628–647. doi:10.1002/oca.2123. (cited on page 32.)

[177] J. Z. Kolter, C. Plagemann, D. T. Jackson, A. Y. Ng, S. Thrun, A probabilistic approach to mixed open-loop and closed-loop control, with application to extreme autonomous driving, in: 2010 IEEE International Conference

on Robotics and Automation, I E E E, Piscataway, Jan. 2010, pp. 839–845. doi:10.1109/ROBOT.2010.5509562. (cited on page 33.)

[178] E. Velenis, P. Tsiotras, J. Lu, Modeling aggressive maneuvers on loose surfaces: The cases of Trail-Braking and Pendulum-Turn, in: ECC 2007, pp. 1233–1240. doi:10.23919/ECC.2007.7068670. (cited on page 33.)

[179] E. Velenis, P. Tsiotras, J. Lu, Optimality Properties and Driver Input Parameterization for Trail-braking Cornering, European Journal of Control 14 (4) (2008) 308–320. doi:10.3166/ejc.14.308-320.
URL http://www.sciencedirect.com/science/article/pii/S0947358008707751 (cited on page 33.)

[180] D. Tavernini, M. Massaro, E. Velenis, D. I. Katzourakis, R. Lot, Minimum time cornering: the effect of road surface and car transmission layout, Vehicle System Dynamics 51 (10) (2013) 1533–1547. doi:10.1080/00423114.2013.813557. (cited on pages 33, 56 and 119.)

[181] C. You, P. Tsiotras, Real-Time Trail-Braking Maneuver Generation for Off-Road Vehicle Racing, in: A. A. C. Conference (Ed.), 2018 Annual American Control Conference (ACC), IEEE, [Piscataway, NJ], 2018, pp. 4751–4756. doi:10.23919/ACC.2018.8431620. (cited on page 33.)

[182] F. Zhang, J. Gonzales, S. E. Li, F. Borrelli, K. Li, Drift control for cornering maneuver of autonomous vehicles, Mechatronics 54 (2018) 167–174. doi:10.1016/j.mechatronics.2018.05.009.
URL http://www.sciencedirect.com/science/article/pii/S0957415818300771 (cited on page 33.)

[183] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, E. A. Theodorou, Information theoretic MPC for model-based reinforcement learning, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 5/29/2017 - 6/3/2017, pp. 1714–1721. doi:10.1109/ICRA.2017.7989202. (cited on page 33.)

[184] H. K. Khalil, Nonlinear systems, 3rd Edition, Prentice Hall and London : Pearson Education, Upper Saddle River, N.J., 2002. (cited on pages 34 and 44.)

[185] F. Borrelli, A. Bemporad, M. Morari, Predictive control for linear and hybrid systems, Cambridge University Press, Cambridge, 2017. (cited on page 34.)

[186] D. P. Bertsekas, Dynamic programming and optimal control, Vol. 2, Athena scientific Belmont, MA, 1995. (cited on pages 34 and 78.)

[187] M. Ghallab, D. Nau, P. Traverso, Automated planning and acting, Cambridge University Press, 2016. (cited on pages 34 and 44.)

[188] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018. (cited on page 34.)

[189] E. Frazzoli, M. A. Dahleh, E. Feron, Real-Time Motion Planning for Agile Autonomous Vehicles, Journal of Guidance, Control, and Dynamics 25 (1) (2002) 116–129. doi:10.2514/2.4856. (cited on page 35.)

[190] Y. Kuwata, J. Teo, S. Karaman, G. Fiore, E. Frazzoli, J. How, Motion Planning in Complex Environments Using Closed-loop Prediction, in: Guidance, Navigation, and Control and Co-located Conferences, [publisher not identified], [Place of publication not identified], 2008, p. 661. doi:10.2514/6.2008-7166. (cited on pages 35, 39 and 40.)

[191] R. Tedrake, I. R. Manchester, M. Tobenkin, J. W. Roberts, LQR-trees: Feedback Motion Planning via Sums-of-Squares Verification, The International Journal of Robotics Research 29 (8) (2010) 1038–1052. doi:10.1177/0278364910369189. (cited on page 35.)

[192] J. Lygeros, K. H. Johansson, S. N. Simic, J. Zhang, S. S. Sastry, Dynamical properties of hybrid automata, IEEE Transactions on Automatic Control 48 (1) (2003) 2–17. doi:10.1109/TAC.2002.806650. (cited on page 36.)

[193] L. P. Kaelbling, T. Lozano-Pérez, Integrated task and motion planning in belief space, The International Journal of Robotics Research 32 (9-10) (2013) 1194–1227. doi:10.1177/0278364913484072. (cited on page 36.)

[194] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, P. Abbeel, Combined task and motion planning through an extensible planner-independent interface layer, in: 2014 IEEE international conference on robotics and automation (ICRA), 2014, pp. 639–646. (cited on page 36.)

[195] M. Toussaint, Logic-geometric programming: An optimization-based approach to combined task and motion planning, in: Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015. (cited on page 36.)

[196] R. J. Dakin, A tree-search algorithm for mixed integer programming problems, The Computer Journal 8 (3) (1965) 250–255. doi:10.1093/comjnl/8.3.250. (cited on page 36.)

[197] M. Fox, D. Long, Modelling mixed discrete-continuous domains for planning, Journal of Artificial Intelligence Research 27 (2006) 235–297. (cited on page 36.)

[198] J. Löhr, P. Eyerich, T. Keller, B. Nebel, A planning based framework for controlling hybrid systems, in: Twenty-Second International Conference on Automated Planning and Scheduling, 2012. (cited on page 36.)

[199] S. Bogomolov, D. Magazzeni, A. Podelski, M. Wehrle, Planning as Model Checking in Hybrid Domains.
URL https://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/download/8494/8682 (cited on page 36.)

[200] M. Helmert, G. Röger, et al., How Good is Almost Perfect?, in: AAAI, Vol. 8, 2008, pp. 944–949. (cited on page 36.)

[201] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Mastering the game of go without human knowledge, Nature 550 (7676) (2017) 354.
URL https://www.nature.com/articles/nature24270?sf123103138=1
(cited on page 36.)

[202] R. E. Korf, Real-time heuristic search, Artificial Intelligence 42 (2-3) (1990) 189–211. doi:10.1016/0004-3702(90)90054-4. (cited on page 36.)

[203] A. G. Barto, S. J. Bradtke, S. P. Singh, Learning to act using real-time dynamic programming, Artificial Intelligence 72 (1-2) (1995) 81–138. doi:10.1016/0004-3702(94)00011-O. (cited on page 36.)

[204] A. Tamar, Y. I. WU, G. Thomas, S. Levine, P. Abbeel, Value Iteration Networks, in: D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29, Curran Associates, Inc, 2016, pp. 2154–2162.
URL http://papers.nips.cc/paper/6046-value-iteration-networks.pdf
(cited on page 36.)

[205] A. Guez, T. Weber, I. Antonoglou, K. Simonyan, O. Vinyals, D. Wierstra, R. Munos, D. Silver, Learning to Search with MCTSnets.
URL http://arxiv.org/pdf/1802.04697v2 (cited on page 36.)

[206] B. Amos, I. Jimenez, J. Sacks, B. Boots, J. Z. Kolter, Differentiable MPC for End-to-end Planning and Control, in: Advances in Neural Information Processing Systems, 2018, pp. 8289–8300. (cited on page 36.)

[207] T. Weber, S. Racanière, D. P. Reichert, L. Buesing, A. Guez, D. J. Rezende, A. P. Badia, O. Vinyals, N. Heess, Y. Li, R. Pascanu, P. Battaglia, D. Hassabis, D. Silver, D. Wierstra, Imagination-Augmented Agents for Deep Reinforcement Learning.
URL http://arxiv.org/pdf/1707.06203v2 (cited on page 36.)

[208] K. Lowrey, A. Rajeswaran, S. Kakade, E. Todorov, I. Mordatch, Plan Online, Learn Offline: Efficient Learning and Exploration via Model-Based Control.
URL http://arxiv.org/pdf/1811.01848v3 (cited on page 36.)

[209] B. Kim, L. P. Kaelbling, T. Lozano-Perez, Learning to guide task and motion planning using score-space representation, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 5/29/2017 - 6/3/2017, pp. 2810–2817. doi:10.1109/ICRA.2017.7989327. (cited on page 36.)

[210] M. Bhardwaj, S. Choudhury, S. Scherer, Learning Heuristic Search via Imitation.
URL http://arxiv.org/pdf/1707.03034v1 (cited on page 36.)

[211] C. Zhang, J. Huh, D. D. Lee, Learning Implicit Sampling Distributions for Motion Planning, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 10/1/2018 - 10/5/2018, pp. 3654–3661. doi:10.1109/IROS.2018.8594028. (cited on page 36.)

[212] B. Ichter, J. Harrison, M. Pavone, Learning Sampling Distributions for Robot Motion Planning, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 21.05.2018 - 25.05.2018, pp. 7087–7094. doi:10.1109/ICRA.2018.8460730. (cited on page 36.)

[213] E. Groshev, A. Tamar, M. Goldstein, S. Srivastava, P. Abbeel, Learning generalized reactive policies using deep neural networks, in: 2018 AAAI Spring Symposium Series, 2018. (cited on pages 36 and 131.)

[214] S. Choudhury, M. Bhardwaj, S. Arora, A. Kapoor, G. Ranade, S. Scherer, D. Dey, Data-driven planning via imitation learning, The International Journal of Robotics Research 37 (13-14) (2018) 1632–1672. doi:10.1177/0278364918781001. (cited on page 36.)

[215] O. Sundstrom, L. Guzzella, A generic dynamic programming Matlab function, in: I. Staff (Ed.), 2009 IEEE International Conference on Control Applications, I E E E, [Place of publication not identified], 2009, pp. 1625–1630. doi:10.1109/CCA.2009.5281131. (cited on page 37.)

[216] M. Likhachev, G. J. Gordon, S. Thrun, ARA*: Anytime A* with provable bounds on sub-optimality, in: Advances in neural information processing systems, 2004, pp. 767–774. (cited on pages 41 and 104.)

[217] J. Rasmussen, Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models, IEEE Transactions on Systems, Man, and Cybernetics SMC-13 (3) (1983) 257–266. doi:10.1109/TSMC.1983.6313160. (cited on page 45.)

[218] E. DONGES, ASPEKTE DER AKTIVEN SICHERHEIT BEI DER FUEHRUNG VON PERSONENKRAFTWAGEN, in: AUTOMOB-IND, 1982. (cited on page 45.)

[219] A. R. HALE, J. STOOP, J. HOMMELS, Human error models as predictors of accident scenarios for designers in road transport systems, Ergonomics 33 (10-11) (1990) 1377–1387. doi:10.1080/00140139008925339. (cited on page 45.)

[220] A. Muigg, Implizites Workloadmanagement: Konzept einer zeitlich-situativen Informationsfilterung im Automobil, 2009. (cited on pages 45 and 167.)

[221] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, M. Maurer, Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving, in: 2015 IEEE 18th International Conference on Intelligent Transportation Systems (ITSC), IEEE, Piscataway, NJ, 2015, pp. 982–988. doi:10.1109/ITSC.2015.164. (cited on pages 46, 47 and 163.)

[222] Audi Cars Tell You How Fast to Go to Catch All Green Lights.
URL https://www.wired.com/story/audi-glosa-green-light-connected/
(cited on page 50.)

[223] L. Guzzella, A. Sciarretta, Vehicle Propulsion Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. doi:10.1007/978-3-642-35913-2. (cited on page 56.)

[224] G. Genta (Ed.), Motor vehicle dynamics: Modeling and simulation / Giancarlo Genta, Vol. v.43 of Series on Advances in Mathematics for Applied Sciences, WORLD SCIENTIFIC, Singapore and London, 1997. doi:10.1142/SAMAS. (cited on page 57.)

[225] D. Yang, L. Zhu, B. Ran, Y. Pu, P. Hui, Modeling and Analysis of the Lane-Changing Execution in Longitudinal Direction, IEEE Transactions on Intelligent Transportation Systems 17 (10) (2016) 2984–2992. doi:10.1109/TITS.2016.2542109. (cited on page 58.)

[226] E. Velenis, D. Katzourakis, E. Frazzoli, P. Tsiotras, R. Happee, Steady-state drifting stabilization of RWD vehicles, Control Engineering Practice 19 (11) (2011) 1363–1376. doi:10.1016/j.conengprac.2011.07.010.
URL http://www.sciencedirect.com/science/article/pii/S0967066111001614 (cited on page 59.)

[227] H. B. Pacejka, I. Besselink, H. B. T. Pacejka, vehicle dynamics, Tire and vehicle dynamics, 3rd Edition, Butterworth-Heinemann, Amsterdam and London, 2012. (cited on page 59.)

[228] M. Werling, S. Kammel, J. Ziegler, L. Gröll, Optimal trajectories for time-critical street scenarios using discretized terminal manifolds, The International Journal of Robotics Research 31 (3) (2012) 346–359. doi:10.1177/0278364911423042. (cited on pages 62 and 148.)

[229] K. Kant, S. W. Zucker, Toward Efficient Trajectory Planning: The Path-Velocity Decomposition, The International Journal of Robotics Research 5 (3) (1986) 72–89. doi:10.1177/027836498600500304. (cited on page 63.)

[230] M. Werling, J. Ziegler, S. Kammel, S. Thrun, Optimal trajectory generation for dynamic street scenarios in a Frenét Frame, in: 2010 IEEE International Conference on Robotics and Automation, I E E E, Piscataway, Jan. 2010, pp. 987–993. doi:10.1109/ROBOT.2010.5509799. (cited on page 67.)

[231] S. J. Anderson, S. C. Peters, T. E. Pilutti, K. Iagnemma, An optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios, International Journal of Vehicle Autonomous Systems 8 (2-4) (2010) 190–216. (cited on page 67.)

[232] A. Rizaldi, M. Althoff, Formalising Traffic Rules for Accountability of Autonomous Vehicles, in: 2015 IEEE 18th International Conference on Intelligent Transportation Systems (ITSC), IEEE, Piscataway, NJ, 2015, pp. 1658–1665. doi:10.1109/ITSC.2015.269. (cited on page 67.)

[233] J. Ziegler, C. Stiller, Fast collision checking for intelligent vehicle motion planning, in: 2010 IEEE intelligent vehicles symposium (IV 2010), La Jolla, California, USA, 21-24 June 2010, IEEE, Piscataway, 2010, pp. 518–522. doi:10.1109/IVS.2010.5547976. (cited on page 72.)

[234] T. Fraichard, H. Asama, Inevitable collision states—A step towards safer robots?, Advanced Robotics 18 (10) (2004) 1001–1024. (cited on page 104.)

[235] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, M. Likhachev, Multi-Heuristic A*, The International Journal of Robotics Research 35 (1-3) (2016) 224–243. (cited on pages 104 and 155.)

[236] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, Self-normalizing neural networks, in: Advances in neural information processing systems, 2017, pp. 971–980. (cited on page 131.)

[237] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034. (cited on page 131.)

[238] D. P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization.
URL http://arxiv.org/pdf/1412.6980v9 (cited on page 132.)

[239] H. Beglerovic, S. Metzner, M. Horn, Challenges for the Validation and Testing of Automated Driving Functions, in: C. Zachäus, B. Müller, G. Meyer (Eds.), Advanced microsystems for automotive applications 2017, Lecture Notes in Mobility, Springer, Cham, Switzerland, 2018, pp. 179–187. doi:10.1007/978-3-319-66972-4{\textunderscore}15. (cited on pages 138 and 139.)

[240] S. A. Sajadi-Alamdari, H. Voos, M. Darouach, Ecological Advanced Driver Assistance System for Optimal Energy Management in Electric Vehicles, IEEE Intelligent Transportation Systems Magazine (2018) 1doi:10.1109/MITS.2018.2880261. (cited on page 151.)