Universität der Bundeswehr München
Fakultät für Luft- und Raumfahrttechnik
Institut für Mathematik und Rechneranwendung

Verification of Collision Avoidance Systems using Optimal Control and Sensitivity Analysis

Ilaria Xausa

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

Doktors der Ingenieurswissenschaften (Dr.-Ing.)

eingereichten Dissertation

Diese Dissertation wurde am 18. März 2015 bei der Universität der Bundeswehr München, 85577 Neubiberg eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 3. Juni 2015 angenommen.

Tag der Prüfung: 4. Juni 2015

Zusammenfassung

Die Analyse von Verkehrsszenarien auf mögliche Konfliktsituationen ist wegen der hohen Komplexität solcher Szenarien äußerst herausfordernd. Insbesondere sollten gewisse Sicherheits- und Leistungsanforderungen garantiert werden. In dieser Dissertation werden das komplexe Umfeld und die Sicherheitsanforderungen als Niveaumengen einer Lipschitz-Funktion respektive als Durchschnitt von Niveaumengen glatter Funktionen definiert. Die Leistungsanforderungen werden durch Minimierung einer Kostenfunktion erreicht. Somit kann das Problem als Problem der optimalen Steuerung formuliert werden. Dies erlaubt uns, eine weite Bandbreite an Straßengeometrien, polygonalen Hindernissen und Fahrzeugdynamiken zu analysieren. Sobald das Optimalsteuerungsproblem formalisiert wurde, können numerische Simulationen durch direkte Methoden (mit OCPID-DAE1 Software) oder Hamilton-Jacobi-Methoden (mit ROC-HJ-Software) erhalten werden. Ausweichmanöver, die Menge erreichbarer Punkte und die Menge der Anfangszustände, von welchen eine Kollision verhindert werden kann, können für Fahrzeugdynamiken mit bis zu sieben Zustandsvariablen berechnet werden.

Eine Sensitivitätsanalyse wird dann durchgeführt, um die Robustheit der Trajektorien und der Menge erreichbarer Punkte zu überprüfen. Ein erster Aspekt, bereits eingehend behandelt in der Fachliteratur, konzentriert sich auf die Berechnung einer Echtzeit-Näherung von Ausweichmanövern und der Menge erreichbarer Punkte unter Verwendung einer gegebenen Referenztrajektorie. Ein zweiter Aspekt versucht, die Effekte von Messfehlern auf Rechenergebnisse vorherzusagen und damit die Robustheit gegen Störungen in den Anfangswerten zu überprüfen. Ein dritter Aspekt ist durch das Bedürfnis von Sensorentwicklern inspiriert, die Genauigkeitsanforderungen von Sensoren zu ermitteln. Zu diesem Zweck nähert man den maximal zulässigen Sensorfehler, der es noch ermöglicht, Sicherheit zu garantieren.

Gemäß diesen Schritten wird ein Simulationsrahmen zur Überprüfung bestehender Kollisionsvermeidungsalgorithmen vorgeschlagen. Beispiele für Prüfungsverfahren werden implementiert und erfolgreich auf Kollisionsvermeidung durch Brems- und Lenkalgorithmen getestet.

Abstract

The mathematical definition of a car traffic scenario, where a conflict situation may occur, is a challenging topic in collision avoidance due to the complexity of the scenario which must satisfy safety and performance requirements. In this thesis, the complex environment and the safety requirements are simply defined as level sets of some Lipschitz function or as intersections of level sets of smooth functions. The performance criteria are then achieved by minimizing a cost function, referencing the structure of an optimal control problem. This allows to deal with a variety of road geometries, considering a significative number of polygonal obstacles and several choices of vehicle motion models. Once the optimal control problem has been formalized, the numerical simulations are obtained via direct methods (with OCPID-DAE1 software), or Hamilton-Jacobi methods (with ROC-HJ software). Avoidance trajectories, sets of reachable points and sets of initial points from which it is possible to avoid a collision are computed for vehicle dynamics with a number of states up to seven.

Sensitivity analysis is then implemented to verify robustness of trajectories and reachable sets, in a triple challenge. A first aspect, well known in literature, focuses on the computation of real-time approximation of avoidance trajectories and reachable sets using a given reference trajectory. A second novel aspect aims to predict measurement error effects on the computational results, verifying robustness with respect to perturbations in the initial data. A third novel aspect is inspired by the necessity for sensor developers to understand how precise must be a sensor. To this purpose the estimation of the maximum sensor error for a given trajectory is derived, preserving safety.

Through these steps a simulation framework to verify existing collision avoidance algorithms is proposed. Examples of verification procedures are implemented and successfully tested on collision avoidance by braking and steering algorithms.

Acknowledgements

I would like to express my sincere gratitude to my thesis advisor, Prof. Dr. Matthias Gerdts, for accepting me as a Ph.D. student and for supporting me in the past three years. I have appreciated his availability, his calmness and his encouragment. Thanks for the time you have devoted to me.

My sincere gratitude goes to my second advisor Dr. Robert Baier, for providing precious advices and support. I am grateful to Dr. Olivier Bokanowski, for welcoming me in Paris for six months. The discussions with him widely contributed to improve the content of this thesis.

A special acknowledgement goes to the Driver Assistance System department leader Thomas Ruchatz and subdepartment leader Hans-Christian Schaub at Volkswagen Research. Their professional experience and their awareness were fundamental for the success of the project. Furthemore, I thank the CABS team at Volkwagen Research. The weekly meetings and the evening talks were source of inspiration for my work. In particular, I address my thank to Dr. Daniel Toepfer who, as experienced researcher, provided essential technical and organizational suggestions.

I am most grateful to all the coordinators and the members of the Marie Curie ITN-SADCO Network. The focus on collaborative research and training was an occasion to develop career opportunities, to exchange ideas and projects and to sharpen the training in optimal control. A special thanks to the SADCO fellows, for being motivating colleagues and cheerful friends.

Finally, I heartily thank my friends and family for the support throughout all my studies at University and during my staying at Volkswagen. I am particularly grateful to Daniel and Miriam: in these years of Ph.D. they shared with me the good and the bad times, cheering me up and encouraging me.

Contents

Chapter I

Introduction

Contents

1 Advanced safety and driver assistance systems

1.1 Safety and driver assistance systems in automotive industry

The history of “safety” in automotive industry started in 1903 when the first seat belt was patented. An important step to reduce the number of accidents was provided in the 1940s with the invention of the Driver Airbag System, a so called passive safety system. In the modern perspective, safety means to help the driver to avoid accidents through active safety systems, like Anti-lock Braking System (ABS), and driver assistance systems, like danger alerts systems occurring whenever the driver’s reaction delays or fails. Safety and Driver Assistance Systems (DAS), within their functional limits, concretize this modern concept of safety by analyzing the traffic situation, estimating dangers correctly and taking appropriate actions. Future developments are named Advance Driver Assistance Systems (ADAS) and evolve in the direction of automated motor vehicles. The introduction of ADAS in the automotive market suffers, however, of a double impedance. By law the responsibility of the behavior of the vehicle in road traffic situations has to rely on the driver only. This legal issue is nowadays a core discussion at an international level (United Nations Economic Commission for Europe, Vehicle Infrastructure Integration Consortium in the US, Advanced Safety Vehicle Project 3 in Japan, [31]) and requirements for missing new legal regulations are preventing ADAS technologies from entering the automotive market. The second challenge originates from technological aspects. The advancement of new algorithms has to deal with the complexity of the environment and the limits of on board sensors.

An overview of last generation DAS is provided in [101] and only a small sample of developments is listed here.

- **Cruise Control System (CCS) with speed limiter** limits the speed of the vehicle to a preset maximum speed.
- **Adaptive Cruise Control (ACC)** is equipped with a radar sensor in the front bumper to monitor the traffic situation in front of the vehicle. If the radar sensor detects another vehicle driving in front in the same lane, then ACC will decelerate and maintain a preset distance from the vehicle driving in front.
- **Front Assist** helps to avoid rear-end collisions. The information provided by the front radar and the front camera (depending on equipment) is used to keep monitoring the traffic situation in front of the vehicle, and alerts the driver to any critical situations. The aim is to minimize the stopping distance in critical situations. The system operates in a speed range from around 5 [km/h] to 210 [km/h] and at a distance of up to 120 meters. When a dangerous situation is identified the driver is warned visually and acoustically. If the driver reacts by braking too gently, the vehicle will automatically generate the brake pressure required for the situation. More specifications on this function are in [99, 100].
- **Multicollision brake** triggers an automatic braking maneuver when it identifies an initial collision.
- **Traffic sign recognition** is a graphical notation tool for traffic signal as in Figure 1.1.
- **Dynamic Light Assist (DLA)** adapts the headlamp range to prevailing conditions.
- **Lane departure warning (Lane Assist)** helps the vehicle to maintain its lane in many different traffic situations.
- **Parking systems** allows the driver to park in spaces 90 degrees to the lane, parallel to the lane, and to the left and right of the lane.
- **Driver Alert System (DAS)** analyzes the way the driver is steering the vehicle. If the system identifies that the driver is about to fall asleep, then an acoustic warning is issued and the suggestion of a break is displayed.

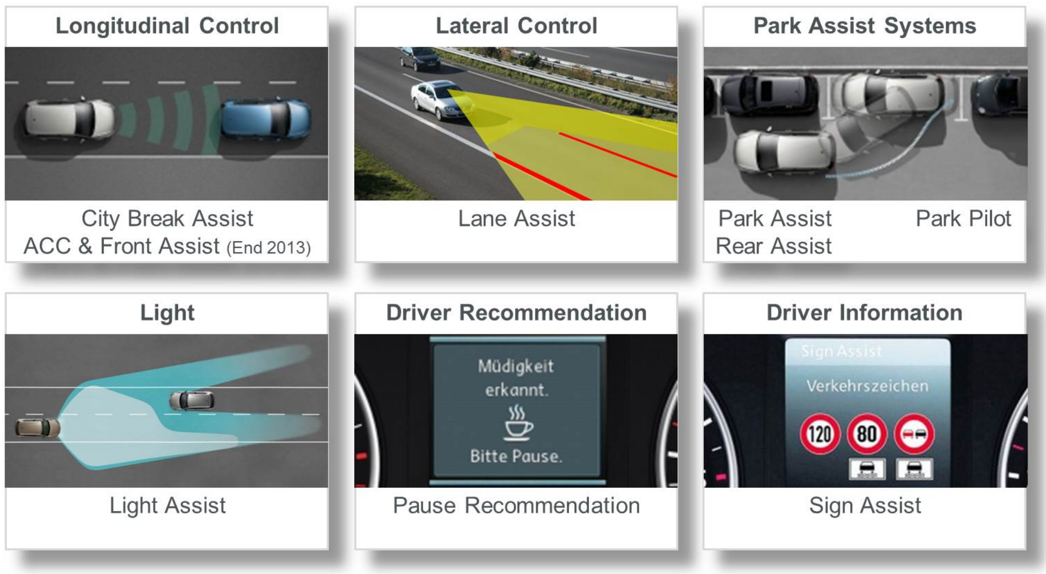


Figure 1.1: Today's driver assistance systems.

1.2 Collision avoidance algorithms

Particular attention in this thesis is given to DAS involved in collision avoidance, such as Front Assist [99, 100]. Most state-of-the-art collision avoidance systems use a similar algorithm. Systems published by Mazda and Honda are based on works of [35, 44, 88] since the early 1990s. In [35] the authors present the study of the effectiveness of a rear-end collision avoidance system capable of working on both straight and curved sections of highway. In [44] the authors propose a radar-based automatic braking system to prevent the vehicle from a rear-end collision or to reduce the impact speed without adverse effects on normal driving. Moreover, in [88] the authors present a collision warning algorithm using parameters estimated from an estimation scheme for tire-road friction.

Despite algorithms implemented in today's DAS, collision avoidance has been in general an active area of research. Contributors inspired by industrial problems, cover a wide range of subjects, such as robotic [60, 95], aerospace industry [32], and automotive industry [1, 31, 57, 63]. In [60] Neuro-fuzzy approaches are developed to determine time-optimal, collision-free path of a car-like mobile robot navigating in a dynamic environment. In [95] the problem involves a robot avoidance collision, where the robot behaves like a charged particle that is attracted by a target position and repelled by the obstacles. In [32] a collision avoidance Unmanned Aerial Vehicle (UAV) is treated by identifying capture sets. In [1] the set of all reachable point is computed efficiently by linearizing the nonlinear system and then computing zonotopes, which can be viewed as a Minkowski sum of line segments, giving an overapproximation of the real reachable set. In [31] a two-vehicle collision avoidance problem in a roundabout is studied. In order to guarantee safety, the authors employ an approach based on the computation of the unsafe set as capture set of an hybrid automaton system, formalizing mathematically the automobile structure. In [57] the minimal distance to an obstacle within which the vehicle cannot avoid a collision is computed. They use an algorithm based on second-order cone programming and sequential-quadratic programming. In [63] the authors study the merging into the traffic and a lane changing maneuver. The way of approaching a collision avoidance problem, extend to a big variety of tools, including optimal [67, 81], hybrid [58, 96], differential game [66], and stochastic [2] frameworks. The focus of the stochastic and hybrid work has been on the computation of safe regions in a collision situation, the optimal approach optimizes performance criteria over a finite time horizon, while the differential game approach considers the objects as players of a pursuit-evasion differential game.

1.3 Requirements for collision avoidance systems

The existing literature on the development of collision avoidance algorithms in car traffic scenarios reveals a major technical challenges. Reliable systems are requested by the automotive industry, to trade off the minimization rate of nuisance alarms and the minimization rate of missed detection. The false warnings (nuisance alarms and missing detections) can cause the failure of the collision avoidance system that will annoy or confuse the driver instead of helping her/him. Nuisance/false alarms may desensitize the driver and cause future warnings to be ignored, or can start an automatic braking maneuver whenever it is not necessary, risking that the driver loses the vehicle control. For example, let a path be driven following a road surface, say a curved roadway, different from the one implemented in the system, say a straight road. In this case objects outside the vehicle path are detected as dangerous situation. Or an obstacle in an adjacent lane is detected as to be on the vehicle's path when the road is curving. Furthermore, a collision avoidance system suffering of missing detections creates a false protection feeling on the driver. The driver's attention may decrease due to the confidence on the collision avoidance system, even in dangerous situations where the missing detection occurs.

False warnings can be a consequence of limitations of the algorithm considering a small range of traffic scenario types, or too strong simplifications in the complexity of the human behavior, or low quality data used to model the environment on which the algorithm is based. Thus to avoid false warnings a collision avoidance system has to be designed to meet the following criteria.

- A collision avoidance system must encompass a wide variety of driving situations and human characteristics, to be useful to the driver and not interfere with normal driving habits. The effect of individuals driving styles, the complexity of the scenario are elements that have to be taken into account in the development of the model behind the collision avoidance algorithm.

- The instrumentation quality data used to model the warning timings criteria can be subject to measurement errors due to the limitations and imprecisions of the currently available technology on environment perception. Thus, a collision avoidance algorithm must satisfy the robustness property under measurements errors. Environment perception means to identify road lanes and geometry and motion of obstacles. An overview on the modern literature on obstacle perception is in [89]. While algorithm for lane detection based on a probabilistic representation, are discussed in [64] and [90]. Technologies for environment perception must encounter many required performance standards, such as capabilities in detecting a large field-of-view, sensibility to precipitation, real-time capabilities to provide advanced notice of critical situations, and costs in the commercial automotive market. Among all the sensor systems considered in literature, see the overview in [62], the ones currently implemented on commercial automobiles are radar, mono-vision, stereo-vision. Examples of sensor systems are given in Figure 1.2 and in [89].

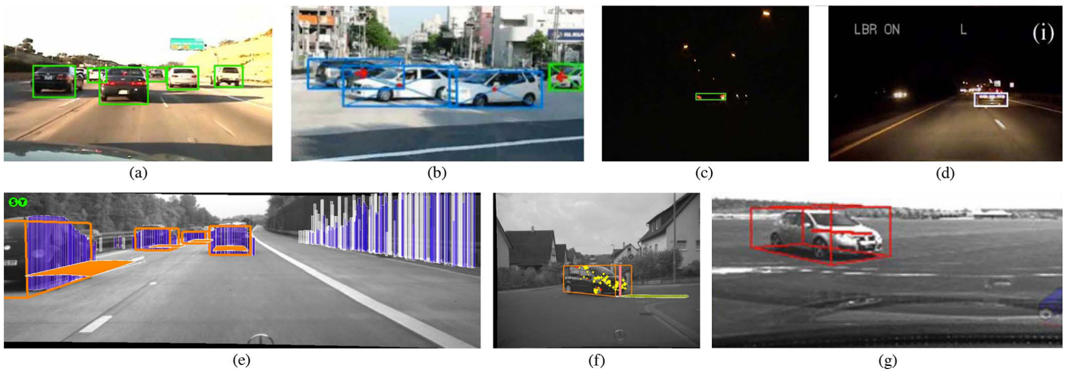


Figure 1.2: Sensor systems: (a, b, c) monocular, (d, e, f) stereovision, see [89].

The above criteria deal with the complexity of a real world scenario, where unexpected events, and rough measurement approximations can occur. A good compromise between the desire to guarantee safety and the limited technological tools needs to be agreed. A prudential point of view, such that to design an algorithm which prevents or reduce the severity of collisions more than aiming to avoid all collisions, is fulfilling this agreement.

Besides technical criteria, modern driver assistance systems require high standard security qualifications before entering the market. These qualifications in legal regulations demand intense testing which aims to find mistakes in the software packages, considering the sensor accuracy. Currently, exhaustive tests in real world scenarios are necessary, demanding high costs. Release testing of todays DAS requires up to 2 million test km and 1.000 test drivers. Forecast for high automation is about 100 million km with costs for such release testing of several 100 million EUR [102]. This is the reason why nowadays the automotive industry researchers look for cost efficient alternatives to this kind of test methodology, to verify weather a collision avoidance algorithm matches the standard qualifications. Some common cost efficient verification methods for collision avoidance algorithms will be discussed in the next section.

2 Verification of collision avoidance systems

The benefit of formal methods to verify collision avoidance algorithm and guarantee safety, has been shown in many application areas and projects. Mathematical tools are widely involved for such purposes.

2.1 Related work

Verification is usually performed by evaluating test cases for specific trajectories given a particular scenario. This is done using real vehicles, as in [34, 59]. More precisely, in [59] on-road tests are run to verify collision avoidance warnings by using vehicular communication. In [68] statistical methods are implemented on small samples of naturalistic driving data to meet the legal requirements for evaluation of collision avoidance algorithms. Simulation environments are a good alternative to real vehicle tests. Simulation techniques are studied for instance in [29, 106]. In [29] they develop appropriate alerting thresholds based on performance results and implement them in rear-end collision mitigations. In [106] they use simulators to reproduce the test on road, where sensors, actuators, driver, vehicle dynamics and traffic environment are incorporated. Techniques based on the computation of safety sets are used to provide general conditions for the absence of incorrect decisions for a given collision avoidance function, see [36, 72, 74]. In particular, in [36] a finite number of well-chosen trajectories will suffice to prove correctness by checking that the system avoids a bad set. Approximation of trajectories of the system by sensitivity analysis is developed and implemented in the context of numerical integration. In [72], safety analysis for a very general dynamical system is studied by looking at forward or backward reachable sets or tubes, i.e. the set of all reachable points and the set of all starting points from which it is possible to avoid a collision. Moreover, sensitivity analysis is used to prove the numerical stability of this algorithm. Eventually, in [74] a verification method of a collision avoidance algorithm is analyzed, by computing the backward reachable set of a specific car traffic scenario. The robustness to measurement errors is also discussed, using the definitions of robust minimal reachable set and uncertain minimal reachable set.

The approach discussed in this thesis may be classified as the simulation environment framework presented in [29, 106], by using optimal and reachability framework.

Many of the above approaches, are considering a specific car traffic scenario [29, 68, 74, 106]. Instead, the approach presented here will consider the rich diversity of real world car traffic scenarios. This generalization is achieved by a novel method for modelling general road geometries and polygonal or circular obstacles, implementing several physical models for the vehicle motion. The computation of an optimal avoidance trajectory which satisfies safety and performance requirements is one of the main tools for verification in this thesis. A second tool is the computation of the reachable set (forward or backward), as the set of points (final points or initial points) reached by an avoidance trajectory which meets the safety requirements. Differently from [32] the dynamics implemented here is high dimensional (up to 7 states) and highly nonlinear. While in [1] an approximation of the reachable set is computed and the nonlinear dynamics is linearized, in this thesis the exact reachable set is computed with nonlinear dynamics. These aspects are challenging from the modelization and computational point of view, but they give results closer to the reality. As seen above, sensitivity analysis with respect to initial data perturbation is a tool widely used in literature [36, 72, 74]. Sensitivity analysis is here employed in a novel way to compute robust trajectories and robust sets of reachable points. The aim is to specify sensor requirements, a huge challenge in application development. Most importantly, numerical simulations are given in Chapter IV, and examples of verification procedures are successfully tested on collision avoidance by braking and steering algorithms in Chapter V.

2.2 Contribution of this thesis

In this thesis a tool for validation of collision avoidance systems is constructed. These systems (see references in Section 1.2) can be classified in two classes: Collision Avoidance using Braking and Steering maneuvers (CABS) and Collision Avoidance using only Braking maneuvers (CAB), such as rear-end collision avoidance algorithms. In particular, the activation times of CABS and CAB systems are accessed, which may be too early or too late due to sensor errors or algorithmic errors, as motivated in Section 1.3. Usually this is studied by collecting naturalistic driving data and adjusting the activation timing criteria on this. The aim of this thesis is to develop a software called Virtual Test Maker (VTM) to shift the tests from on-road based to simulation based. As said in Section 1.3, two criteria must be satisfied by the collision avoidance system to prevent false warnings, and thus verified: the complexity of human and environmental factors has to be included in the model for the development of the algorithm, and sensor measurement errors need to be taken into account to generate robust algorithms.

The first criterion is verified by creating a virtual environment, modelling a wide range of naturalistic car traffic scenario where a collision is likely to happen. In this environment avoidance trajectories and reachable sets are computed and compared with CABS and CAS output. To provide a virtual environment, the mathematical model is based on the fact that the naturalistic scenario is perceived in its complexity as a whole by the system, reacting to unexpected circumstances, and planning its time evolution to achieve its mission. In this thesis, the mathematical formalization of the environment starts by modelling the evolution of the vehicle motion as solution of a system of ordinary differential equations (ODEs). Several non linear systems are here solved numerically and compared, from the 3-dimensional kinematic model, till the 7-dimensional single track model. Thus the need of physical simplifications in the model and of the analytical calculation playing a central role in [65] is decreased by the approaches considered here. This formalization is especially designed to deal with the constraints imposed by limited vehicle velocities and accelerations, because it is derived directly from the motion dynamics of the vehicle. Among the admissible accelerations and velocities, a combination is chosen that guarantees safety, maximizing an objective function. The objective function includes a measure of progress towards a goal location. For instance, the minimization of the time interval in which to perform the avoidance maneuver, or the maximization of the distance to the obstacles for the whole vehicle path. Safety is met whenever the vehicle reaches a target without being involved in a collision. By combining these aspects, the vehicle trades off its desire to move fast towards the goal and its desire to steer around obstacles, leading to an optimal control problem. Herein, accelerations and velocities (classified as optimal controls and optimal solutions of the problem) are computed such that performance criteria are maximized through an objective function, and safety is guarantee by imposing inequality constraints. This approach has the advantage that it is derived directly from the motion dynamics of the vehicle, taking into account the drag and tire forces acting on the vehicle. Moreover safety conditions, such as to follow a road path or to avoid moving or fixed obstacles or to reach a safe target zone, are naturally modelled as equations and inequalities. From this model two possible tools for verification are implemented. The first method is evaluating test cases for specific trajectories and sets of reachable points (forward reachable sets) in an optimal framework. A second method is based on the computation of the unsafe set of initial states, where the collision avoidance/mitigation function must be activated, as complement of the safe set of initial states from which is possible to avoid a collision (backward reachable sets). In particular, forward reachable sets are numerically computed successfully up to 7-dimensional dynamical models, while backward reachable sets up to 6-dimensional dynamical models.

Robustness of the collision avoidance algorithm with respect to sensor measurement errors is considered for verification by studying sensitivity analysis of trajectories and of reachable sets. A first type of sensitivity (named ODE-Sensitivity, see [73, 105]) is implemented to predict how sensor measurement errors in initial data affect the given collision avoidance maneuver or the computed reachable set. A similar approach is used in [73], where the sensitivities are used to study the behaviour of numerical errors in the algorithm. The authors compute a first order Taylor approximation of the solution by linearization around a nominal solution for perturbation of the initial state. The derivatives of the solution with respect to the initial state are computed as in [2] by fixing the control. A second type of sensitivity generates real-time approximations of collision avoidance maneuvers or reachable sets when errors in the initial data occur. Such sensitivity based on parametric sensitivity analysis is performed under the name of Fiacco-Sensitivity [23, 24, 41, 49, 52]. In this case, the dependence of both the optimal trajectory and the optimal control on perturbations in the initial state is incorporated in the Taylor approximations, beyond the scopes of the approach in [73]. Estimation of sensor errors such that the collision avoidance maneuver is still collision free is also an important aspect for sensor developers, and thus studied here.

3 Outline

The next chapter mathematically defines car traffic scenarios which occur in conflict situations. By imposing safety and performance requirements, a natural modellization though an optimal control approach is derived. Chapter III recalls notions optimal control theory used in this thesis. Numerical solutions of optimal control problems are discussed via direct methods [14, 49] and via Hamilton-Jacobi methods [11, 17, 18, 73]. Such notions are the theoretical background of the software packages used in Chapter IV. In Chapter IV numerical simulations of optimal trajectories and reachable sets are computed for car traffic scenarios. By using OCPID-DAE1 software [48], based on direct methods, an optimal collision-free trajectory, set of points reachable from a given initial position and the associate trajectories are computed for a given conflict situation. With the ROC-HJ software [16], based on Hamilton-Jacobi methods, a safety set of initial states is computed in a car traffic scenario and minimum time trajectories are constructed. An entire Section is devoted to sensitivity analysis where the optimal trajectory and the reachable set are estimated when perturbations in the initial state occur. Moreover an estimation of the initial state error is given for particular car traffic scenarios. Chapter V shows how the simulation of Chapter IV can be implemented for verification purposes. The verification is classified for collision avoidance by braking algorithms, and collision avoidance by braking and steering algorithms. A special attention is given to the analysis of sensor measurement errors in trajectories computed with the verified software packages. Finally, in Chapter VI future work directions are discussed. The chapter opens with an overview on challenges for implementing optimal control trajectories in a real car. The remaining discussion is focused on alternative models to consider error measurements or, more in general, uncertainties in the environment perception. Such modelization is based on game theory or stochastic optimal control.

Chapter II

Modelization

Contents

The main tasks of collision avoidance systems are to reliably indicate future collisions and - if possible - to provide escape trajectories if such exist. The collision avoidance system is supposed to control a reference vehicle that is one of the elements of a car traffic scenario. For this specific scenario the goals are to guarantee safety by providing a collision free maneuver for the reference vehicle and to satisfy some performance criteria for such a maneuver. This chapter suggests an optimal control based method that has the potential of fulfilling these two goals. In particular the focus is to show how the modeling process of a car traffic scenario leads to an optimal control problem once the safety and performance conditions are imposed.

The car traffic scenario involves the motion of the reference vehicle (described by ODEs) and the mathematical characterization of the environment where it evolves (defined by state constraints and boundary constraints). A precise model as the one in [46] includes a 7-dimensional state and highly nonlinear differential equations. On the other hand models with less states and simpler equations (for instance the 4-dimensional point mass model or the 3-dimensional kinematic model) allow a wider range of vehicle maneuvers, shrinking the set of points where the collision avoidance system activation occurs. The other physical entities, such as road obstacles and target sets, are modeled as the level set of some Lipschitz function or as an intersection of sets with smooth boundaries. Once the car traffic scenario has been described, the safety property is defined by imposing that the vehicle trajectory has to reach the target area within a fixed time interval, avoiding the obstacles and staying on the road. Eventually, performance criteria are provided knowing that the safety property has to hold. Firstly, an optimal trajectory is required where the term "optimal" addresses to a fast or driver friendly or extreme vehicle trajectory. Secondly, the set of all points that a vehicle maneuver can reach is determined. Thirdly, the set of all initial points from which it is possible to avoid the obstacles by staying on the road are defined. In conclusion, the methodology presented involves two steps: analyzing the safety of the system, and then optimizing a desired performance goal within the safe region of operation. The computations presented in this chapter are obtained with OCPID-DAE1 software (see Section 1 of Chapter IV for more details). Classical results on optimal control as well as state of the art techniques to solve optimal control problems will be treated in Chapter III.

1 Reference vehicle model

The motion of the reference vehicle has the property to evolve in time obeying physical laws described by Ordinary Differential Equations (ODEs):

$$\begin{cases} \dot{z}(t) = f(t, z(t), u(t)) \text{ a.e. } t \in [t_0, t_f] \subset \mathbb{R}, \\ z(t_0) = z_0, \end{cases} \tag{1.1}$$

given the *initial state* $z_0 \in \mathbb{R}^{n_z}$ and the function (*dynamics*) $f : [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}$. The curve $z : [t_0, t_f] \to \mathbb{R}^{n_z}$ is unknown and it is the dynamical evolution of the *state* of the *dynamical system* in (1.1), moreover $\mathbb{R}^{n_z}$ will be called *n_z-dimensional state space.* The function $u : [t_0, t_f] \to \mathbb{R}^{n_u}$ is the *control* that influences the function z. The solution of (1.1) for some control u and for a given initial value z_0 is denoted with $z^u_{z_0}$. The peculiarity of the system presented in this section is that the vehicle motion is controllable by a collision avoidance system or more simply by a driver that wants to preform an avoidance maneuver. Experience teaches that this controllability is accessible through parameter values that affect the braking force and steering performance. In the mathematical formalism such parameter values are denoted by the function u as it appears in (1.1).

Several levels of complexity enter the ODE system, depending on the simplifying assumptions made on the physical system. A comparison of such models is given in the next subsections, starting from the highly nonlinear 7-dimensional single track model, till the 3-dimensional kinematic model. Moreover the identification of state z and control u will be underlined for each physical model.

1.1 The single track “7D” model and the “6D” model

The single-track car model (a detailed presentation is provided in [47]) is derived by assuming that the rolling and pitching behavior of the car body can be neglected, and thus the roll angle and the pitch angle are small. These assumptions give a double advantage: the two wheels on the front and rear axle are replaced by a virtual wheel located in the center of the respective axle and the car’s center of gravity is located on the roadway which allows to consider the motion of the car in the horizontal plane only. The geometrical description of the car for the single track model is depicted in Figure 1.1.

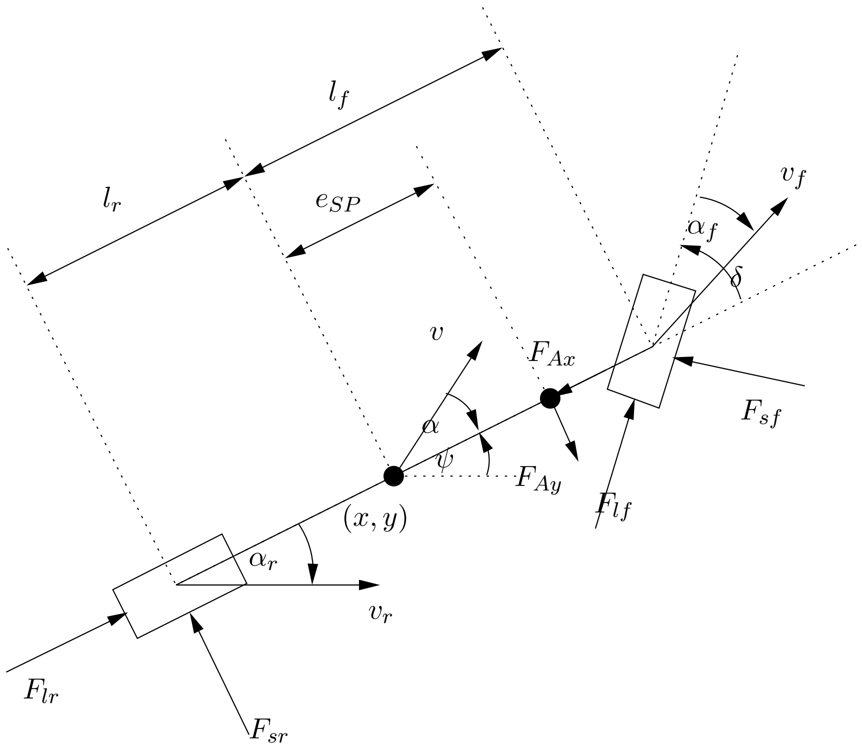


Figure 1.1: Geometrical description of the single-track car model [46].

Herein, some notation is introduced:

$r_C = (x_{car}, y_{car})^\top$	center of gravity in a reference coordinate system
$v_C = (v_x, v_y)^\top$	velocity at the center of gravity
v_f, v_r	velocities of the front and rear wheel
δ	steering angle
α	side slip angle
α_f, α_r	slip angles at front and rear wheel
ψ	yaw angle
F_{sf}, F_{sr}	lateral tire forces (side forces) on front and rear wheel
F_{lf}, F_{lr}	longitudinal tire forces on front and rear wheel
l_f, l_r	distances from the center of gravity to the front and rear wheel
e_{SP}	distance from the center of gravity to the drag mount point
F_{Ax}, F_{Ay}	drag force on the car due to air resistance and side wind
m	mass of the car

The points r_R, r_F at the front and at the rear wheels and their velocity are defined as

$$\begin{aligned} r_R &= r_C + S(\psi)\begin{pmatrix} -l_r \\ 0 \end{pmatrix} = \begin{pmatrix} x - l_r\cos\psi \\ y - l_r \sin\psi \end{pmatrix}, \\ r_F &= r_C + S(\psi)\begin{pmatrix} l_f \\ 0 \end{pmatrix} = \begin{pmatrix} x + l_f\cos\psi \\ y + l_f \sin\psi \end{pmatrix}, \end{aligned} \tag{1.2}$$

where $S(\psi)$ is the anti-clockwise rotation matrix of angle ψ, defined as:

$$S(\psi) = \begin{pmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{pmatrix}. \tag{1.3}$$

The velocity $v_C = (v_x, v_y)$ has module $v = \sqrt{v_x^2 + v_y^2}$ and is rotated by a side slip angle $-\alpha$ from the car axle joining the front and the rear wheel

$$v_C = S(\psi - \alpha)\begin{pmatrix} v \\ 0 \end{pmatrix} = \begin{pmatrix} v\cos(\psi-\alpha) \\ v\sin(\psi-\alpha) \end{pmatrix}, \tag{1.4}$$

which gives differential Equations (1.26a) and (1.26b).

By combining the fact that $v_R = r'_R$ (where f' of a function f denotes the derivative with respect to time of f) and v_R has direction rotated by $\psi - \alpha_r$ from the x-axel and module $\tilde{v}_R$, it holds

$$v_R = r'_R = \begin{pmatrix} x' + l_r\psi'\sin\psi \\ y' - l_r\psi'\cos\psi \end{pmatrix}, \qquad \begin{pmatrix} \tilde{v}_R \\ 0 \end{pmatrix} = S(\psi - \alpha_r)^T \cdot v_R. \tag{1.5}$$

From the second component of (1.5), and using the trigonometric identities, it follows:

$$\begin{aligned}
&-\sin(\psi-\alpha_r)(v\cos(\psi-\alpha)+l_r\psi'\sin\psi)+\\
&\qquad+\cos(\psi-\alpha_r)(v\sin(\psi-\alpha)-l_r\psi'\cos\psi)=0\Rightarrow\\
&v(\sin(\psi-\alpha)\cos(\psi-\alpha_r)-\cos(\psi-\alpha)\sin(\psi-\alpha_r))+\\
&\qquad-l_r\psi'(\cos\psi\cos(\psi-\alpha_r)+\sin\psi\sin(\psi-\alpha_r))=0\Rightarrow\\
&v\sin\alpha_r\cos\alpha-v\cos\alpha_r\sin\alpha-l_r\psi'\cos\alpha_r=0\Rightarrow\\
&\tan\alpha_r=\tfrac{v\sin\alpha+l_r\psi'}{v\cos\alpha}.
\end{aligned}\tag{1.6}$$

Likewise for the front wheel $v_F = r'_F$ and v_F has direction rotated by $\psi+\delta-\alpha_f$ from the x-axle and module $\tilde{v}_F$:

$$v_F = r'_F = \begin{pmatrix} x' - l_f\psi'\sin\psi \\ y' + l_f\psi'\cos\psi \end{pmatrix}, \qquad \begin{pmatrix} \tilde{v}_F \\ 0 \end{pmatrix} = S(\psi+\delta-\alpha_f)^T\cdot v_F. \tag{1.7}$$

Thus

$$\begin{aligned}
&-\sin(\psi+\delta-\alpha_f)(v\cos(\psi-\alpha)-l_f\psi'\sin\psi)+\\
&\qquad+\cos(\psi+\delta-\alpha_f)(v\sin(\psi-\alpha)+l_f\psi'\cos\psi)=0\Rightarrow\\
&v(\sin(\psi-\alpha)\cos(\psi+\delta-\alpha_f)-\cos(\psi-\alpha)\sin(\psi+\delta-\alpha_f))+\\
&\qquad+l_f\psi'(\cos\psi\cos(\psi+\delta-\alpha_f)+\sin\psi\sin(\psi+\delta-\alpha_f))=0\Rightarrow\\
&v\sin(\alpha_f-\delta)\cos\alpha-v\cos(\alpha_f-\delta)\sin\alpha+l_f\psi'\cos(\alpha_f-\delta)=0\Rightarrow\\
&\tan(\alpha_f-\delta)=\tfrac{v\sin\alpha-l_f\psi'}{v\cos\alpha}.
\end{aligned}\tag{1.8}$$

The physical system described above should satisfy the second Newton law:

$$F = ma, \tag{1.9}$$

with F being the force acting on the system, m the mass of the object, a the acceleration. The physical system described above should also satisfy the equilibrium of momentum law:

$$M = I\dot{\omega}, \qquad M = \left(\sum_{i=1,\ldots,n} F_i r_i \sin\theta_i\right), \tag{1.10}$$

with M being the momentum of force, I the moment of inertia, $\dot{\omega}$ the angular acceleration. The momentum M is defined as the sum of the n scalar products between the force F_i acting on the car and the vector r_i, the latter being the distance from the action point of force F_i to the center of gravity (x, y), where the angle between r_i and F_i is θ_i, $i = 1, \ldots, n$.

A description of the forces acting on the system is given straightforward.

$$\begin{aligned}
F_{sf} &= D_f \sin(C_f \arctan(B_f \alpha_f + \\
&\quad -E_f(B_f \alpha_f - \arctan(B_f \alpha_f)))) \\
&\equiv F_{sf}(\alpha_f), && \text{(1.11a)} \\
F_{sr} &= D_r \sin(C_r \arctan(B_r \alpha_r + \\
&\quad E_r(B_r \alpha_r - \arctan(B_r \alpha_r)))) \\
&\equiv F_{sr}(\alpha_r), && \text{(1.11b)} \\
F_{lf} &= -F_{Bf}(F_B) - F_{Rf}(v) \\
&\equiv F_{lf}(v; F_B), && \text{(1.11c)} \\
F_{lr} &= -F_{Br}(F_B) - F_{Rr}(v) \\
&\equiv F_{lr}(v; F_B), && \text{(1.11d)} \\
F_x &= F_{lr} - F_{Ax} + F_{lf}\cos(\delta) - F_{sf}\sin(\delta) \\
&\equiv F_x(v, F_B, \alpha_f, \delta), && \text{(1.11e)} \\
F_y &= F_{sr} - F_{Ay} + F_{lf}\sin(\delta) + F_{sf}\cos(\delta) \\
&\equiv F_y(v, F_B, \alpha_f, \alpha_r, \delta), && \text{(1.11f)}
\end{aligned}$$

where $D_f, D_r, C_f, C_r, B_f, B_r, E_f, E_r$ are constants. Forces F_{sf}, F_{sr} are derived by Pacejka's Magic Formula [78], while formulas for the braking forces F_{Bf}, F_{Br} and the rolling resistance forces F_{Rf}, F_{Rr} as well as the air resistance forces F_{Ax}, F_{Ay} are derived straightforward. The slide slip angles centered in the center of gravity, in the front wheel and in the rear wheel are given by

$$\begin{aligned}
\alpha &:= \psi - \arctan(y'/x') \\
&\equiv \alpha(v_x, v_y, \psi), && \text{(1.12a)} \\
\alpha_f &:= \delta - \arctan\left(\frac{l_f \psi' - v\sin(\alpha)}{v\cos(\alpha)}\right) \\
&\equiv \alpha_f(v_x, v_y, \psi, \psi', \delta), && \text{(1.12b)} \\
\alpha_r &:= \arctan\left(\frac{l_r \psi' + v\sin(\alpha)}{v\cos(\alpha)}\right) \\
&\equiv \alpha_r(v_x, v_y, \psi, \psi'). && \text{(1.12c)}
\end{aligned}$$

The (1.12a) derives from (1.4), while (1.12b) and (1.12c) derive from (1.6) and (1.8). Drag forces:

$$\begin{aligned}
F_{Ax} &= \frac{1}{2} \cdot c_\omega \cdot \rho \cdot A \cdot v^2, \quad c_\omega, \rho, A \text{ contants}, && \text{(1.13a)} \\
F_{Ay} &= 0. && \text{(1.13b)}
\end{aligned}$$

The braking forces:

$$F_{Bf} = \begin{cases} \frac{l_r}{l_f+l_r} F_B, & \text{if } F_B > \Delta, \\ \frac{1}{2}\frac{l_f+2l_r}{l_f+l_r} F_B + \frac{3}{4\Delta}\frac{l_f}{l_f+l_r} F_B^2 + \frac{1}{4\Delta^3}\frac{l_f}{l_f+l_r} F_B^4, & \text{if } |F_B| \le \Delta, \\ F_B, & \text{otherwise,} \end{cases} \tag{1.14a}$$

$$F_{Br} = \begin{cases} \frac{l_f}{l_f+l_r} F_B, & \text{if } F_B > \Delta, \\ \frac{l_f}{l_f+l_r}\left(\frac{2}{\Delta}F_B^2 - \frac{1}{\Delta^2}F_B^3\right), & \text{if } 0 < F_B \le \Delta, \\ 0, & \text{otherwise,} \end{cases} \tag{1.14b}$$

for a small positive $\Delta = 0.01$. If $\Delta = 0$, Equations (1.14a)-(1.14b) become:

$$F_{Bf}(F_B) \equiv \min(F_B, \frac{l_r}{l_f + l_r} F_B), \tag{1.15a}$$

$$F_{Br}(F_B) \equiv \max(0, \frac{l_f}{l_f + l_r} F_B). \tag{1.15b}$$

The rolling resistance forces:

$$F_{Rf} = f_R(v) \cdot F_{zf}, \tag{1.16a}$$

$$F_{Rr} = f_R(v) \cdot F_{zr}, \tag{1.16b}$$

with friction coefficient

$$f_R(v) = f_{R0} + f_{R1}\frac{v}{100} + f_{R4}\left(\frac{v}{100}\right)^4 \qquad (v \text{ in } [km/h]), \tag{1.17}$$

and constants

$$\begin{aligned} F_{zf} &= \tfrac{m \cdot l_r \cdot g}{l_f+l_r}, \\ F_{zr} &= \tfrac{m \cdot l_f \cdot g}{l_f+l_r}. \end{aligned} \tag{1.18}$$

Equations (1.9) and (1.11) yeld Equations (1.21d) and (1.21e):

$$\begin{pmatrix} v_x' \\ v_y' \end{pmatrix} = \frac{1}{m}\left[S(\psi)\begin{pmatrix} F_{lr} \\ F_{sr} \end{pmatrix} - S(\psi)\begin{pmatrix} F_{Ax} \\ F_{Ay} \end{pmatrix} + S(\psi+\delta)\begin{pmatrix} F_{lf} \\ F_{sf} \end{pmatrix}\right]. \tag{1.19}$$

Equations (1.10) and (1.11) yeld Equations (1.21c) and (1.21f):

$$\begin{aligned} \psi'' &= \tfrac{1}{I_{zz}}\left[F_{Ax}e_{SP}\sin\pi + F_{lr}(-l_r)\sin 0 \right. \\ &\quad + F_{Ay}e_{SP}\sin\left(-\tfrac{\pi}{2}\right) + F_{sr}(-l_r)\sin\left(\tfrac{\pi}{2}\right) \\ &\quad \left. + F_{lf}l_f\sin\delta + F_{sf}l_f\sin\left(\tfrac{\pi}{2}+\delta\right)\right] \\ &= \tfrac{1}{I_{zz}}\left[-F_{Ay}e_{SP} - F_{sr}l_r + F_{lf}l_f\sin\delta + F_{sf}l_f\cos\delta\right], \end{aligned} \tag{1.20}$$

where I_{zz} is the moment of inertia and $M = I_{zz}\psi''$ by definition of momentum of forces.

Thus the vehicle motion is described by the Ordinary Differential Equations (ODEs):

Single track 7D model with controls w_δ, F_B and states v_x and v_y:

$$x' = v_x \tag{1.21a}$$
$$y' = v_y \tag{1.21b}$$
$$\psi' = w_\psi \tag{1.21c}$$
$$v_x' = \frac{1}{m}(F_x \cos(\psi) - F_y \sin(\psi)) \tag{1.21d}$$
$$v_y' = \frac{1}{m}(F_x \sin(\psi) + F_y \cos(\psi)) \tag{1.21e}$$
$$w_\psi' = \frac{1}{I_{zz}}(F_{sf} l_f \cos(\delta) - F_{sr} l_r - \underbrace{F_{Ay}}_{0} e_{SP} + F_{lf} l_f \sin(\delta)) \tag{1.21f}$$
$$\delta' = w_\delta \tag{1.21g}$$

for almost every t in the fixed time interval $[t_0, t_f] \subset \mathbb{R}$ and with control parameters $|w_\delta(t)| \leq w_{\delta max}$ and $F_B(t) \in [F_{Bmin}, F_{Bmax}]$. Therefore by following the notation in (1.1), $z := (x, y, \psi, v_x, v_y, w_\psi, \delta)$ and $u := (w_\delta, F_B)$.

Alternative formulation of the system of differential equations for the "7D" model comes by deriving

$$v^2 = v_x^2 + v_y^2 \quad \Rightarrow \quad 2vv' = 2v_x v_x' + 2v_y v_y'. \tag{1.22}$$

Using (1.4),(1.21d),(1.21e) and (1.12a):

$$\begin{array}{ll} v_x' = \frac{1}{m}(F_x \cos(\psi) - F_y \sin(\psi)), & v_x = v\cos(\psi - \alpha), \\ v_y' = \frac{1}{m}(F_x \sin(\psi) + F_y \cos(\psi)), & v_y = v\sin(\psi - \alpha), \end{array} \tag{1.23}$$

and thus

$$v' = \frac{1}{m}(F_x \cos\alpha - F_y \sin\alpha), \tag{1.24}$$

with F_x, F_y and α defined in (1.11e), (1.11f) and (1.12a).

Moreover differentiating (1.12a) and using (1.23):

$$\alpha' = w_\psi - \frac{F_x \sin\alpha + F_y \cos\alpha}{mv}, \tag{1.25}$$

with F_x and F_y defined in (1.11e), (1.11f).

"7D" model with controls w_δ, F_B and states α and v:

$$x' = v\cos(\psi - \alpha) \tag{1.26a}$$
$$y' = v\sin(\psi - \alpha) \tag{1.26b}$$
$$\psi' = w_\psi \tag{1.26c}$$
$$v' = \frac{1}{m}(F_x \cos\alpha - F_y \sin\alpha) \tag{1.26d}$$
$$\alpha' = w_\psi - \frac{F_x \sin\alpha + F_y \cos\alpha}{mv} \tag{1.26e}$$
$$w'_\psi = \frac{1}{I_{zz}}(F_{sf} l_f \cos(\delta) - F_{sr} l_r + F_{lf} l_f \sin(\delta)) \tag{1.26f}$$
$$\delta' = w_\delta \tag{1.26g}$$

with controls s.t. $|w_\delta(t)| \leq w_{\delta max}$ and $F_B(t) \in [F_{Bmin}, F_{Bmax}]$, $l_f, l_r, \frac{1}{I_{zz}}, e_{SP}$ are constants and system of forces described in (1.11). Herein $z := (x, y, \psi, v, \alpha, w_\psi, \delta)$ and $u := (w_\delta, F_B)$.

Observation 1.1. *By taking Equations* (1.21) *and considering* $z := (x, y, \psi, v_x, v_y, w_\psi)$ *and* $u := (\delta, F_B)$*:*

"6D" model with controls w_δ, F_B and states v_x and v_y:

$$x' = v_x \tag{1.27a}$$
$$y' = v_y \tag{1.27b}$$
$$\psi' = w_\psi \tag{1.27c}$$
$$v'_x = \frac{1}{m}(F_x \cos(\psi) - F_y \sin(\psi)) \tag{1.27d}$$
$$v'_y = \frac{1}{m}(F_x \sin(\psi) + F_y \cos(\psi)) \tag{1.27e}$$
$$w'_\psi = \frac{1}{I_{zz}}(F_{sf} l_f \cos(\delta) - F_{sr} l_r + F_{lf} l_f \sin(\delta)) \tag{1.27f}$$

Equations (1.26) *by considering* $z := (x, y, \psi, v, \alpha, w_\psi)$ *and* $u := (\delta, F_B)$ *give:*

"6D" model with controls w_δ, F_B and states α and v:

$$x' = v\cos(\psi - \alpha) \tag{1.28a}$$
$$y' = v\sin(\psi - \alpha) \tag{1.28b}$$
$$\psi' = w_\psi \tag{1.28c}$$
$$v' = \frac{1}{m}(F_x\cos\alpha - F_y\sin\alpha) \tag{1.28d}$$
$$\alpha' = w_\psi - \frac{F_x\sin\alpha + F_y\cos\alpha}{mv} \tag{1.28e}$$
$$w'_\psi = \frac{1}{I_{zz}}(F_{sf}l_f\cos(\delta) - F_{sr}l_r + F_{lf}l_f\sin(\delta)) \tag{1.28f}$$

For all the systems presented in this subsection the following condition has to be imposed

$$F_{sf}^2 + F_{lf}^2 \le g^2 m_f^2, \qquad F_{sr}^2 + F_{lr}^2 \le g^2 m_r^2, \tag{1.29}$$

where $g = 9.81[m/s^2]$ and m_f, m_r is the mass at the front and rear wheel respectively defined as:

$$m_f = \frac{m\cdot l_r}{l_f + l_r}, \quad m_r = \frac{m\cdot l_f}{l_f + l_r} \tag{1.30}$$

To compare the dynamics presented in the next subsections of this chapter, Problem 1.2 is optimized with OCPID-DAE1 software, details will be treated in Section 1 of Chapter IV.

Problem 1.2. *Assuming the notation in* (1.1), *find* $z = (z_1, z_2, \dots, z_{n_z})$ *and* u *such that:*

$$\begin{aligned} \min_u \quad & c_1 t_f + c_2\bar{x} \\ s.t. \quad & z(t) = f(t, z(t), u(t)) \text{ for a.e. } t \in [t_0, t_f], \\ & z(t_0) = z_0, \\ & z_1(t_f) = \bar{x}, \quad z_2(t_f) > \bar{y}, \quad z_3(t_f) = 0, \\ & u \in \mathcal{U} := \{u : \mathbb{R} \to U \subset \mathbb{R}^{n_u}, \text{ measurable}\}, \end{aligned}$$

where $c = (c_1, c_2)^\top$ *is a constant vector,* $\bar{x}$, $\bar{y}$ *are constants or optimized parameters,* $z_1(t)$ *denotes the x-position of the reference vehicle at time* t, $z_2(t)$ *denotes the y-position of the reference vehicle at time* t, *and* $z_3(t)$ *denotes the yaw angle of the reference vehicle at time* t.

1.2 The "4D" point mass model

Taken the model described in (1.26), let F_B in (1.14) be the only force acting on the system i.e. the drag forces, rolling resistance forces and lateral tire forces are neglected. Therefore the vehicle is identified with its center of gravity and the side slip angle α is equal to zero, i.e. the steering angle δ coincides with the yaw angle ψ, as shown in Figure 1.2.

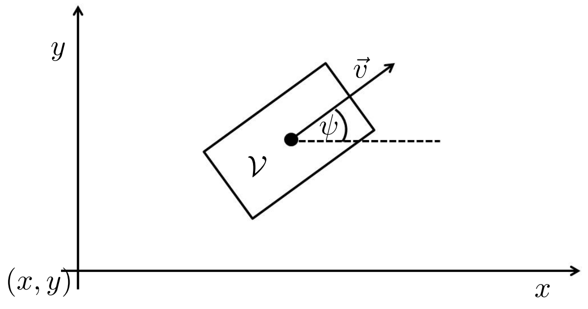


Figure 1.2: Geometrical description of the point mass car model.

Taking into account Equation (1.8), the system of differential equations is:

"4D" point mass model with controls w_ψ, F_B:

$$x' = v\cos(\psi) \tag{1.31a}$$
$$y' = v\sin(\psi) \tag{1.31b}$$
$$\psi' = w_\psi \tag{1.31c}$$
$$v' = -\frac{F_B}{m} \tag{1.31d}$$

with $z := (x, y, \psi, v)$ and $u := (w_\psi, F_B)$.

In Figure 1.3 the point mass model of system in (1.31) is compared with the single track model of system in (1.26). Given an initial x-position, the reference vehicle has to bypass as quick as possible the obstacle and reach the same x-position as the obstacle with null velocity in y direction. The depicted trajectories are the path of the reference vehicle and they start from all different initial x-positions. The initial vehicle velocity is the same for each trajectory and a fixed obstacle of width $2[m]$ is in $x = 50[m]$. The point mass model allows the vehicle to go closer to the obstacle (trajectories are drawn up to a distance of about $25[m]$ to the obstacle), while the single track allows trajectory only about $35[m]$ distant. This observation highlights the fact that the point mass model allows a broader range of maneuvers in this scenario than the single track model. Therefore, unavoidable collisions for the point mass model are also unavoidable for the single track model. In this sense adopting the point mass model for the vehicle motion is a prudential point of view for modeling the decision of unavoidable collision. This concept is useful in deciding the activation point of collision avoidance systems, where for legal issues it has to activate only when a collision is surely unavoidable.

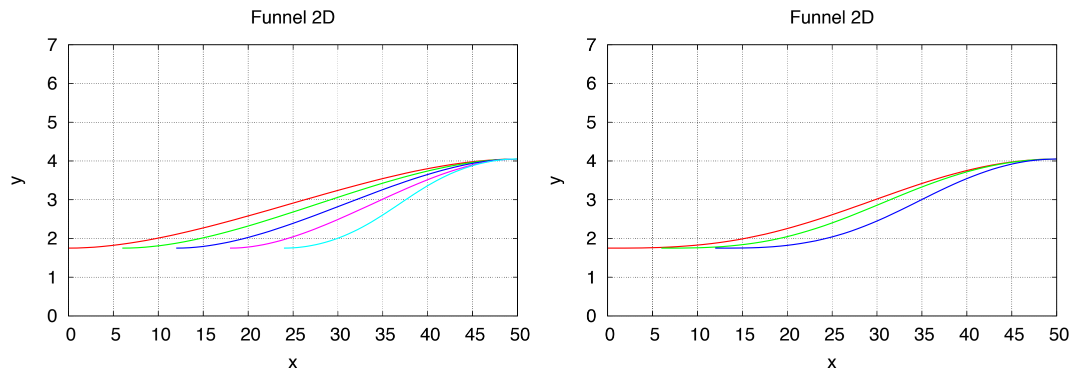


Figure 1.3: Minimal time trajectories obtained by solving Problem 1.2 from several initial x-position are considered with initial velocity $v(t_0) = 25[m/s]$. At the left picture, the system in (1.31) for the 4D point mass model is considered, at the right one it is system in (1.26) for the 7D model with α, v as states.

1.3 The kinematic single track model "3D" or "5D".

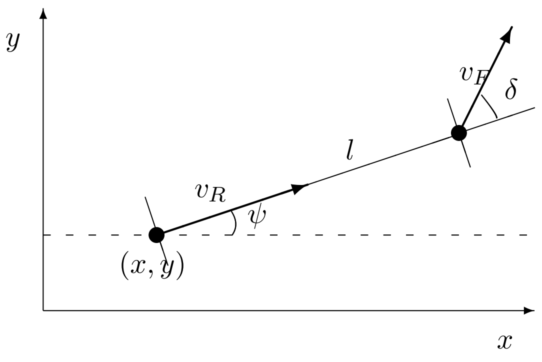


Figure 1.4: Geometrical description of the kinematic car model.

Simplifying assumptions are the low lateral acceleration and the lateral tire forces are negligible (the lateral velocity component of the tire-road contact point vanishes). Let $l = l_r + l_f$ be the distance from rear axle to front axle and let (x, y) be the midpoint position of rear axle of car, see Figure 1.4. The midpoint position and velocity of the rear axle of the car compute to:

$$r_R = \begin{pmatrix} x \\ y \end{pmatrix}, \quad v_R = \begin{pmatrix} x' \\ y' \end{pmatrix}. \tag{1.32}$$

In the front axle of the car the midpoint position and velocity compute to:

$$r_F = \begin{pmatrix} x_F \\ y_F \end{pmatrix} = r_R + S(\psi)\begin{pmatrix} l \\ 0 \end{pmatrix} = \begin{pmatrix} x + l\cos\psi \\ y + l\sin\psi \end{pmatrix},$$

$$\tag{1.33}$$

$$v_F = \begin{pmatrix} x'_F \\ y'_F \end{pmatrix} = \begin{pmatrix} x' - l\psi'\sin\psi \\ y' + l\psi'\cos\psi \end{pmatrix},$$

where $S(\psi)$ is the anti-clockwise rotation matrix of angle ψ defined in Definition 1.3. Under the assumption that the lateral velocity component at rear axle vanishes, then the velocity v_R with module v is

$$v_R = S(\psi)\begin{pmatrix} v \\ 0 \end{pmatrix} = \begin{pmatrix} v\cos\psi \\ v\sin\psi \end{pmatrix}. \tag{1.34}$$

This and (1.32) lead to the differential equations for the position (x, y) of the midpoint of the rear axle:

$$x'(t) = v(t)\cos\psi(t), \quad y'(t) = v(t)\sin\psi(t). \tag{1.35}$$

To derive the differential equation for the yaw angle ψ, under the assumption that the lateral velocity component at front axle vanishes, it holds:

$$v_F = S(\psi + \delta)\begin{pmatrix} \tilde{v} \\ 0 \end{pmatrix} \quad \Rightarrow \quad \psi'(t) = \frac{v(t)}{l}\tan\delta(t), \tag{1.36}$$

with $\tilde{v}$ module of v_F. Summarizing, if $z := (x, y, \psi)$ and the velocity $v(t)$ and the steering angle $\delta(t)$ are known (i.e. $u := (v, \delta)$), the position of the vehicle is given by the differential equations:

<u>"3D" kinematic single track model with controls δ, v:</u>

$$x'(t) = v(t)\cos\psi(t) \tag{1.37a}$$
$$y'(t) = v(t)\sin\psi(t) \tag{1.37b}$$
$$\psi'(t) = \frac{v(t)}{l}\tan\delta(t) \tag{1.37c}$$

Observation 1.3. *If $u := (\delta, F_B)$ and $z = (x, y, \psi, v)$ then:*

<u>"4D" kinematic single track model with controls δ, F_B:</u>

$$x'(t) = v(t)\cos\psi(t) \tag{1.38a}$$
$$y'(t) = v(t)\sin\psi(t) \tag{1.38b}$$
$$\psi'(t) = \frac{v(t)}{l}\tan\delta(t) \tag{1.38c}$$
$$v'(t) = -F_B(t)/m \tag{1.38d}$$

where the additional state constraint $|\delta| \leq 0.5263[rad]$ has to hold. If the control parameters are the braking force $F_B(t)$ and the steering angular velocity $w_\delta(t)$ and $z := (x, y, \psi, v, \delta)$ then:

<u>"5D" kinematic single track model with controls ω_δ, F_B:</u>

$$x'(t) = v(t)\cos\psi(t) \tag{1.39a}$$
$$y'(t) = v(t)\sin\psi(t) \tag{1.39b}$$
$$\psi'(t) = \frac{v(t)}{l}\tan\delta(t) \tag{1.39c}$$
$$v'(t) = -F_B(t)/m \tag{1.39d}$$
$$\delta'(t) = \omega_\delta(t) \tag{1.39e}$$

where the additional state constraints $|\delta| \leq 0.5263[rad], |\delta'| \leq 0.5[rad/s]$ have to be satisfied.

In Figure 1.5 the three kinematic models are presented. Between the "3D" and the "4D" model no relevant difference appears, however the "5D" model shows a smoother behavior in the steering function than the other two kinematic models, which is the expected behaviour imposed by Equation (1.39e).

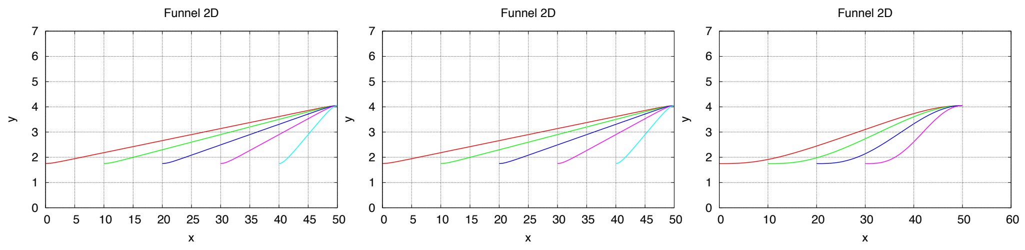


Figure 1.5: Minimal time trajectories for the kinematic systems in (1.37), (1.38), and (1.39) (reading from left to right), obtained by solving Problem 1.2 from several initial x-position, are considered with initial velocity $v(t_0) = 35[m/s]$.

1.4 Conclusions

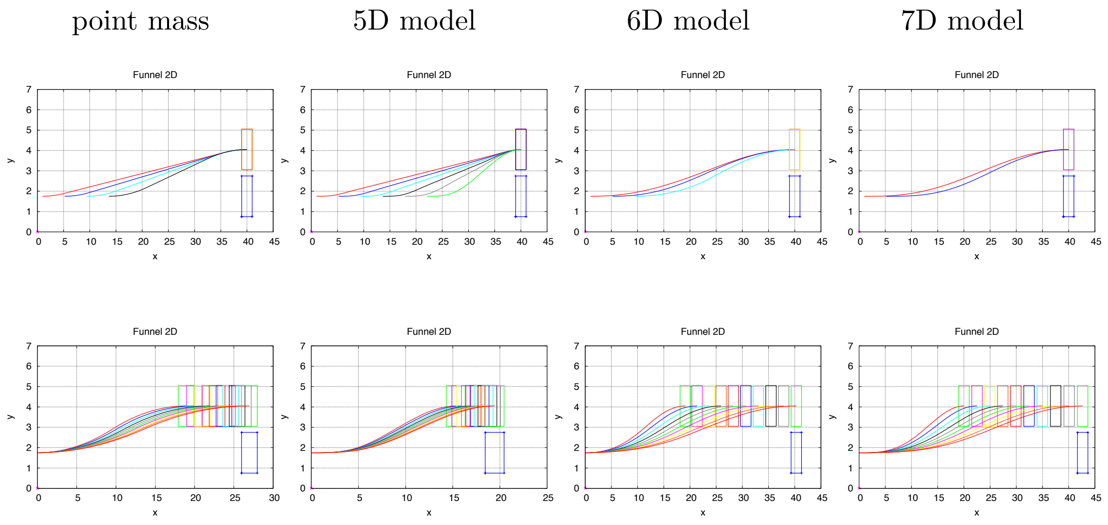


Figure 1.6: A comparison of the 4D point mass model in system in (1.31) (left), the kinematic 5D system in (1.39) (middle left), the single track 6D system in (1.28) (middle right) and the single track 7D system in (1.26) (right) is shown. First row shows minimal time trajectories from several initial x-positions. Second row plots trajectories minimazing the initial x distance to the obstacle from several initial velocities.

Several vehicle dynamics were given in this section. In the evaluation of a good model three criteria should be taken into account. The first criteria concerns the capabilities of the model to represent well the real motion of a vehicle. Keeping this in mind, a prudential point of view should be assumed whenever a collision decision has to be taken. The automotive industry wants to be sure that a collision is happening in order to avoid false activations of the collision avoidance systems. In this case a vehicle model that allows a larger number of maneuvers than a real car is attractive, since a collision decision for it implies a collision decision for the less agile real cars. The third criteria is looking at a computational point of view, where lower dimensional model speed up the numerical calculations, while higher dimensional and highly non-linear models are difficult to handle.

Among all the kinematic models presented in Figure 1.5 a good approximation of capabilities of real vehicles is only the 5D model. Indeed by looking at smoothness of the steering effect and at the minimum x initial distance to an obstacle positioned at $50[m]$, it appears that 3D and 4D kinematic models give an avoidance trajectory starting at up to $10[m]$ distance to the obstacle, instead the 5D kinematic model allows only trajectories starting at up to $20[m]$ distance to the obstacle. This means that for the 5D model is more difficult to avoid a collision than for the lower dimensional kinematic models, in agreement with real life vehicle behaviors. With respect to the 5D model, the 4D point mass model has both the advantages of being one dimension lower and performing trajectories even closer to the results given by complex models as 7D and 6D, see Figure 1.6, and it is easy to handle numerically.

Complex single track 6D and 7D models in (1.21), (1.26), (1.27), and (1.28) should be considered for implementation of trajectories in real cars, thanks to their adherence to reality. The choice between 6D and 7D models can be based on the computational time limits or on the preference of the controls considered (steering velocity and braking force for the 7D, instead of steering angle and braking force of the 6D). Often models in (1.21), (1.27) are preferred over models in (1.26), (1.28) to avoid regularity problems whenever the velocity is taking zero value.

2 Car traffic scenarios and modeling of constraints

Investigation and classification of conflict situations before a collision has been investigated in the project GIDAS (German In Depth investigation Accident Study). This is a joint project between FAT (Forschungsvereinigung Automobiltechnik or Automotive Industry Research Association) and BASt (Bundesanstalt für Straßenwesen or the Federal Road Research Institute) and it started on July 1999. They developed a database where accidents are classified by level of injuries, scenarios, and occurrence. On the statistical methods they use to classify accidents and the data source, refer to [77]. The level of injuries is based on the MAIS scale, where MAIS2+ are considered severe injuries (see [77] for details).

The department K-EFFS/G of Volkswagen Research, underlined a general classification in Figures 2.1, 2.2, 2.3, and 2.4. In particular in Figures 2.1-2.2, there are situations of conflict before a collision, where all level of injuries (all MAIS) and severe level of injuries (MAIS2+) are considered on **occupants of the reference vehicle**, for all environments, including urban and highways. While in Figures 2.3-2.4, there are situations of conflict before a collision, where all level of injuries (all MAIS) and severe level of injuries (MAIS2+) are considered on **opponents of reference vehicle (bicycle, motorcycle, pedestrian)**, for all environments, including urban.

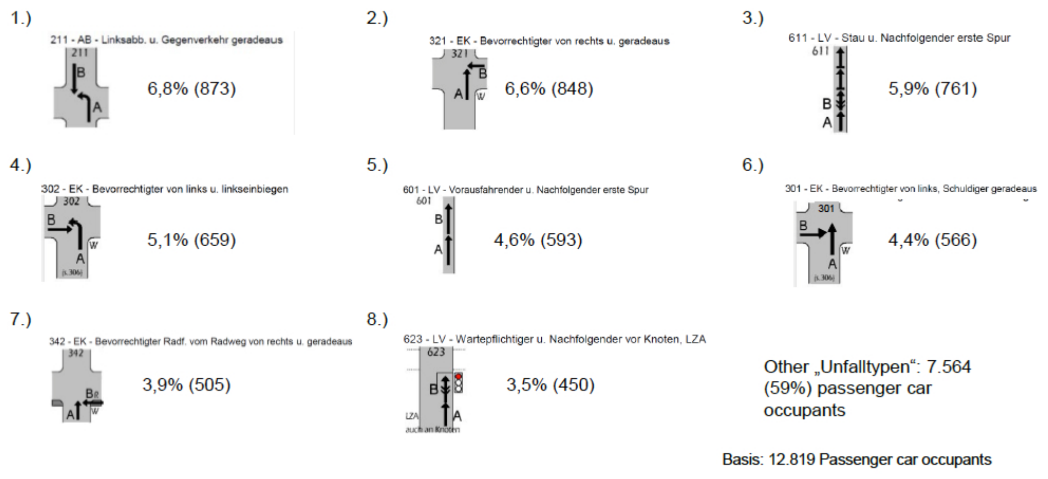

1.) 211 - AB - Linksabb. u. Gegenverkehr geradeaus 6,8% (873)
2.) 321 - EK - Bevorrechtigter von rechts u. geradeaus 6,6% (848)
3.) 611 - LV - Stau u. Nachfolgender erste Spur 5,9% (761)
4.) 302 - EK - Bevorrechtigter von links u. linkseinbiegen 5,1% (659)
5.) 601 - LV - Vorausfahrender u. Nachfolgender erste Spur 4,6% (593)
6.) 301 - EK - Bevorrechtigter von links, Schuldiger geradeaus 4,4% (566)
7.) 342 - EK - Bevorrechtigter Radf. vom Radweg von rechts u. geradeaus 3,9% (505)
8.) 623 - LV - Wartepflichtiger u. Nachfolgender vor Knoten, LZA 3,5% (450)
Other „Unfalltypen“: 7.564 (59%) passenger car occupants
Basis: 12.819 Passenger car occupants

Figure 2.1: From the department K-EFFS/G of Volkswagen Research: Most frequent situations of conflict for all environments, for all level of injuries (all MAIS) are considered on occupants of the reference vehicle.

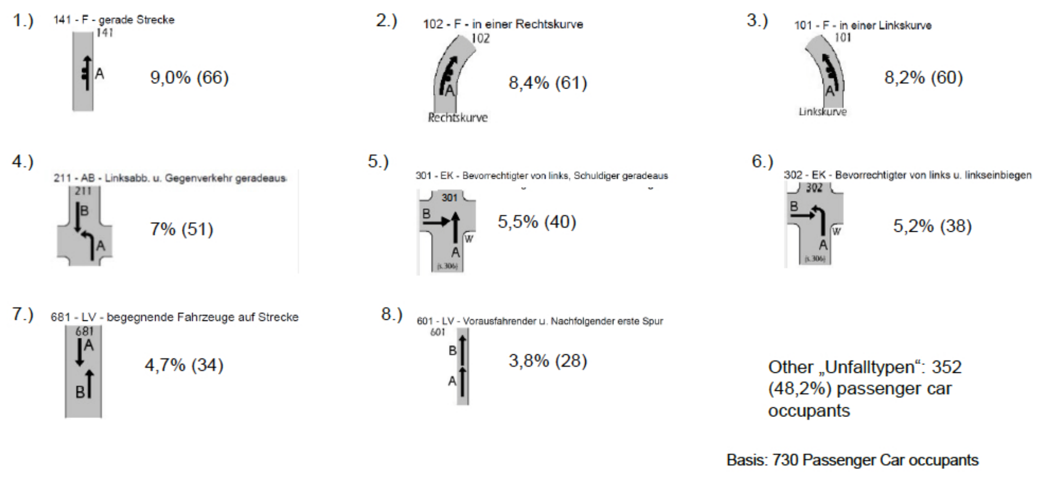
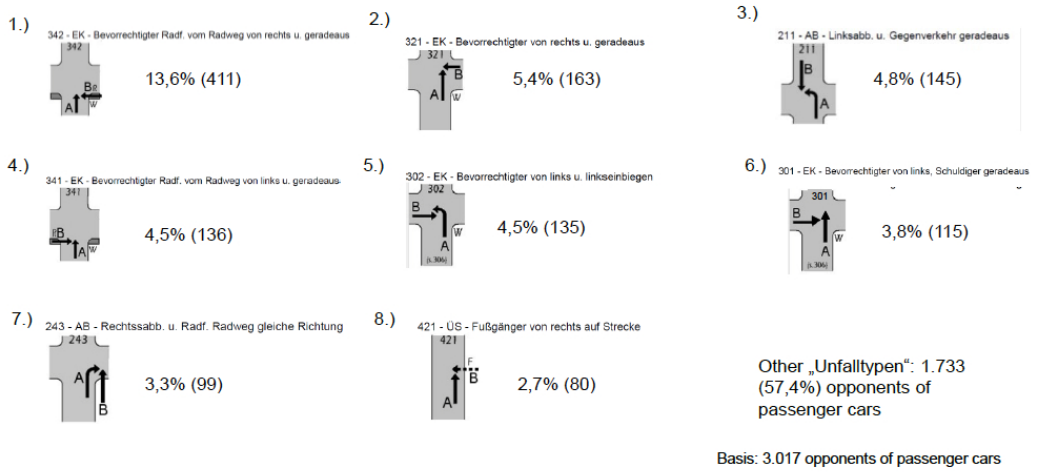
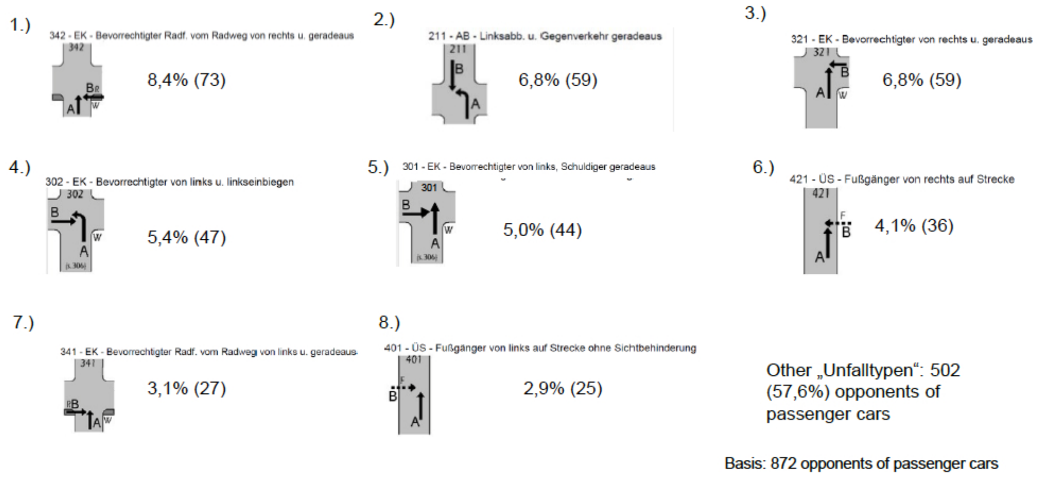

1.) 141 - F - gerade Strecke 9,0% (66)
2.) 102 - F - in einer Rechtskurve 8,4% (61)
3.) 101 - F - in einer Linkskurve 8,2% (60)
4.) 211 - AB - Linksabb. u. Gegenverkehr geradeaus 7% (51)
5.) 301 - EK - Bevorrechtigter von links, Schuldiger geradeaus 5,5% (40)
6.) 302 - EK - Bevorrechtigter von links u. linkseinbiegen 5,2% (38)
7.) 681 - LV - begegnende Fahrzeuge auf Strecke 4,7% (34)
8.) 601 - LV - Vorausfahrender u. Nachfolgender erste Spur 3,8% (28)
Other „Unfalltypen“: 352 (48,2%) passenger car occupants
Basis: 730 Passenger Car occupants

Figure 2.2: From the department K-EFFS/G of Volkswagen Research: Most frequent situations of conflict for all environments, for severe level of injuries (MAIS2+) are considered on occupants of the reference vehicle.



1.)
342 - EK - Bevorrechtigter Radf. vom Radweg von rechts u. geradeaus
13,6% (411)
2.)
321 - EK - Bevorrechtigter von rechts u. geradeaus
5,4% (163)
3.)
211 - AB - Linksabb. u. Gegenverkehr geradeaus
4,8% (145)
4.)
341 - EK - Bevorrechtigter Radf. vom Radweg von links u. geradeaus
4,5% (136)
5.)
302 - EK - Bevorrechtigter von links u. linkseinbiegen
4,5% (135)
6.)
301 - EK - Bevorrechtigter von links, Schuldiger geradeaus
3,8% (115)
7.)
243 - AB - Rechtssabb. u. Radf. Radweg gleiche Richtung
3,3% (99)
8.)
421 - ÜS - Fußgänger von rechts auf Strecke
2,7% (80)
Other „Unfalltypen": 1.733 (57,4%) opponents of passenger cars
Basis: 3.017 opponents of passenger cars

Figure 2.3: From the department K-EFFS/G of Volkswagen Research: Most frequent situations of conflict for all environments, for all level of injuries (all MAIS) are considered on opponents of reference vehicle (bicycle, motorcycle, pedestrian).



1.)
342 - EK - Bevorrechtigter Radf. vom Radweg von rechts u. geradeaus
8,4% (73)
2.)
211 - AB - Linksabb. u. Gegenverkehr geradeaus
6,8% (59)
3.)
321 - EK - Bevorrechtigter von rechts u. geradeaus
6,8% (59)
4.)
302 - EK - Bevorrechtigter von links u. linkseinbiegen
5,4% (47)
5.)
301 - EK - Bevorrechtigter von links, Schuldiger geradeaus
5,0% (44)
6.)
421 - ÜS - Fußgänger von rechts auf Strecke
4,1% (36)
7.)
341 - EK - Bevorrechtigter Radf. vom Radweg von links u. geradeaus
3,1% (27)
8.)
401 - ÜS - Fußgänger von links auf Strecke ohne Sichtbehinderung
2,9% (25)
Other „Unfalltypen": 502 (57,6%) opponents of passenger cars
Basis: 872 opponents of passenger cars

Figure 2.4: From the department K-EFFS/G of Volkswagen Research: Most frequent situations of conflict for all environments, for severe level of injuries (MAIS2+) are considered on opponents of reference vehicle (bicycle, motorcycle, pedestrian).

The car traffic scenarios analyzed in this thesis are collected among the most frequent according to GIDAS scenario, for all level of injuries and all environments. Figures 2.1, 2.2, 2.3, and 2.4 show that the road geometries involved are mainly showing that crossing, straight roads, curves and roundabout, with a maximum of four vehicles per scenario.

As physical entities, the notion of car traffic scenario includes the vehicle shape and motion, road boundaries, obstacle shape and motion and target area. The goal of this chapter is to unify these problems and provide a general framework for modeling such entities through a level set approach. To this purpose the notion of level set function plays a crucial role in this chapter. The function $h : \mathbb{R}^n \to \mathbb{R}$ is a level set function for a set $\mathcal{H} \subset \mathbb{R}^n$ if it holds

$$X \in \mathcal{H} \quad \iff \quad h(X) \leq 0. \tag{2.1}$$

There always exists such a function since one can use the signed distance function to $\mathcal{H}$, i.e. $h(X) = d_{\mathcal{H}}(X)$ such that

$$d_{\mathcal{H}}(X) := \begin{cases} d(X, \mathcal{H}) & \text{if } X \notin \mathcal{H} \\ -d(X, \complement\mathcal{H}) & \text{if } X \in \mathcal{H} \end{cases} \tag{2.2}$$

where $\complement\mathcal{H} = \mathbb{R}^n \backslash \mathcal{H}$. For the problem considered in this thesis, at least Lipschitz continuity for the level set function is required. In some cases it is possible to define the set $\mathcal{H}$ as the intersection of sets with smooth boundaries, i.e. level sets of smooth functions. This becomes important for sensitivity analysis issues and to find numerical solutions via direct methods of Section 2.1 in Chapter III.

2.1 Target

Let $z = (z_1, z_2, \ldots, z_{n_z})$ be the state for the system (1.1) with dynamics f described in Section 1, and let (z_1, z_2) be the evolution in time of the position of the vehicle, while z_3 is the yaw angle evolution in time. The target set Ω represents the area in a car traffic scenario where the vehicle will be at time t_f. Such a set can be thought of as the set of points in the space where the driver would take back the control of the car if any collision avoidance system was activated, or the collision warnings would stop. Therefore the target set would be:

$$\Omega = \left\{ \begin{array}{l} z \in \mathbb{R}^{n_z} \ \mid \ az_1 + bz_2 + c \leq 0, a, b, c \in \mathbb{R} \\ \qquad \text{and } |z_3(t_f) - z_3(t_0) - \beta| \leq \epsilon, \beta \in [-\pi/2, \pi/2] \end{array} \right\}, \tag{2.3}$$

meaning that the vehicle has to reach the half-space $\mathcal{H} = \{(z_1, z_2) \in \mathbb{R}^2 \,|\, az_1 + bz_2 + c \leq 0, a, b, c \in \mathbb{R}\}$ and with a certain orientation given by the yaw angle condition. For instance for straight road scenarios the $|z_3(t_f) - z_3(t_0)| \leq \epsilon$ has to hold for a small threshold $\epsilon > 0$. For a curve or a crossing scenario, $|z_3(t_f) - z_3(t_0) - \beta| \leq \epsilon, \beta \in [-\pi/2, \pi/2]$.

The associated Lipschitz continuous function $\varphi : \mathbb{R}^{n_z} \to \mathbb{R}$ is

$$\varphi(z) := \max\Big((az_1 + bz_2 + c), (|z_3(t_f) - z_3(t_0) - \beta| - \epsilon)\Big), \tag{2.4}$$

where $a, b, c \in \mathbb{R}$ and $\beta \in [-\pi/2, \pi/2]$.

Observation 2.1. *The set Ω is an intersection of level sets with smooth boundaries, i.e.*

$$\Omega = \left\{ z \in \mathbb{R}^{n_z} \mid \widetilde{\varphi}(z) \le 0_{n_{\widetilde{\varphi}}} \right\}, \tag{2.5}$$

with $\widetilde{\varphi} : \mathbb{R}^{n_z} \to \mathbb{R}^{n_{\widetilde{\varphi}}}$ smooth with $n_{\widetilde{\varphi}} = 3$, such that

$$\widetilde{\varphi} := \begin{pmatrix} az_1 + bz_2 + c \\ z_3(t_f) - z_3(t_0) - \beta - \epsilon \\ -z_3(t_f) + z_3(t_0) + \beta - \epsilon \end{pmatrix}, \tag{2.6}$$

where $a, b, c \in \mathbb{R}$, $\beta \in [-\pi/2, \pi/2]$, $\epsilon \in \mathbb{R}$ small.

2.2 Road configurations

As presented in the introduction to this chapter, relevant road geometries are summarized in four categories: straight, curve, crossing, roundabout. Our aim is to define the road as a mathematical object according to its shape. A general road setting defined only by a list of points will also be considered. This generalization finds its motivation in the data structure of sensor output (see [89]), where the sensor detects only some points that are likely to belong to the road and returns such a list to the user. A simple idea on how to generate the set describing a straight road and a roundabout opens this section.

- A straight road of the type in Figure 2.5

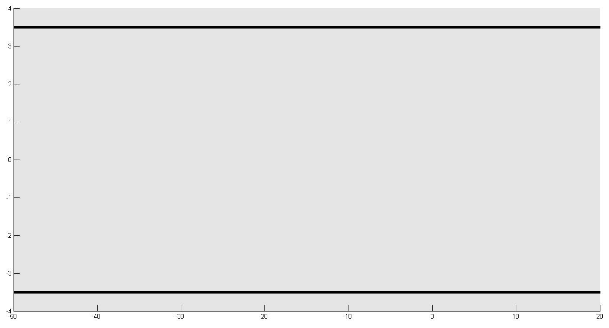


Figure 2.5: Straight road geometry.

is characterized by the set of points

$$\mathcal{K}_r = \Big\{ (x, y) \in \mathbb{R}^2 \,|\, y_{down} \le y \le y_{up} \Big\}, \tag{2.7}$$

with y_{down}, y_{up} constants. The set in (2.7) is the intersection of the two half-spaces of $\mathbb{R}^2$ with smooth boundaries

$$\begin{aligned} \mathcal{K}_{r_1} &= \Big\{(x,y) \in \mathbb{R}^2 \,|\, y_{down} - y \leq 0\Big\}, \\ \mathcal{K}_{r_2} &= \Big\{(x,y) \in \mathbb{R}^2 \,|\, y - y_{up} \leq 0\Big\}. \end{aligned} \tag{2.8}$$

- The roundabout shown in Figure 2.6

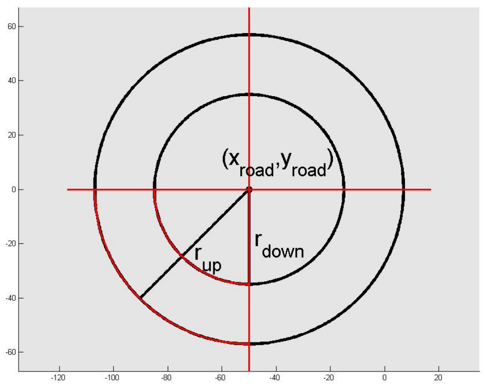


Figure 2.6: Roundabout-curved road geometry.

is defined as

$$\mathcal{K}_r = \left\{(x,y) \in \mathbb{R}^2 \,|\, r_{down}^2 \leq (x - x_{road})^2 + (y - y_{road})^2 \leq r_{up}^2\right\}, \tag{2.9}$$

where (x_{road}, y_{road}) is the center of the roundabout and $r_{up} - r_{down}$ is the constant width of the road. Again the set in (2.9) is the intersection of sets with smooth boundaries

$$\begin{aligned} \mathcal{K}_{r_1} &= \Big\{(x,y) \in \mathbb{R}^2 \,|\, (x - x_{road})^2 + (y - y_{road})^2 - r_{up}^2 \leq 0\Big\}, \\ \mathcal{K}_{r_2} &= \Big\{(x,y) \in \mathbb{R}^2 \,|\, -(x - x_{road})^2 - (y - y_{road})^2 + r_{down}^2 \leq 0\Big\}. \end{aligned} \tag{2.10}$$

To characterize the set $\mathcal{K}_r$, a level set function denoted as g_r is used, and whenever possible, a characterization through a smooth function $\widetilde{g}_r$ is given. For instance, let g_{r_1}, g_{r_2} be functions from $\mathbb{R}^2$ to $\mathbb{R}$, then

- in the case of (2.7),

$$\begin{aligned} \mathcal{K}_{r_1} \ni (x,y) &\iff g_{r_1}(x,y) = y_{down} - y \leq 0, \\ \mathcal{K}_{r_2} \ni (x,y) &\iff g_{r_2}(x,y) = y - y_{up} \leq 0, \end{aligned} \tag{2.11}$$

and

$$\begin{aligned}\mathcal{K}_{r_1} \cap \mathcal{K}_{r_2} = \mathcal{K}_r \ni (x,y) \iff \\ g_r(x,y) = \max\left(g_{r_1}(x,y), g_{r_2}(x,y)\right) \leq 0,\end{aligned} \tag{2.12}$$

where g_r Lipschitz continuous, or alternatively,

$$\mathcal{K}_r \ni (x,y) \iff \widetilde{g}_r(x,y) := \begin{pmatrix} y_{down} - y \\ y - y_{up} \end{pmatrix} \leq 0_{n_{\widetilde{g}_r}}, \tag{2.13}$$

where $\widetilde{g}_r : \mathbb{R}^2 \to \mathbb{R}^{n_{\widetilde{g}_r}}$ is smooth and $n_{\widetilde{g}_r} = 2$;

- in the case of (2.9),

$$\begin{aligned}\mathcal{K}_{r_1} \ni (x,y) \iff \\ g_{r_1}(x,y) = (x - x_{road})^2 + (y - y_{road})^2 - r_{up}^2 \leq 0, \\ \\ \mathcal{K}_{r_2} \ni (x,y) \iff \\ g_{r_2}(x,y) = -(x - x_{road})^2 - (y - y_{road})^2 + r_{down}^2 \leq 0,\end{aligned} \tag{2.14}$$

and

$$\begin{aligned}\mathcal{K}_{r_1} \cap \mathcal{K}_{r_2} = \mathcal{K}_r \ni (x,y) \iff \\ g_r(x,y) = \max\left(g_{r_1}(x,y), g_{r_2}(x,y)\right) \leq 0,\end{aligned} \tag{2.15}$$

where g_r Lipschitz continuous, or alternatively, $\mathcal{K}_r \ni (x,y)$ if and only if

$$\widetilde{g}_r(x,y) := \begin{pmatrix} (x - x_{road})^2 + (y - y_{road})^2 - r_{up}^2 \\ -(x - x_{road})^2 - (y - y_{road})^2 + r_{down}^2 \end{pmatrix} \leq 0_{n_{\widetilde{g}_r}}, \tag{2.16}$$

where $\widetilde{g}_r : \mathbb{R}^2 \to \mathbb{R}^{n_{\widetilde{g}_r}}$ is smooth and $n_{\widetilde{g}_r} = 2$.

Remark 2.2. *A general and simple rule for constructing Lipschitz continuous level set functions is the following. Assume that g_1 (resp. g_2) are Lipschitz continuous level set functions for the set $\mathcal{K}_1$ (resp. $\mathcal{K}_2$), that is, $g_i(x) \leq 0 \Leftrightarrow x \in \mathcal{K}_i$, for $i = 1, 2$. Then*

$$\max(g_1(x), g_2(x)) \leq 0 \quad \Leftrightarrow \quad x \in \mathcal{K}_1 \cap \mathcal{K}_2, \tag{2.17a}$$

$$\min(g_1(x), g_2(x)) \leq 0 \quad \Leftrightarrow \quad x \in \mathcal{K}_1 \cup \mathcal{K}_2. \tag{2.17b}$$

Hence $\max(g_1, g_2)$ *(resp.* $\min(g_1, g_2)$*) is Lipschitz continuous and can be used as a level set function for $\mathcal{K}_1 \cap \mathcal{K}_2$ (resp. $\mathcal{K}_1 \cup \mathcal{K}_2$). Then more complex structures can be coded by combining* (2.17a) *and* (2.17b) *following well-known techniques in computational geometry (see e.g. [40, 55]).*

A generalization to other road geometries is given straightforward.

- A straight road with varying widths as in Figure 2.7

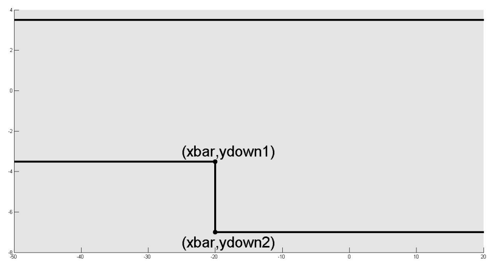


Figure 2.7: Straight road geometry with varying width.

is modelled by the set

$$\mathcal{K}_r := \Big\{(x,y) \in \mathbb{R}^2 \,|\, y_{down}(x) \le y \le y_{up}\Big\}, \tag{2.18}$$

where

$$y_{down}(x) = \begin{cases} y_{down_1} & \text{if } x \le \bar{x}, \\ y_{down_2} & \text{if } x > \bar{x}. \end{cases} \tag{2.19}$$

$y_{up}, y_{down_1}, y_{down_2}, \bar{x}$ constants. Observing that $\mathcal{K}_r = \mathcal{K}_{r_1} \cap \mathcal{K}_{r_2}$,

$$\begin{aligned} \mathcal{K}_{r_1} &= \Big\{(x,y) \in \mathbb{R}^2 \,|\, y_{down}(x) - y \le 0\Big\}, \\ \mathcal{K}_{r_2} &= \Big\{(x,y) \in \mathbb{R}^2 \,|\, y - y_{up} \le 0\Big\}. \end{aligned} \tag{2.20}$$

To define the level set function of (2.18) it would be natural to use the function (2.19)

$$\begin{gathered} g_r(x,y) = \max\left(g_{r_1}(x,y), g_{r_2}(x,y)\right) \le 0, \text{ with} \\ g_{r_1}(x,y) = y_{down}(x) - y \le 0, \quad g_{r_2}(x,y) = y - y_{up} \le 0. \end{gathered} \tag{2.21}$$

However due to the discontinuity of $g_{r_1}(x,y)$, the function $g_r(x,y)$ is not Lipschitz. The desired definition which yields a Lipschitz level set function is instead

$$\begin{aligned} g_r(x,y) = \max\Big(& \min\big(-(y - y_{down_1}), -(x - \bar{x})\big), \\ & -(y - y_{down_2}), (y - y_{up}) \Big). \end{aligned} \tag{2.22}$$

The set $\mathcal{K}_r$ should not be written as intersection of level sets with smooth boundaries and thus cannot be used for models where direct methods are applied.

- A crossing with corner points given by $(x_i, y_i), i = 0, \ldots, 3$, as in Figure 2.9:

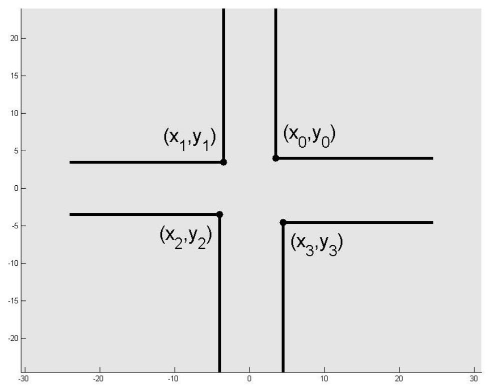


Figure 2.8: Crossing road geometry with smooth boundaries.

is modelled as the set

$$\begin{aligned}\mathcal{K}_r := & \bigcap_{i=0,\ldots,3} \mathcal{K}_{r_i} \\ = & \bigcap_{i=0,\ldots,3} \Big\{ \quad (x,y) \in \mathbb{R}^2 \,|\, (-1)^{\lceil \frac{i}{2} \rceil}(y-y_i) \le 0 \\ & \qquad \text{or } (-1)^{\lfloor \frac{i}{2} \rfloor}(x-x_i) \le 0 \quad \Big\},\end{aligned} \tag{2.23}$$

where $\lfloor x \rfloor$ denotes the biggest integer less or equal than a real number x and $\lceil x \rceil$ denotes the smallest integer grater or equal than x. Thus the level set function is

$$g_r(x,y) = \max_{i=0,\ldots,3} \left(\min\left((-1)^{\lceil \frac{i}{2} \rceil}(y-y_i),\, (-1)^{\lfloor \frac{i}{2} \rfloor}(x-x_i) \right) \right). \tag{2.24}$$

Also in this case the set $\mathcal{K}_r$ cannot be defined as intersection of sets with smooth boundaries. However an alternative way to model the crossing road such that it satisfies this criteria, is obtained considering its boundaries as four circles (Figure 2.9).

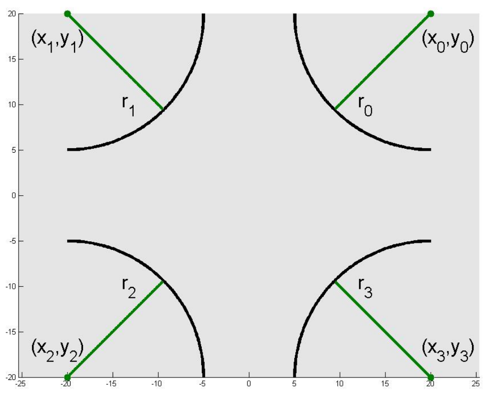


Figure 2.9: Crossing road geometry.

In this case

$$\begin{aligned}\mathcal{K}_r &:= \bigcap_{i=0,\ldots,3} \mathcal{K}_{r_i} \\ &= \bigcap_{i=0,\ldots,3} \Big\{ \quad (x,y) \in [\underline{x}, \bar{x}] \times [\underline{y}, \bar{y}] \subset \mathbb{R}^2 \mid \\ &\qquad\qquad -(x-x_i)^2 - (y-y_i)^2 + r_i^2 \leq 0 \quad \Big\},\end{aligned} \tag{2.25}$$

where $\underline{x} = \max(x_1, x_2), \bar{x} = \min(x_0, x_3)$ and $\underline{y} = \max(y_2, y_3), \bar{y} = \min(y_0, y_1)$. The Lipschitz continuous level set function is

$$\begin{aligned}g_r := \max\Big(&\max_{i=0,\ldots,3} \left(-(x-x_i)^2 - (y-y_i)^2 + r_i^2 \leq 0\right), \\ &(x-\bar{x}), -(x-\underline{x}), (y-\bar{y}), -(y-\underline{y}) \quad \Big).\end{aligned} \tag{2.26}$$

A characterization of the set (2.25) through a smooth function $\widetilde{g}_r : \mathbb{R}^2 \to \mathbb{R}^{n_{\widetilde{g}_r}}$ is:

$$\mathcal{K}_r \ni (x,y) \iff \widetilde{g}_r \leq 0_{n_{\widetilde{g}_r}}, \tag{2.27}$$

where $n_{\widetilde{g}_r} = 8$ and

$$\widetilde{g}_r(x,y) := \begin{pmatrix} -(x-x_1)^2 - (y-y_1)^2 + r_1^2 \\ -(x-x_2)^2 - (y-y_2)^2 + r_2^2 \\ -(x-x_3)^2 - (y-y_3)^2 + r_3^2 \\ -(x-x_4)^2 - (y-y_4)^2 + r_4^2 \\ x - \bar{x} \\ -x + \underline{x} \\ y - \bar{y} \\ -y + \underline{y} \end{pmatrix}. \tag{2.28}$$

- A curved road that starts and ends straight, as in Figure 2.10 with center (x_{road}, y_{road})

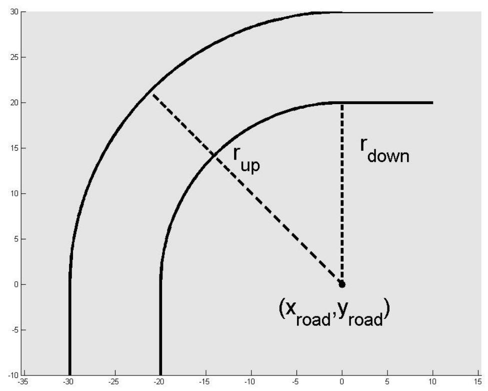


Figure 2.10: Curved road geometry.

is modeled as the union

$$\mathcal{K}_r := \mathcal{K}_{r_1} \cup \mathcal{K}_{r_2} \cup \mathcal{K}_{r_3}, \tag{2.29}$$

with

$$\begin{aligned} \mathcal{K}_{r_1} &= \left\{(x,y) \in \mathbb{R}^2 \,|\, x_{down} \le x \le x_{up},\ \text{if } y \le y_{road}\right\}, \\ \mathcal{K}_{r_2} &= \left\{(x,y) \in \mathbb{R}^2 \,|\, y_{down} \le y \le y_{up},\ \text{if } x \ge x_{road}\right\}, \\ \mathcal{K}_{r_3} &= \Bigg\{(x,y) \in \mathbb{R}^2 \,|\quad r_{down}^2 \le (x-x_{road})^2 + (y-y_{road})^2 \le r_{up}^2, \\ &\qquad \text{if } x \le x_{road} \text{ and } y \ge y_{road}\Bigg\}. \end{aligned} \tag{2.30}$$

where

$$\begin{aligned} y_{up} &= y_{road} + r_{up}, \\ y_{down} &= y_{road} + r_{down}, \\ x_{up} &= x_{road} - r_{down}, \\ x_{down} &= x_{road} - r_{up}, \end{aligned}$$

are constants. The level set function is defined straightforward

$$\begin{aligned} g_r(x,y) = \min\Bigg(&\max\big((x-x_{up}),\, -(x-x_{down}),\, (y-y_{road})\big), \\ &\max\big((y-y_{up}),\, -(y-y_{down}),\, -(x-x_{road})\big), \\ &\max\Bigg(\big((x-x_{road})^2 + (y-y_{road})^2 - r_{up}^2\big), \\ &\qquad -(x-x_{road})^2 - (y-y_{road})^2 + r_{down}^2, \\ &\qquad -(y-y_{road}),\, (x-x_{road}) \Bigg) \Bigg). \end{aligned} \tag{2.31}$$

An alternative formulation for a curve road (see red curve line in Figure 2.6) through a smooth function $\widetilde{g}_r$ is obtained by intersecting the set (2.9) with one of the four possible intersection of two half spaces in $\mathbb{R}^2$ defined by the hyperplanes $x = x_{road}$ and $y = y_{road}$ (red cross in Figure 2.6).

$$\begin{aligned}\mathcal{K}_r = \quad & \left\{(x,y) \in \mathbb{R}^2 \,|\, r_{down}^2 \leq (x - x_{road})^2 + (y - y_{road})^2 \leq r_{up}^2\right\} \bigcap \\ & \left\{(x,y) \in \mathbb{R}^2 \,|\, -1^{\lceil \frac{i}{2} \rceil + 1}(x - x_{road}) \leq 0\right\} \bigcap \\ & \left\{(x,y) \in \mathbb{R}^2 \,|\, -1^{\lfloor \frac{i}{2} \rfloor + 1}(y - y_{road}) \leq 0\right\},\end{aligned} \tag{2.32}$$

where i is given in the set $\{0, 1, 2, 3\}$ and it denotes one of the four subspaces defined by $x = x_{road}$ and $y = y_{road}$ starting from the upper-right and counting anti-clockwise, (x_{road}, y_{road}) is the center of the circle modeling the curve and r_{up}, r_{down} are the radius of the two circles forming the curve boundaries.

– Let $g_r : \mathbb{R}^2 \to \mathbb{R}$ be defined as

$$g_r(x,y) := \max\left(\begin{array}{l} (x - x_{road})^2 + (y - y_{road})^2 - r_{up}^2, \\ -(x - x_{road})^2 - (y - y_{road})^2 + r_{down}^2, \\ -1^{\lceil \frac{i}{2} \rceil + 1}(x - x_{road}), \quad -1^{\lfloor \frac{i}{2} \rfloor + 1}(y - y_{road}) \end{array} \right) \tag{2.33}$$

then g_r is Lipschitz continuous and

$$\mathcal{K}_r \ni (x,y) \iff g_r(x,y) \leq 0; \tag{2.34}$$

– Let $\widetilde{g}_r : \mathbb{R}^2 \to \mathbb{R}^{n_{\widetilde{g}_r}}$ be defined as

$$\widetilde{g}_r(x,y) := \begin{pmatrix} +(x - x_{road})^2 + (y - y_{road})^2 - r_{up}^2 \\ -(x - x_{road})^2 - (y - y_{road})^2 + r_{down}^2 \\ -1^{\lceil \frac{i}{2} \rceil + 1}(x - x_{road}) \\ -1^{\lfloor \frac{i}{2} \rfloor + 1}(y - y_{road}) \end{pmatrix} \tag{2.35}$$

then $\widetilde{g}_r$ is a smooth function with $n_{\widetilde{g}_r} = 4$ and

$$\mathcal{K}_r \ni (x,y) \iff \widetilde{g}_r(x,y) \leq 0_{n_{\widetilde{g}_r}}. \tag{2.36}$$

A more general way to construct level set functions for roads delimited by polygonal lines is shown in Figure 2.11.

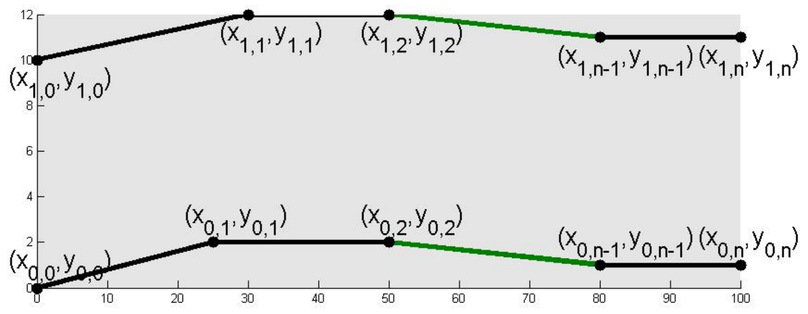


Figure 2.11: Road defined by polynomials interpolating the sensor detected points that belong to the road boundaries.

The set describing the road is characterized as

$$\mathcal{K}_r = \left\{(x,y) \in \mathbb{R}^2 \,|\, p_1(x,y) \leq 0 \text{ and } p_2(x,y) \geq 0\right\}. \tag{2.37}$$

For $j = 1, 2$, $p_j : \mathbb{R}^2 \to \mathbb{R}$ are interpolation functions of the points $(x_{j,i}, y_{j,i})$, $i \in \{0, \ldots, n\}$ detected by the sensor as points belonging to the upper ($j = 1$) and lower ($j = 2$) bounds of the road, as in Figure 2.11. Moreover, for each $j \in \{1, 2\}$

$$p_j = \begin{cases} q_{j,0}(x,y), & \forall x \in [x_{j,0}, x_{j,1}], \quad \forall y \in [y_{j,0}, y_{j,1}] \\ \vdots & \\ q_{j,n-1}(x,y), & \forall x \in [x_{j,n-1}, x_{j,n}], \quad \forall y \in [y_{j,n-1}, y_{j,n}] \end{cases} \tag{2.38}$$

where $q_{j,i} : \mathbb{R}^2 \to R$ are the polynomials of degree at most 1, interpolating the pair of knots $(x_{j,i}, y_{j,i})$ and $(x_{j,i+1}, y_{j,i+1})$ for $i = 0, \ldots, n-1$ and $j = 1, 2$. The level set function associated to set (2.37) is:

$$\begin{aligned} g_r(x,y) = \;& \min_{j=1,2} g_{r_j}(x,y), \\ g_{r_j}(x,y) = \;& \max_{i=0,\ldots,n-1} \Bigg(\min \big(\; (x - x_{j,i+1}), \\ & \qquad -(x - x_{j,i}), \\ & \qquad (y - y_{j,i+1}), \\ & \qquad -(y - y_{j,i}), \\ & \qquad (-1^{j+1} q_{j,i}(x,y)) \; \big) \Bigg). \end{aligned} \tag{2.39}$$

Observation 2.3. *In the particular case where the functions p_j are piecewise linear and convex, the level set function of* (2.37) *is*

$$\begin{aligned} g_r :=& \max_{j=1,2} g_{r_j}, \\ g_{r_j} :=& \max_{i=0,\ldots,n-1} \left(-1^{j+1} q_{j,i}\right). \end{aligned} \tag{2.40}$$

Moreover a characterization of the set (2.37) *through a smooth function* $\widetilde{g}_r : \mathbb{R}^2 \to \mathbb{R}^{n_{\widetilde{g}_r}}$ *is possible*

$$\mathcal{K}_r \ni (x,y) \iff \widetilde{g}_r(x,y) \leq 0_{n_{\widetilde{g}_r}}, n_{\widetilde{g}_r} = 2n, \tag{2.41}$$

with

$$\widetilde{g}_r(x,y) := \begin{pmatrix} +q_{1,0}(x,y) \\ -q_{2,0}(x,y) \\ \vdots \\ +q_{1,n-1}(x,y) \\ -q_{2,n-1}(x,y) \end{pmatrix}. \tag{2.42}$$

Literature on models of the road via piecewise defined cubic polynomials or B-splines is discussed in [51].

2.3 Obstacles geometry and motion

In this subsection equations for modeling the obstacle shape and motion are derived. The obstacle geometry is approximated by circles or rectangles equipped by linear or circular motion. The obstacle i is denoted with $\mathcal{O}_i$ and the "obstacle-free set" is referring to $\mathcal{K}_o$ which is the complement of the union of all $\mathcal{O}_i$, denoting an area that is collision free. Let $X_i(t) = (x_i(t), y_i(t))$ denote the center of obstacle i, which may depend of the time $t \in [t_0, t_f] \subset \mathbb{R}$.

Circular obstacles. The obstacle is approximated by a closed ball:

$$\mathcal{O}_i := \overline{B}(X_i(t), \ell_i), \tag{2.43}$$

centered at $X_i(t) = (x_i(t), y_i(t))$ and with given fixed radius $\ell_i > 0$, for $i = 1, \ldots, k$. Thus the obstacle-free set is defined as:

$$\begin{aligned} \mathcal{K}_o(t) &:= \complement \left(\bigcup_{i=1,\ldots,k} \overline{B}(X_i(t), \ell_i) \right) \\ &= \bigcap_{i=1,\ldots,k} \left\{ (x,y) \in \mathbb{R}^2 \mid (x - x_i(t))^2 + (y - y_i(t))^2 - \ell_i^2 > 0 \right\}. \end{aligned} \tag{2.44}$$

It holds that

$$\begin{aligned} &(x,y) \in \mathcal{K}_o(t) \iff \\ g_o((x,y),t) &:= \max\nolimits_{i=1,\ldots,k} \left(-(x - x_i(t))^2 - (y - y_i(t))^2 + \ell_i^2 \right) < 0, \end{aligned} \tag{2.45}$$

with $g_o : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$ Lipschitz. Moreover

$$(x,y) \in \mathcal{K}_o(t) \iff$$
$$\widetilde{g}_o((x,y),t) := \begin{pmatrix} -(x-x_1(t))^2 - (y-y_1(t))^2 + \ell_1^2 \\ \vdots \\ -(x-x_i(t))^2 - (y-y_i(t))^2 + \ell_4^2 \end{pmatrix} < 0_{n_{\widetilde{g}_o}}, \tag{2.46}$$

with $\widetilde{g}_o : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^{n_{\widetilde{g}_o}}$ smooth, $n_{\widetilde{g}_o} = 4$.

Rectangular obstacles. In the case of rectangular obstacles the following introductory observation is needed.

Observation 2.4. *The rectangle $\mathcal{R} = [-\ell_x, \ell_x] \times [-\ell_y, \ell_y]$ with center in $(0,0)$, orientation $\psi = 0$ and half dimensions $\ell = [\ell_x, \ell_y]$ is a set*

$$\mathcal{R} := \left\{(x,y) \in \mathbb{R}^2 \,|\, -(\ell_x - |x|) \leq 0 \text{ and } -(\ell_y - |y|) \leq 0\right\}. \tag{2.47}$$

Thus it holds

$$(x,y) \notin [-\ell_x, \ell_x] \times [-\ell_y, \ell_y] \iff$$
$$d_\ell(x,y) := \min(\ell_x - |x|, \ell_y - |y|) < 0. \tag{2.48}$$

The obstacle $\mathcal{O}_i(t)$ can also be modelled as an affinely transformed rectangle with center in $X_i(t)$, rotated of an angle $\psi_i(t)$, i.e.

$$\mathcal{O}_i(t) := R_{\psi_i(t)} \mathcal{R}_i + X_i(t), \tag{2.49}$$

where $\mathcal{R}_i$ rectangle of the form in (2.47) with half lengths $\ell_i = (\ell_{i_x}, \ell_{i_y})^\top$, and

$$R_{\psi_i(t)} := \begin{pmatrix} \cos(\psi_i(t)) & -\sin(\psi_i(t)) \\ \sin(\psi_i(t)) & \cos(\psi_i(t)) \end{pmatrix}. \tag{2.50}$$

Given a point $Y \in \mathbb{R}^2$ the condition for avoiding the obstacle $\mathcal{O}_i(t)$ at time t is

$$Y \notin \mathcal{O}_i(t) \iff$$
$$d_{X_i,\psi_i}(Y,t) := d_{\ell_i}(R_{\psi_i(t)}(Y - X_i(t))) < 0, \tag{2.51}$$

where d_{ℓ_i} is defined in (2.48) and

$$R_{\psi_i(t)} := \begin{pmatrix} \cos(\psi_i(t)) & -\sin(\psi_i(t)) \\ \sin(\psi_i(t)) & \cos(\psi_i(t)) \end{pmatrix}. \tag{2.52}$$

Thus if the obstacle set is defined as $\mathcal{K}_o(t) := \complement(\cup_{i=1,\dots,k} \mathcal{O}_i(t))$ the condition for avoidance of the obstacle region is given by the following condition

$$Y \notin \mathcal{O}_i(t), \forall i \iff Y \in \mathcal{K}_o(t) \iff$$
$$g_o(Y,t) := \max_{i=1,\dots,k} d_{X_i,\psi_i}(Y,t) < 0, \tag{2.53}$$

with $g_o : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$.

Observation 2.5. *The four corners* $(X_{ij}(t))_{1\leq j\leq 4}$ *of obstacle* $\mathcal{O}_i(t)$ *are given by*

$$X_{ij}(t) := X_i(t) + R_{-\psi_i(t)} T_j \ell_i, \quad 1 \leq j \leq 4, \tag{2.54}$$

with

$$T_j = \begin{pmatrix} (-1)^{j-1} & 0 \\ 0 & (-1)^{\lfloor \frac{j-1}{2} \rfloor} \end{pmatrix}, \tag{2.55}$$

where $1 \leq j \leq 4$ *and* $\lfloor x \rfloor$ *denotes the integer part of a real* x.

In this rectangular case, it is not possible to derive a condition to characterized the obstacle-free set through a smooth function.

Moving obstacles. The center $X_i(t)$ and the orientation $\psi_i(t)$ of obstacle $\mathcal{O}_i(t)$ can depend on time t. This happens when the obstacle moves. A linear accelerated motion along a straight path is considered:

$$\begin{aligned}
x_i(t) &= x_{i0} + v_{ix0}t + \frac{1}{2}a_{ix0}t^2, && \text{(2.56a)}\\
y_i(t) &= y_{i0} + v_{iy0}t + \frac{1}{2}a_{iy0}t^2, && \text{(2.56b)}\\
v_{ix}(t) &= v_{ix0} + a_{ix0}t, && \text{(2.56c)}\\
v_{iy}(t) &= v_{iy0} + a_{iy0}t, && \text{(2.56d)}\\
\psi_i(t) &= \psi_i(t_0) = \arctan\frac{v_{iy0}}{v_{ix0}}, && \text{(2.56e)}
\end{aligned}$$

where (a_{ix0}, a_{iy0}) is the constant acceleration of the obstacle i, (v_{ix0}, v_{iy0}) is the initial velocity, (x_{i0}, y_{i0}) the initial position and $\psi_i(t_0)$ is the initial yaw angle which is constant during the motion. It can also be that the obstacle motion is along a curved road:

$$\begin{aligned}
\theta_i(t) &= \theta_{i0} + \omega_{i0}t + \frac{1}{2}\alpha_{i0}t^2, && \text{(2.57a)}\\
\omega_i(t) &= \omega_{i0} + \alpha_{i0}t, && \text{(2.57b)}\\
x_i(t) &= a\cos(\theta_i(t)) + x_c, && \text{(2.57c)}\\
y_i(t) &= a\sin(\theta_i(t)) + y_c, && \text{(2.57d)}\\
\psi_i(t) &= \theta_i(t) - \frac{\pi}{2}, && \text{(2.57e)}
\end{aligned}$$

with α_{i0} constant angular acceleration, ω_{i0} initial angular velocity, θ_{i0} the initial angle position, (x_c, y_c) center of motion and a is the initial distance to the center of the motion.

General obstacle motion along a curve C is possible via parametrization of C by a function modeling the evolution of the path of the obstacle in time.

2.4 Vehicle and corresponding level set functions

Let $X(z(t),t) = (z_1(t), z_2(t))$ denote the center of gravity of the vehicle with motion described in Section 1, from which it appears the dependency on the time t and on the state $z = (z_1, z_2, \dots, z_{n_z})$. Indeed for some systems presented in Section 1 conditions on z hold, for instance the condition of Kamm's circle in Equation (1.29). Herein, the tire forces F_{lr}, F_{lf} depend on the state as shown in Equations (1.11d) and (1.11c), while the tire forces F_{sr}, F_{sf} depend on the side slip angles α_r, α_f (see Equations (1.11b), (1.11a)) which depend on several states (see Equations (1.12c), (1.12b)). Such conditions are satisfied if and only if

$$z(t) \in \mathcal{K}_d(t) = \{z(t) \in \mathbb{R}^{n_z} \mid \begin{array}{l} F_{sf}(z(t))^2 + F_{lf}(z(t))^2 \leq g^2 m_f^2, \\ F_{sr}(z(t))^2 + F_{lr}(z(t))^2 \leq g^2 m_r^2\}, \end{array} \tag{2.58}$$

and it holds the equivalence

$$\begin{array}{l} z(t) \in \mathcal{K}_d(t) \iff \\ g_d(z(t)) := \max\big(\begin{array}{l} (F_{sf}(z(t))^2 + F_{lf}(z(t))^2 - g^2 m_f^2), \\ (F_{sr}(z(t))^2 + F_{lr}(z(t))^2 - g^2 m_r^2) \end{array} \big) \leq 0. \end{array} \tag{2.59}$$

Moreover given $\widetilde{g}_d : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^{n_{\widetilde{g}_d}}$ smooth, $n_{\widetilde{g}_d} = 2$, then

$$z(t) \in \mathcal{K}_d(t) \iff \widetilde{g}_d(z(t)) := \begin{pmatrix} (F_{sf}(z(t))^2 + F_{lf}(z(t))^2 - g^2 m_f^2) \\ (F_{sr}(z(t))^2 + F_{lr}(z(t))^2 - g^2 m_r^2) \end{pmatrix} \leq 0_2. \tag{2.60}$$

The shape of the vehicle can be modeled as a circle

$$\mathcal{V}(z(t),t) := \overline{B}(X(z(t),t), \ell), \tag{2.61}$$

where $\overline{B}(X(z(t),t),\ell)$ is a ball in $\mathbb{R}^2$ centered in $X(z(t),t)$ of radius $\ell \in \mathbb{R}$. Or the vehicle can be modelled as a rectangle, using (2.47):

$$\mathcal{V}(z(t),t) := \big\{ \begin{array}{l} X(z(t),t) = (z_1(t), z_2(t)) \in \mathbb{R}^2 \mid \\ -(\ell_x - |z_1(t)|) \leq 0 \text{ and} \\ -(\ell_y - |z_2(t)|) \leq 0 \quad \big\}. \end{array} \tag{2.62}$$

Here $X(z(t),t)$ is the rectangle center, $\ell = (\ell_x, \ell_y)^\top$ half lengths and vertices

$$X_j(z(t),t) = X(z(t),t) + R_{-z_3(t)} T_j \ell, \quad j = 1, \dots, 4, \tag{2.63}$$

with the matrices T_j as in (2.55), R as in (2.52), and z_3 being the yaw angle of the vehicle.

Let $\mathcal{V}(z(t),t)$ be the vehicle set evolving in time with state $z(\cdot)$, solution of the system in (1.1) and let $\mathcal{K}_d$ as in (2.58). Let Ω be a target set, let $\mathcal{K}_r$ be the road set and let $\mathcal{O}_i(t), i = 0, \dots, k$ be the obstacles with associated obstacle-free set $\mathcal{K}_o$.

Then in the time interval $[t_0, t_f]$ the reference vehicle has to drive a safe trajectory which means that $z(\cdot)$ has to satisfy the following condition

$$z(t) \in \mathcal{K}_d(t), \quad \forall t \in [t_0, t_f] \tag{2.64a}$$

$$\mathcal{V}(z(t), t) \cap \complement\left(\mathcal{K}_r \cup \mathcal{K}_o(t)\right) = \emptyset, \quad \forall t \in [t_0, t_f], \tag{2.64b}$$

$$z(t_f) \in \Omega. \tag{2.64c}$$

While Conditions (2.64a) and (2.64c) can be represented by inequalities involving level set functions or smooth functions, (see (2.59) and (2.60) for condition (2.64a), and see (2.4) and (2.6) for condition (2.64c)), for (2.64b) there is no characterization. In the next paragraphs characterizations of the sets $\mathcal{V}(z(t), t)$, $\mathcal{K}_r$, and $\mathcal{K}_o(t)$ will give an at least necessary condition for (2.64b).

Observation 2.6. *If the vehicle is identified with its center of gravity $X(z(t), t)$, condition* (2.64b) *is equivalent to one of the following:*

- *given g_o and g_r as in Subsections 2.3 and 2.2, Lipschitz regular,*

  $$X(z(t), t) \in \mathcal{K}(t) \iff g(z(t), t) \leq 0, \textit{ for each } t \in [t_0, t_f], \tag{2.65}$$

  with $g(z(t), t) := \max(g_r(z(t)), g_o(z(t), t) + \eta)$, $\eta > 0$ small (such that the equality can hold);

- *given $\widetilde{g}_o$ and $\widetilde{g}_r$ as in Subsections 2.3 and 2.2, smooth functions,*

  $$X(z(t), t) \in \mathcal{K}(t) \iff g(z(t), t) \leq 0_{\mathbb{R}^{n_{\widetilde{g}}}}, \textit{ for each } t \in [t_0, t_f], \tag{2.66}$$

  with

  $$\widetilde{g}(z(t), t) := \begin{pmatrix} \widetilde{g}_r(z(t), t) \\ \widetilde{g}_o(z(t), t) + \eta \end{pmatrix} < 0_{n_{\widetilde{g}}}, \tag{2.67}$$

  $\eta > 0$ small;

with $\mathcal{K}(t) := \mathcal{K}_r \cap \mathcal{K}_o(t)$, $\mathcal{K}_r$ and $\mathcal{K}_o(t)$ defined in Subsections 2.3 and 2.2.

If the vehicle is defined as a circle (2.61) or a rectangle (2.62), then a sufficient condition for (2.64b) has to be derived. In particular Equation (2.64b) is equivalent to

$$\mathcal{V}(z(t), t) \subset \mathcal{K}_r, \tag{2.68a}$$

$$\mathcal{V}(z(t), t) \subset \mathcal{K}_o(t). \tag{2.68b}$$

Condition for (2.68a) is satisfied redefining the lower (upper) bound of the road by increasing (decreasing) the values of y_{down}, r_{down} (y_{up}, r_{up}) by the maximum dimension of the vehicle. In the next paragraphs condition (2.68b) is analyzed depending on the obstacle shape.

Circular vehicle. If the vehicle $\mathcal{V}$ is modeled as the ball (2.61) and the obstacles $\mathcal{O}_i$ are described by (2.43) for each $i = 1, \dots, k$, then (2.68b) is equivalent to:

$$B(X(z(t),t),\ell) \cap \bigcup_{1\le i\le k} B(X_i(t),\ell_i) = \emptyset. \tag{2.69}$$

and since for each $i = 1, \dots, k$

$$B(X(z(t),t),\ell) \cap B(X_i(t),\ell_i) = \emptyset \iff \|X(z(t),t) - X_i(t)\|_2 > \ell + \ell_i, \tag{2.70}$$

then condition (2.69) is equivalent to one of the following:

- given $g_o : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}$ Lipschitz,
$$g_o(z(t),t) := \max_{1\le i\le k} -(\|X(z(t),t) - X_i(t)\|_2 - \ell - \ell_i) < 0; \tag{2.71}$$
- given $\widetilde{g}_o : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}^{n_{\widetilde{g}_o}}$ smooth, $n_{\widetilde{g}_o} = k$,
$$\widetilde{g}_o(z(t),t) := \begin{pmatrix} -(z_1(t)-x_1(t))^2 - (z_2(t)-y_1(t))^2 + (\ell+\ell_1)^2 \\ \vdots \\ -(z_1(t)-x_k(t))^2 - (z_2(t)-y_k(t))^2 + (\ell+\ell_k)^2 \end{pmatrix} < 0_{n_{\widetilde{g}_o}}. \tag{2.72}$$

If the obstacles $\mathcal{O}_i(t)$ are the affinely transformed rectangles in (2.49), the following conditions hold simultaneously

$$\begin{aligned} &\forall i = 1,\dots,k, \forall j = 1,\dots,4, \quad X_{i,j}(t) \notin \overline{B}(X(z(t),t),\ell) \iff \\ &\quad \max_{1\le i\le k,\ 1\le j\le 4} -(\|X(z(t),t) - X_{i,j}(t)\|_2 - \ell) < 0, \\ &\forall i = 1,\dots,k, \quad X_i(t) \notin \overline{B}(X(z(t),t),\ell) \iff \\ &\quad \max_{1\le i\le k} -(\|X(z(t),t) - X_i(t)\|_2 - \ell) < 0, \end{aligned} \tag{2.73}$$

where $X_{i,j}(t)$ are the vertices of obstacle i defined in Observation 2.5. Condition (2.73) assures that neither a vertex nor the center of any of the obstacles lie inside the vehicle. The following are equivalent necessary conditions for (2.68b):

- given $g_o : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}$ Lipschitz,
$$g_o(z(t),t) := \max_{1\le i\le k} \left(\max_{\tilde{X}(t)\in\{X_{i,1}(t),\dots,X_{i,4}(t),X_i(t)\}} -\|X(z(t),t) - \tilde{X}(t)\|_2 + \ell \right) < 0; \tag{2.74}$$
- given $\widetilde{g}_o : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}^{n_{\widetilde{g}_o}}$ smooth, $n_{\widetilde{g}_o} = 5k$,
$$\widetilde{g}_o(z(t),t) := \begin{pmatrix} -(z_1(t)-x_{1,1}(t))^2 - (z_2(t)-y_{1,1}(t))^2 + \ell^2 \\ -(z_1(t)-x_{1,2}(t))^2 - (z_2(t)-y_{1,2}(t))^2 + \ell^2 \\ \vdots \\ -(z_1(t)-x_k(t))^2 - (z_2(t)-y_k(t))^2 + \ell^2 \end{pmatrix} < 0_{n_{\widetilde{g}_o}}. \tag{2.75}$$

Rectangular vehicle. If the vehicle is modelled as a rectangle, see (2.62), then two case have to be considered. If the obstacles $\mathcal{O}_i(t)$, $i = 1, \dots, k$ are circles described by (2.43), the following conditions hold simultaneously

$$\begin{aligned}\forall i = 1,\dots,k, \forall j = 1,\dots,4, \quad X_j(z(t),t) \notin \overline{B}(X_i(t),\ell_i) \iff \\ \max_{1\le i\le k,\ 1\le j\le 4} -(\|X_j(z(t),t) - X_i(t)\|_2 - \ell_i) < 0, \\ \forall i = 1,\dots,k, \quad X(z(t),t) \notin \overline{B}(X_i(t),\ell_i) \iff \\ \max_{1\le i\le k} -(\|X(z(t),t) - X_i(t)\|_2 - \ell_i) < 0,\end{aligned} \tag{2.76}$$

where $X_j(z(t),t)$ are the vertices of the vehicle defined in (2.63). Condition (2.76) assures that neither a vertex of the vehicle nor its center lie inside any of the obstacles. The following are equivalent necessary conditions for (2.68b):

- given $g_o : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}$ Lipschitz,

$$g_o(z(t),t) := \max_{1\le i\le k}\left(\max_{\tilde{X}\in\{X_1,\dots,X_4,X\}} -\|X_i(t) - \tilde{X}(z(t),t\|_2 + \ell_i\right) < 0; \tag{2.77}$$

- given $\widetilde{g}_o : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}^{n_{\widetilde{g}_o}}$ smooth, $n_{\widetilde{g}_o} = 5k$,

$$\widetilde{g}_o(z(t),t) := \begin{pmatrix} -(x_1(t) - z_{1,1}(t))^2 - (y_1(t) - z_{2,1}(t))^2 + \ell_1^2 \\ -(x_1(t) - z_{1,2}(t))^2 - (y_1(t) - z_{2,2}(t))^2 + \ell_1^2 \\ \vdots \\ -(x_k(t) - z_1(t))^2 - (y_1(t) - z_2(t))^2 + \ell_k^2 \end{pmatrix} < 0_{n_{\widetilde{g}_o}}. \tag{2.78}$$

Figure 2.12 shows the case where also the obstacles are rectangles and they are described by (2.49).

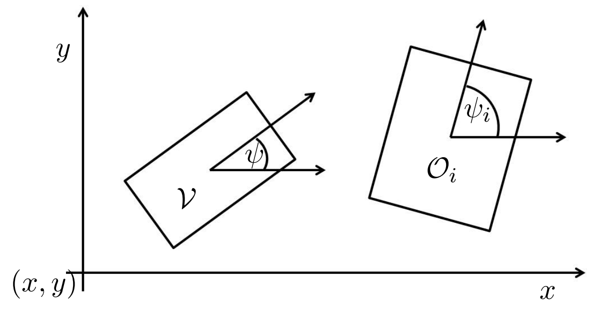


Figure 2.12: Vehicle and obstacles.

Then, as in Equation (2.51), given a point $Y \in \mathbb{R}^2$, the following

$$\begin{aligned} Y \notin \mathcal{V}(z(t),t) \iff \\ d_{X,\psi}(Y) := d_\ell(R_{-\psi(z(t),t)}(Y - X(z(t),t))) < 0\end{aligned} \tag{2.79}$$

gives a condition for the avoidance of the obstacles, where function $d_\ell(\cdot)$ is defined in (2.48). Thus, the following

$$g_o(z(t),t) := \max_{1\le i\le k}\Big(\max_{\tilde{X}_i\in\{X_{i,1},\dots,X_{i,4},X_i\}} d_{X,\psi}(\tilde{X}_i(t)), \quad \max_{\tilde{X}\in\{X_1,\dots,X_4,X\}} d_{X_i,\psi_i}(\tilde{X}(z(t),t)) \Big) < 0, \tag{2.80}$$

is a necessary condition for (2.68b).

Presently, by definition the function $g_o : \mathbb{R}^{n_z}\times\mathbb{R}\to\mathbb{R}$ is Lipschitz continuous and

$$g_o(z,t) < 0 \quad \Longleftrightarrow \quad \begin{array}{l} \forall i,j:\ X_j(z(t),t), X(z(t),t) \notin \mathcal{O}_i(t) \text{ and} \\ \forall i,j:\ X_{i,j}(t), X_i(t) \notin \mathcal{V}(z(t),t) \end{array} \tag{2.81}$$

However, the aim is to characterize the fact that the obstacle and vehicle are disjoints, i.e.,

$$\mathcal{V}(z,t)\cap\Big(\bigcup_{1\le i\le k}\mathcal{O}_i(t)\Big) = \emptyset. \tag{2.82}$$

In general, the condition $g_o(z(t),t) < 0$ (that is, condition (2.81)) is not sufficient to ensure that (2.82) holds, as shown by the counter-example illustrated in Figure 2.13.

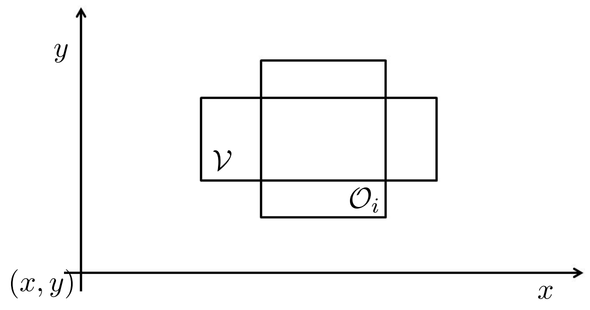


Figure 2.13: Vehicle and obstacles.

This condition can be solved by using Farkas' Lemma, see [50], adding an additional control. This will however increase the complexity of the problem.

To summarize, in this section for a given car traffic scenario defined as a quintuple $(\mathcal{K}_d,\mathcal{K}_r,\mathcal{K}_o,\mathcal{V},\Omega)$, the associated Lipschitz functions g_d, g_r, g_o, φ or the associated smooth functions $\widetilde{g}_d, \widetilde{g}_r, \widetilde{g}_o, \widetilde{\varphi}$ were derived by imposing property (2.64). Let now $g : \mathbb{R}^{n_z}\times\mathbb{R}\to\mathbb{R}$ be defined as

$$g(z(t),t) := \max\{g_d(z(t),t), g_r(z(t),t), g_o(z(t),t)+\eta\}, \tag{2.83}$$

and let $\widetilde{g} : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}^{n_{\widetilde{g}}}$ be the vector function such that

$$\widetilde{g}(z(t),t) := \begin{pmatrix} \widetilde{g}_d(z(t),t) \\ \widetilde{g}_r(z(t),t) \\ \widetilde{g}_o(z(t),t) + \eta \end{pmatrix}, \tag{2.84}$$

with $\eta > 0$ small. Thus equivalent sufficient conditions for (2.64) to hold, are given by

- in this case the level set functions are only Lipschitz,

$$\varphi(z(t_f)) \leq 0, \text{ and } g(z(t),t) \leq 0 \ \forall t \in [t_0, t_f]; \tag{2.85}$$

- in this case the level set functions are smooth,

$$\begin{gathered} \widetilde{\varphi}(z(t_f)) := \begin{pmatrix} \widetilde{\varphi}_1(z(t_f)) \\ \vdots \\ \widetilde{\varphi}_{n_{\widetilde{\varphi}}}(z(t_f)) \end{pmatrix} \leq 0, \text{ and} \\ \widetilde{g}(z(t),t) := \begin{pmatrix} \widetilde{g}_1(z(t),t) \\ \vdots \\ \widetilde{g}_{n_{\widetilde{g}}}(z(t),t) \end{pmatrix} \leq 0 \ \forall t \in [t_0, t_f]; \end{gathered} \tag{2.86}$$

- if for instance Ω and $\mathcal{K}$ are the zero-level sets of functions φ and g respectively, or equivalently if Ω and $\mathcal{K}$ are the intersection sets of the zero-level sets of functions $\widetilde{\varphi}_i$, $i = 1, \ldots, n_{\widetilde{\varphi}}$, and $\widetilde{g}_j$, $i = 1, \ldots, n_{\widetilde{g}}$ respectively, then (2.85) and (2.86) are equivalent to (2.87)

$$z(t_f) \in \Omega, \text{ and } z(t) \in \mathcal{K}(t) \ \forall t \in [t_0, t_f], \tag{2.87}$$

Conditions (2.85)-(2.87) are called *constraints*. In particular if there is time dependency they are called *state constraints*, otherwise they are named *boundary constraints*.

3 Definition of objectives used in collision avoidance systems

The model presented in this chapter addresses to two main objectives which can be thought of as safety and performance goals. The reference vehicle trajectory, modelled in Section 1 as the function $z_{z_0}^u(\cdot)$ solution of System (1.1), must achieve such objectives. Guaranteeing safety is the first priority, satisfying performance criteria the second. Safety has been discussed in Section 2 and a mathematical condition to impose it, is given in one of Conditions (2.85)-(2.87).

In this Section the performance criteria will be defined as the functional

$$\varphi_0(z^u_{z_0}(\tau), \tau, w) + \int_0^\tau f_0(z^u_{z_0}(t), u(t))dt. \tag{3.1}$$

To satisfy such performance criteria means to find a control strategy $u \in \mathcal{U} := \{u : [0, \infty) \to U \subset \mathbb{R}^{n_u} \mid u \text{ measurable}\}$ and parameters $w \in W, \tau \in [t_0, t_f]$ such that the functional (3.1) is minimized. Once the performance criteria have been defined it becomes natural to write the optimal control problem modelling the car traffic scenario where safety and performance requirements are guaranteed.

Problem 3.1. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed time points $t_0 < t_f$ and*

$$\begin{aligned} \varphi_0 &: \mathbb{R}^{n_z} \times [t_0, t_f] \times \mathbb{R}^{n_w} \to \mathbb{R}, \\ f_0 &: \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}, \\ f &: [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}, \end{aligned}$$

be maps. Let $(\mathcal{K}_t)_{t \geq 0}$ and Ω be closed set of $\mathbb{R}^{n_z}$. Find states $z : \mathbb{R} \to \mathbb{R}^{n_z}$ absolutely continuous, controls $u \in \mathcal{U} = \{u \,|\, u : [0, +\infty) \to U \subset \mathbb{R}^{n_u}$ meaurable.$\}$ and parameters $w \in W \subset \mathbb{R}^{n_w}$, $\tau \in [t_0, t_f]$ such that:

$$\begin{aligned} \min_{z, u \in \mathcal{U}, w \in W, \tau \in [t_0, t_f]} \quad & J := \varphi_0(z(\tau), \tau, w) + \int_{t_0}^\tau f_0(z(t), u(t))dt && \text{(3.2a)} \\ s.t. \quad & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, && \text{(3.2b)} \\ & z(t_0) = z_0, && \text{(3.2c)} \\ & z(t) \in \mathcal{K}(t), \forall t \in [t_0, \tau], && \text{(3.2d)} \\ & z(\tau) \in \Omega. && \text{(3.2e)} \end{aligned}$$

The (3.2a) is called *objective function*, Equation (3.2b) is an *Ordinary Differential Equation* (ODE), Equations (3.2c) and (3.2e) are *boundary constraints*, while (3.2d) is a *state constraint.* Conditions (3.2d)-(3.2e) can be written as (2.85) or (2.86), due to the equivalence with (2.87).

By solving Problem 3.1, for a specific characterization of the functional (3.1), the following three main questions will be answered in the next paragraphs.

1. Does it exist a collision-free trajectory for the reference vehicle, minimizing the final time to achieve the target or minimizing the steering effort during the path or starting the maneuver on the last possible point such that it is possible to avoid a rear-end collision?
2. Given an initial point for the vehicle maneuver, which is the set of points that the vehicle can reach, avoiding a collision?
3. Which points of the car traffic scenario are "safe starting points", in the sense that the reference vehicle starting a maneuver from those points can avoid a collision and reach the target set?

Question 1. Compute an optimal trajectory from a given initial position to a secure final state.

Definition 3.2. *The state solution of Problem 3.1 associated to the control solution u, for a given initial state $z_0 \in \mathbb{R}^{n_z}$, is denoted by $z^u_{z_0}$ and it is called* optimal trajectory.

In this case the objective function is defined to meet the following requirements.

- The minimization of the final time τ expects to perform the vehicle maneuver z as fast as possible.
- Driver friendly trajectories with not extreme control values are also requested, this is achieved for instance by minimizing the steering effort term. The controls appear to be more regular than for minimization of the initial distance to an obstacle where the controls are expected to be more extreme since are the shortest possible trajectories.
- In a rear-end collision scenario, the distance vehicle-obstacle at time t_0 is denoted by w and it will determine the vehicle initial position, i.e. $z(t_0) = z_0(w)$. The minimization of w represents the last possible $z_0(w)$ from which the vehicle can start a maneuver that avoids a rear-end collision, in the sense that if $\tilde{w} > w$ then the maneuver starting from $z_0(\tilde{w})$ does not satisfy property (3.2d) and (3.2e).

Therefore,

$$\varphi_0(z^u_{z_0}(\tau), \tau, w) + \int_0^\tau f_0(z^u_{z_0}(t), u(t))dt := c_1\tau + c_2 w + c_3 \int_0^\tau u(t)^2 dt \tag{3.3}$$

with appropriate constants $c_1, c_2, c_3 \geq 0$. The objective functional is a linear combination of the final time τ, the steering effort and the initial distance w to obstacle $\mathcal{O}_i$. Since sometimes it is not clear whether a collision can be avoided at all, a constraint violation minimization technique is employed, for instance conditions (3.2d) and (3.2e) are minimized instead of being hard constraints.

Question 2. Compute the reachable set from a given initial position.

Definition 3.3. *The* reachable set *in the time interval $[t_0, t_f]$ associated to the dynamics f for a given initial state $z_0 \in \mathbb{R}^{n_z}$, for Problem 3.1, is defined as*

$$\begin{aligned}\mathcal{FR}^f_{t_f} := \Big\{ z_f \in \mathbb{R}^{n_z} \mid & \textit{ for a given } z_0 \in \mathbb{R}^{n_z}, \exists u \in \mathcal{U} \textit{ and } \exists \tau \in [t_0, t_f] : \\ & z^u_{z_0} \textit{ feasible trajectory of Problem 3.1 and } z^u_{z_0}(\tau) = z_f \Big\}.\end{aligned} \tag{3.4}$$

The set of all trajectories that reach the points of the reachable set is called trajectory funnel *and it is defined as*

$$\mathcal{TF}^f_{t_f} := \Big\{ z^u_{z_0} : \mathbb{R} \to \mathbb{R}^{n_z} \mid \textit{for a given } z_0 \in \mathbb{R}^{n_z}, \exists u \in \mathcal{U} \textit{ and } \exists \tau \in [t_0, t_f] : \\ z^u_{z_0} \textit{ feasible trajectory of Problem 3.1 and } z^u_{z_0}(\tau) \in \mathcal{FR}^f_{t_f} \Big\}. \tag{3.5}$$

A more general and complete definition of reachable set is given in [7, Section 10.4].

Let $\mathbb{G} = \{g_h\}_{h \in N \subset \mathbb{N}}$ be a grid on the state space $\mathbb{R}^{n_z}$. The reachable set is characterized by distance functions minimizing the distance of the endpoint $z(\tau)$ of a trajectory to g_h plus a regularization term involving the steering effort, for each grid point g_h. Thus

$$\varphi_0(z^u_{z_0}(\tau), \tau, w) + \int_0^\tau f_0(z^u_{z_0}(t), u(t))dt := c_1 \|z(\tau) - g_h\|_2 + c_2 \int_0^\tau u(t)^2 dt \tag{3.6}$$

is the minimized function over u, w, τ for each g_h in $\mathbb{G}$, with appropriate constants $c_1, c_2 \geq 0$.

Question 3. Compute the backward reachable set within time t_f.

Definition 3.4. *The* backward reachable set *in the time interval* $[t_0, t_f]$ *associated to the dynamics f, for Problem 3.1, is defined as*

$$\mathcal{BR}^f_{t_f} := \Big\{ z_0 \in \mathbb{R}^{n_z} \mid \exists u \in \mathcal{U} : z^u_{z_0} \textit{ feasible trajectory of Problem 3.1} \Big\}. \tag{3.7}$$

A definition of backward reachable set is given for instance in [17].

Let $\mathbb{G} = \{g_h\}_{h \in N \subset \mathbb{N}}$ be a grid on the state space $\mathbb{R}^{n_z}$. For each g_h the Problem 3.1 is soved with $z_0 = g_h$ and the functional

$$\varphi_0(z^u_{z_0}(\tau), \tau, w) + \int_0^\tau f_0(z^u_{z_0}(t), u(t))dt \tag{3.8}$$

is minimized.

4 Errors in data detection by sensors

Measurements errors due to sensors enter the problem by assuming that any sensor data is composed of the true value plus some random error value. This is important to investigate false warnings of collision avoidance systems as explained in Section 1.3 of Chapter I. The purpose of this section is to show how this can be modeled to predict changes in the solution of the Optimal Control Problem 3.1 by studying the Perturbed Optimal Control Problem 4.1.

Problem 4.1. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed time points $t_0 < t_f$ and*

$$\begin{aligned} \varphi_0 &: \mathbb{R}^{n_z} \times [t_0, t_f] \times \mathbb{R}^{n_w} \to \mathbb{R}, \\ f_0 &: \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}, \\ f &: [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}, \end{aligned}$$

be maps. Let $(\mathcal{K}_t)_{t\geq 0}$ and Ω be closed set of $\mathbb{R}^{n_z}$. Find states $z : \mathbb{R} \to \mathbb{R}^{n_z}$ absolutely continuous, controls $u \in \mathcal{U} = \{u \,|\, u : [0, +\infty) \to U \subset \mathbb{R}^{n_u}$ meaurable$\}$ and parameters $w \in W \subset \mathbb{R}^{n_w}$, $\tau \in [t_0, t_f]$ such that:

$$\min_{z,u\in\mathcal{U},w\in W,\tau\in[t_0,t_f]} \quad J := \varphi_0(z(\tau), \tau, w) + \int_{t_0}^{\tau} f_0(z(t), u(t))dt \tag{4.1a}$$

$$s.t. \quad \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, \tag{4.1b}$$

$$z(t_0) = z_0(p), \tag{4.1c}$$

$$z(t) \in \mathcal{K}(t), \forall t \in [t_0, \tau], \tag{4.1d}$$

$$z(\tau) \in \Omega, \tag{4.1e}$$

where $p \in P \subset \mathbb{R}^{n_p}$ fixed.

The perturbation parameter $p \in P \subset \mathbb{R}^{n_p}$ models sensor perturbations that enter the mathematical model owing to measurement errors in the initial values. The vector p is used for sensitivity analysis and will denote perturbation parameters that enter the problem, but are not optimized.

Let the following map be considered

$$L^\infty([0, t_f], \mathbb{R}^{n_u}) \times \mathbb{R}^{n_p} \ni (u, p) \mapsto z^u_{z_0(p)}(\cdot) \in W^{1,\infty}([t_0, t_f], \mathbb{R}^{n_z}), \tag{4.2}$$

denoting the control and parameter to state mapping, which maps a given control u and a given parameter p to the corresponding state trajectory $z^u_{z_0(p)}$. Two types of sensitivity are then investigated.

- Let $\hat{z} := z^{\hat{u}}_{z_0}$ the nominal optimal solution of Problem 4.1 with nominal optimal control $\hat{u} := \hat{u}(\hat{p})$ and nominal perturbation $\hat{p}$, such that the initial value is $z_0 := z_0(\hat{p})$. The aim is to investigate the dependence of the solution $\hat{z}$ and of the control $\hat{u}$ on p to find an optimal solution of Problem 4.1. The **Fiacco-Sensitivity** approach is used to approximate such optimal solution of Problem 4.1, by using a Taylor expansion around $\hat{p}$ and a parametric sensitivity analysis (see Section 2.1 of Chapter III) of the nominal solution $\hat{z}$ and of $\hat{u}$ with respect to p (references on parametric sensitivity analysis are in [41, 52]).

- Let $\hat{z} := z_{z_0}^{\hat{u}}$ the nominal optimal solution of Problem 3.1 with nominal optimal control $\hat{u} := \hat{u}(\hat{p})$, nominal perturbation $\hat{p}$ and initial value $z_0 := z_0(\hat{p})$. The aim is to investigate the dependence of the solution $\hat{z}$ with respect to p by fixing the control $\hat{u}$ (a reference for this sensitivity is [105]). This means to find the solution of the following initial value problem for $p \in B(\hat{p}, r)$.

$$\begin{aligned} \dot{z}(t) &= f(z(t), \hat{u}(t)), \text{ a.e. in } [t_0, t_f], \\ z(t_0) &= z_0(p) =: \xi. \end{aligned} \tag{4.3}$$

  The goal of this model is to understand how an optimal solution for Problem 3.1 changes by perturbing the initial value but keeping the same controls. This will model situations where unknown errors from the sensors are given to the driver or to the collision avoidance system, that will calculate an optimal control taking such values as correct ones. For this second case the approach called **ODE-Sensitivity** is applied and it investigates the dependence of the solution of the initial value problem (4.3) with respect to the initial state. Thus the perturbed solution will not be optimal since $\hat{u}$ is fixed.

The sensitivities here introduced will be computed in Section 3 of Chapter IV for nominal optimal trajectories and for nominal reachable sets. Moreover an estimation of the maximum possible perturbation p is given, such that the perturbed trajectory, solution of (4.3) is still safe (i.e. it satisfies (3.2d) and (3.2e)) even if it is not optimal. This finds its motivation in the search of the lowest sensor measurement precision, such that robustness for the collision avoidance system is guarantee.

Chapter III

Theoretical Background

Contents

In this chapter some results in optimal control theory are recalled from existing literature. A brief overview in reachability and optimal control theory is given in Section 1, based on the introductions of [4, 5, 97, 98]. Section 2 is presenting some of the numerical methods discussed in literature for solving reachability and optimal control problems. It is based on the work presented in [14, 17, 18, 49, 92]. Particular attention is oriented to numerical methods implemented in the software packages OCPID-DAE (such as direct methods, see [48] and Subsection 2.1 of this thesis) and ROC-HJ (such as methods for solving the Hamilton Jacobi Belmann PDE, see [16] and Subsection 2.2 of this thesis), used to compute the results in Chapters IV and V.

1 Optimal Control Theory

Control theory analyses the properties of controlled systems, i. e. dynamical systems on which one can operate through a control. The aim is to bring the system from a given initial state to a certain final state, respecting specific criteria. From a mathematical point of view a control system is a dynamical system depending on a control input, which is often subject to constraints. Examples of dynamical systems are differential systems, discrete systems, systems with noise, systems with delay,... To model such system, mathematical tools from several domains are used, such as differential equations, partial differential equations, integral equations, functional equations, stochastic equations,... An example of a control system, which will be addressed in the next sections, is the parametrized dynamical system in Problem 1.1.

Problem 1.1. *Let the following dynamical system be given:*

$$\begin{aligned} &\dot{z}(t) = f(t, z(t), u(t)), \quad \textit{for a.e. } t \in [t_0, t_f], \\ &z(t_0) = z_0, \\ &z(t) \in \mathcal{K}, \quad \forall t \in [t_0, t_f), \end{aligned} \tag{1.1}$$

where the elements that define the control system are:

$t_f, t_0 \in \mathbb{R}$,	*are given initial time and the final time, respectively*;
$z : \mathbb{R} \to \mathbb{R}^{n_z}, n_z > 0$,	*is an absolutely continuous function called the state*;
$z_0 \in \mathbb{R}^{n_z}$,	*is the initial state*;
$u : \mathbb{R} \to U \subset \mathbb{R}^{n_u}, n_u > 0$,	*is a measurable function called the control*;
$f : \mathbb{R} \times \mathbb{R}^{n_z} \times U \to \mathbb{R}^{n_z}$,	*is the dynamics map*;
$\mathcal{K} \subset \mathbb{R}^{n_z}$,	*a closed set called state-constraints set.*

Under mild hypothesis (see [7]), given a measurable function u, the control system 1.1 admits a unique solution which is an absolutely continuous function denoted as $z^u_{z_0}$ called state of the control system.

Reachability analysis studies the ability to find a control law to bring a given control system from an initial state to a final state. The main questions in reachability are to determine for a given control system the set of all final states that can be reached by the system starting from a given initial state (forward reachable set) and vice versa the set of all initial states from which the system can start and reach a given final state (backward reachable set). Often in industrial applications the concept of robust reachable sets and perturbed reachable sets arise, see [5] and Chapter IV of this thesis. It is important to mention that methods for the approximation of reachable sets often use set-valued calculus and the formulation of control problems as corresponding differential inclusions, see [5, 7]. Therefore, systems of the form (1.1) can be written as the initial value problems (or Cauchy problems) associated with the differential inclusion

$$\begin{aligned} &z'(t) \in F(t, z(t)), \quad \text{for almost all } t \in [t_0, t_f], \\ &z(t_0) = z_0, \\ &z(t) \in \mathcal{K}, \quad \forall t \in [t_0, t_f), \end{aligned} \tag{1.2}$$

where $F : \mathbb{R} \times \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is a set-valued map. Equivalence of the solutions of Systems (1.1) and (1.2) was proven by Filippov in 1959 under mild assumptions (see [7, Chapter 10]). In Section 2.2 an algorithm based on the solution of the Hamilton Jacobi Belmann equation to compute these sets is shown. Algorithms for computing reachable sets are discussed in [6, 8, 87] and [19, 20, 43, 70, 83].

Optimal control deals with the problem of finding a control function for a given control system to go from an initial state to a final state minimizing certain criteria. Its roots are in calculus of variations, studying problems of the type described in Problem 1.2.

Problem 1.2. *Find a function z such that:*

$$\begin{array}{l} \min_z \int_{t_0}^{t_f} L(t, z(t), \dot{z}(t))dt, \\ z(t_0) = z_0, \quad z(t_f) = z_f, \end{array}$$

given a time interval $[t_0, t_f]$, a function $L : [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}$, and z_0, z_f points in $\mathbb{R}^{n_z}$.

An example of Problem 1.2 that initially attracted attention was to determine the curve designing the path that brings a point-like body from one place to another, with zero initial speed, in the least amount of time. It is known as the Brachistochrone problem, studied by Galileo in 1638. Consequently Jean Bernoulli in 1696 found the solution, an arc of a cycloid starting from the vertical tangent, see [98]. Variational principles of classical mechanics, such as Fermat's principle, Dirichlet's principle, Euler Lagrange equations, were widely used techniques for solving problems in calculus of variations.

Optimal control is an extension of the calculus of variations which imposes a new kind of constraints, such as dynamical constraints and pathwise constraints, leading to the formulation of Problem 1.3.

Problem 1.3. *Find a measurable function $u : [t_0, t_f] \to \mathbb{R}^{n_u}$ and a corresponding absolutely continuous function $z : [t_0, t_f] \to \mathbb{R}^{n_z}$ such that:*

$$\begin{array}{l} \min_{z,u} \varphi_0(z(t_0), z(t_f)), \\ \dot{z} = f(t, z(t), u(t)) \ \textit{a.e. in} \ [t_0, t_f], \\ u(t) \in U \subset \mathbb{R}^{n_u}, \\ \varphi(z(t_0), z(t_f)) = 0, \end{array}$$

given a time interval $[t_0, t_f]$, functions $f : [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}$, $\varphi_0 : \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}$, $\varphi : \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_\varphi}$ and z_0, z_f points in $\mathbb{R}^{n_z}$.

The study of optimal control problems attained more and more attention from the late 1950s when two important advances where made. The first, the Pontryagin maximum principle, is a generalization of the Euler Lagrange equation to deal with control constraints and differential equations. It is a set of necessary conditions for a control function to be optimal [80]. The second, the dynamic programming principle, is a procedure that links the search for an optimal control function to the search for the solution of a partial differential equation called Hamilton-Jacobi equation [13].

Key advances in optimal control arose in the 1970s when two important developments (explained below) took place for solving problems with state-constraints, as given in Problem 1.4, which were not covered by the previous available version of the maximum principle.

Problem 1.4. *Find a measurable function $u : [t_0, t_f] \to \mathbb{R}^{n_u}$ and an absolutely continuous function $z : [t_0, t_f] \to \mathbb{R}^{n_z}$ such that:*

$$\begin{aligned}
&\min_{z,u} \varphi_0(z(t_0), z(t_f)),\\
&\dot{z} = f(t, z(t), u(t)) \text{ a.e. in } [t_0, t_f],\\
&u(t) \in U \subset \mathbb{R}^{n_u},\\
&\varphi(z(t_0), z(t_f)) = 0,\\
&z(t) \in \mathcal{K} \subset \mathbb{R}^{n_z}, \quad \forall t \in [t_0, t_f),
\end{aligned}$$

given a time interval $[t_0, t_f]$, functions $f : [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}$, $\varphi_0 : \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}$, $\varphi : \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_\varphi}$ and z_0, z_f points in $\mathbb{R}^{n_z}$, $\mathcal{K}$ a compact set.

The field of nonsmooth analysis and in particular the theory of generalized gradients developed by F.H. Clarke (see [28]), provided necessary conditions of optimality for such nonsmooth variational problems. In this way it was possible to solve problems as 1.4 involving state constraints as generalized problems in calculus of variations with discontinuous derivative of the cost function. Related references on the derivation of necessary conditions of optimality are [84, 85, 98]. An alternative approach is based on geometric control [93] by interpreting the necessary conditions for optimality as geometric conditions about reachable sets. Higher order necessary conditions are discussed in [107].

The second key advance is the concept of viscosity solution due to M.G. Crandall and P.-L. Lions. It provides a framework for existence and uniqueness of generalized solutions to Hamilton-Jacobi equations arising in optimal control. In particular it built a relation between the value function (map that associates any initial time t_0 and initial state z_0 with the optimal value of Problems 1.3 and 1.4) and the solution of the Hamilton-Jacobi equation. For the unconstrained case (Problem 1.3), several works have been devoted to the characterization of the value function as a continuous viscosity solution of a Hamilton-Jacobi equation [11, 38]. In the presence of state constraints, the continuity of this value function is no longer satisfied unless the dynamics satisfy assumptions on the boundary of the state constraints, called controllability assumptions (inward pointing qualification condition (IQ) by Soner in [91] and outward pointing qualification condition (OQ) by Frankowska in [42]). For the characterization of the value function without controllability assumptions see for instance [17] and the references therein and for a characterization based on viability theory, see [26, 27].

Numerical methods for solving optimal control problems are discussed in Section 2 together with the analysis of the dependence of minimizers on parameters (sensitivity analysis). The main motivation for investigating sensitivity analysis comes from engineering applications. The optimal performance is degraded by parameter perturbations and the implementation of nominal controls in real-world processes does not give the nominal expected trajectory. Therefore, the dependency of the solution of an optimal control problem on parameter values in the initial state components it must be investigated. By adopting the dynamic programming approach, such dependence is supplied by the value function, solution of the Hamilton-Jacobi equation (whenever existence hypothesis are satisfied). Moreover computational schemes are available for computing the optimal state and control, satisfying the maximum principle. In this case, it is possible to provide gradients or sensitivity information about this dependence near the nominal initial data as done in [23, 24, 41, 49] and recalled in Section 2.1. In this way an alternative method to the computation of a new optimal solution is given, providing a real-time approximation of the perturbed optimal solution via Taylor expansion.

2 Numerical methods

Numerical solutions of optimal control problems can be categorized into three main approaches: dynamic programming approach, indirect method and direct method.

The dynamic programming approach (DPP) is based on the dynamic programming principle. The approach solves the Hamilton Jacobi Equation associated with the given optimal control problem by computing the level sets of the value function [11, 17, 18, 73]. Such methods require to solve a partial differential equation in potentially high state dimensions, and the design of related computational algorithms is still under development, see for instance [16, 71]. However first results [32] on application to industrial problems are promising.

The indirect method solves numerically a multi-point boundary value problem for the state and adjoint variables by using the maximum principle [3, 53, 56, 61, 79]. This approach leads to highly accurate numerical solutions but it often requires good knowledges on necessary and sufficient conditions to set up the optimality system. Moreover a good initial guess for the approximate solution is needed in order to guarantee convergence [33, 75, 92].

The direct method is based on discretization of state and/or control variables over time, so that the resulting finite dimensional optimization problem is solved numerically by suitable methods for nonlinear programming. This approach has the advantage that it is easy to manage for users that do not have a deep insight in optimal control theory and it is capable to handle large scale problems. Direct methods have been presented in detail by many authors, in particular see [14, 49].

2.1 Direct methods

This subsection is based on the work presented in [14, 49, 92].

The transcription method

Direct methods, also known as transcription methods, received huge attention in the numerical optimal control community in the last twenty years, with many applications to complex optimal control problems in industry. The idea is based on discretization of the infinite-dimensional optimal control problem and on optimization of the resulting finite-dimensional optimization problem with state of art software packages for Nonlinear Programming (NLP). Also known as “first discretize and then optimize”, transcription methods are identified in two phases:

1. transcribe the infinite-dimensional problem into a finite-dimensional problem with finite set of NLP variables;
2. optimize the finite-dimensional problem using an NLP solver.

The focus of this Chapter is to summarize two main techniques to discretize the optimal control problem 2.1: the collocation method and the direct shooting method.

Gradient methods for continuous optimal control problems are also classified as direct methods. The algorithm works in the same spaces in which the optimal control problem is defined and only later it introduces a suitable discretization to solve numerically the resulting initial value problem. However only simple state constraints, like box constraints, can be included [49, Chapter 8].

Among the optimal control problems of the form 1.4, the following type is considered here.

Problem 2.1 (OCP)**.** *Given $I := [t_0, t_f] \subset \mathbb{R}$ a non-empty compact time interval with $t_0 < t_f$ fixed,*

$$\begin{aligned} \varphi_0 &: \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}, \\ f &: I \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}, \\ g &: I \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_g}, \\ \varphi &: \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_\varphi}, \end{aligned}$$

sufficiently smooth functions, the following problem is defined:

$$\begin{array}{lll} \min & \varphi_0(z(t_0), z(t_f)) & \\ w.\ r.\ t. & z \in \mathcal{Z}, u \in \mathcal{U} & \\ s.\ t. & \dot{z}(t) = f(t, z(t), u(t)), & a.e.\ in\ I \\ & g(t, z(t), u(t)) \leq 0, & in\ I \\ & \varphi(z(t_0), z(t_f)) = 0, & \end{array} \tag{2.1}$$

where $\mathcal{Z} = W^{n_z}_{1,\infty}(I)$ is the state space and $\mathcal{U} = L^{n_u}_{\infty}(I)$ is the control space. The notation $W^{n_z}_{1,\infty}(I)$ denotes the space of absolutely continuous vector-valued functions *from I in $\mathbb{R}^{n_z}$ with essentially bounded first derivative. The notation $L_\infty^{n_u}(I)$ denotes the space of essentially bounded vector-valued functions from I to $\mathbb{R}^{n_z}$.*

Bolza problems with cost function of the type

$$\varphi_0(z(t_0), z(t_f)) + \int_{t_0}^{t_f} f_0(t, z(t), u(t))dt \tag{2.2}$$

with $f_0 : I \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}$ sufficiently smooth, can be written in Mayer form defining the additional state z_{new}:

$$\dot{z}_{new}(t) = f_0(t, z(t), u(t)), z_{new}(t_0) = 0. \tag{2.3}$$

Moreover problems with free final time can be also transformed to Problem 2.1 by applying the time transformation

$$t(s) = t_0 + s(t_f - t_0), s \in [0, 1]. \tag{2.4}$$

Time discretization: The time discretization is defined by a grid on the time interval $[t_0, t_f] \subset \mathbb{R}$:

$$\begin{gathered} \mathbb{G} = \{t_0 < t_1 < \cdots < t_N = t_f\}, \\ \text{with step size } h_j = t_{j+1} - t_j, j = 0, \ldots, N-1 \\ \text{and mesh-size } h := \max_{j=0,\ldots,N-1} h_j. \end{gathered} \tag{2.5}$$

If $\mathbb{G}$ is an equidistant grid then the step size is constant $h = (t_f - t_0)/N$ and the grid points are $t_j = t_0 + jh, j = 0, \ldots, N$. To estimate the accuracy of a solution means to investigate how well the approximate solution matches with the solution of Problem 2.1. Algorithms for automatic and adapted step-size selection are required to improve efficiency and accuracy of discretization algorithms [14]. Strategies for step-size selection are based on numerical estimates of the local discretization error in order to keep this error below a given tolerance.

Control discretization: Let us take a time grid $\mathbb{G}$ defined as in (2.5), following the notation of [49] the discretized control is:

$$u_{\text{app}}(\cdot) = \sum_{i=0}^{M} w_i B_i(\cdot) \tag{2.6}$$

where u_{app} belongs to the $(M+1)$-dimensional subspace of the essentially bounded functions from $[t_0, t_f]$ to $\mathbb{R}^{n_u}$ with basis $\mathbb{B} := \{B_0, \ldots, B_M\}$ and basis functions $B_0, \ldots, B_M$. The components of the vector $w := (w_0, \ldots, w_M)$ are the new control parameters for the discretized problem. Some examples of control discretization are here described:

1. Let $w_i = u(t_i), i = 0, \dots, N-1$ be the control variables at each node of the time grid $\mathbb{G}$, the controls are chosen as piecewise constant functions:

$$u_{\text{app}}(t) = w_i, \quad t_i \le t < t_{i+1} \tag{2.7}$$

2. Let $w_i = u(t_i), i = 0, \dots, N-1$ be the control variables at each node of the time grid $\mathbb{G}$, the controls are chosen as piecewise linear interpolating functions between w_i and w_{i+1}:

$$u_{\text{app}}(t) = w_i + \frac{t - t_i}{h_i}(w_{i+1} - w_i), \quad t_i \le t < t_{i+1} \tag{2.8}$$

3. As in [49, Section 5.1] the control can be parametrized by functions with local support called B-spline functions:

$$u_{\text{app}}(\cdot) := \sum_{i=1}^{N+k-1} w_i B_i^k(\cdot) \tag{2.9}$$

where for $i = 1, \dots, N+k-1$, $w_i \in \mathbb{R}^{n_u}$ and $B_i^k(\cdot)$ B-splines of order $k \in \mathbb{N}$ defined by recursion:

$$\begin{aligned} B_i^1(t) &:= \begin{cases} 1 & \text{if } \tau_i \le t < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases} \\ B_i^k(t) &:= \tfrac{t-\tau_i}{\tau_{i+k-1}-\tau_i} B_i^{k-1}(t) + \tfrac{\tau_{i+k}-t}{\tau_{i+k}-\tau_{i+1}} B_{i+1}^{k-1}(t) \end{aligned} \tag{2.10}$$

on the auxiliary grid $\mathbb{G}_{aux} := \{\tau_i | i = 1, \dots, N+2k-1\}$, with

$$\tau_i := \begin{cases} t_0 & \text{if } 1 \le i \le k \\ t_{i-k} & \text{if } k+1 \le i \le N+k-1 \\ t_N & \text{if } N+k \le i \le N+2k-1 \end{cases} \tag{2.11}$$

4. In [86] the control is parametrized on the time grid $\mathbb{G}$ by N-degree Lagrange interpolating polynomials:

$$u_{\text{app}}(\cdot) := \sum_{i=0}^{N} w_i L_i(\cdot) \tag{2.12}$$

where for $i = 1, \dots, N$, $w_i := u(t_i)$ at the Legendre-Gauss-Lobatto points t_i and $L_i(\cdot)$ are Lagrange interpolating polynomials of order $N \in \mathbb{N}$. Such control discretization is used for the pseudospectral methods, a class of transcription methods in which both, control and state, are approximated by N-degree Lagrange interpolating polynomials. The advantage of pseudospecral method is that it can have a rate of convergence faster than polynomials (exponential or spectral) and a good accuracy, however oscillations for non-differentiable trajectories are a disadvantage.

Integration schemes

Discretization methods for ODE are an important tool for finding numerical solutions of optimal control problem. They are categorized as explicit or implicit, one-step methods or multi-step methods. Let us consider the initial value problem:

$$\dot{z}(t) = f(t, z(t)), \quad z(t_0) = z_0 \tag{2.13}$$

and let $\mathbb{G}_z$ denote the time grid defined in (2.5).

Definition 2.2. *Let $F_n, n \in \{1,2\}$, continuous functions with values in $\mathbb{R}^{n_z}$. An* explicit method *for finding the approximate solution of* (2.13) *is defined as:*

$$\begin{aligned} &z(t_0) = z_0, \\ &z(t_{i+m+1}) = F_1(t_i, \dots, t_{i+m}, z(t_i), \dots, z(t_{i+m}), h_i, \dots, h_{i+m}), \\ &\qquad\qquad\qquad\qquad\qquad\qquad i+m \in \{0, \dots, N-1\}. \end{aligned} \tag{2.14}$$

An implicit method *for finding the approximate solution of* (2.13) *is defined as:*

$$\begin{aligned} &z(t_0) = z_0, \\ &F_2(t_i, \dots, t_{i+m+1}, z(t_i), \dots, z(t_{i+m+1}), h_i, \dots, h_{i+m+1}) = 0, \\ &\qquad\qquad\qquad\qquad\qquad\qquad i+m \in \{0, \dots, N-1\}. \end{aligned} \tag{2.15}$$

If $m > 0$ the method is called multi-step method*, if $m = 0$ it is called* one-step method.

At each integration step, for implicit methods, a nonlinear system of equations is solved using Newtown method or fixed point iteration. Therefore implicit methods require a higher computational effort than explicit methods, but they lead to better stability properties. Some references discussing consistency, stability and convergence of integration schemes are [25, 49].

A prominent class of one-step explicit methods is the k-stage Runge-Kutta scheme.

Definition 2.3 (see [49, Section 4.1])**.** *The k-stage Runge-Kutta method is defined as:*

$$z(t_{i+1}) = z(t_i) + h_i \sum_{j=1}^{k} b_j f_j(t_i, z(t_i), h_i), \tag{2.16}$$

where

$$f_j(t_i, z(t_i), h_i) = f(t_i + c_j h_i, z(t_i) + h_i \sum_{l=1, l<j} a_{jl} f_l(t_i, z(t_i), h_i)) \tag{2.17}$$

for $1 \leq j \leq k$ and $k \in \mathbb{N}$. The known constants a_{jl}, b_j, c_j are defined using the Butcher array:

$$\begin{array}{c|cccc} c_1 & & & & \\ c_2 & a_{21} & & & \\ \vdots & \vdots & & & \\ c_k & a_{k1} & \dots & a_{kk-1} & \\ \hline & b_1 & \dots & b_{k-1} & b_k \end{array} \tag{2.18}$$

If $a_{jl} = 0$ for all $l \geq j$ the scheme is called explicit, otherwise implicit.

Explicit Euler method is a 1-stage Runge-Kutta method defined by the Butcher array:

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array} \tag{2.19}$$

and

$$z(t_{i+1}) = z(t_i) + h_i f(t_i, z(t_i)). \tag{2.20}$$

Direct collocation

The direct collocation method aims to transform the optimal control problem into a nonlinear optimization problem via full discretization (both control and state are discretized in time) [14, 54].

State discretization: Once the control is discretized as in (2.6), the solution of the Ordinary Differential Equation (ODE) over the interval $[t_i, t_i + h_i]$ for $i = 0, \dots, N-1$, with t_i points of the state grid $\mathbb{G}$ defined as in (2.5), is approximated by a polynomial $p : [t_i, t_i + h_i] \to \mathbb{R}^{n_z}$ of degree k which satisfies the collocation conditions:

- collocation points are defined as $t_i \leq \tau_{i1} < \cdots < \tau_{ik} \leq t_{i+1} = t_i + h_i$; z_{app} denotes the function obtained by discretization of state z and $z_i = z_{app}(t_i)$ and $u_i = u_{app}(t_i; w)$;
- collocation conditions are given by imposing the initial condition and the differential equations of Problem 2.1 at all collocation points:

$$\begin{aligned} p(t_i) &= z_i, \\ \dot{p}(\tau_{ij}) &= f(\tau_{ij}, p(\tau_{ij}), u_{app}(\tau_{ij}; w)), \quad j = 1, \dots, k; \end{aligned} \tag{2.21}$$

- define $z_{i+1} = p(t_{i+1})$ and denote $z_{ij} = p(\tau_{ij})$.

There are different schemes for choosing the collocation points, the most common are Gauss, Radau and Lobatto methods. Gauss, Lobatto and Radau are implicit Runge-Kutta schemes.

Discretized OCP: Discretization of state and control constraints yields the following discretized optimal control problem.

Problem 2.4. *The discretized optimal control problem in Mayer form via collocation method is:*

$$\begin{array}{lll} \min & J(z) = \varphi_0(z_0, z_N) & \\ w.\ r.\ t. & z = (z_0, z_{01}, \dots, z_{0k}, z_1, \dots, & \\ & \quad z_{N-1}, z_{N-1\ 0}, \dots, z_{N-1\ k}, z_N, & \\ & \quad w_0, \dots, w_M) & (2.22) \\ & \quad \in \mathbb{R}^{(n_z(N+1)+n_zNk+n_u(M+1))} & \\ s.\ t. & H(z) = 0_{\mathbb{R}^{Nn_z+n_\varphi}} & \\ & G(z) \leq 0_{\mathbb{R}^{(N+1)n_g}} & \end{array}$$

where

$$H(z) = \begin{pmatrix} z_1 - p(t_1) \\ \vdots \\ z_N - p(t_N) \\ \varphi(z_0, z_N) \end{pmatrix} = 0_{\mathbb{R}^{Nn_z+n_\varphi}}, \quad G(z) = \begin{pmatrix} g(t_0, z_0, u_0) \\ \vdots \\ g(t_N, z_N, u_N) \end{pmatrix} \leq 0_{\mathbb{R}^{(N+1)n_g}}.$$
(2.23)

Problem 2.4 is a large and sparse NLP problem with optimization variable z, which can be solved using sequential quadratic programming or interior-point methods, [103].

Gradients: NLP methods require gradients of the functions involved.

Collocation methods are specific Runge-Kutta methods, thus they can be written as in Definition 2.3

$$z_{i+1} = z_i + h_i F(t_i, z_i, w, h_i), \tag{2.24}$$

where $F : \mathbb{R} \times \mathbb{R}^{n_z} \times \mathbb{R}^{(M+1)n_u} \times \mathbb{R} \to \mathbb{R}^{n_z}$ is continuous. Problem 2.4 is large and sparse, thus derivatives of the functions J, H, G with respect to the optimization variable $z = (z_0, \dots, z_{N-1}, w_0, \dots, w_M)$ are easy to compute.

$$\frac{\partial J(z)}{\partial z} = \left(\frac{\partial \varphi_0(z_0,z_N)}{\partial z_0}, 0_{\mathbb{R}^{n_z}}, \dots, 0_{\mathbb{R}^{n_z}}, \frac{\partial \varphi_0(z_0,z_N)}{\partial z_N}, 0_{\mathbb{R}^{M+1}} \right)$$

$$\frac{\partial H(z)}{\partial z} = \left(\begin{array}{ccccc|c} M_0 & I_{n_z} & & & & h_0 \frac{\partial F(t_0,z_0,w,h_0)}{\partial w} \\ & & \ddots & & & \vdots \\ & & & M_N & I_{n_z} & h_N \frac{\partial F(t_N,z_N,w,h_N)}{\partial w} \\ \frac{\partial \varphi(z_0,z_N)}{\partial z_0} & 0_{\mathbb{R}^{n_z} \times \mathbb{R}^{n_\varphi}} & \dots & 0_{\mathbb{R}^{n_z} \times \mathbb{R}^{n_\varphi}} & \frac{\partial \varphi(z_0,z_N)}{\partial z_N} & 0_{\mathbb{R}^{M+1} \times \mathbb{R}^{n_\varphi}} \end{array} \right) \tag{2.25}$$

where $M_i := I_{n_z} + h_i \partial F(t_i, z_i, w, h_i)/\partial z$, $i = 0, \dots, N$.

$$\frac{\partial G(z)}{\partial z} = \left(\begin{array}{ccc|ccc} \frac{\partial g(t_0,z_0,u_0)}{\partial z_0} & & & \frac{\partial g(t_0,z_0,u_0)}{\partial u_0}\frac{\partial u_0}{\partial w_0} & \cdots & \frac{\partial g(t_0,z_0,u_0)}{\partial u_0}\frac{\partial u_0}{\partial w_M} \\ & \ddots & & \vdots & & \vdots \\ & & \frac{\partial g(t_N,z_N,u_N)}{\partial z_N} & \frac{\partial g(t_N,z_N,u_N)}{\partial u_N}\frac{\partial u_N}{\partial w_0} & \cdots & \frac{\partial g(t_N,z_N,u_N)}{\partial u_N}\frac{\partial u_N}{\partial w_M} \end{array}\right) \tag{2.26}$$

Control parametrization through functions with basis functions with local support, for instance B-spline in (2.9), will lead to sparsity of the derivative $\frac{\partial u(\cdot,w)}{\partial w}$. This will further increase the simplification of the structure of the derivatives.

Direct single shooting

Shooting methods are classically used for solving Boundary Value Problems (BVP) in indirect methods. However applications of such methods to direct methods has been studied and implemented in [15, 21, 22, 37, 45, 49, 82, 94]. They are classified as reduced discretization methods, so that the control is discretized and then the state is obtained recursively by numerical integration.

State discretization: Given a time grid $\mathbb{G}$ defined by (2.5), z_{app} denotes the discretization of state z and $z_i = z_{app}(t_i)$ $i = 0, \dots, N$, and $u_i = u_{app}(t_i; w)$. Single Shooting method can be summarized as follow:

- guess an initial value $z_0 = z(t_0)$ and a control parametrization;
- integrate the state equation from t_0 to t_f using methods in Section 2.1, for instance for one-step methods:

$$\begin{aligned} &z_0 = z(t_0) \\ &z_{i+1} = z_i + h_i F(t_i, z_i, w, h_i), \quad i = 0, \dots, N-1 \end{aligned} \tag{2.27}$$

  where $F : \mathbb{R} \times \mathbb{R}^{n_z} \times \mathbb{R}^{(M+1)n_u} \times \mathbb{R} \to \mathbb{R}^{n_z}$ continuous, and for $i = 1, \dots, N$, z_i are not optimization variables but are functions of z_0 and w, $z_i(z_0, w)$.

For a given initial condition the state trajectory is uniquely defined by the control parametrization w. The main advantage of this method with respect to direct collocation is the small amount of NLP variables. However it is very sensitive to a small change in the initial value which may have a large influence on the state, specially with highly non-linear differential equations.

Discretized OCP:

Problem 2.5. *The discretized optimal control problem in Mayer form via the single shooting method is:*

$$\begin{array}{ll} \min & J(z) = \varphi_0(z_0, z_N) \\ w.\ r.\ t. & z = (z_0, w_0, \dots, w_M) \in \mathbb{R}^{n_z + n_u(M+1)} \\ s.\ t. & H(z) = 0_{\mathbb{R}^{n_\varphi}} \\ & G(z) \leq 0_{\mathbb{R}^{(N+1)n_g}} \end{array} \tag{2.28}$$

where

$$H(z) = \ \varphi(z_0, z_N) \ = 0_{\mathbb{R}^{n_\varphi}}, \quad G(z) = \begin{pmatrix} g(t_0, z_0, u_0) \\ \vdots \\ g(t_N, z_N, u_N) \end{pmatrix} \leq 0_{\mathbb{R}^{(N+1)n_g}} \tag{2.29}$$

Gradients: Reduced discretization methods lead to more complex derivatives than full discretization methods, since they access the sensitivity information of the numerical solution with respect to parameters.

$$\frac{\partial J(z)}{\partial z} = \left(\frac{\partial \varphi_0(z_0,z_N)}{\partial z_0} + \frac{\partial \varphi_0(z_0,z_N)}{\partial z_N}\frac{\partial z_N}{\partial z_0}, \frac{\partial \varphi_0(z_0,z_N)}{\partial z_N}\frac{\partial z_N}{\partial w} \right)$$

$$\frac{\partial H(z)}{\partial z} = \left(\frac{\partial \varphi(z_0,z_N)}{\partial z_0} + \frac{\partial \varphi(z_0,z_N)}{\partial z_N}\frac{\partial z_N}{\partial z_0}, \frac{\partial \varphi(z_0,z_N)}{\partial z_N}\frac{\partial z_N}{\partial w} \right)$$

$$\frac{\partial G(z)}{\partial z} = \left(\begin{array}{c|c} \frac{\partial g(t_0,z_0,u_0)}{\partial z_0} & \frac{\partial g(t_0,z_0,u_0)}{\partial u_0}\frac{\partial u_0}{\partial w} \\ \frac{\partial g(t_1,z_1,u_1)}{\partial z_1}\frac{\partial z_1}{\partial z_0} & \frac{\partial g(t_1,z_1,u_1)}{\partial z_1}\frac{\partial z_1}{\partial w} + \frac{\partial g(t_1,z_1,u_1)}{\partial u_1}\frac{\partial u_1}{\partial w} \\ \vdots & \vdots \\ \frac{\partial g(t_N,z_N,u_N)}{\partial z_N}\frac{\partial z_N}{\partial z_0} & \frac{\partial g(t_N,z_N,u_N)}{\partial z_N}\frac{\partial z_N}{\partial w} + \frac{\partial g(t_N,z_N,u_N)}{\partial u_N}\frac{\partial u_N}{\partial w} \end{array} \right) \tag{2.30}$$

The sensitivities $\frac{\partial z_i(z_o,w)}{\partial z_0}$ and $\frac{\partial z_i(z_o,w)}{\partial w}$ for $i = 1, \dots, N-1$ can be computed by finite differences. However to improve accuracy and efficiency, approaches as sensitivity differential equation, adjoint equation and algorithmic differentiation are used. To exploit such methods, see [49] and the references therein.

Direct multiple shooting

In order to reduce instability problems of the single shooting method, one approach is to divide the problem into shorter steps. For each time subinterval a single shooting subproblem is computed. In this way the size of the problem for the NLP solver (more variables than single shooting but still less variables than collocation) is increased, but the structure of the gradient is with a similar pattern as collocation methods.

State discretization: Let us consider the time grid for state discretization $\mathbb{G}_z$ as in (2.5) which does not coincide with the grid $\mathbb{G}_u$ used for the discretization of the control (2.6):

- shooting nodes are defined as $t_0 \leq t_1 < \cdots < t_N \leq t_f$, which is not the same discretization of the control grid;
- for each segment in $[t_i, t_{i+1}], i = 0, \ldots, N-1$ guess the initial state $s_i = z(t_i)$;
- integrate the state equation in each interval from t_i to t_{i+1} and denote with $z_i(t, s_i, w)$ the solution of the initial value problem

$$\begin{aligned} z(t_i) &= s_i, \\ \dot{z}(t) &= f(t, z(t), u_{app}(t; w)), \quad t \in [t_i, t_{i+1}]; \end{aligned} \tag{2.31}$$

- impose the continuity conditions at each shooting node:

$$z_i(t_{i+1}, s_i) - s_{i+1} = 0, \quad i = 0, \ldots, N-1. \tag{2.32}$$

Multiple shooting method is considered a hybrid method in between collocation and single shooting methods because it divides the problem into shorter single shooting subproblems, imposing continuity at the shooting nodes. The uncoupling structure between the multiple shooting segments allows a parallelized implementation improving efficiency.

Discretized OCP:

Problem 2.6. *The discretized optimal control problem in Mayer form via multiple shooting method is:*

$$\begin{array}{ll} \min & J(z) = \varphi_0(z_0, z_N) \\ w.\ r.\ t. & z = (s_0, \ldots, s_N, w_0, \ldots, w_M) \in \mathbb{R}^{(n_z(N+1)+n_u(M+1))} \\ s.\ t. & H(z) = 0_{\mathbb{R}^{Nn_z+n_\varphi}} \\ & G(z) \leq 0_{\mathbb{R}^{(N+1)n_g}} \end{array} \tag{2.33}$$

where

$$H(z) = \begin{pmatrix} z_0(t_1, s_0, w) - s_1 \\ \vdots \\ z_{N-1}(t_N, s_{N-1}, w) - s_N \\ \varphi(s_0, s_N) \end{pmatrix} = 0_{\mathbb{R}^{Nn_z + n_\varphi}},$$

$$G(z) = \begin{pmatrix} g(t_0, s_0, u_{app}(t_0; w)) \\ \vdots \\ g(t_N, s_N, u_{app}(t_N; w)) \end{pmatrix} \leq 0_{\mathbb{R}^{(N+1)n_g}}. \tag{2.34}$$

Parametric sensitivity analysis

The discretized optimal control problem obtained in the previous paragraph is here written in parametrized form:

Problem 2.7. *Let $p \in \mathbb{R}^{n_p}$ be a given parameter and $J, H_i, G_j : \mathbb{R}^{n_z} \times \mathbb{R}^{n_p} \to \mathbb{R}$, $i = 1, \ldots, n_H$, $j = 1, \ldots, n_G$, sufficiently smooth functions.*

$$\min_{z \in \mathbb{R}^{n_z}} \quad J(z, p) \tag{2.35a}$$

$$s.t. \quad H_i(z, p) = 0, \quad i = 1, \ldots, n_H, \tag{2.35b}$$

$$G_j(z, p) \leq 0, \quad j = 1, \ldots, n_G. \tag{2.35c}$$

The nominal parameter $\hat{p}$ is fixed and $\hat{z} = z(\hat{p})$ denotes an optimal solution of Problem 2.7 for the nominal parameter $\hat{p}$.

Under some regularity assumptions on J, H_i, G_j, Theorem 2.9 states that if $\hat{z}$ is a strongly regular local minimum for Problem 2.7 with $p = \hat{p}$, then Problem 2.7 with p in a neighborhood of $\hat{p}$ has a unique strongly regular local minimum $z(p)$ and the sensitivities are given (i.e. the differential with respect to p of z and of the Lagrange multipliers, computed in $\hat{p}$).

Definition 2.8 (see [49, Definition 6.1.2]). *A strongly regular local minimum $\hat{z}$ of the Problem 2.7, is a point in the state space $\mathbb{R}^{n_z}$ such that*

- *$\hat{z}$ is a local minimum for Problem 2.7, i.e.*

$$\begin{gathered} \exists \varepsilon > 0 : \quad J(\hat{z}) \leq J(z), \\ \forall z \in B(\hat{z}, \varepsilon) \text{ that satisfies (2.35b) and (2.35c);} \end{gathered} \tag{2.36}$$

- *The gradients*

$$\nabla_z H_i(\hat{z}, \hat{p}), i = 1, \ldots, n_H, \nabla_z G_j(\hat{z}, \hat{p}), j \in A(\hat{z}, \hat{p}), \tag{2.37}$$

are linearly independent, with

$$A(z,p) := \{j = 1,\dots,n_G \,|G_j(z,p) = 0\} \tag{2.38}$$

the set of active inequality constraints;

- *Let* $\hat{\mu} \in \mathbb{R}^{n_H}$, $\hat{\lambda} \in \mathbb{R}^{n_G}$ *such that* $(\hat{z},\hat{\mu},\hat{\lambda})$ *satisfies the following Karush-Kuhn-Tucker (KKT, see [49, Theorem 2.3.33]) conditions:*

$$\hat{\mu} \geq 0, \tag{2.39a}$$
$$\hat{\mu}^\top G(\hat{z},\hat{p}) = 0, \tag{2.39b}$$
$$\nabla_z J(\hat{z},\hat{p}) + \nabla_z G(\hat{z},\hat{p})^\top \hat{\mu} + \nabla_z H(\hat{z},\hat{p})^\top \hat{\lambda} = 0 \tag{2.39c}$$

  where $\hat{\mu},\hat{\lambda}$ *are called Lagrange multiplier for the inequality and equality constraints, respectively, of Problem 2.7;*

- *The conditions*

$$\hat{\mu}_j - G_j(\hat{z},\hat{p}) > 0, \quad \forall j = 1,\dots,n_G \tag{2.40}$$

  hold true;

- *Let the critical cone of Problem 2.7 defined as*

$$T_C(z,p) := \Bigg\{ \begin{array}{l} d \in \mathbb{R}^{n_z} \mid \\ \nabla_z H_i(z,p)(d) = 0, i = 1,\dots,n_H, \\ \nabla_z G_j(z,p)(d) \leq 0, j \in A(z,p), \mu = 0, \\ \nabla_z G_j(z,p)(d) = 0, j \in A(z,p), \mu > 0 \end{array} \Bigg\} \tag{2.41}$$

  and let the Lagrange function of Problem 2.7 defined as

$$L(z,\mu,\lambda,p) := J(z,p) + \mu^\top G(z,p) + \lambda^\top H(z,p) \tag{2.42}$$

  then, for all $d \in T_C(\hat{z},\hat{p})$ *with* $d \neq 0_{\mathbb{R}^{n_z}}$,

$$\nabla^2_{zz} L(\hat{z},\ell_0,\hat{\mu},\hat{\lambda},\hat{p})(d,d) > 0. \tag{2.43}$$

Theorem 2.9 (see [49, Theorem 6.1.4]). *Let* $J, H_i, G_j : \mathbb{R}^{n_z} \times \mathbb{R}^{n_p} \to \mathbb{R}$, $i = 1,\dots,n_H$, $j = 1,\dots,n_G$ *be twice continuously differentiable and* $\hat{p}$ *a nominal parameter. Let* $\hat{z}$ *be a strongly regular local minimum of Problem 2.7 for* $p = \hat{p}$, *with Lagrange multipliers* $\hat{\mu}$ *and* $\hat{\lambda}$.

Then there exist neighborhoods $B_\epsilon(\hat{p})$ *and* $B_\delta(\hat{p},\hat{\mu},\hat{\lambda})$, *such that Problem 2.7 has a unique strongly regular local minimum*

$$(z(p),\mu(p),\lambda(p)) \tag{2.44}$$

for each $p \in B_\epsilon(\hat{p})$, *and*

$$A(\hat{z},\hat{p}) = A(z(p),p) \tag{2.45}$$

where $A(z,p)$ is in (2.38) *Moreover, $(z(p),\mu(p),\lambda(p))$ is continuously differentiable with respect to p and*

$$\begin{pmatrix} \frac{dz}{dp}(\hat{p}) \\ \frac{d\mu}{dp}(\hat{p}) \\ \frac{d\lambda}{dp}(\hat{p}) \end{pmatrix} = - \begin{pmatrix} \nabla^2_{zz}L & \nabla_z G^\top & \nabla_z H^\top \\ \hat{\Xi}\cdot\nabla_z G & \hat{\Lambda} & \Theta \\ \nabla_z H & \Theta & \Theta \end{pmatrix}^{-1} \cdot \begin{pmatrix} \nabla^2_{pz}L \\ \hat{\Xi}\cdot\nabla_z G \\ \nabla_z H \end{pmatrix}, \tag{2.46}$$

where

$$\hat{\Xi} := diag(\hat{\mu}_1,\ldots,\hat{\mu}_{n_G}), \hat{\Lambda} := diag(G_1,\ldots,G_{n_G}). \tag{2.47}$$

All functions and their derivatives are evaluated at $(\hat{z},\hat{\mu},\hat{\lambda},\hat{p})$.

Therefore,

$$\frac{dz}{dp}(\hat{u},\hat{p})(\cdot) \tag{2.48}$$

can be computed using the linearized necessary Karush-Kuhn-Tucker conditions in an optimal solution $(\hat{z},\hat{u})$. An approximation to the optimal perturbed trajectory is given by

$$z(\hat{u}(p),p)(\cdot) \approx \hat{z}(\cdot) + \frac{dz}{dp}(\hat{u},\hat{p})(\cdot)(p-\hat{p}). \tag{2.49}$$

2.2 Hamilton Jacobi approach

The Hamilton Jacobi approach described here is a numerical algorithm to compute the backward reachable set and the minimal time trajectory for a constrained control problem of the type 1.1. The technique used is based on [17] where they consider a target problem for a nonlinear system under state constraints. Reachability problems are linked with the solution of the Hamilton Jacobi Equation in the framework of viscosity solutions. Therefore, optimal times and backward reachable sets are characterized with a level-set approach. Several papers in the literature deal with the link between reachability and optimal control problems, see [17, 69] and the references therein. This subsection is published in [104].

The following assumptions (H1)-(H4) will be needed:

(H1) $f : \mathbb{R}^{n_z} \times U \to \mathbb{R}^{n_z}$ is a continuous function and Lipschitz continuous in z uniformly in u, i.e. $\exists L \geq 0, \forall z_1, z_2$, $\forall u \in U$: $|f(z_1,u) - f(z_2,u)| \leq L|z_1 - z_2|$.

(H2) For all $z \in \mathbb{R}^{n_z}$ the velocity set $f(z,U)$ is convex.

(H3) Ω is a nonempty closed set of $\mathbb{R}^{n_z}$. Let $\varphi : \mathbb{R}^{n_z} \to \mathbb{R}$ be Lipschitz continuous and a level set function for the target, i.e.

$$\varphi(z) \leq 0 \iff z \in \Omega. \tag{2.50}$$

(H4) $(\mathcal{K}_t)_{t\in[t_0,t_f]}$ are a family of subsets of $\mathbb{R}^{n_z}$ such that there exists Lipschitz continuous level set function $g : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}$ with

$$g(z,t) \leq 0 \quad \Longleftrightarrow \quad z \in \mathcal{K}_t \quad \forall t \in [t_0,t_f], z \in \mathbb{R}^{n_z}. \tag{2.51}$$

The focus is now to compute backward reachable sets associated to the dynamics f. By Definition 3.4 in Chapter II, if $z^u_{z_0}(\cdot)$ denotes the absolutely continuous function, solution of the initial value problem:

$$\begin{aligned} &\dot{z}(t) = f(z(t),u(t)) \qquad \text{a.e. } t \in [t_0,t_f] \subset \mathbb{R}, \\ &z(t_0) = z_0, \end{aligned} \tag{2.52}$$

then the *backward reachable sets reaching the target* Ω *at times* $\tau \leq t_f$ for the dynamics f with time-dependent sets $(\mathcal{K}_t)$ for the state constraints is defined by

$$\begin{aligned} \mathcal{BR}^f_t := \Big\{ z_0 \in \mathbb{R}^{n_z} \mid \;& \exists \tau \in [t_0,t_f], \exists u \in \mathcal{U} : \; z^u_{z_0}(\tau) \in \Omega \\ &\text{and } z^u_{z_0}(t) \in \mathcal{K}_t \text{ for all } t \in [t_0,\tau] \Big\}. \end{aligned} \tag{2.53}$$

Remark 2.10. *Let the* capture basin *be defined as:*

$$\begin{aligned} Cap^f_{\Omega,(\mathcal{K}_t)}(t_f) := \Big\{ z_0 \in \mathbb{R}^{n_z} \mid \;& \exists u \in \mathcal{U}, \; z^u_{z_0}(t_f) \in \Omega \\ &\textit{and } z^u_{z_0}(t) \in \mathcal{K}_t \textit{ for all } t \in [t_0,t_f] \Big\} \end{aligned} \tag{2.54}$$

(following in particular [5, Subsec. 1.2.1.2]). When there is no time dependancy, $\mathcal{K}_t \equiv \mathcal{K}$*, it is known that the set* (2.53) *is a capture basin:*

$$\mathcal{BR}^f_{t_f} \equiv Cap^{\tilde{f}}_{\Omega,\mathcal{K}}(t_f) \tag{2.55}$$

associated with the dynamics $\tilde{f}(z_0,(u,\lambda)) := \lambda f(z_0,u)$ *for* $(u,\lambda) \in \tilde{U} = U \times [0,1]$ *(see for instance [73]). Here, a new virtual control* $\lambda(\cdot)$ *with* $\lambda(t) \in [0,1]$ *is introduced.*

Now let the ***value function*** v be defined by

$$v(z_0,t_f) := \inf_{u\in\mathcal{U}} \max\left(\varphi(z^u_{z_0}(t_f)), \max_{t\in[t_0,t_f]} g(t, z^u_{z_0}(t)) \right), \tag{2.56}$$

where $g(z,t) \equiv g(z)$ in the case $\mathcal{K}_t \equiv \mathcal{K}$. Such value function involving a supremum cost have been studied by Barron and Ishii in [12]. Then the function v is a level set function for $Cap^{\tilde{f}}_{\Omega,\mathcal{K}}(t_f)$ in the sense that the following holds (see [17])

$$Cap^{\tilde{f}}_{\Omega,\mathcal{K}}(t_f) = \{ z_0 \in \mathbb{R}^{n_z} \mid v(z_0,t_f) \leq 0 \}. \tag{2.57}$$

In particular assumptions (H3) and (H4) are essential for (2.57) to hold. Furthermore, v is the unique continuous *viscosity solution* (in the sense of [11]) of the following Hamilton-Jacobi (HJ) Equation

$$\min\big(\partial_{t_f} v + \mathcal{H}(z, \nabla_z v),\ v - g(z)\big) = 0, \qquad t_f > t_0,\ z \in \mathbb{R}^{n_z}, \tag{2.58a}$$

$$v(z, t_0) = \max(\varphi(z), g(z)), \qquad z \in \mathbb{R}^{n_z}, \tag{2.58b}$$

where

$$\mathcal{H}(z, p) := \max_{u \in U}(-f(z, u) \cdot p), \quad p \in \mathbb{R}^{n_z}, \tag{2.59}$$

is the *Hamiltonian.*

For the computation of backward reachable sets with time dependent state constraints, following the approach of [18], the new state variable $\xi := (z, t)$ and the "augmented" dynamics with values in $\mathbb{R}^{n_z+1}$ are introduced:

$$F(\xi, u) := \begin{pmatrix} f(z, u) \\ 1 \end{pmatrix}. \tag{2.60}$$

Let also be $\xi_0 := (z_0, t_0)$ and trajectories $\xi^u_{\xi_0}$ associated to F, fulfilling

$$\dot{\xi}(t) = F(\xi(t), u(t)) \text{ and } \xi(t_0) = \xi_0. \tag{2.61}$$

For a fixed $t_f > t_0$, let

$$\tilde{\Omega} := \bigcup_{t \in [t_0, t_f]} \Omega \times \{t\} \ \equiv\ \Omega \times [t_0, t_f] \tag{2.62}$$

and

$$\tilde{\mathcal{K}} := \bigcup_{t \in [t_0, t_f]} \mathcal{K}_t \times \{t\}. \tag{2.63}$$

Then it holds:

Proposition 2.11. *For all $t_f \geq t_0$,*

$$\forall t \in [t_0, t_f],\ z_0 \in Cap^f_{\Omega, (\mathcal{K}_t)}(t_f) \iff (z_0, t_0) \in Cap^F_{\tilde{\Omega}, \tilde{\mathcal{K}}}(t_f). \tag{2.64}$$

The definition of φ by $\varphi(z, t) := \varphi(z)$ is extended, so that for any $\xi_0 \in \mathbb{R}^n \times \mathbb{R}$ and $\tau \geq t_0$ the value p is:

$$p(\xi_0, \tau) := \inf_{u \in \mathcal{U}} \max\left(\varphi(\xi^u_{\xi_0}(\tau)), \max_{t \in [t_0, \tau]} g(\xi^u_{\xi_0}(t))\right) \tag{2.65}$$

Then, one can verify that p is Lipschitz continuous and the following theorem holds:

Theorem 2.12. *Assume (H1)-(H4).*
(i) *For every $\tau \geq t_0$:*

$$Cap^f_{\Omega,(\mathcal{K}_t)}(\tau) = \{z \in \mathbb{R}^{n_z},\ p((z,t_0),\tau) \leq 0\}\,. \tag{2.66}$$

(ii) *p is the unique continuous viscosity solution of*

$$\begin{cases} \min(\partial_\tau p + \mathcal{H}((z,t_f),(\nabla_z p, \partial_{t_f} p)),\ p((z,t_f),\tau) - g(z)) = 0, \\ \qquad\qquad\qquad\qquad\qquad\qquad \tau > t_0,\ (z,t_f) \in \mathbb{R}^{n_z+1} \\ p((z,t_f),t_0) = \max(\varphi(z), g(z,t_f)), \quad (z,t_f) \in \mathbb{R}^{n_z+1}. \end{cases} \tag{2.67}$$

where for any $\xi = (z,t_f)$ and $(p_z, p_{t_f}) \in \mathbb{R}^{n_z} \times \mathbb{R}$:

$$\mathcal{H}((z,t_f),(p_z,p_{t_f})) := \max_{u\in U}(-f(z,u)\cdot p_z - p_{t_f}). \tag{2.68}$$

Once the backward reachable set is characterized by a viscosity solution of (2.67), it is possible to use a PDE solver to find the solution on a grid in the state space.

Minimal time function and optimal trajectory reconstruction.

In the case of fixed state constraints (i.e. $\mathcal{K}_t \equiv \mathcal{K}$), the *minimum time function*, denoted by $\mathcal{T}$, is defined by:

$$\mathcal{T}(z_0) := \inf\{\tau \in [t_0,t_f]\,|\,\exists u \in \mathcal{U} : z^u_{z_0}(\tau) \in \Omega \text{ and } z^u_{z_0}(t) \in \mathcal{K} \text{ for all } t \in [t_0,\tau]\}, \tag{2.69}$$

and if no such time τ exists then $\mathcal{T}(z_0) = \infty$. It is easy to see that the function satisfies

$$\mathcal{T}(z_0) = \inf\{\tau \in [t_0,t_f],\ v(z_0,\tau) \leq 0\}. \tag{2.70}$$

Notice that $\mathcal{T}$ can be discontinuous even though v is always Lipschitz continuous. No controllability assumptions are used in the present approach.

The optimal trajectory reconstruction is then obtained by minimizing the minimal time function along possible trajectories (see for instance [39]). More precisely, assume that the starting point is z_0 and that $z_n := z^u_{z_0}(t_n) \in \Omega$ at some future time

$t_n > t_0$. This is equivalent to require $\mathcal{T}(z_n) = 0$.

```
input  : $t_0 = 0$, time step $\Delta t > 0$, small threshold $\eta > 0$, a control
         discretization of the set $U$, say $(u_k)_{k=1,\dots,N_u} \subset U$, and $t_n := n\Delta t$,
         $n = 0, \dots, N-1$.
output : $(z_n)_{n\in\{0,\dots,N-1\}}$ grid function approximating the minimum time
         trajectory
while $n < N$ and $\mathcal{T}(z_n) \geq \eta$ do
    Find $k^* := \mathrm{argmin}_{k=1,\dots,N_u} \mathcal{T}\big(\bar{z}_{z_n}^{u_k}(\Delta t)\big)$;
    Set $z_{n+1} := \bar{z}_{z_n}^{u_{k^*}}(\Delta t)$;
    $n := n + 1$;
end
```

Algorithm 1: Minimum time trajectiory with ROC-HJ software.

Where the notation $\bar{z}_{z_n}^{u_k}(h)$ denotes a *one-step second-order Runge-Kutta approximation* of the trajectory with fixed control u_k on $[t_n, t_{n+1}]$. The *Heun scheme* ***with piecewise constant selections*** uses the iteration:

$$\bar{z}_{z_n}^{u_k}(h) := z_n + \frac{h}{2}(f(z_n, u_k) + f(z_n + hf(z_n, u_k), u_k)). \tag{2.71}$$

It is also possible to do smaller time steps between $[t_n, t_n + \Delta t]$ in order to improve the precision for a given control u_k. Nevertheless, **numerical observations are showing that** the approximation is in general more sensitive to the control discretization $(u_k)_{k=1,\dots,N_u}$ of the set U.

In the time-dependant case this minimal time function can be defined in a similar way from the value p (see [18] for details), and the optimal trajectory reconstruction follows the same lines with the "augmented" dynamics (2.60).

Chapter IV

Computational Results

Contents

Investigating performance criteria for a trajectory in a car traffic scenario preserving safety is the main goal of this chapter. The investigation process began in Section 3 of Chapter II where three questions were posed and the optimal control model was given in Problem 3.1 of Chapter II. Here the algorithm behind such requests is presented and numerical solutions are shown. Two different software packages for several vehicle models and scenarios are tested. The package OCPID-DAE1 [48] with a Fortran 90 interface is designed for the numerical solution of optimal control problems and parameter identification problems. The ROC-HJ [16] Solver for solving Hamilton-Jacobi Bellman equations can be used for reachable sets computations and optimal trajectory reconstruction. Numerical difficulties can arise from the high dimensionality of the vehicle model or from the regularity restrictions on the functions describing the car traffic scenario.

As explained in Section 1 of Chapter II lower dimensional models are preferred, since they have higher maneuverability capability with lower CPU times. A trade off between low dimension and the precision of the collision decision, by keeping adherence to real capabilities of the vehicle, is a key value in the choice of the reference vehicle model. The 4D point mass model in (1.31) of Chapter II is matching this requirement as motivated in Section 1 of Chapter II. On the other side, to compute trajectories close to real life car motions and for being able to implement them in real cars, it is necessary to consider more complex dynamics, therefore the model in (1.21) is used. The vehicle model is widely discussed in Section 1 of Chapter II and the related notation is reminded here. The control set is denoted as

$$\mathcal{U} := \{u : [t_0, t_f) \to U,\ u \text{ measurable}\},$$

with U a nonempty compact subset of $\mathbb{R}^{n_u}$, $n_u \geq 1$. Let $n_z \geq 1$ and the dynamics describing the evolution of the vehicle motion is

$$f : \mathbb{R}^{n_z} \times U \to \mathbb{R}^{n_z} \text{ Lipschitz continuous with respect to } (z, u),$$

where f is described in one of the (1.21), (1.26), (1.27), (1.28), (1.31), (1.37), (1.38), (1.39) depending on the chosen vehicle model. Given an initial state $z_0 \in \mathbb{R}^{n_z}$, let $z^u_{z_0}$ denote the *absolutely continuous* solution of the following dynamical system

$$\begin{aligned} &\dot{z}(t) = f(z(t), u(t)) \quad \text{for a.e. } t \geq 0, \\ &z(0) = z_0. \end{aligned} \tag{0.1}$$

Depending on the chosen dynamics, the state z might have to obey state constraints (1.29) of Chapter II.

A car traffic scenario is described in Section 2 of Chapter II as a quintuple

$$(\mathcal{K}_d, \mathcal{K}_r, \mathcal{K}_o, \mathcal{V}, \Omega), \tag{0.2}$$

where $\mathcal{K}_d$ is defined by the dynamic constraints (1.29) of Chapter II, $\mathcal{K}_r$ is the road set, $\mathcal{K}_o$ is the obstacle-free set, $\mathcal{V}$ is the vehicle set, and Ω is the target set. As shown there, by imposing (2.64), the associated Lipschitz functions g_d, g_r, g_o, φ or the associated smooth functions $\widetilde{g}_d, \widetilde{g}_r, \widetilde{g}_o, \widetilde{\varphi}$ are derived, and the optimal control problem 3.1 is given. In this chapter numerical simulations obtained by solving such model show optimal trajectories (see Definition 3.2 in Chapter II), reachable sets/funnel trajectories (see Definition 3.3 in Chapter II), and backward reachable sets (see Definition 3.4 in Chapter II), for several car traffic scenarios. In this chapter the (0.2) is defined for each numerical simulation, followed by the optimal control problem in the form of Problem 3.1 of Chapter II. A brief explanation of the algorithm implemented is then explained, before plotting the computations.

The last section is dedicated to sensitivity analysis as required in Section 4 of Chapter II. Firstly, parametric sensitivity analysis (Fiacco-Sensitivity) is adopted in order to investigate the influence of inaccurate sensor measurements, in both trajectories and reachable sets. A condition to update the nominal optimal solution to the optimal one of the perturbed problem is derived, whenever a deviation between the desired and the actual state of the process occurs. Secondly, a sensitivity analysis with fixed optimal control (ODE-Sensitivity) is computed to investigate the influence of inaccurate sensor measurements, in the solution of the initial value problem (and thus trajectories and reachable sets), being the control equal to the known nominal optimal control of the non perturbed dynamics. Hence, the definition of a robust reachable set is given, as the set of reachable points by the trajectories, which are solutions of the nominal and the perturbed problem. Thirdly, the definition of perturbed backward reachable set is given as the set of all initial state from which is possible to drive a collision free nominal trajectory such that the nominal trajectory is also admissible for perturbations in a neighborhood of the nominal initial state, being the control taken as the nominal control. The perturbed backward reachable set will be used to derive an estimation of the maximum possible error around a given initial data, such that the nominal trajectory, staring from the perturbed initial data and being the control the nominal one, is still collision-free. This means to give a characterization of the parameter perturbations such that the nominal solution is admissible even if such perturbations occur.

1 A direct methods software

The numerical solution of the optimal control problems related to the three question of Section 3 of Chapter II are here provided. For doing so the software package OCPID-DAE1 is used. It aims to find a numerical solution of the optimal control problems via direct methods, using B-spline interpolation of the controls and a multiple shooting method for the approximation of the states, see Section 2.1 of Chapter III for details. Mainly a 4D point mass model and a 7D single track model are used, however comparison with other kinematic models is also given.

1.1 Case study: straight road with a fixed circular obstacle

The case study used in this subsection expects a stationary obstacle located at the position (x_1, y_1) on a straight road that has to be avoided by the reference vehicle driving at a prescribed speed:

- The reference vehicle is a circle $\mathcal{V}(z(t), t) = B(X(z(t), t), \ell)$ of radius $\ell = 1$ and center $X(z(t), t) = (x(t), y(t))$. It evolves with dynamics f describing a 7D single track model as in (1.21) of Chapter II, with

$$\begin{array}{rcl} z & = & (x, y, \psi, v_x, v_y, \omega_\psi, \delta) \text{ states}, \\ u & = & (\omega_\psi, F_B) \text{ controls}, \\ z_0 & = & (x_0, y_0, \psi_0, v_{x_0}, v_{y_0}, \omega_{\psi_0}, \delta_0) \\ & = & (0.0, 1.75[m], 0.0, 70[km/h].0, 0.0, 0.0, 0.0) \text{ initial states}. \end{array} \tag{1.1}$$

  The state must obey condition in (1.29) of Chapter II.

- $\Omega = \{z \in \mathbb{R}^{n_z},\ z_1 \geq x_1,\ z_3 = 0.0\}$ according to (2.3) of Chapter II.

- Road: $\mathcal{K}_r = \left\{ (x,y) \in \mathbb{R}^2 \,|\, 0 - w_2 \leq y \leq 7 + w_2 \right\}$, where $w_2 \in \mathbb{R}$ is a parameter that needs to be optimized or fixed to zero, as in (2.7) of Chapter II.

- Obstacle parameters: one fixed circular obstacle

$$\mathcal{O}_1 = B(X_1(w_1), \ell_1(w_2)), \tag{1.2}$$

with radius $\ell_1(w_2) = 1.0 - w_2$ centered at $X_1 = (x_1(w_1), y_1) = (w_1, 1.75)$, where $w = (w_1, w_2) \in \mathbb{R}^2$ is a parameter that needs to be optimized or which is fixed to the value $w = (30.0, 0.0)$. If w_1 is optimized then it means that the initial distance between reference vehicle and obstacle is minimized, if w_2 is optimize, i.e. $w_2 \neq 0$, then a constraints minimization technique is implemented. The obstacle set is defined as

$$\mathcal{K}_o = \{ Y \in \mathbb{R}^2 \,|\, \|Y - X_1(w)\|_2{}^2 \geq \ell_1(w_2)^2 \}. \tag{1.3}$$

Using (2.70) of Chapter II the state z has to satisfy property in (2.87) of Chapter II, with

- state constraints defined for:

$$\mathcal{K} = \left\{ z \in \mathbb{R}^{n_z} \mid \underline{g} \leq \widetilde{g}(z) \leq \overline{g}, \underline{g}, \overline{g} \in \mathbb{R}^{n_{\widetilde{g}}} \right\}, \tag{1.4}$$

with

$$\begin{aligned} \widetilde{g}(z) &:= \begin{pmatrix} z_2 \\ (z_1 - x_1(w))^2 + (z_2 - y_1)^2 \end{pmatrix}, \\ \underline{g} &:= \begin{pmatrix} 0 + \ell + \eta - w_2 \\ (\ell_1(w_2) + \ell + \eta)^2 \end{pmatrix}, \\ \overline{g} &:= \begin{pmatrix} 7 - \ell - \eta + w_2 \\ +\infty \end{pmatrix}, \end{aligned} \tag{1.5}$$

according to (2.13), and (2.72) of Chapter II. Therein $n_z = 7$ is the state and dynamics dimension and $\eta = 0.3[m]$ is a safety margin.

- boundary constraints defined for:

$$\Omega = \left\{ z \in \mathbb{R}^{n_z} \mid \underline{\varphi} \leq \widetilde{\varphi}(z) \leq \overline{\varphi}, \underline{\varphi}, \overline{\varphi} \in \mathbb{R}^{n_{\widetilde{\varphi}}} \right\}, \tag{1.6}$$

with

$$\widetilde{\varphi}(z) := \begin{pmatrix} z_1 \\ z_3 \end{pmatrix}, \underline{\varphi} := \begin{pmatrix} x_1(w) + 10 \\ 0 \end{pmatrix}, \overline{\varphi} := \begin{pmatrix} +\infty \\ 0 \end{pmatrix}, \tag{1.7}$$

according to (2.6) of Chapter II.

Thus the associated optimal control problem as in 3.1 of Chapter II is given straightforwardly.

Problem 1.1. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with $t_0 < t_f$ fixed. Let $\widetilde{g}$ be a smooth function defined in* (1.5) *and let $\widetilde{\varphi}$ be a smooth function defined in* (1.7). *Find states $z : \mathbb{R} \to \mathbb{R}^{n_z}$ absolutely continuous, controls $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u}$ meaurable$\}$ and parameters $w \in W \subset \mathbb{R}^{n_w}$, $\tau \in [t_0, t_f]$ such that:*

$$
\begin{array}{ll}
\min\limits_{z, u \in \mathcal{U}, w \in W, \tau \in [t_0, t_f]} & \varphi_0(z(\tau), \tau, w) + \int_0^\tau f_0(z(t), u(t))dt \\
s.t. & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, \\
& z(t_0) = z_0, \\
& \underline{g} \le \widetilde{g}(z(t)) \le \overline{g}, \quad \forall t \in [t_0, \tau], \quad \Longleftrightarrow \\
& \begin{cases} (0 + \ell + \eta - w_2) \le z_2(t) \le (7 - \ell - \eta + w_2) \\ (z_1(t) - x_1(w))^2 + (z_2(t) - y_1)^2 \ge (\ell_1(w_2) + \ell + \eta)^2 \end{cases} \\
& \underline{\varphi} \le \widetilde{\varphi}(z(\tau)) \le \overline{\varphi}, \quad \Longleftrightarrow \\
& \begin{cases} z_1(\tau) \ge x_1(w) + 10 \\ z_3(\tau) = 0.0 \end{cases}
\end{array}
\tag{1.8}
$$

where φ_0, f_0 are maps, $\overline{g}, \underline{g}, \overline{\varphi}, \underline{\varphi}$ are in (1.5) *and* (1.7). *The function $f : [t_0, t_f] \times \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}^7$ is the single track dynamics given in* (1.21) *of Chapter II, with $z = (x, y, \psi, v_x, v_y, w_\psi, \delta), u = (w_\delta, F_B)$.*

The software used in this section requires smooth state constraints, this is why in Problem 1.1, condition in (2.86) of Chapter II is used over the equivalent conditions in (2.87), (2.85) of Chapter II.

Optimal trajectory

To compute an optimal trajectory z (see Definition (3.2) of Chapter II) for Problem 1.1, the following functional has to be minimize:

$$
c_1 \tau + c_2^\top w + c_3 \int_0^\tau u(t)^2 dt,
$$

with $c_1, c_3 \in \mathbb{R}$, $c_2 \in \mathbb{R}^2$ positive and the optimized parameter $w = (w_1, w_2) \in \mathbb{R}^2$. The Algorithm 2 is used and the simulations are here presented.

```
input  : road geometry, obstacle data (geometry and motion), vehicle
         data (model, geometry, initial state $z_0$), target data (position
         and yaw angle).
output: trajectory $z$, control $u$, parameters $w$.

define $c$ and objective function as in (3.3) Chapter II;
solve the Optimal Control Problem 1.1;
```

Algorithm 2: Optimal trajectory with OCPID-DAE1 software.

A comparison of single track trajectories with several objective functions weights ($c \in \mathbb{R}^4$ such that $c_i(j) = \delta_{ij}$, where δ is the Kronecker delta) is give in Figures 1.1, 1.2, 1.3. The top row always refers to the trajectory in the (x, y) plane that the reference vehicle is supposed to drive. The middle row shows the steering angular velocity ω_δ and the braking force F_B. The parameters in the bottom row are the final time τ, the initial x-distance w_1 between reference vehicle and obstacle, and the constraint violation term w_2. In Figure 1.1 a minimum time trajectory (left column) and a minimum steering effort trajectory (right column) are compared. The latter shows smother controls that makes the trajectory driver friendly, however minimum time trajectory do not have extreme controls as it is the case for minimum distance trajectories (Figure 1.3). Figure 1.2 plots the trajectory obtained by minimizing w_2, if non-positive then the state constraints are satisfied. Moreover if w_2 is negative, the obtained trajectory is as far as possible from the obstacles and from the border of the road, if w_2 is positive then the state constraints are not satisfied for any possible control and a collision scenario occurs. The Figure 1.3 compares minimization of w_1 for different initial velocities of the reference vehicle. Highly extreme controls appear and as it will be shown in Section 3 a small error in the initial data will lead to a crash due to the fact that such trajectory approaches the obstacle very close. The right picture shows that for an initial velocity of $70[km/h]$ the minimum initial x-distance between obstacle and vehicle is $16[m]$. For the same initial velocity in Figure 1.2, the left picture has an initial distance grater than $16[m]$ and this is why in that case the constraint violation w_2 (third parameter plot) is negative. While in the right picture the initial distance lower than $16[m]$ and this is why in that case the constraint violation w_2 (third parameter plot) is grater than zero, leading to a collision.

Minimum time and minimum steering effort.

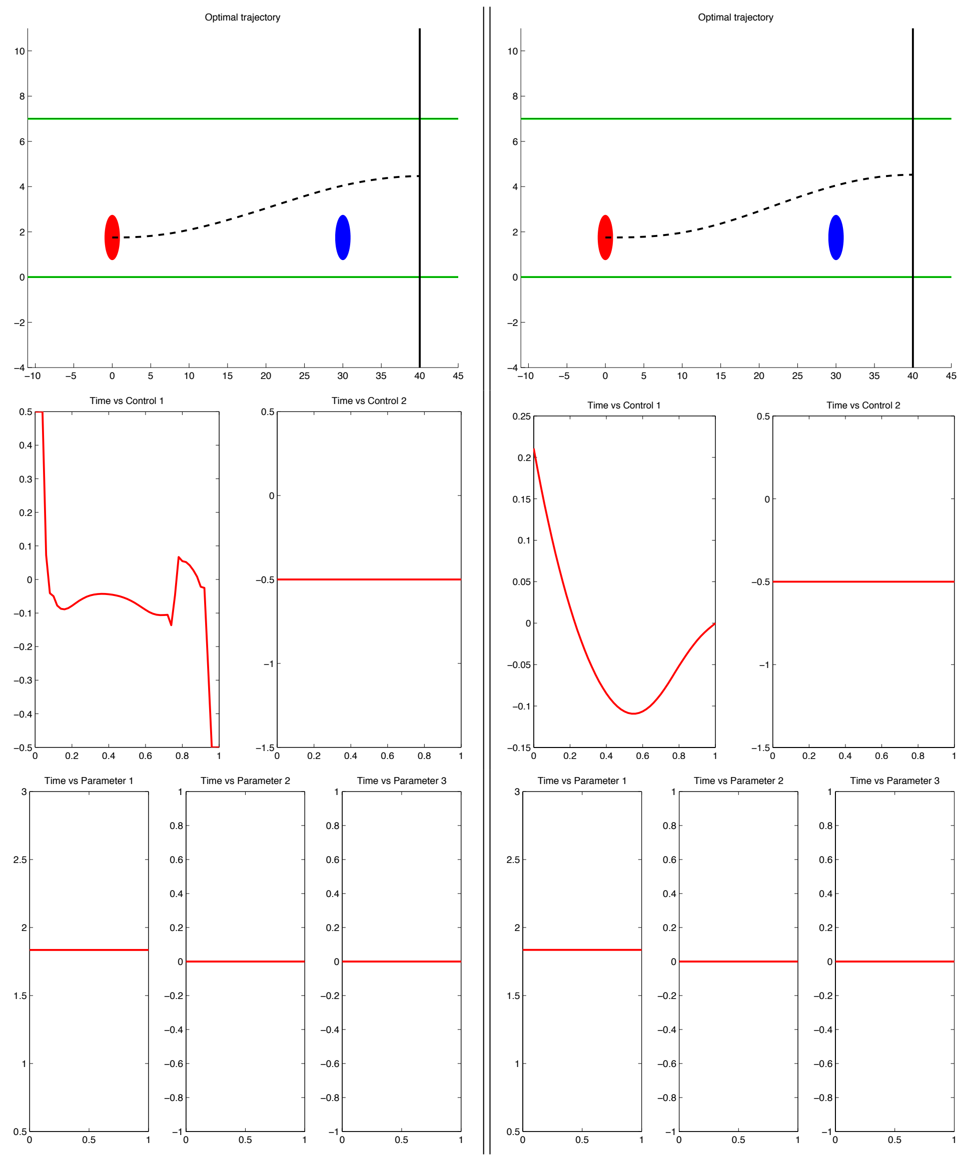


Figure 1.1: The rows (from top to bottom) contain the optimal trajectory, controls (steering velocity and braking force), and parameters (final time, initial x-distance between vehicle and obstacle, constraint violation). In the left column the final time is minimized with $c = (1, 0, 0, 0)$. In the right column the steering effort is minimized with $c = (0, 0, 0, 1)$ and initial velocity $v_{x0} = 70[km/h]$. CPU time: $0.05[s] - 0.07[s]$.

Minimum constraint violation.

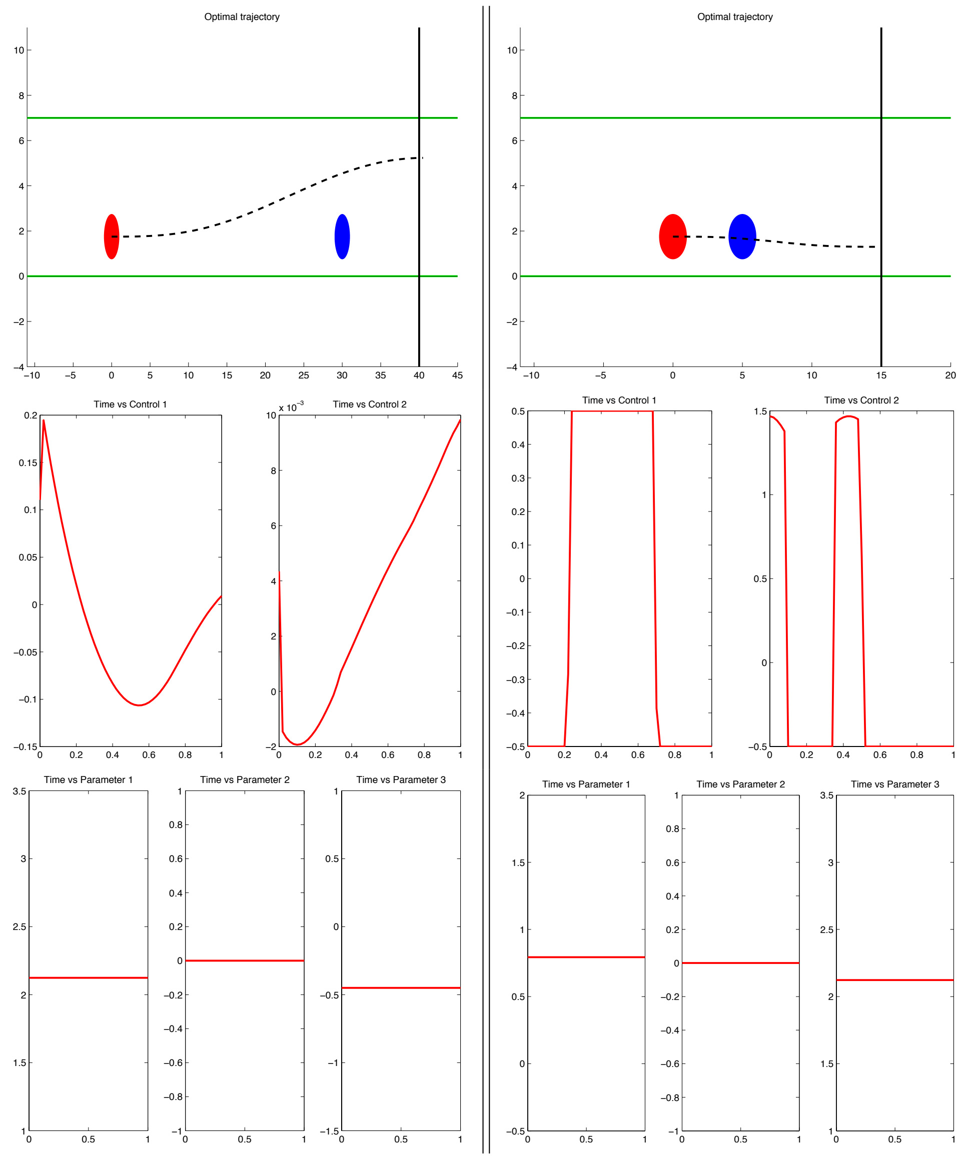


Figure 1.2: The rows (from top to bottom) contain the optimal trajectory, controls (steering velocity and braking force), and parameters (final time, initial x-distance between vehicle and obstacle, constraint violation). The constraint violation is minimized with $c = (0, 0, 1, 0)$ and initial velocity $v_{x0} = 70[km/h]$. In the left column the initial distance is fixed to $w_1 = 30[m]$, in the right column it is fixed to $w_1 = 5[m]$. CPU time: $0.05[s] - 0.07[s]$.

Minimum initial distance.

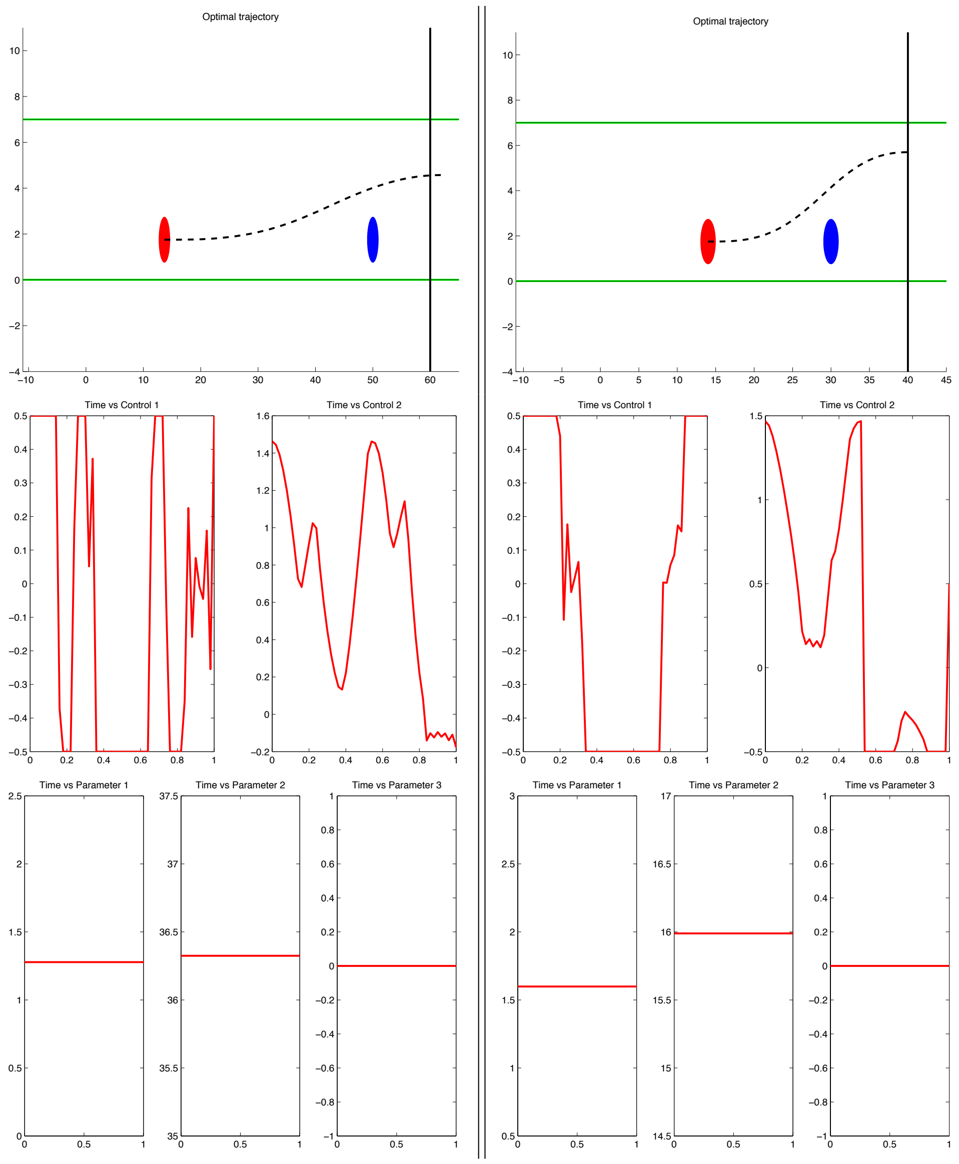


Figure 1.3: The rows (from top to bottom) contain the optimal trajectory, controls (steering velocity and braking force), and parameters (final time, initial x-distance between vehicle and obstacle, constraint violation). The initial x-distance w_1 between vehicle and obstacle is minimized with $c = (0, 1, 0, 0)$; in the left column with initial vehicle velocity $v_{x_0} = 150[km/h]$, in the right column with $v_{x_0} = 70[km/h]$. CPU time: $0.05[s] - 0.07[s]$.

Reachable set and trajectory funnel

The reachable set $\mathcal{FR}^f_{t_f}$ and the trajectory funnel $\mathcal{TF}^f_{t_f}$ associated to dynamics f and to the final time $\tau \in [t_0, t_f]$ are in Definition 3.3 of Chapter II. Let $z^{\hat{u}}_{z_0}$ denote the optimal trajectory, solution of Problem 1.1 with objective function given by (3.6) in Chapter II, i.e. where the term $\|z^{\hat{u}}_{z_0}(\tau) - g_h\|_2$ is minimized, with $\mathbb{G} = \{g_h\}_{h\in\{1,\dots,k\}}$ is a grid in the state space. Therefore, an approximation of the reachable set is given by the union of all grid points g_h sufficiently close to $\hat{z}^u_{z_0}(\tau)$,

$$\mathcal{FR}^f_{t_f} \approx \bigcup_{g_h : \|z^{\hat{u}}_{z_0}(\tau) - g_h\|_2 \le Ch} \{g_h\}, \ C > 0 \text{ suitable}, \tag{1.9}$$

i.e. those belonging to an $O(h)$-neighborhood of g_h, see [9, 10].

```
input  : road geometry, obstacle data (geometry and motion), vehicle
         data (model, geometry, initial state $z_0$), target data (position
         and yaw angle).
output: $G \subset \mathbb{G}$,
         $U$ matrix,
         $Z$ matrix.

define a grid $\mathbb{G} = \{g_h\}_{h=1,\dots,k}, k > 0$ in the state space;
for h=1,...,k do
    define $c$ and objective function as in (3.6) Chapter II;
    solve the Optimal Control Problem 1.1;
    if solution found then
        $G := G \cup \{g_h\}$;
        $U := U \cup \{u_h(t)\}_{t\in[t_0,\tau]}$;
        $Z := Z \cup \{z_h(t)\}_{t\in[t_0,\tau]}$;
    end
end
```

Algorithm 3: Reachable set and trajectory funnel with OCPID-DAE1 software.

The reachable set and the trajectory funnel are calculated for an initial velocity of $v = 126[km/h]$ in Figure 1.4. The dotted points of the reachable set correspond to different free final times. Due to the initial speed and the end condition $\psi(\tau) = 0[rad]$, no other grid points from the dashed bounding box can be reached by the avoiding car.

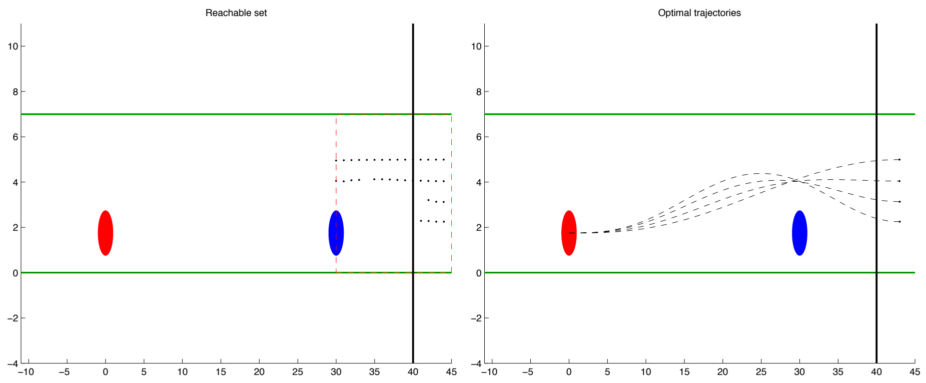


Figure 1.4: Graph to the left shows the reachable set, graph to the right refers to the reachable set with segment parallel to y-axel and its trajectory funnel (Definition 3.3 of Chapter II).

Backward reachable set

The backward reachable set $\mathcal{BR}^f$ associated to dynamics f is in Definition 3.4 Chapter II. It is the set of all initial states of a reference vehicle dynamics such that the safety property (2.87) in Chapter II holds. Let $\mathbb{G} = \{g_h\}_{h\in\{1,\ldots,k\}}$ be a grid on the state space. An approximation of the backward reachable set is then given by the union of all grid points g_h such that it exists $z^{\hat{u}}_{g_h}$ optimal trajectory of Problem 1.1 with initial state g_h,

$$\mathcal{BR}^f_{t_f} \approx \bigcup_{g_h : z^{\hat{u}}_{g_h}(t) \text{ solves Problem 1.1.}} \{g_h\}. \tag{1.10}$$

```
input  : road geometry, obstacle data (geometry and motion), vehicle
         data (model, geometry, initial state $z_0$), target data (position
         and yaw angle).
output: $G \subset \mathbb{G}$, $Z$, $U$.
define a grid $\mathbb{G} = \{g_h\}_{h=1,\dots,k}$, $k > 0$ in the state space $\mathbb{R}^2$;
for $h$=1,...,$k$ do
    set $z_0 := g_h$;
    define the objective function as a constant;
    solve the Optimal Control Problem 1.1;
    if solution found then
        $G := G \cup \{g_h\}$;
        $Z(h) := \{z_h(t)\}_{t\in[t_0,\tau]}$;
        $U(h) := \{u_h(t)\}_{t\in[t_0,\tau]}$;
    end
end
```

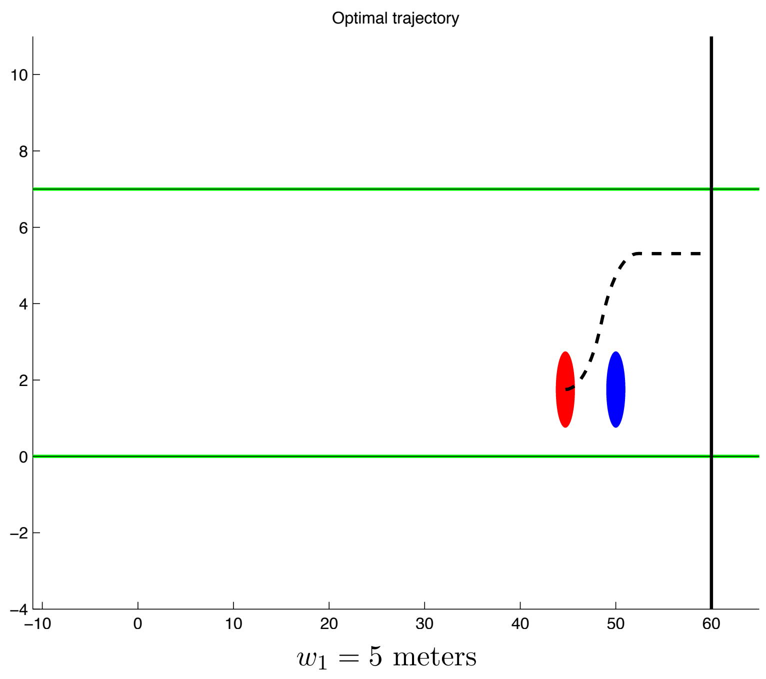
Algorithm 4: Backward reachable set with OCPID-DAE1 software.

Numerical simulations for the backward reachable set with OCPID-DAE software are given in Subsection 2.1 where a comparison with backward reachable sets compute with ROC-HJ software takes place.

1.2 Comparison with other car models

Comparison of several car models was already shown in Section 1 of Chapter II. However here some key points are underlined by showing optimal trajectories, trajectory funnels and reachable sets for such models.

In Figures 1.5, 1.6, 1.7, and 1.8, the minimum distance trajectory of the kinematic 4D and 5D is compared with the point mass model and single track 7D. The kinematic 4D model is not affected by changes in the initial velocity for the computation of the minimum initial distance. This is due to the fact that the steering angle is a control and not a state as in the other models where the steering angular velocity is instead a control.



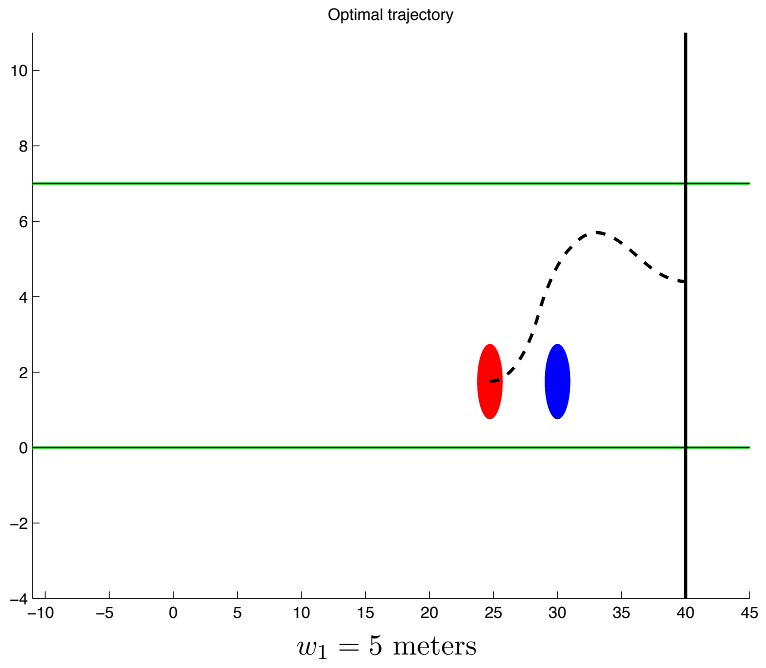
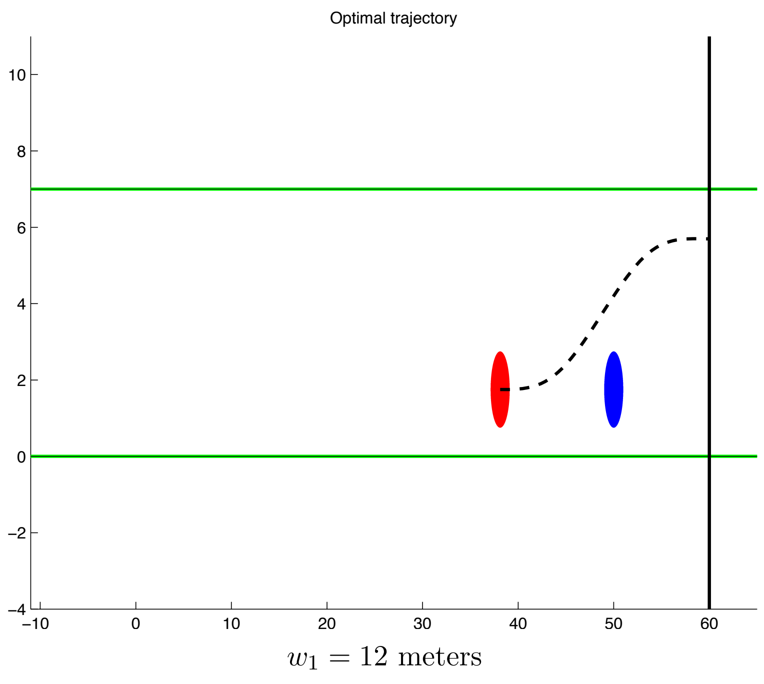


Figure 1.5: Optimal trajectories for Problem 1.1 with $c = (0, 1, 0, 0)$ are shown for the kinematic 4D car model in (1.38) of Chapter II. In the top graph with initial vehicle velocity $v_{x_0} = 150[km/h]$, in the bottom graph with $v_{x_0} = 70[km/h]$. CPU time: $0.05[s] - 0.07[s]$.



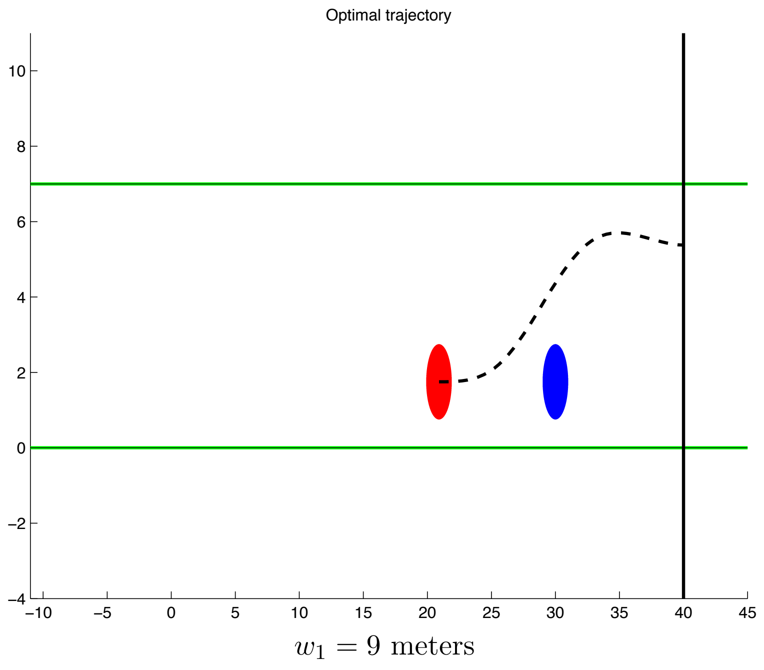
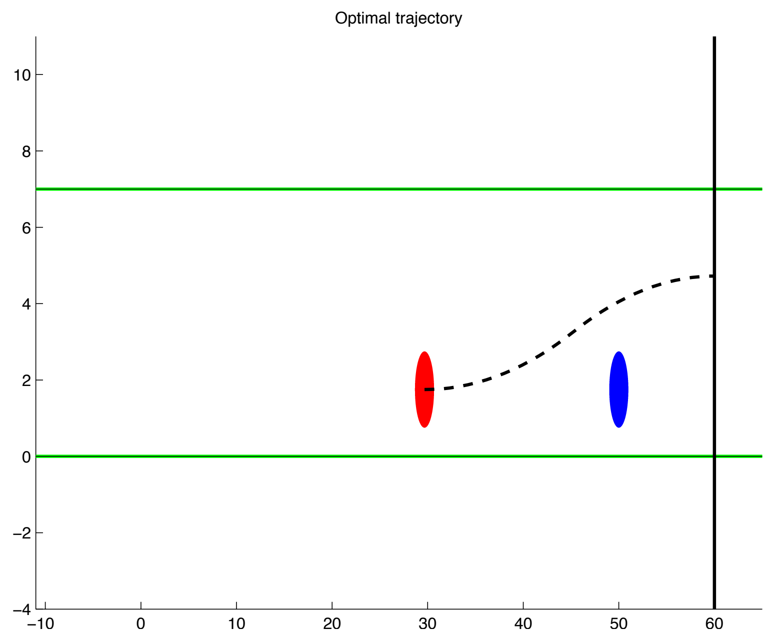
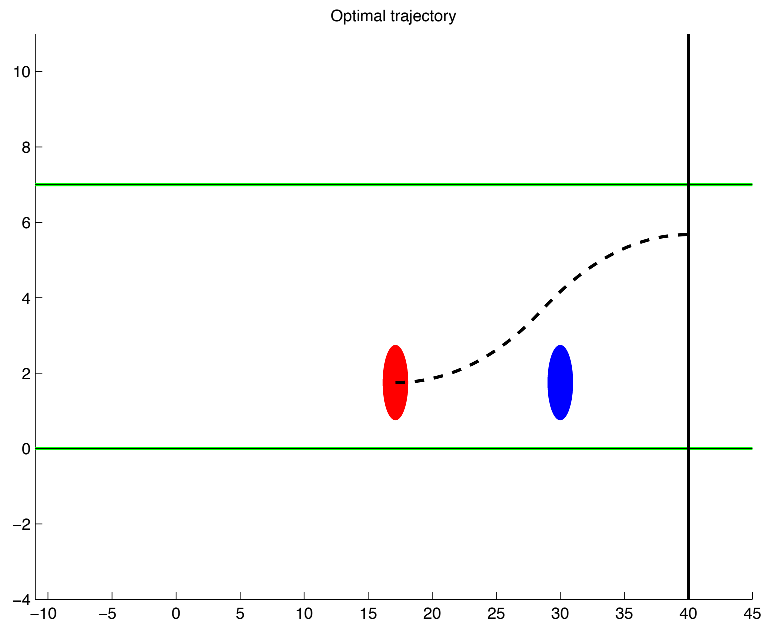


Figure 1.6: Optimal trajectories for Problem 1.1 with $c = (0, 1, 0, 0)$ are shown for the kinematic 5D car model in (1.39) of Chapter II. In the top graph with initial vehicle velocity $v_{x_0} = 150[km/h]$, in the bottom graph with $v_{x_0} = 70[km/h]$. CPU time: $0.05[s] - 0.07[s]$.

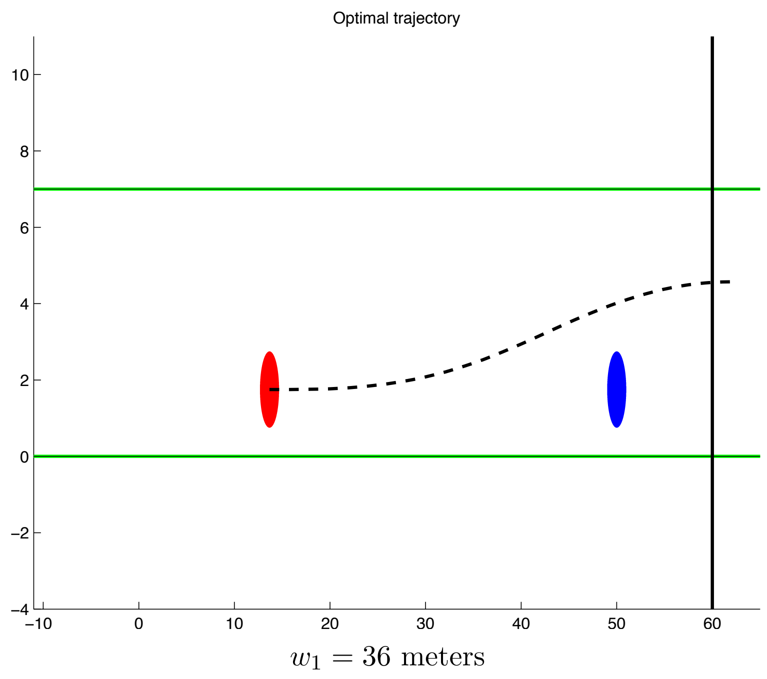


$w_1 = 20$ meters



$w_1 = 13$ meters

Figure 1.7: Optimal trajectories for Problem 1.1 with $c = (0, 1, 0, 0)$ are shown for the point mass car model in (1.31) of Chapter II. In the top graph with initial vehicle velocity $v_{x_0} = 150[km/h]$, in the bottom graph with $v_{x_0} = 70[km/h]$. CPU time: $0.05[s] - 0.07[s]$.



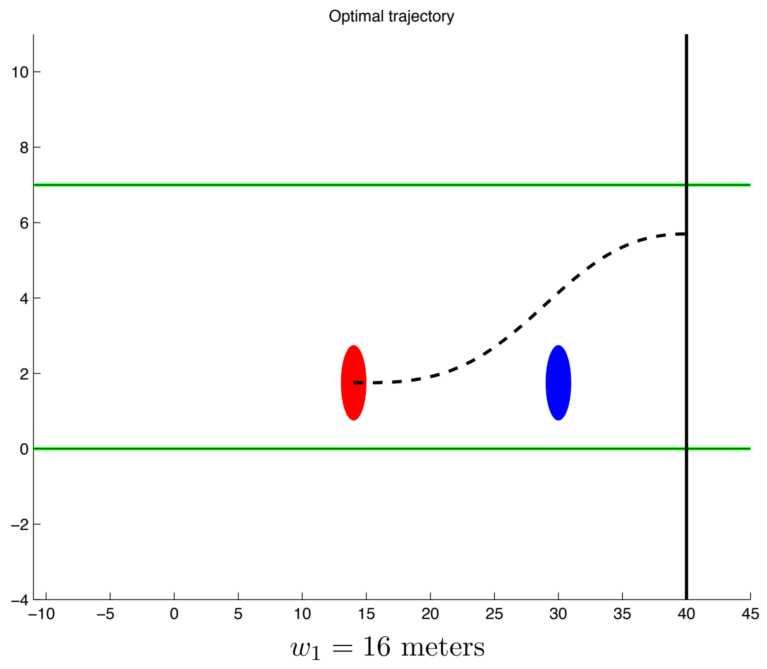


Figure 1.8: Optimal trajectories for Problem 1.1 with $c = (0, 1, 0, 0)$ are shown for the single track car model in (1.21) of Chapter II. In the top graph with initial vehicle velocity $v_{x_0} = 150[km/h]$, in the bottom graph with $v_{x_0} = 70[km/h]$. CPU time: $0.05[s] - 0.07[s]$.

The reachable set evaluated for the point mass model in Figure 1.9 does not lead to considerably differences with the single track reachable set of Figure 1.4. However the kinematic 3D model reachable set in Figure 1.10 is larger than the single track one. This is well understood by looking at the trajectory funnel where changes in the steering angle have immediate consequence on the trajectory directions, since the angle is a control and not controlled through its derivative, as in the single track dynamics.

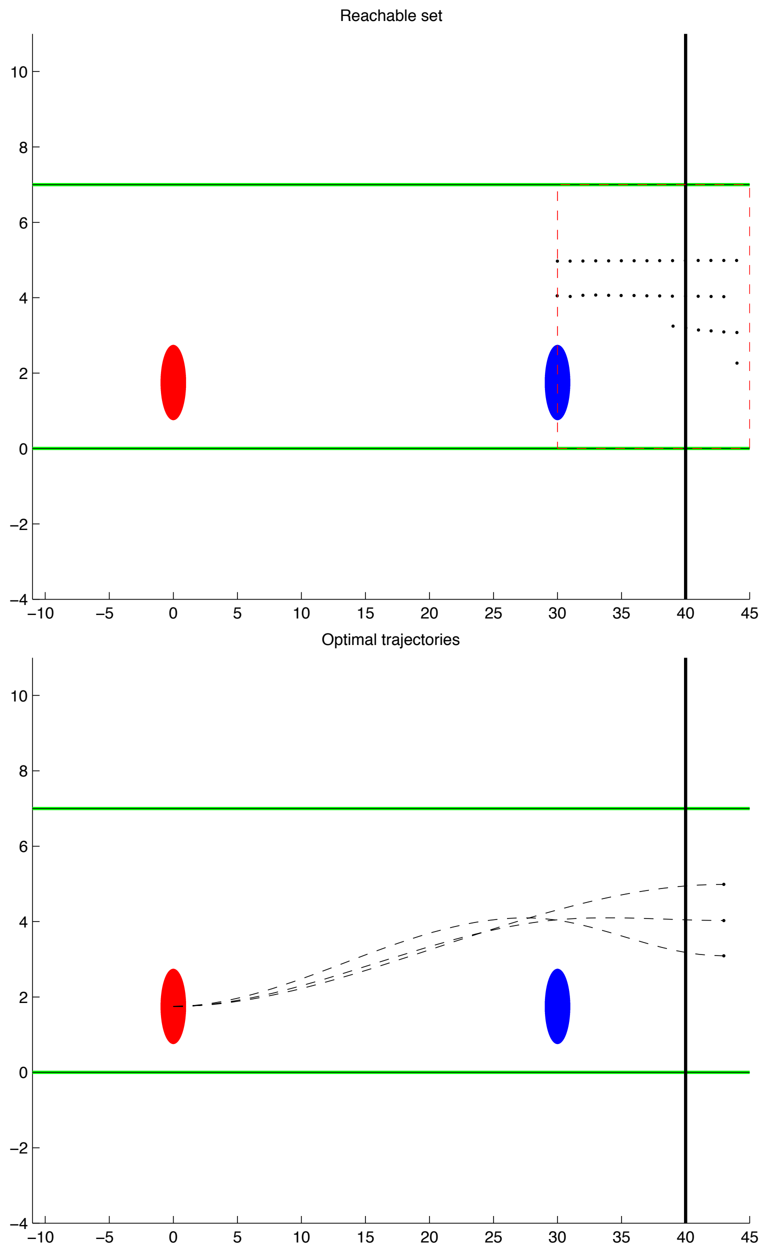


Figure 1.9: Figure to the top shows the reachable set, figure to the bottom refers to the reachable set with segment parallel to y-axis and its trajectory funnel.

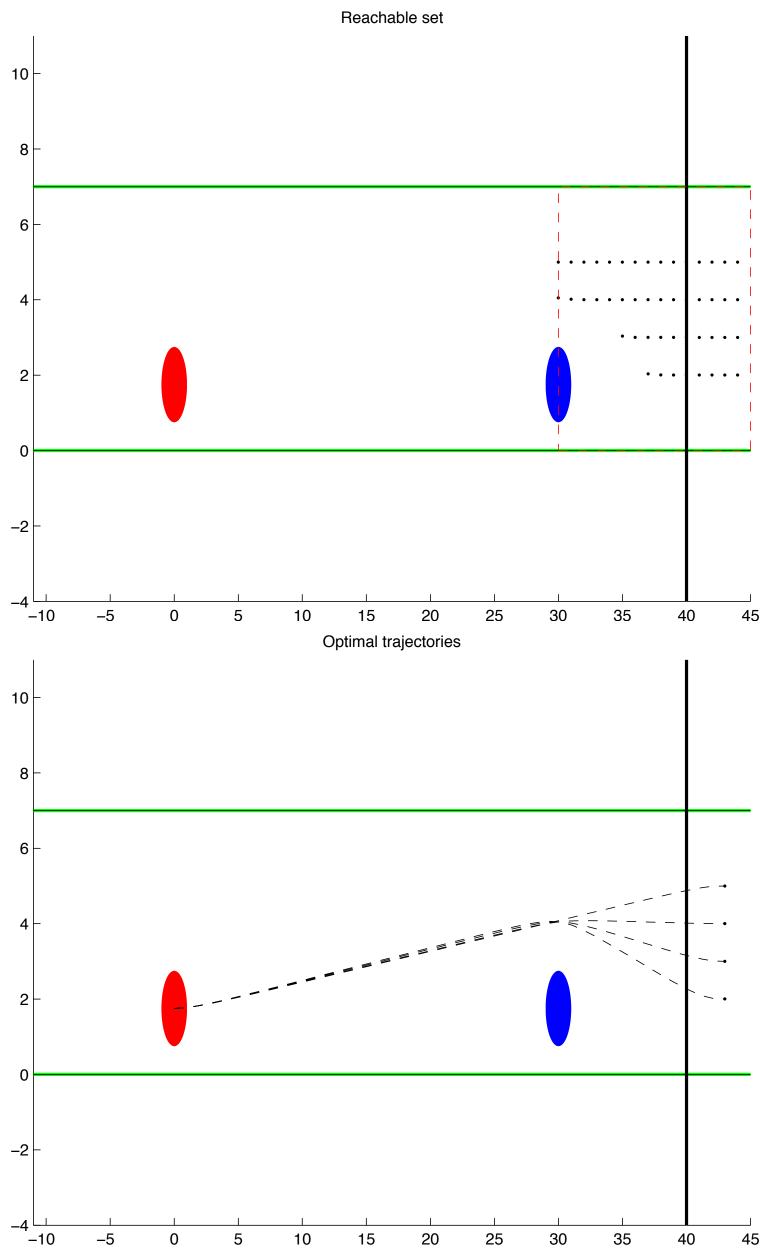


Figure 1.10: Figure to the top shows the reachable set, figure to the bottom refers to the reachable set with segment parallel to y-axis and its trajectory funnel.

1.3 Optimal trajectories for a curve scenario

This subsection is considering an overtaking maneuver of a moving obstacle in a curve road.

- The reference vehicle is a circle $\mathcal{V}(z(t),t) = B(X(z(t),t),\ell)$ of radius $\ell = 1[m]$ and center $X(z(t),t) = (x(t),y(t))$. It evolves with dynamics f describing a 7D single track model as in (1.21) of Chapter II, with

$$\begin{aligned} z &= (x, y, \psi, v_x, v_y, \omega_\psi, \delta) \text{ states}, \\ u &= (\omega_\psi, F_B) \text{ controls}, \\ z_0 &= (x_0, y_0, \psi_0, v_{x_0}, v_{y_0}, \omega_{\psi_0}, \delta_0) \\ &= (0.0, -25.25[m], 0.0, 70[km/h], 0.0, 0.0, 0.1[rad]) \\ &\quad \text{ initial states}, \end{aligned} \tag{1.11}$$

  The state z must obey condition in (1.29) of Chapter II.

- Target: $\Omega = \left\{z \in \mathbb{R}^{n_z} \mid z_2 \geq 0,\ z_3 = \frac{\pi}{2}\right\}$ according to (2.3) of Chapter II.

- Road: $\mathcal{K}_r = \left\{(x,y) \in \mathbb{R}^2 \,|\, (r_{curve}+0)^2 \leq x^2+y^2 \leq (r_{curve}+7)^2\right\}$, where $r_{curve} \in \mathbb{R}$ is the radius of the curved road and in this case it is equal to $20[m]$,as in (2.32) of Chapter II.

- Obstacle parameters: one circular obstacle

$$\mathcal{O}_1 = B(X_1(w,t),\ell_1), \tag{1.12}$$

  with radius $\ell_1 = 1.0$ centered at

$$X_1 = (x_1(w,t_0), y_1(w,t_0)) = (5.25\cos(w), 5.25\sin(w)), \tag{1.13}$$

  where $w \in \mathbb{R}$ is a parameter that needs to be optimized or which is fixed to the value $w = 0.57[rad]$. If w is optimized then it means that the initial distance between reference vehicle and obstacle is minimized. The obstacle set is defined as

$$\mathcal{K}_o(t) = \{(x,y) \in \mathbb{R}^2 \,|\, (x-x_1(w,t))^2 + (y-y_1(w,t))^2 \geq \ell_1^2\}, \tag{1.14}$$

  for each $t \in [t_0, t_f]$. The dependence on t of $\mathcal{O}_1$ means that the obstacle is moving with a circular motion described in (2.57) with $\omega_{10} = 5[rad/s]$ and $\alpha_{10} = -5[rad/s^2]$.

Using (2.70) of Chapter II, the state z has to satisfy property in (2.87) of Chapter II with

- state constraints defined for:

$$\mathcal{K} = \left\{(z,t) \in \mathbb{R}^{n_z} \times [t_0,t_f] \mid \underline{g} \leq \widetilde{g}(z,t) \leq \overline{g}, \underline{g}, \overline{g} \in \mathbb{R}^{n_{\widetilde{g}}}\right\}, \tag{1.15}$$

  with

$$\begin{aligned} \widetilde{g}(z,t) &:= \begin{pmatrix} z_1^2 + z_2^2 \\ (z_1 - x_1(w,t))^2 + (z_2 - y_1(w,t))^2 \end{pmatrix}, \\ \underline{g} &:= \begin{pmatrix} (0+\ell+\eta+r_{curve})^2 \\ (\ell_1+\ell+\eta)^2 \end{pmatrix}, \\ \overline{g} &:= \begin{pmatrix} (7-\ell-\eta+r_{curve})^2 \\ +\infty \end{pmatrix}, \end{aligned} \tag{1.16}$$

according to (2.35), (2.36), and (2.72) of Chapter II. Therein $n_z = 7$ is the state and dynamics dimension and $\eta = 0.3[m]$ is a safety margin.

- boundary constraints defined for:

$$\Omega = \left\{z \in \mathbb{R}^{n_z} \mid \underline{\varphi} \le \widetilde{\varphi}(z) \le \overline{\varphi}, \underline{\varphi}, \overline{\varphi} \in \mathbb{R}^{n_{\widetilde{\varphi}}}\right\}, \tag{1.17}$$

with

$$\widetilde{\varphi}(z(\tau)) := \begin{pmatrix} z_2 \\ z_3 \end{pmatrix}, \underline{\varphi} := \begin{pmatrix} 0 \\ \frac{\pi}{2} \end{pmatrix}, \overline{\varphi} := \begin{pmatrix} +\infty \\ \frac{\pi}{2} \end{pmatrix}, \tag{1.18}$$

according to (2.6) of Chapter II.

Thus the associated optimal control problem is given straightforwardly.

Problem 1.2. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with $t_0 < t_f$ fixed. Let $\widetilde{g}$ be a smooth function defined in* (1.16) *and let $\widetilde{\varphi}$ be a smooth function defined in* (1.18)*. Find states $z : \mathbb{R} \to \mathbb{R}^{n_z}$ absolutely continuous, controls $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u}$ meaurable.$\}$ and parameters $w \in W \subset \mathbb{R}^{n_w}$, $\tau \in [t_0, t_f]$ such that:*

$$\begin{array}{ll}
\min\limits_{z, u \in \mathcal{U}, w \in W, \tau \in [t_0, t_f]} & \varphi_0(z(\tau), \tau, w) + \int_{t_0}^{\tau} f_0(z(t), u(t)) dt \\
s.t. & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, \\
& z(t_0) = z_0, \\
& \underline{g} \le \widetilde{g}(z(t), t) \le \overline{g}, \quad \forall t \in [t_0, \tau], \quad \Longleftrightarrow \\
& \begin{cases} (20 + \ell + \eta)^2 \le z_1(t)^2 + z_2(t)^2 \le (27 - \ell - \eta)^2 \\ (z_1(t) - x_1(w, t))^2 + (z_2(t) - y_1(w, t))^2 \ge (\ell_1 + \ell + \eta)^2 \end{cases} \\
& \underline{\varphi} \le \widetilde{\varphi}(z(\tau)) \le \overline{\varphi}, \quad \Longleftrightarrow \\
& \begin{cases} z_2(\tau) \ge 0 \\ z_3(\tau) = \frac{\pi}{2} \end{cases}
\end{array} \tag{1.19}$$

where φ_0, f_0 are maps, $\overline{g}, \underline{g}, \overline{\varphi}, \underline{\varphi}$ are in (1.16) *and* (1.18)*. The function $f : [t_0, t_f] \times \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}^7$ is the single track dynamics given in* (1.21) *of Chapter II, with $z = (x, y, \psi, v_x, v_y, w_\psi, \delta), u = (w_\delta, F_B)$.*

The software used in this section requires smooth state constraints, this is why in Problem 1.2, condition in (2.86) of Chapter II is used over the equivalent conditions in (2.87), (2.85) of Chapter II.

In Figure 1.11 a minimum time trajectory (found by solving Problem 1.2 minimizing τ) and a minimum initial distance trajectory (found by solving Problem 1.2 minimizing $z_1(t_0) - x_1(w, t_0)$) are shown. For the minimum time trajectory (left column) the controls are less extreme and thus driver friendly. In the minimum initial distance trajectory (right column) the vehicle path seems to hit the obstacle, this however is not true if one thinks that the obstacle is moving forward.

Minimum final time and minimum initial distance.

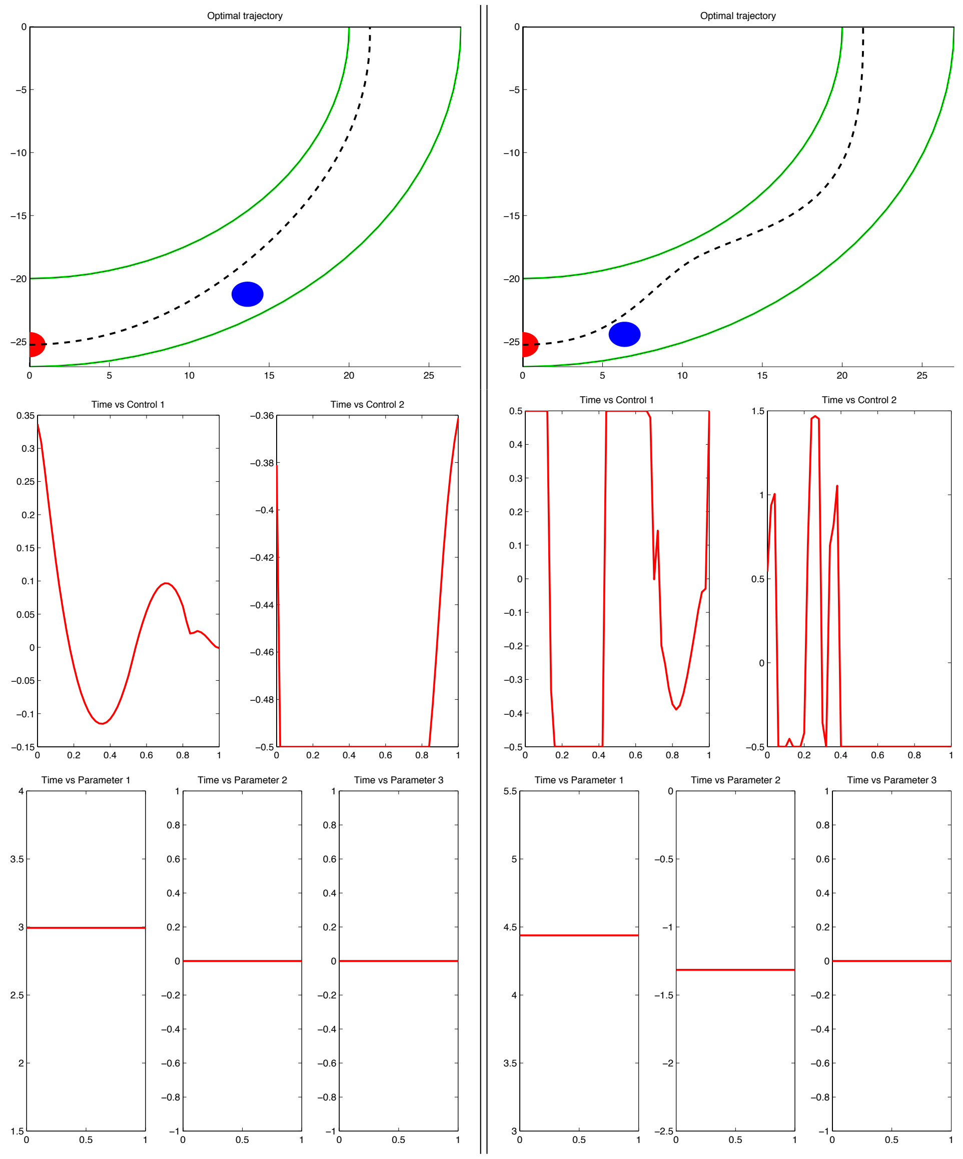


Figure 1.11: The top row contains the optimal trajectory, the middle row contains the controls (steering velocity and braking force), and the bottom row contains the parameters (final time, initial x-distance between vehicle and obstacle, constraint violation). In the left column the final time is minimized, in the right column the initial x-distance between vehicle and obstacle is minimized. CPU time: $0.05[s] - 0.07[s]$.

2 A software for solving the Hamilton-Jacobi equation

The search of a trajectory performed within some minimal time criteria and the set of all safety points from which the reference vehicle can drive a trajectory is here computed and numerical simulations for different scenarios are given. The backward reachable set $\mathcal{BR}^f_{t_f}$ associated to dynamics f with final time $\tau \in [t_0, t_f] \in \mathbb{R}$, is in Definition 3.4 Chapter II. It is the set of all initial states of a reference vehicle dynamics such that Property (2.87) of Chapter II holds. The approach used here to compute the backward reachable set and implemented in ROC-HJ software [16] is based on the work of [17] and it is recalled in Subection 2.2 of Chapter III. Throughout the section, the 4D point mass model (1.31) of Chapter II is used. In the present section ROC-HJ software is used for solving the Hamilton-Jacobi Equation (2.67) in Chapter III. A second order ENO finite difference scheme for partial differential equations of Hamilton-Jacobi type is implemented. More precisely an ENO2 spatial discretization method as described in [76] is used, coupled with an Euler forward RK1 scheme in time.

2.1 Case study: straight road with a fixed rectangular obstacle.

Let the following configuration be considered with a fixed obstacle:

- Target: $\Omega = \{(x, y, \psi),\ x \geq 0,\ |\psi| \leq 0.1\}$ according to (2.3) of Chapter II.
- Road: $\mathcal{K}_r = \left\{(x, y) \in \mathbb{R}^2 \,|\, -3.5 \leq y \leq 3.5\right\}$ as in (2.7) of Chapter II.
- Obstacle parameters: one fixed rectangular obstacle $\mathcal{O}_1$ of size $\ell_{1_x} = \ell_{1_y} = 1.0$ centered at $X_1 = (x_1, y_1) = (-10.0,\ -1.5)$.
- Reference vehicle evolves with dynamics f describing a 4D point mass model as in (1.31) of Chapter II, with

$$\begin{array}{rcl} z &=& (x, y, \psi, v) \text{ states}, \\ u &=& (\omega_\psi, F_B) \text{ controls}, \\ z_0 &=& (x_0, y_0, \psi_0, v_0) \\ &=& (-40.0[m], -1.5[m], 0.0, 35.0[m/s]) \text{ initial states}, \end{array} \tag{2.1}$$

  It is a rectangle $\mathcal{V}(z, t)$ of dimensions $\ell_{1_x} = \ell_{1_y} = 1.0[m]$ and center at $X(z, t) = (x(t), y(t))$.

Therefore the state z has to satisfy Property (2.87) of Chapter II, with

$$\begin{array}{rcl} \mathcal{K} &=& \{z \in \mathbb{R}^{n_z} \,|\, |z_2| \leq (3.5 - \ell - \eta), g_o(z) + \eta \leq 0\}, \\ \Omega &=& \{z \in \mathbb{R}^{n_z} \,|\, z_1 \geq 0,\ |z_3| \leq 0.1\}, \end{array} \tag{2.2}$$

therein g_o real-valued function defined by (2.80) of Chapter II, $n_z = 4$ is the state and dynamics dimension and $\eta = 0.3[m]$ is a safety margin. Thus the associated problem is given straightforwardly.

Problem 2.1. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed time points $t_0 = 0 < t_f$ and*

$$\begin{aligned} f &: [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}, \\ g &: \mathbb{R}^{n_z} \to \mathbb{R}, \\ \varphi &: \mathbb{R}^{n_z} \to \mathbb{R}, \end{aligned}$$

be maps. Find the absolutely continuous function $z^u_{z_0} : \mathbb{R} \to \mathbb{R}^{n_z}$, a control $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u} \text{ measurable}\}$ and a parameter $\tau \in [t_0, t_f]$ such that:

$$\begin{aligned} &\dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, \\ &z(t_0) = z_0, \\ &g(z(t)) := \max(|z_2(t)| - (3.5 - \ell - \eta), g_o(z(t))) \le 0, \\ &\varphi(z(\tau)) := \max(-z_1(\tau), |z_3(\tau)| - 0.1) \le 0, \end{aligned} \tag{2.3}$$

with f as in (1.31) of Chapter II, $z = (x, y, \psi, v)$, $u = (w_\psi, F_B)$, g_o a real-valued function defined by (2.80) in Chapter II, and τ is minimized.

The associate algorithm is 5. Minimal time trajectories are also computed by following Algorithm 1 of Chapter III.

```
input  : road geometry, obstacle data (geometry and motion), vehicle
         data (model, geometry, initial state $z_0$), target data (position
         and yaw angle).
output: $G \subset \mathbb{G}$

define a grid $\mathbb{G} = \{g_h\}_{h=1,\dots,k}, k > 0$ in the state space $\mathbb{R}^{n_z}$;
solve the HJ equation (2.67) in Chapter III associated to (2.3), the
solution is $v(z_0, t_f)$;
for $h=1,\dots,k$ do
    evaluate $v(g_h, \tau)$;
    if $v(g_h, \tau) \le 0$ then
        $G := G \cup \{g_h\}$;
    end
end
```

Algorithm 5: Backward reachable set with ROC-HJ software.

Hence, the reference vehicle (red rectangle) drives from left to right in Figure 2.1 and overtakes the fixed obstacle (blue rectangle). By considering relative velocities, this scenario is enough to model an overtaking maneuver of a slower moving car.

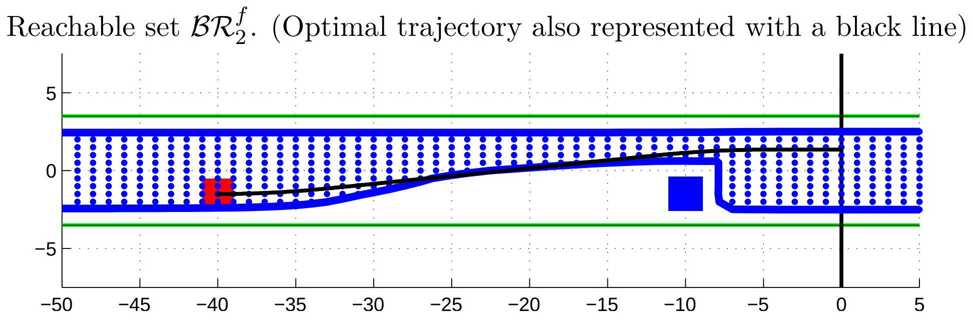


Figure 2.1: Reachable set $\mathcal{BR}_2^f$, obtained with $N_x = 70$ and $N_y = 8$ grid points.

The white part in Figure 2.1 corresponds to an area of starting points from which the red car cannot reach the target, whatever maneuver is undertaken, since its velocity is too high to avoid a collision with the blue car. Hence the trajectories from these points are infeasible and this means the starting points do not belong to the backward reachable set.

In Figure 2.2 the evolution in the time interval $[t_0, t_f] = [0, 2]$ of the $\mathcal{BR}_2^f$ set is shown. For each plot $i = 1, \ldots, 4$, starting from the blue points the reference vehicle can reach the target avoiding the obstacle within t_i seconds.

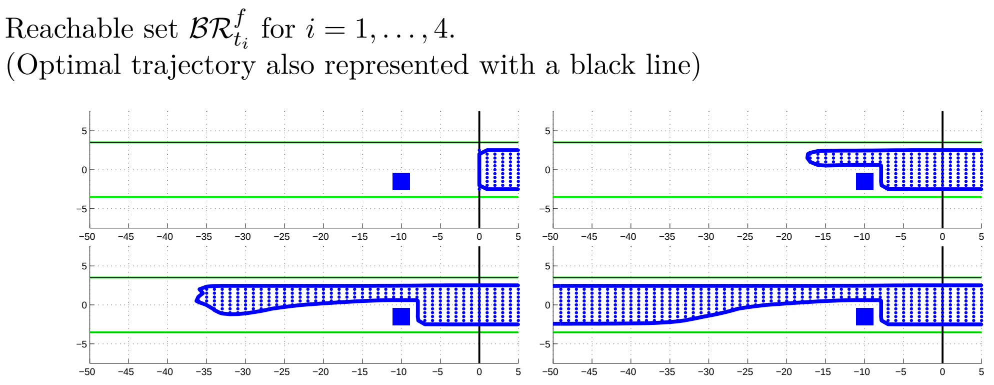


Figure 2.2: Evolution in time ($t_1 = 0[s], t_2 = 0.5[s], t_3 = 1[s], t_4 = 1.5[s]$) of the reachable set $\mathcal{BR}_2^f$, obtained with $N_x = 70$ and $N_y = 8$ grid points.

Convergence test

For testing the stability of the HJ approach, a convergence analysis with respect to mesh grid refinement is given. Let grid on the state space be defined and

$$(x, y, \psi, v) \in [-50, 10] \times [-4, 4] \times [-1, 1] \times [5, 65], \tag{2.4}$$

using a variable number of grid points in the (x, y) variables given by $N_x = 35 \cdot 2^m$ and $N_y = 4 \cdot 2^m$, depending on an integer parameter $m \in \{1, \dots, 5\}$.

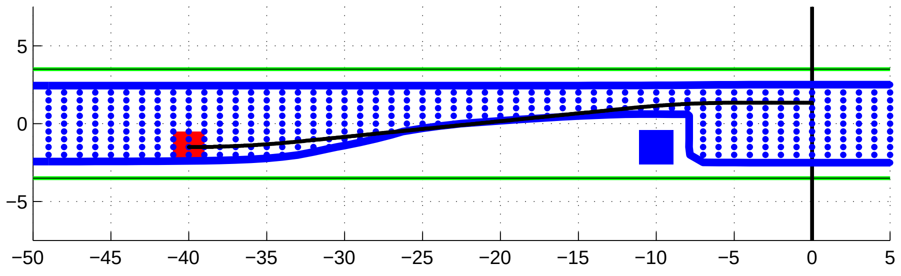


Figure 2.3: Reachable set $\mathcal{BR}_2^f$, and optimal trajectory represented with a black line.

The number of grid points in the ψ and v variables are fixed and given by

$$N_\psi = 20 \quad \text{and} \quad N_v = 6. \tag{2.5}$$

The errors are computed by using a reference value function v_{ex} obtained for $m = 5$ (i.e., $N_x = 1020$, $N_y = 128$). Furthermore a CFL restriction of the type $\Delta t/\Delta x \leq$ *const* is used for the stability of the scheme. The results are given in Table 2.1. The L^∞, L^1 and L^2 errors have been computed as follows. For a given grid mesh (z_i) and corresponding (constant) step space Δz, with $e_i := v(z_i) - v_{ex}(z_i)$ denoting the local error at grid point z_i:

$$e_{L^\infty} := \max_i |e_i|, \quad e_{L^1} := \Delta z \ \textstyle\sum_i |e_i|, \quad e_{L^2} := \left(\Delta z \ \sum_i e_i^2\right)^{1/2}. \tag{2.6}$$

In Table 2.1, in order to evaluate numerically the order of convergence for a given L^p norm, the estimate $\alpha_m := \frac{\log(e^{(m-1)}/e^{(m)})}{\log(2)}$ is used for corresponding values $N_x = 35 \cdot 2^m$ and $N_y = 4 \cdot 2^m$. (i.e. the mesh steps N_x and N_y are refined by 2 between two successive computations).

A convergence of order roughly 2 is observed even for the ENO2-RK1 scheme which in principle is only first order in time. This is due to the fact that the dynamics is close to a linear one in this case (a similar RK2 scheme was also tested, second order in time, which gives similar convergence results on this example).

N_x	N_y	e_{L^∞}	order	e_{L^1}	order	e_{L^2}	order	CPU time (s.)
70	8	0.489	–	1.126	–	6.769	–	0.34
140	16	0.078	2.64	0.337	1.73	2.255	1.58	1.50
280	32	0.026	1.60	0.118	1.52	0.795	1.50	9.20
560	64	0.006	2.07	0.030	1.98	0.207	1.94	69.40

Table 2.1: Error table for varying (N_x, N_y) parameters.

Comparison with a direct method

In order to validate the HJB approach for this scenario, comparison of the results with numerical simulations obtained by using a direct optimal control approach for calculating the reachable set ([9, 10]) are needed. The simulations are obtained by using the OCPID-DAE1 Software [48], following the approach described in [52, 105]. The resulting backward reachable set within time $t_f = 2$ is plotted in Figure 2.4 (top picture) and shows a good correspondence with the HJ approach (backward reachable set in the bottom picture). Notice that both computed reachable sets are expected to be equal up to some accuracy of the order $O(\Delta x)$ (with $\Delta x = 1$ in this figure).

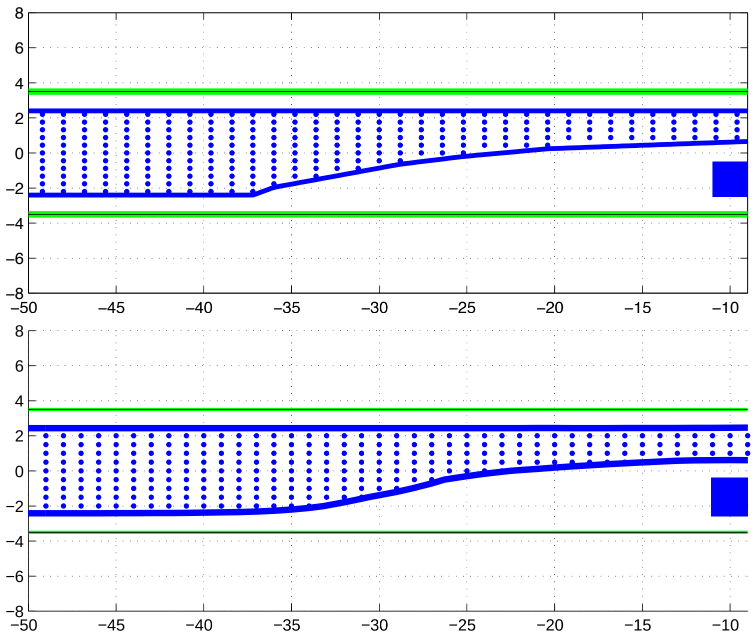


Figure 2.4: Comparison with a direct method (upper picture).

Remark 2.2. *The advantage of using a direct optimal control approach (like the OCPID-DAE1 Software) is that is able to deal with a greater number of states variables, which is necessary whenever a precise car-model is required, close to the behavior of a real car. However the handling of state constraints (in particular obstacles with non-smooth boundaries) sometimes leads to numerical difficulties in order to get feasible trajectories. On the other hand the PDE solver (like ROC-HJ for solving Hamilton-Jacobi equations) is limited, in practice, by the number of state variables, because it requires to solve a PDE with as many dimensions as the number of state variables. However if the dimension can be processed on a given architecture, then the HJ approach requests only the Lipschitz property of the functions describing the dynamics and the state constraints. In particular, there is no problem for dealing with non smooth obstacles such as rectangular obstacles,* *crossing roads scenarios (more generally it could handle polygonal roads or more complex polygonal obstacles).*

Computational considerations

The ROCHJ software is easily dealing with rectangular obstacles and handling many different road geometry due to the very few regularity assumptions made on the functions describing such objects. Indeed only Lipschitz continuity of the state and boundary contraints is requested. This gives the possibility to model the car traffic scenario adherent to the complexity of a real world scenario, leading to correct avoidance trajectories. The limitations of such software are based on the architectures used because the method cannot handle high dimension in the state space. Problems with 5 and 6 states as the 6D single track model have been implemented but they request special techniques to be solved by the computer. For instance, parallelization is extremely useful in these case and some numerical simulations were extremely promising. Another possibility was investigated by combining a computation on multiple threads (depending on the available computer) with a "clever" choice of a state grid, by choosing the least possible number of grid points such that critical state points are considered. For instance the (x, y)-grid has to include the boundaries of the obstacles and of the road, the velocity grid has to meet the projection value and the angle grid can be composed only by three grid points. With the architecture used for the simulations of this Chapter, a MacBookPro computer with two threads, such high dimensional computations were extremely costly.

2.2 Examples for more complex scenarios

In the next examples several road geometries are considered and discussion on interpretation of backward reachable set is explained. For all the examples the avoidance condition between obstacles and vehicle is given by in (2.80) of Chapter II since all objects here are rectangles.

- **Scenario 1: straight road with varying width.** A highway road is considered, with varying width described by the level set (2.18) of Chapter II with function (2.22) of Chapter II. This can be interpreted as an additional exit lane appearing only for a short part of the considered road.

  Two obstacles (blue rectangles) are moving with linear motion (see (2.56) of Chapter II) in the same direction as the reference vehicle (red rectangle). All object widths and lengths are here equal to $1[m]$. The set of blue points depicted in Figure 2.5 and Figure 2.6 is the projection on the (x, y) plane of the backward reachable set with $t_f = 2[s]$, steering angle $\psi(t_0) = 0[rad]$ and velocity $v(t_0) = 35[m/s]$.

The target set is defined according to (2.3), (2.4) of Chapter II:

$$\Omega = \left\{ z \in \mathbb{R}^4 \mid \varphi(z) \leq 0 \right\}, \tag{2.7}$$

with

$$\varphi(z) = \max\left\{-z_1, |z_3| - 0.01\right\}. \tag{2.8}$$

The optimal control problem is given in 2.3.

Problem 2.3. *Let* $[t_0, t_f] \subset \mathbb{R}$ *be a non-empty and bounded interval with fixed time points* $t_0 = 0 < t_f$ *and*

$$\begin{aligned} f &: [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}, \\ g &: [t_0, t_f] \times \mathbb{R}^{n_z} \to \mathbb{R}, \\ \varphi &: \mathbb{R}^{n_z} \to \mathbb{R}, \end{aligned}$$

be maps. Find the absolutely continuous function $z^u_{z_0} : \mathbb{R} \to \mathbb{R}^{n_z}$*, a control* $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u}$ *measurable*$\}$ *and a parameter* $\tau \in [t_0, t_f]$ *such that:*

$$\begin{aligned} &\dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, \\ &z(t_0) = z_0, \\ &g(t, z(t)) := \max\left\{g_r(t, z(t)), g_{o1}(t, z(t)), \ldots, g_{ok}(t, z(t))\right\} \leq 0, \\ &\varphi(z(\tau)) \leq 0, \ \textit{with } \varphi \textit{ defined in } (2.8), \end{aligned} \tag{2.9}$$

with f *as in* (1.31) *of Chapter II,* $z = (x, y, \psi, v)$*,* $u = (w_\psi, F_B)$*,* $g_{oi}, i = 1, \ldots, k$ *are real-valued functions defined by* (2.80) *of Chapter II,* k *is the number of obstacles, and* g_r *is a real-valued function defined by* (2.22) *of Chapter II.*

Scenario 1a: (see Figure 2.5.) In this example an overtaking maneuver is considered with one first (obstacle) car in front of the vehicle, moving forward with velocity $10[m/s]$ and to be overtaken, and a second (obstacle) car next to the reference vehicle also moving forward at velocity $20[m/s]$ and blocking the maneuver. In Figure 2.5, it is the position of the vehicle and of the obstacle cars that is shown at initial time $t_0 = 0$. More precisely the parameters used for this Figure are $(N_x, N_y) = (70, 12)$ grid points; the trajectory (black line) is starting from $(x(0), y(0)) = (-40.0, -1.5)$, $\psi(0) = 0$ and $v(0) = 35[m/s]$, a first obstacle car takes initial values $(x(0), y(0)) = (-10, -1.5)$, with $\psi(0) = 0$ and a constant velocity $v(0) = 10[m/s]$; a second obstacle car takes initial values $(x(0), y(0)) = (-40, 1.5)$, with $\psi(0) = 0$ and a constant velocity $v(0) = 20[m/s]$.

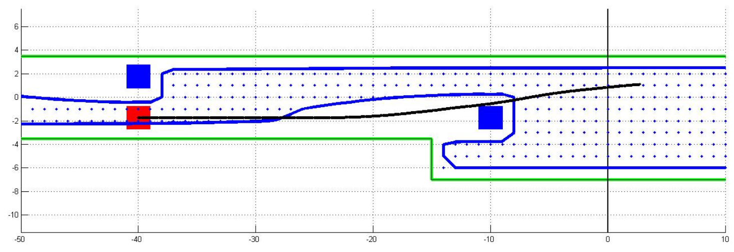


Figure 2.5: (Scenario 2a) Reachable set $\mathcal{BR}_2$.

Scenario 1b: (see Figure 2.6.) This example is similar to the previous one excepted for the fact that the second (obstacle) car is next to the first one at initial time. (More precisely, the second obstacle car now takes initial values $(x(0), y(0)) = (-10, 1.5)$, and other parameters are otherwise unchanged).

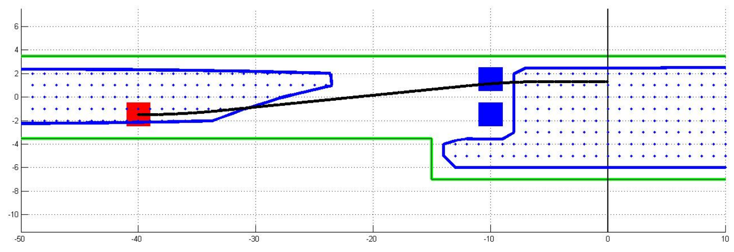


Figure 2.6: (Scenario 1b) The reachable set $\mathcal{BR}_2$ is non connected to the contrary to Scenario 1a.

The backward reachable set is the set of initial points of $\mathbb{R}^4$ (according to the model (1.31) of Chapter II) for which the collision can be avoided avoidable, and target can be reached, within t_f seconds (therefore so that is satisfied the final conditions $\exists \tau \in [t_0, t_f]$ such that $x(\tau) \geq 0$ and $\psi(\tau) = 0$). The set of blue points depicted in Figure 2.6 and Figure 2.5 is the projection on the (x, y) plane of the backward reachable set where the steering angle and the velocity are fixed to the values $\psi(t_0) = 0[rad]$ and $v(t_0) = 35[m/s]$ (which correspond to the initial values of the car when maneuver starts).

By starting in the blue region, the vehicle (in red) might avoid a collision. On the other hand, starting from a point in the white area then the vehicle will either go outside the road or will collide with the obstacle, before being able to reach the safety region Ω. In both figures an extra part of the backward reachable set that lay below second obstacle. This is due to the varying road width, and means that the vehicle is forced to stay on the road. In Figure 2.6 the backward reachable set is not connected, which means that the red rectangle can avoid a collision by starting the maneuver either leaving the obstacles behind (since it is faster no crash will occur) or from a sufficiently high distance behind the two obstacles depending on its y position (about $24[m]$ if the reference vehicle starts in the first lane and $14[m]$ if it starts in the second lane). The optimal trajectories (black line) seem to overlap the obstacles and their trajectories before reaching the target set. This is because the blue rectangles only show the initial position of the obstacles at time $t_0 = 0$, and not the evolution of their linear motion in the time interval $[0, t_f]$.

- **Scenario 2: curved road with fix or moving obstacles.** The road shape is now described by the level set in (2.29) of Chapter II with function (2.31) of Chapter II, with a $7[m]$ width and a road radius of $50[m]$. The target set is again the level set (2.7) with function (2.8).

  The optimal control problem is given in 2.4.

  Problem 2.4. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed time points $t_0 = 0 < t_f$ and*

$$\begin{aligned} f &: [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}, \\ g &: [t_0, t_f] \times \mathbb{R}^{n_z} \to \mathbb{R}, \\ \varphi &: \mathbb{R}^{n_z} \to \mathbb{R}, \end{aligned}$$

  be maps. Find the absolutely continuous function $z_{z_0}^u : \mathbb{R} \to \mathbb{R}^{n_z}$, a control $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u} \text{ measurable}\}$ and a parameter $\tau \in [t_0, t_f]$ such that:

$$\begin{aligned} &\dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, \\ &z(t_0) = z_0, \\ &g(t, z(t)) := \max\{g_r(t, z(t)), g_{o1}(t, z(t)), \ldots, g_{ok}(t, z(t))\} \le 0, \\ &\varphi(z(t_f)) \le 0, \ \textit{with } \varphi \textit{ defined in } (2.8), \end{aligned} \tag{2.10}$$

  with f as in (1.31) of Chapter II, $z = (x, y, \psi, v)$, $u = (w_\psi, F_B)$, $g_{oi}, i = 1, \ldots, k$ are real-valued functions defined by (2.80) in Chapter II, k is the number of obstacles, and g_r is a real-valued function defined by (2.31) in Chapter II.

  Scenario 2a: Two fixed obstacles (blue rectangles with different width and lenght parameters) have to be avoided by the vehicle (red square), see Figure 2.7, and the target Ω has to be reached, if possible, in the time interval $[0, 5]$ seconds. The first obstacle has dimensions $0.5[m]$ and is positioned in

$(-5, 48.25)$. The second obstacle has width $0.5[m]$ and length $1[m]$, its position is $(-25, 45)$.

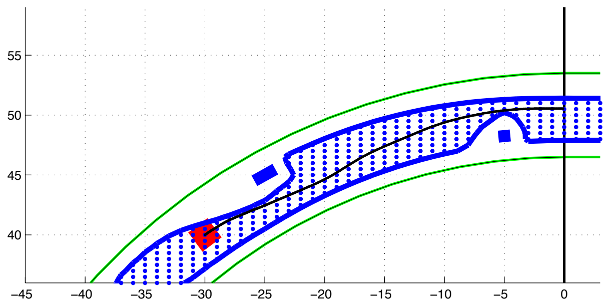


Figure 2.7: (Scenario 2a) Reachable set $\mathcal{BR}_5$ for a curved road with two fixed obstacles.

Scenario 2b: In this scenario depicted in Figure 2.8, the road parameters are similar, there is now only one obstacle but that is furthermore moving with a circular motion at speed of $5[m/s]$.

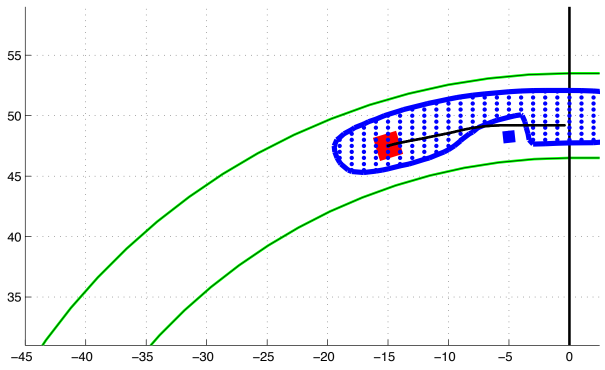


Figure 2.8: (Scenario 2b) Reachable set $\mathcal{BR}^f$ for a curved road with one moving obstacle.

- **Scenario 3: crossing road and moving obstacles.** The following scenario involves a crossing described by the level set in (2.23) of Chapter II, with function (2.24) of Chapter II. The width of the four streets involving a crossing can be different one from each other. Here, the horizontal lower and upper road bounds and the vertical limits are different, as illustrated in Figure 2.9. An object (blue rectangle) of dimensions $1[m]$ is traveling from left to right from position $(-10.0, -2.0)$ meters with speed $5[m/s]$ and deceleration $5[m/s^2]$. A second obstacle (length $1.0[m]$ and width $2.0[m]$) starting from position $(-18, 4)$ is traveling from top to bottom with speed $5[m/s]$ and deceleration $5[m/s^2]$. Within time $t_f = 2.5[s]$ the red square of dimensions $1.0[m]$ has to reach one of the three targets at the end of each road: top (with steering angle

$\frac{\pi}{2}[rad]$), bottom (with steering angle $-\frac{\pi}{2}[rad]$) or right (with steering angle $0[rad]$). Therefore, the level set (2.7) has function:

$$\begin{aligned}\varphi(z(\tau)) \quad = \quad \max\Big\{ & -x(\tau), \min\{|\psi(\tau)-\tfrac{\pi}{2}|, |\psi(\tau)+\tfrac{\pi}{2}|, \\ & |\psi(\tau)|\} - 0.01[rad]\Big\},\end{aligned} \tag{2.11}$$

$\tau \in [t_0, t_f]$. At this speed and for an initial position close to the center of the crossing, the red vehicle is able to leave the crossing before the second obstacle enters the center of the crossing. In this example the optimal trajectory (black line) will steer to avoid the obstacle in the front and will also decelerate to avoid the second obstacle.

The optimal control problem is given in 2.5.

Problem 2.5. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed time points $t_0 = 0 < t_f$ and*

$$\begin{aligned} f &: [t_0, t_f] \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}, \\ g &: [t_0, t_f] \times \mathbb{R}^{n_z} \to \mathbb{R}, \\ \varphi &: \mathbb{R}^{n_z} \to \mathbb{R}, \end{aligned}$$

be maps. Find the absolutely continuous function $z^u_{z_0} : \mathbb{R} \to \mathbb{R}^{n_z}$, a control $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u}$ measurable$\}$ and a parameter $\tau \in [t_0, t_f]$ such that:

$$\begin{aligned} & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, \\ & z(t_0) = z_0, \\ & g(t, z(t)) := \max\{g_r(t, z(t)), g_{o1}(t, z(t)), \dots, g_{ok}(t, z(t))\} \le 0, \\ & \varphi(z(\tau)) \le 0, \ \textit{with } \varphi \textit{ defined in } (2.11), \end{aligned} \tag{2.12}$$

with f as in (1.31) of Chapter II, $z = (x, y, \psi, v)$, $u = (w_\psi, F_B)$, $g_{oi}, i = 1, \dots, k$ are real-valued functions defined by (2.80) of Chapter II, k is the number of obstacles, and g_r is a real-valued function defined by (2.24) of Chapter II.

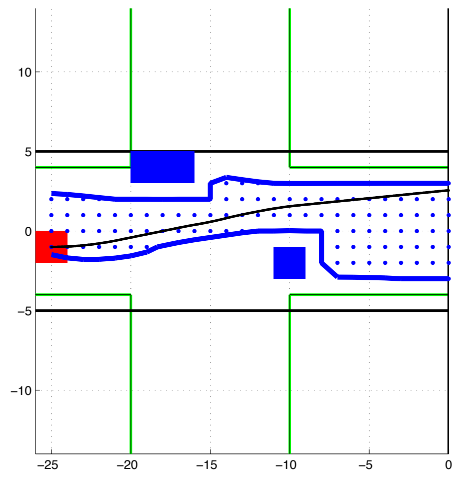


Figure 2.9: (Scenario 3) Reachable set $\mathcal{BR}_{2.5}$ for a crossing with one fixed and one moving obstacle.

3 Sensitivity analysis modeling errors in initial data detection

In this section perturbations in the initial data z_0 enter the Optimal Control Problem 3.1 of Chapter II. The aim is to model sensor errors in measurements of the initial state and to perform a sensitivity analysis for this specific problem to study the influence of parameters on the solution (trajectory and controls) and on the reachable set. Therefore Problem 4.1 of Chapter II, called perturbed optimal control problem is studied. Two different approaches will be considered, Fiacco-Sentitivity, ODE-Sensitivity, as presented in Section 4 of Chapter II. By using a computed nominal optimal trajectory, the Fiacco-Sensitivity tells about changes in the nominal trajectory if the initial values are perturbed in a neighborhood of the nominal initial values. The ODE-Sensitivity is showing how the trajectory changes if the nominal trajectory is driven but with wrong information in the initial data (Section 3.1). In Section 3.2 such concepts will be extended to reachable sets and trajectory funnels. Moreover a maximum error estimation is given in Section 3.3. It is useful to understand the requirements for sensor precision such that the nominal trajectory is still satisfying the state constraints.

3.1 Sensitivity in optimal trajectories

Fiacco-Sensitivity. The first approach called **Fiacco-Sensitivity** is based on a parametric sensitivity analysis of the optimal solution for the Optimal Control

Problem 4.1 of Chapter II with respect to p, compare [41, Sections 3.2 and 4.2]. To this end let $\hat{u} = \hat{u}(\hat{p})$ and $\hat{z} := z_{z_0(\hat{p})}^{\hat{u}(\hat{p})}$ denote the optimal solution for the Optimal Control Problem 4.1 of Chapter II for a nominal parameter $\hat{p}$. The map z defined in (4.2) of Chapter II assignes to $(\hat{u}(\hat{p}), \hat{p})$ the solution $z_{z_0(\hat{p})}^{\hat{u}(\hat{p})}$. Therefore, the Fiacco-Sensitivities of the state and the control are defined as

$$S_F(\cdot) := \frac{dz}{dp}(\hat{u}, \hat{p})(\cdot) = \frac{\partial z}{\partial u}(\hat{u}, \hat{p})(\cdot)\frac{d\hat{u}}{dp}(\hat{p})(\cdot) + \frac{\partial z}{\partial p}(\hat{u}, \hat{p})(\cdot) \quad \text{and} \quad \frac{d\hat{u}}{dp}(\hat{p})(\cdot). \tag{3.1}$$

These sensitivities can be computed using the linearized necessary Karush-Kuhn-Tucker conditions in an optimal solution $(\hat{z}, \hat{u})$. An approximation to the optimal perturbed trajectory is given by

$$z(\hat{u}(p), p)(\cdot) \approx \hat{z}(\cdot) + S_F(\cdot)(p - \hat{p}). \tag{3.2}$$

```
input : $(\hat{z}(t_k), \hat{u}(t_k)), \quad \forall t_k \in [t_0, \tau] \subset \mathbb{R}$,
        $\hat{p}, z_0, r, p$,
        $\xi := z_0(p) \in B(z_0, r)$.
output: $\tilde{z}$
for $t_k \in [t_0, \tau]$ do
    solve system (3.1) and get matrix $S_F(t_k)$;
    $\tilde{z}(t_k) := \hat{z}(t_k) + S_F(t_k)(\hat{p} - p)$
end
```

Algorithm 6: Fiacco-Sensitivity implemented in OCPID-DAE1.

An example of Fiacco-perturbed trajectories according to (3.1) with respect to parameter $p_i, i = 1, \ldots, 5$, is presented in Figure 3.1. The reference trajectory is the minimal time trajectory in the left column of Figure 1.1.

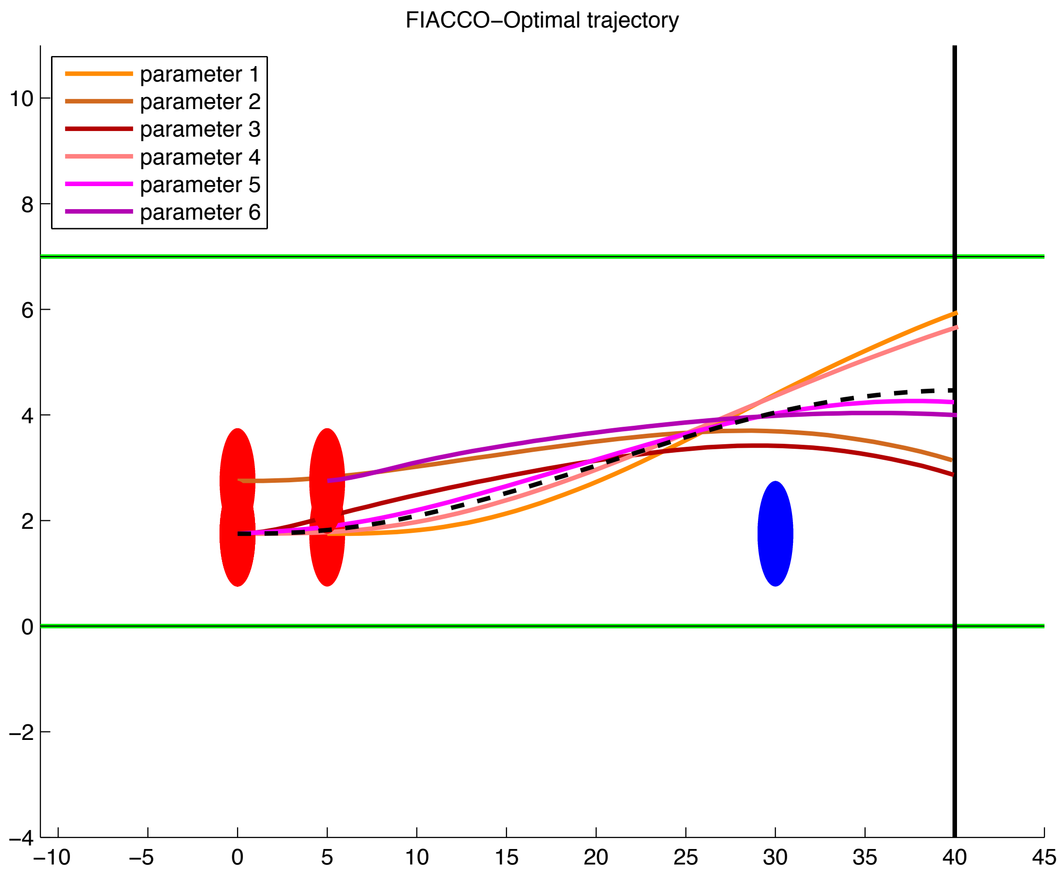


Figure 3.1: The seven trajectories show the nominal trajectory (black dashed line) and its perturbation with respect to $p_1, \dots, p_5$ and w.r.t. all parameters combined (dark magenta line). The perturbation vector is $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$. The solutions (states and controls) or the perturbed optimal control problem for such perturbations of the initial values are well approximated by the Fiacco-Sentitivity which gives feasible trajectories.

ODE-Sensitivity. The second approach called **ODE-Sensitivity** investigates the dependence of the solution of the initial value problem in (4.3) of Chapter II on p in a ball of the nominal parameter $\hat{p}$ and for a fixed (optimal) control $\hat{u}$. To this end let $\hat{u} = \hat{u}(\hat{p})$ be given and let $\hat{z} := z_{z_0(\hat{p})}^{\hat{u}(\hat{p})}$ denote the corresponding solution of the initial value problem

$$z'(t) = f(z(t), \hat{u}(t)), \quad z(0) = z_0(\hat{p}). \tag{3.3}$$

Again the map z defined in (4.2) of Chapter II assignes to $(\hat{u}(\hat{p}), \hat{p})$ the solution $z_{z_0(\hat{p})}^{\hat{u}(\hat{p})}$. Then, the ODE-Sensitivity of the state is defined as

$$S_O(\cdot) := \frac{\partial z}{\partial p}(\hat{u}, \hat{p})(\cdot). \tag{3.4}$$

Note that this is just the partial derivative of the state mapping with respect to p for a fixed control and not the total derivative as in (3.1). An approximation to the perturbed trajectory is obtained similar as in (3.2). The ODE-Sensitivity is given by solving the sensitivity differential equation

$$S_O'(t) = f_z'(\hat{z}(t), \hat{u}(t))S_O(t), \qquad S_O(0) = \frac{dz_0}{dp}(p). \tag{3.5}$$

```
input  : $(\hat{z}(t_k), \hat{u}(t_k)), \quad \forall t_k \in [t_0, \tau] \subset \mathbb{R}$,
         $\hat{p}, z_0, r, p$,
         $\xi := z_0(p) \in B(z_0, r)$.
output: $\tilde{z}$
for $t_k \in [t_0, \tau]$ do
    solve system (3.5) and get matrix $S_O(t_k)$;
    $\tilde{z}(t_k) := \hat{z}(t_k) + S_O(t_k)(\hat{p} - p)$
end
```

Algorithm 7: ODE-Sensitivity implemented in OCPID-DAE1.

An example of ODE-perturbed trajectories with respect to each parameter $p_i, i = 1, \ldots, 5$, is illustrated in Figure 3.2. The reference trajectory is the minimal time trajectory in the left column of Figure 1.1.

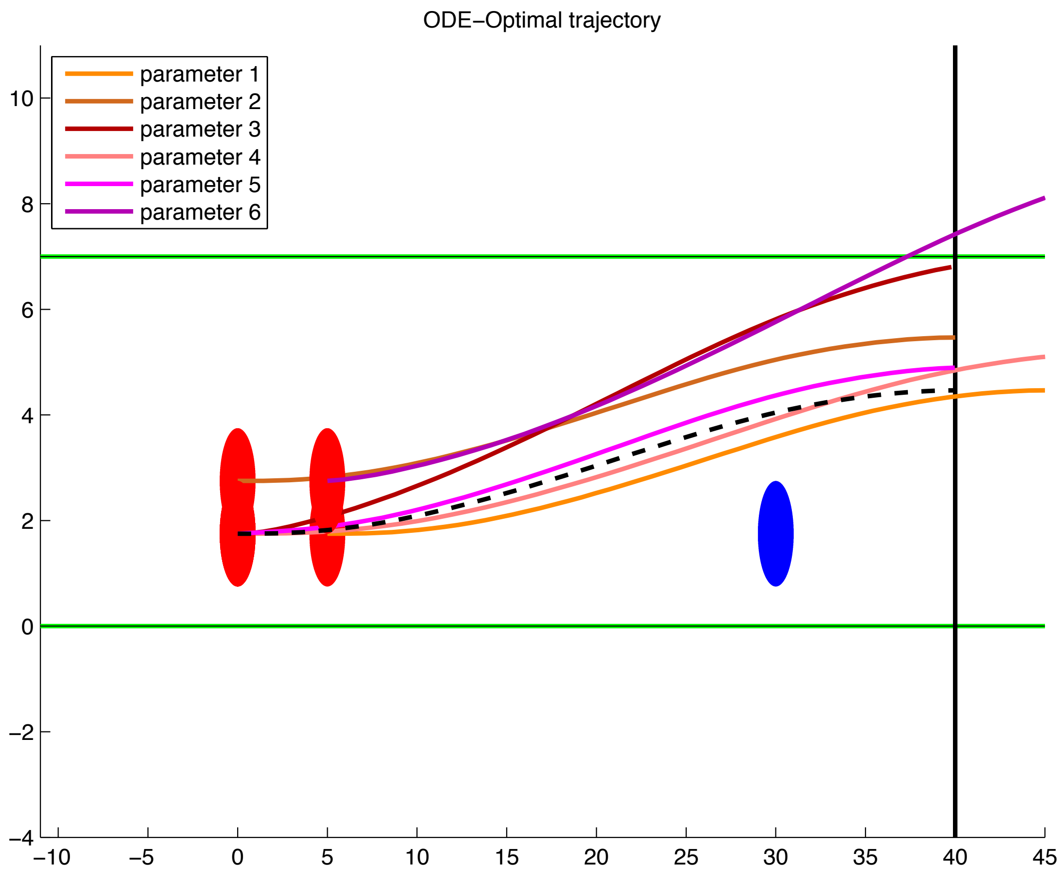


Figure 3.2: The seven trajectories show the nominal trajectory (dashed black line) and its perturbation with respect to $p_1, \ldots, p_5$ and w.r.t. all parameters combined (dark magenta line). The perturbation vector is $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$. Perturbations on x_0 and v_{x_0} values are too big since they lead to a crash. Perturbation in the yaw angle ψ_0 will bring the vehicle out of the road.

The ODE-Sensitivity analysis can be extended from a single perturbation vector $p \in \mathbb{R}^{n_p}$ to a range of perturbations in a ball around the nominal initial state $z_0(\hat{p})$ of radius $r > 0$. Algorithm 8 is using ODE-Sensitivity analysis to compute such trajectories.

```
input  : $(\hat{z}(t_k), \hat{u}(t_k)) \quad \forall t_k \in [t_0, \tau] \subset \mathbb{R}$,
         $\hat{p}, z_0, r$.
output: $\tilde{Z}$ matrix.
define a grid $P := \{p_i\}_{i=1,\dots,h}$ such that $\xi := z_0(p_i) \in B(z_0, r)$;
for $t_k \in [t_0, \tau]$ do
    solve system (3.5) and get matrix $S(t_k)$;
    for $i = 1, \dots, h$ do
        $\tilde{z}_i(t_k) := \hat{z}(t_k) + S(t_k)(\hat{p} - p_i)$;
    end
    $\tilde{Z}(t_k) := (z_1(t_k), \dots, z_h(t_k))$
end
```

Algorithm 8: Perturbed trajectory for a ball around a nominal initial state implemented in MATLAB code.

The perturbed trajectory related to the minimum constraint violation trajectory in the left column of Figure 1.2 is given in Figure 3.3. The perturbed trajectory related to the minimum time trajectory in the left column of Figure 1.1 is given in Figure 3.4. The perturbed trajectories are computed for each entry of the parameter p and for a perturbation taking into account all parameters together. The perturbation intervals are $p_1 \in [-5,\ 5]$ meters, $p_2 \in [-1,\ 1]$ meters, $p_3 \in [-0.1,\ 0.1]$ radiants, $p_4 \in [-5,\ 5]$ meters/seconds and $p_5 \in [-0.5, 0.5]$ meters/seconds. The two figures do not show particular difference, and in both cases such perturbation intervals are too big and lead to a crash.

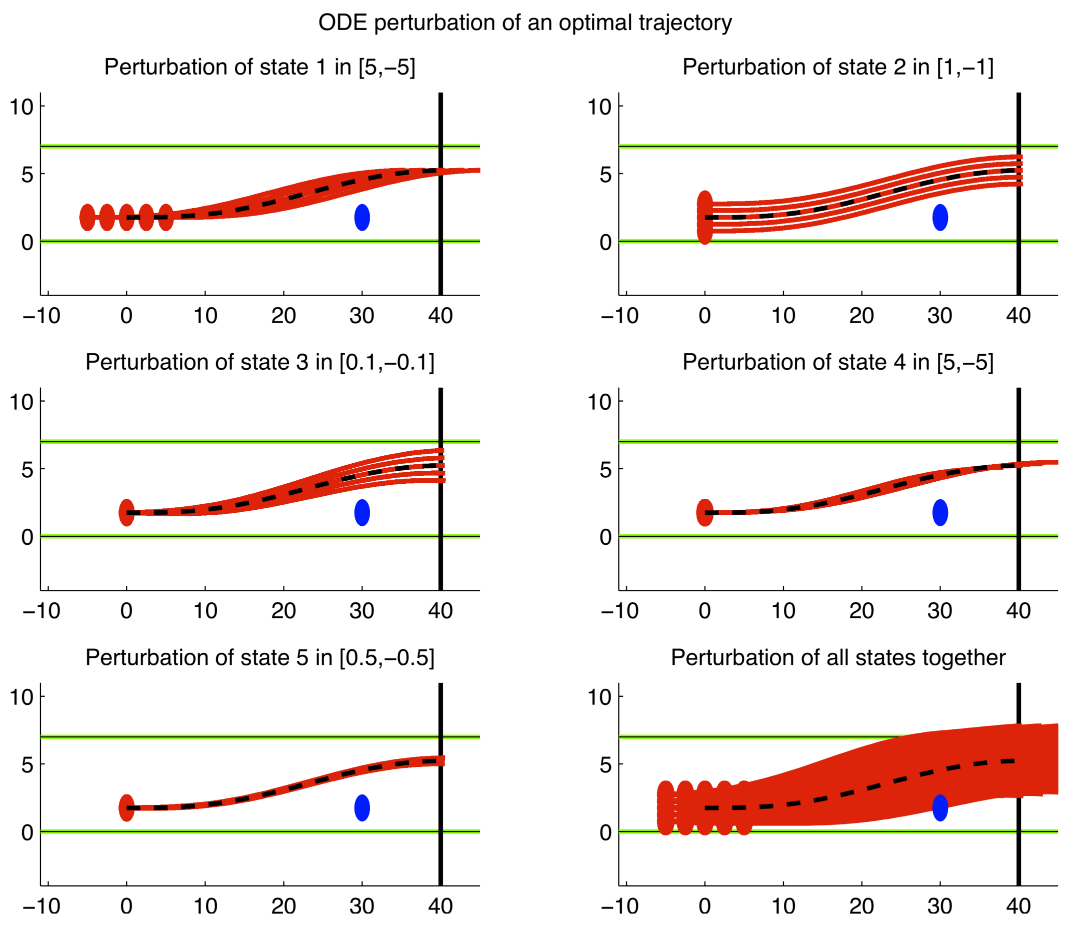


Figure 3.3: Perturbations in the first five initial states are shown in the figure for a minimum constraint violation trajectory. The perturbation value is discretized over the interval shown in the title of each figure to understand the shape of the perturbation set in such interval.

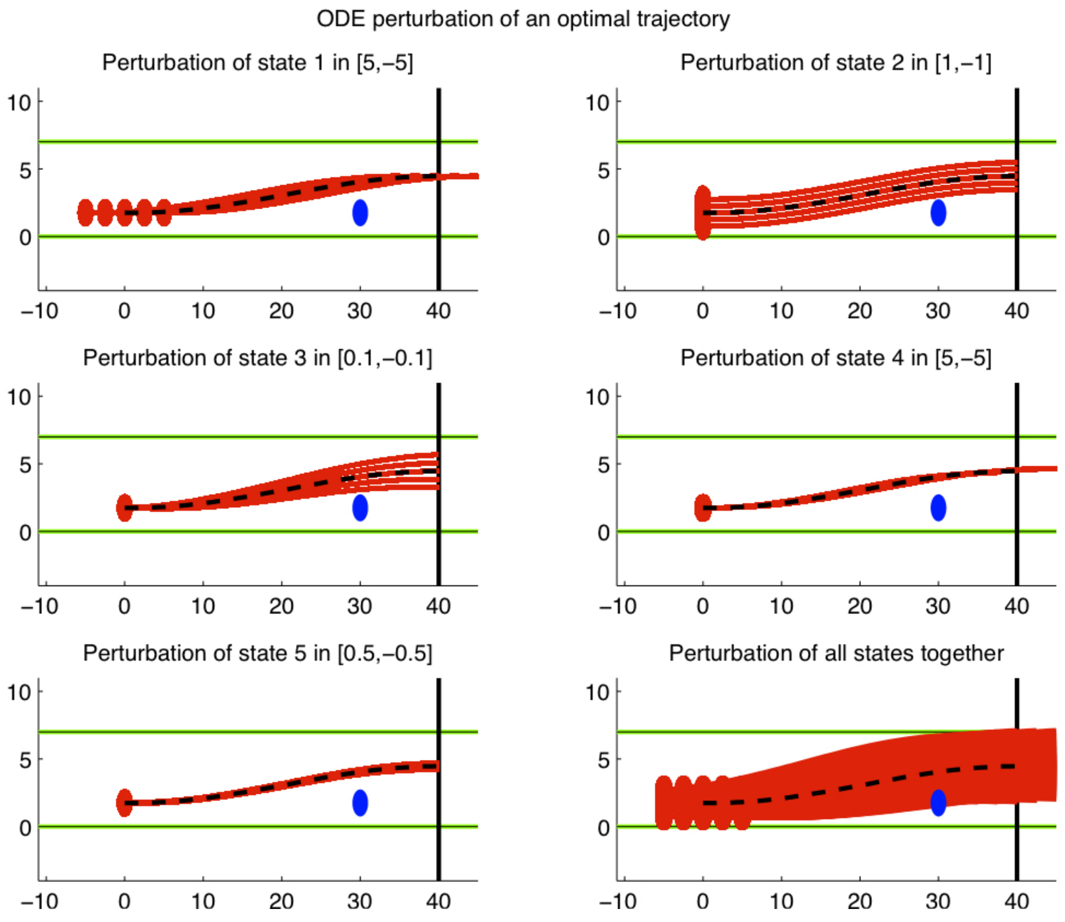
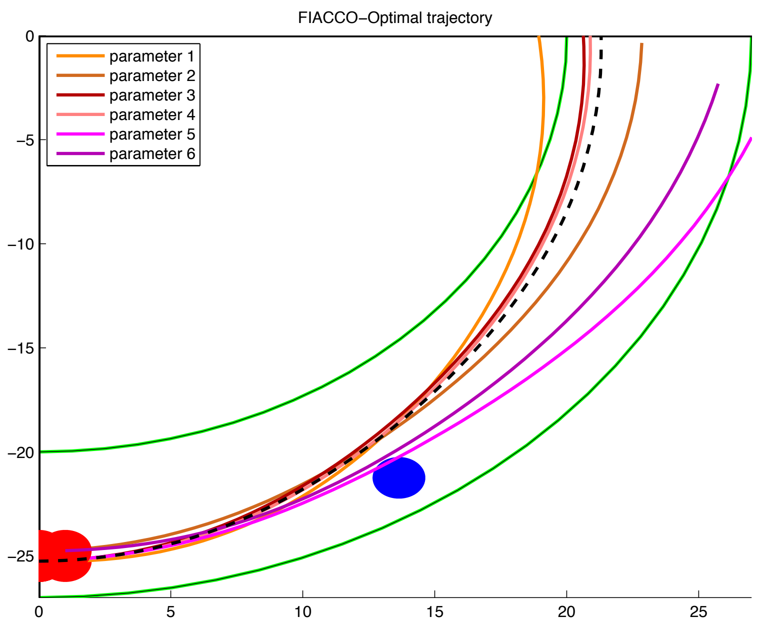


Figure 3.4: Perturbations in the first five initial states are shown in the figure for a minimum time trajectory. The perturbation value is discretized over the interval shown in the title of each figure to understand the shape of the perturbation set in such interval.

Example of Fiacco- and ODE-Sensitivity of an optimal trajectory for a curve road scenario. The Fiacco- and ODE-Sensitivities for the optimal trajectories of Figure 1.11 is given in Figure 3.5 and Figure 3.6, using Algorithms 6 and 7.



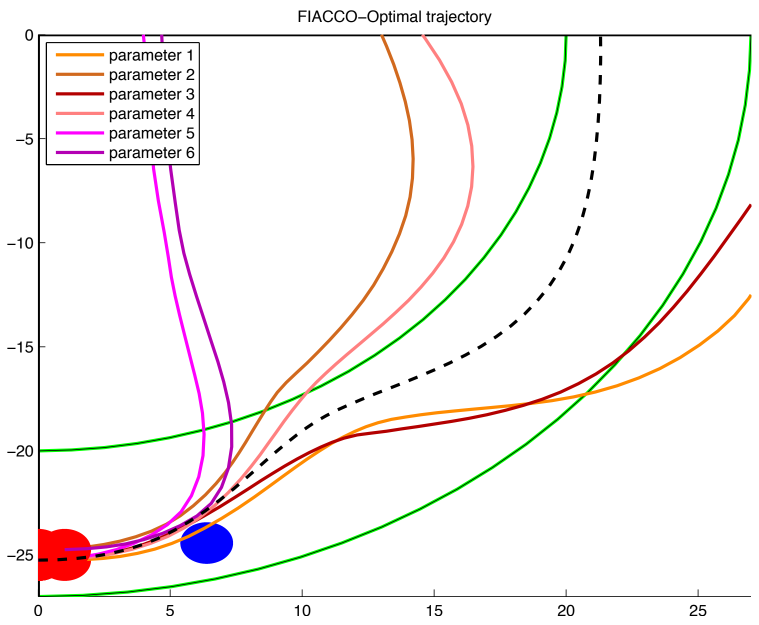
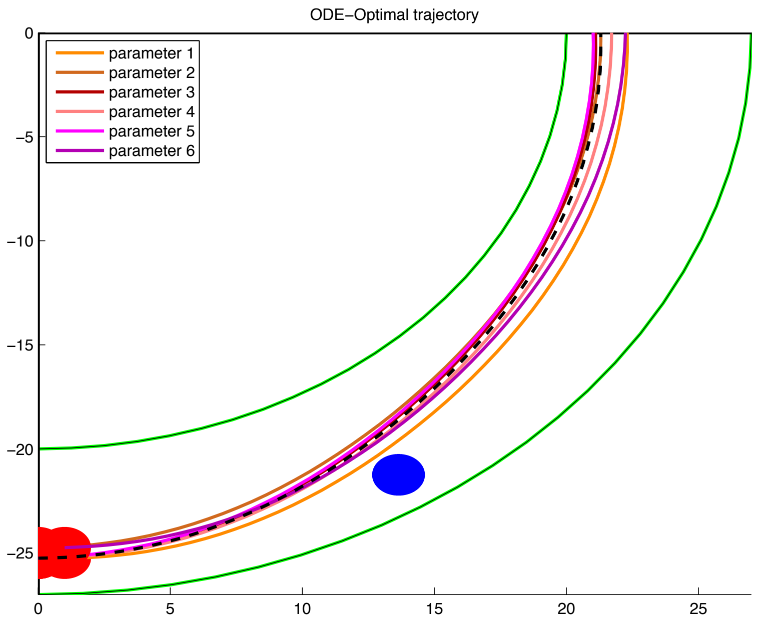


Figure 3.5: Fiacco-Sensitivity: the top picture is the minimum time trajectory of Figure 1.11, the bottom picture is the minimum initial distance trajectory of the same Figure 1.11. The seven trajectories depicted in each picture show the nominal trajectory (dashed black line) and its perturbation with respect to $p_1, \ldots, p_5$ and w.r.t. all parameters combined (dark magenta line). The perturbation vector is $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$.



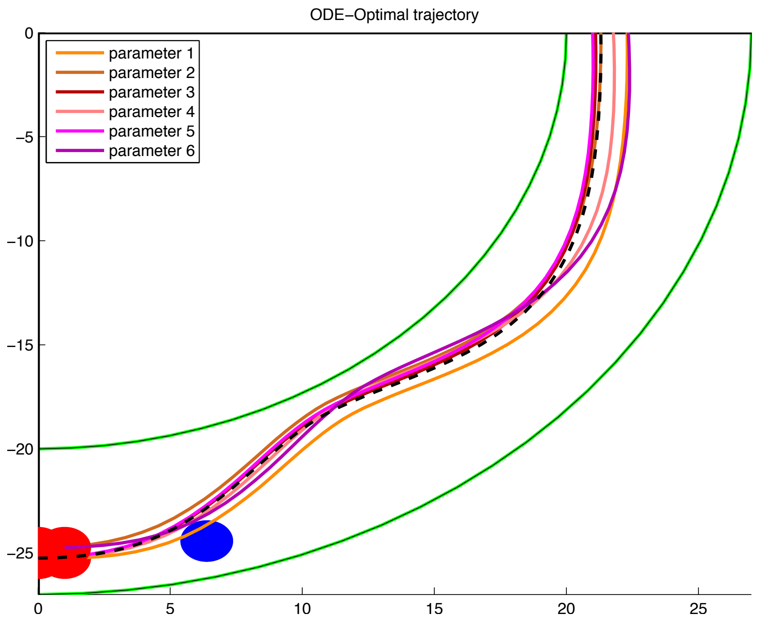


Figure 3.6: ODE-Sensitivity: the top picture is the minimum time trajectory of Figure 1.11, the bottom picture is the minimum initial distance trajectory of the same Figure 1.11. The seven trajectories depicted in each picture show the nominal trajectory (dashed black line) and its perturbation with respect to $p_1, \ldots, p_5$ and w.r.t. all parameters combined (dark magenta line). The perturbation vector is $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$.

The ODE-Sensitivity for perturbations in a ball as in Algorithm 8 is given in Figure 3.7 for the minimum time trajectory and in Figure 3.8 for the minimum initial distance trajectory. Bigger perturbation intervals are allowed for the minimum time trajectory than for the minimum initial distance, because the latter is a more extreme trajectory approaching the obstacle as close as possible.

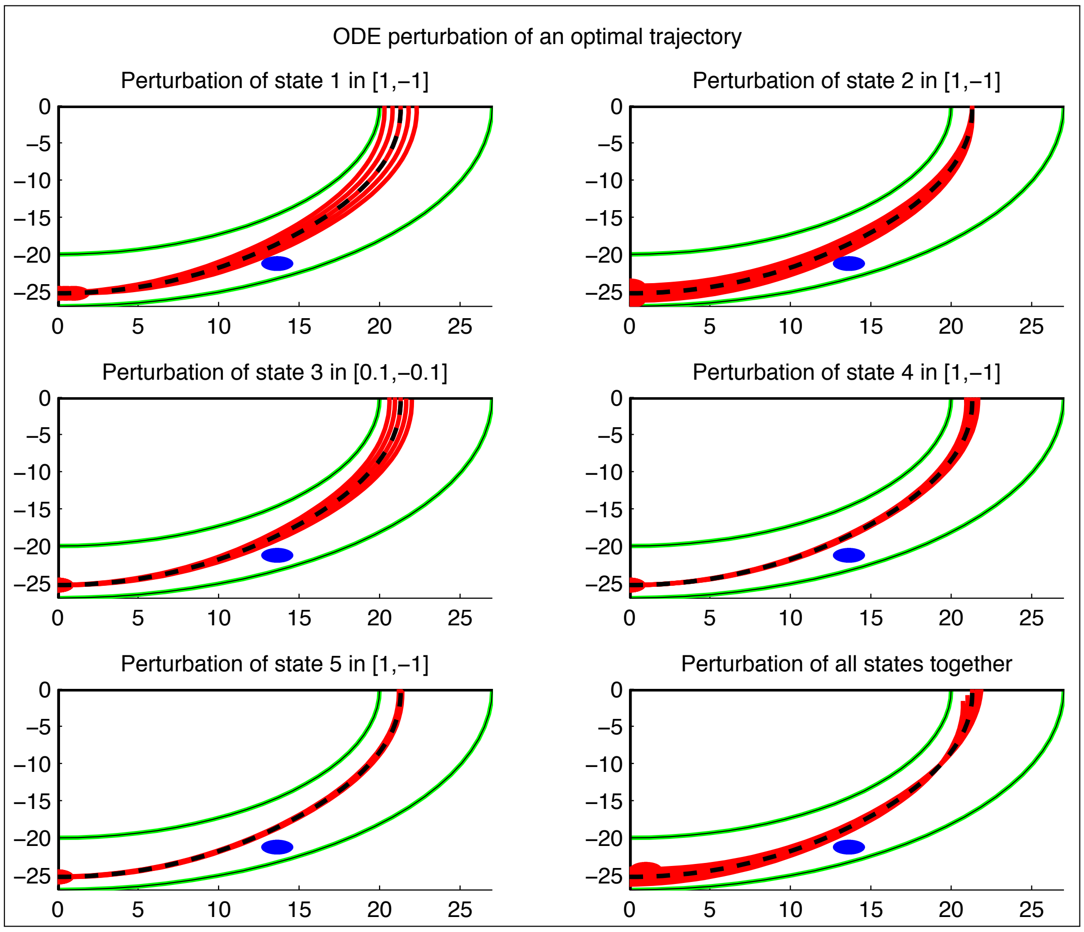


Figure 3.7: Perturbations in the first five initial state are shown in figure for a minimum time trajectory. The perturbation value is discretize over the interval written above each graph.

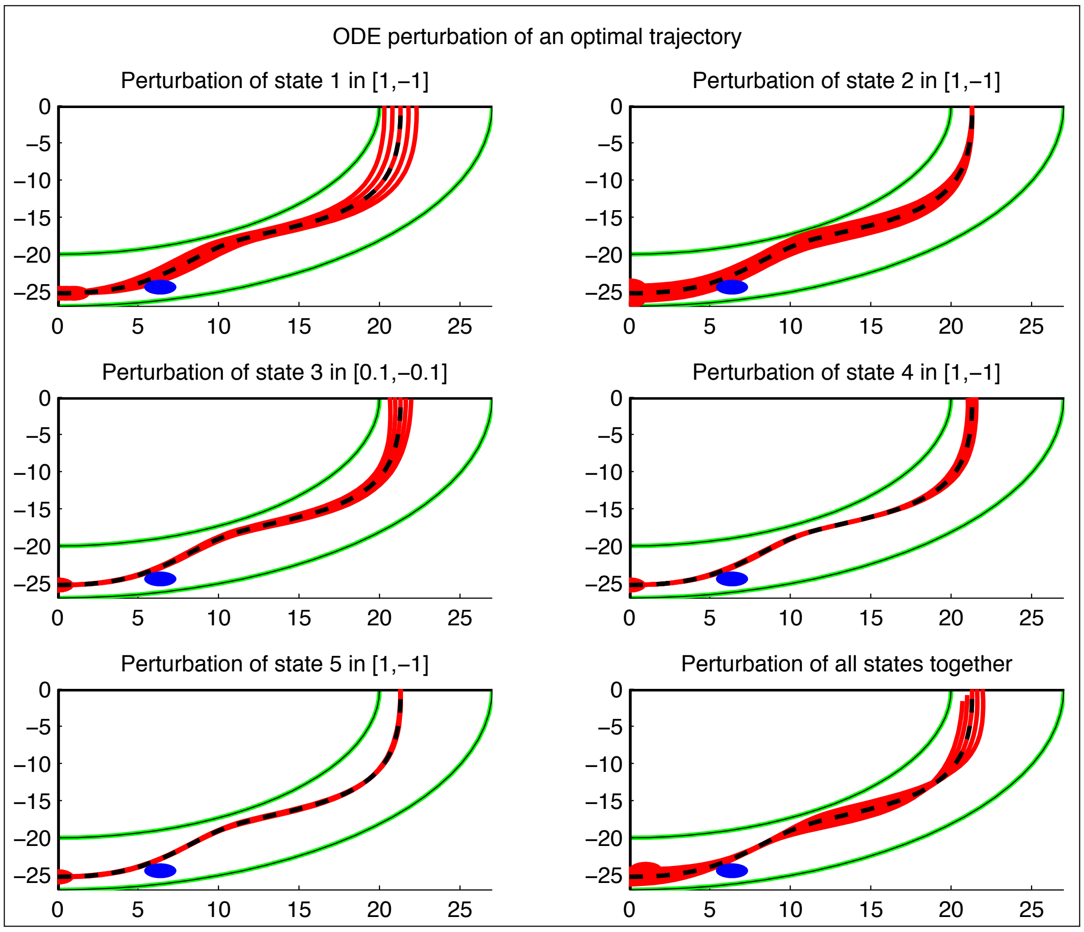


Figure 3.8: Perturbations in the first five initial state are shown in figure for a minimum initial distance trajectory. The perturbation value is discretize over the interval written above each graph.

3.2 Sensitivity in trajectory funnels and reachable sets

In the next first paragraph a way to define the reachable set of the perturbed initial value problem 4.1 of Chapter II, for p in a ball of a nominal parameter $\hat{p}$, for linear dynamics is given. In the second paragraph a method to investigate the dependence of the reachable set on p for the nonlinear system optimal control problem 4.1 of Chapter II is shown. It uses an approximation based on the computation of the reachable sets in Algorithm 3.

Linear initial value problem. Let the linear control system be defined as:

$$\begin{aligned} &\dot{z}(t) = A(t)z(t) + B(t)u(t) \text{ a.e. in } (t_0, t_f) \\ &z(t_0) = z_0(\hat{p}), \hat{p} \in \mathbb{R}^{n_z} \end{aligned} \tag{3.6}$$

for almost every t in $[t_0, t_f]$. The system (3.6) has a unique solution (see [30, Chapter 1]) for every:

$$\begin{aligned} &u \in \mathcal{U} := L^1((t_0, t_f); \mathbb{R}^{n_u}) \\ &A, B \in L^\infty((t_0, t_f); \mathbb{R}^{n_z \times n_z}) \\ &z_0(\hat{p}) \in \mathbb{R}^{n_z}. \end{aligned}$$

The solution is the continuous function such that:

$$\hat{z}(t) = \Phi(t)z_0(\hat{p}) + \Phi(t)\int_{t_0}^{t} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau, u \in \mathcal{U}, \tag{3.7}$$

with $\Phi(\cdot)$ the solution of the following matrix ordinary differential equation with given initial value:

$$\begin{aligned} &\dot{z}(t) = A(t)z(t) \\ &z(t_0) = \mathbb{I}_{n_z} \end{aligned} \tag{3.8}$$

where $\mathbb{I}_{n_z}$ is the $n_z \times n_z$ identity matrix.

Moreover the reachable set is:

$$\mathcal{FR}^f_{t_f, z_0(\hat{p})} = \{\Phi(t_f)z_0(\hat{p}) + \Phi(t_f)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau \mid u \in \mathcal{U}\} \tag{3.9}$$

The perturbed control system is defined as:

$$\begin{aligned} &\dot{z}(t) = A(t)z(t) + B(t)u(t), \text{ a.e. in } (t_0, t_f) \\ &z(t_0) = z_0(\tilde{p}), \text{ where } \tilde{p} \in B(\hat{p}, r), r > 0. \end{aligned} \tag{3.10}$$

The solution can be written as:

$$\tilde{z}(t) = \Phi(t)z_0(\tilde{p}) + \Phi(t)\int_{t_0}^{t} \Phi^{-1}(\tau)B(\tau)\tilde{u}(\tau)d\tau, u \in \mathcal{U} \tag{3.11}$$

and the reachable set is:

$$\mathcal{FR}^f_{t_f, z_0(\tilde{p})} = \{\Phi(t_f)z_0(\tilde{p}) + \Phi(t_f)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau \mid u \in \mathcal{U}\}. \tag{3.12}$$

Observation 3.1. *It holds that*

$$\mathcal{FR}^f_{t_f, z_0(\hat{p})} = \{\Phi(t_f)(z_0(\hat{p}) - z_0(\tilde{p}))\} + \mathcal{FR}^f_{t_f, z_0(\tilde{p})} \tag{3.13}$$

where the sum is a Minkowski sum. Indeed if $x \in \mathcal{FR}^f_{t_f,z_0(\hat{p})}$ *then* x *can be written in the form:*

$$x = \Phi(t_f)z_0(\hat{p}) + \Phi(t_f)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau$$

it exists $u \in \mathcal{U}$. *Therefore,*

$$x = \Phi(t_f)z_0(\hat{p}) - \Phi(t_f)z_0(\tilde{p}) + \Phi(t)z_0(\tilde{p}) + \Phi(t_f)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau. \quad (3.14)$$

Since

$$\bar{x} := \Phi(t_f)z_0(\tilde{p}) + \Phi(t_f)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau$$

with $u \in \mathcal{U}$ *arbitrary, then* $\bar{x} \in \mathcal{FR}^f_{t_f,z_0(\tilde{p})}$. *Thus,*

$$x = \Phi(t_f)(z_0(\hat{p}) - z_0(\tilde{p})) + \bar{x} \in \{\Phi(t_f)(z_0(\hat{p}) - z_0(\tilde{p}))\} + \mathcal{FR}^f_{t_f,z_0(\tilde{p})}.$$

Vice versa if $\bar{x} \in \mathcal{FR}^f_{t_f,z_0(\tilde{p})}$ *then* $\bar{x}$ *can be written in the form:*

$$\bar{x} = \Phi(t_f)z_0(\tilde{p}) + \Phi(t_f)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau$$

it exists $u \in \mathcal{U}$. *Therefore,*

$$\bar{x} = \Phi(t_f)z_0(\tilde{p}) - \Phi(t_f)z_0(\hat{p}) + \Phi(t_f)z_0(\hat{p}) + \Phi(t)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau. \quad (3.15)$$

Since

$$\bar{x} + \Phi(t_f)(z_0(\hat{p}) - z_0(\tilde{p})) = \Phi(t_f)z_0(\hat{p}) + \Phi(t_f)\int_{t_0}^{t_f} \Phi^{-1}(\tau)B(\tau)u(\tau)d\tau =: x$$

with $u \in \mathcal{U}$ *arbitrary, then* $x \in \mathcal{FR}^f_{t_f,z_0(\hat{p})}$.

Definition 3.2. *Let* X *and* Y *be two non-empty subsets of* $\mathbb{R}^n$ *and* d *a distance function of* $\mathbb{R}^n$ *associated to the norm* $\|\cdot\|$. *The Hausdorff distance between the sets* X, Y *is denoted by* $d_H(X,Y)$ *and it is defined by:*

$$d_H(X,Y) := \max\left\{\sup_{z\in X}\inf_{\zeta\in Y} d(z,\zeta),\ \sup_{\zeta\in Y}\inf_{z\in X} d(z,\zeta)\right\}. \quad (3.16)$$

Moreover, the norm of a set $X \subset \mathbb{R}^n$ *is defined as:*

$$\|X\| := \max_{x\in X}\|x\| = d_H(X,\{0\}). \quad (3.17)$$

Observation 3.3. *Denoting with d_H the Hausdorff distance then*

$$\begin{aligned} & d_H(\mathcal{FR}^f_{t_f,z_0(\hat{p})},\ \mathcal{FR}^f_{t_f,z_0(\tilde{p})}) = \\ &= d_H(\{\Phi(t_f)(z_0(\hat{p}) - z_0(\tilde{p}))\} + \mathcal{FR}^f_{t_f,z_0(\tilde{p})},\ \mathcal{FR}^f_{t_f,z_0(\tilde{p})}) = \\ &= \max_{x \in \mathcal{FR}^f_{t_f,z_0(\tilde{p})}} \min_{\bar{x} \in \mathcal{FR}^f_{t_f,z_0(\tilde{p})}} d(x, \bar{x} + \Phi(t_f)(z_0(\hat{p}) - z_0(\tilde{p}))) \leq \\ &\leq \|\Phi(t_f)\| \|z_0(\hat{p}) - z_0(\tilde{p})\|. \end{aligned} \tag{3.18}$$

Observations 3.1 and 3.3 show stability of the reachable set with respect to parameters in the initial value.

Nonlinear perturbed optimal control problem. Let $\mathcal{FR}^f_{t_f,\hat{p}}$ and $\mathcal{TF}^f_{t_f,\hat{p}}$ denote the reachable set and the trajectory funnel (respectively) for Problem 4.1 of Chapter II with nominal parameter $\hat{p}$. An approximation of $\mathcal{FR}^f_{t_f,p}$ and $\mathcal{TF}^f_{t_f,p}$, for $p \neq \hat{p}$ can be obtained by linearization from $\mathcal{FR}^f_{t_f,\hat{p}}$ and $\mathcal{TF}^f_{t_f,\hat{p}}$ using the sensitivity analysis in (3.1) and (3.5). In particular given a grid $\mathbb{G}$ in the state space, then the sensitivity analysis is performed with respect to the perturbation parameter $p \in B(\hat{p}, r)$, $r > 0$ for each optimal solution $\hat{z} := z^{\hat{u}}_{z_0(\hat{p})}$ such that $\exists \tau \in [t_0, t_f]$ such that $\hat{z}(\tau)$ is close enough to a grid point (i.e. $\hat{z}(\tau) \in \mathcal{FR}^f_{t_f,\hat{p}}$).

$$\mathcal{FR}^f_{t_f,p} \approx \bigcup_{g_h : \|\hat{z}(\hat{u},\hat{p})(\tau) + \frac{\partial \hat{z}(\hat{u},\hat{p})(\tau)}{\partial p}(p-\hat{p}) - g_h\|_2 \leq Ch} \{g_h\}, \tag{3.19}$$

Moreover the approximation of the trajectory funnel is given by

$$\mathcal{TF}^f_{t_f,p} \approx \bigcup_{g_h : \|\hat{z}(\hat{u},\hat{p})(\tau) + \frac{\partial \hat{z}(\hat{u},\hat{p})(\tau)}{\partial p}(p-\hat{p}) - g_h\|_2 \leq Ch} \{z(\hat{u},\hat{p})(\cdot) + \frac{\partial \hat{z}(\hat{u},\hat{p})(\cdot)}{\partial p}(p - \hat{p})\}, \tag{3.20}$$

for $C > 0$ suitable and $\frac{\partial \hat{z}(\hat{u},\hat{p})(\tau)}{\partial p}$ denotes one of the previously discussed Fiacco- or ODE-Sensitivities.

```
input : Let $G$, $U$ and $Z$ be the output of Algorithm 3 such that
        $Z(t_k) = (\hat{z}_i(t_k))_{i:g_i \in G}, U(t_k) = (\hat{u}_i(t_k))_{i:g_i \in G} \quad \forall t_k \in [t_0, \tau]$,
        $\hat{p}, z_0, r, p, \xi := z_0(p) \in B(z_0, r)$.
output: $\tilde{G}$, $\tilde{Z}$.
for $t_k \in [t_0, t_f]$ do
    for $g_i \in G$ do
        solve system (3.1) or (3.5) and get matrix $S_i(t_k)$;
        $\tilde{z}_i(t_k) := \hat{z}_i(t_k) + S_i(t_k)(\hat{p} - p)$;
    end
    $\tilde{G} := \tilde{G} \cup \{g_i\}$;
    $\tilde{Z}(k) := (\tilde{z}_i(t_k))_{i:g_i \in G}$;
end
```

Algorithm 9: Sensitivity in reachable set and trajectory funnel.

Figures 3.9 and 3.10 show the reachable set and the trajectory funnel of Problem 4.1 of Chapter II for perturbations on p in the initial state (Fiacco-Sensitivity), while Figures 3.11 and 3.12 show the reachable set and the trajectory funnel if a perturbations p occur in the initial state, but with fixed control (ODE-Sensitivity). The Fiacco-Sensitivity is much more sensitive to perturbations and for big perturbations does not give the desired solution. The ODE-Sensitivity aims to understand how the reachable sets change if a perturbation occur in the initial data, but the control is fixed. The reference reachable set and trajectory funnel are in Figure 1.4. The perturbations are written on top of each graph.

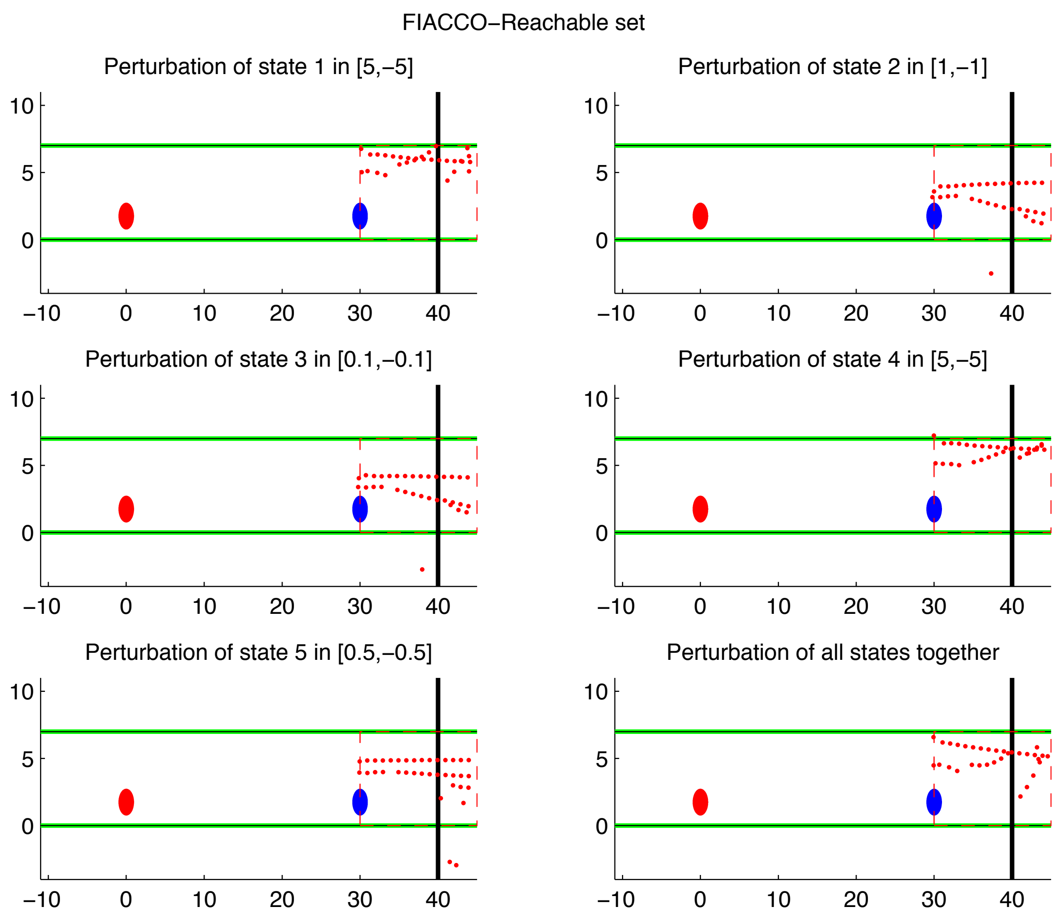


Figure 3.9: Reachable sets approximations by Fiacco-Sensitivity, perturbed of $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$ and with a perturbation of all parameters together.

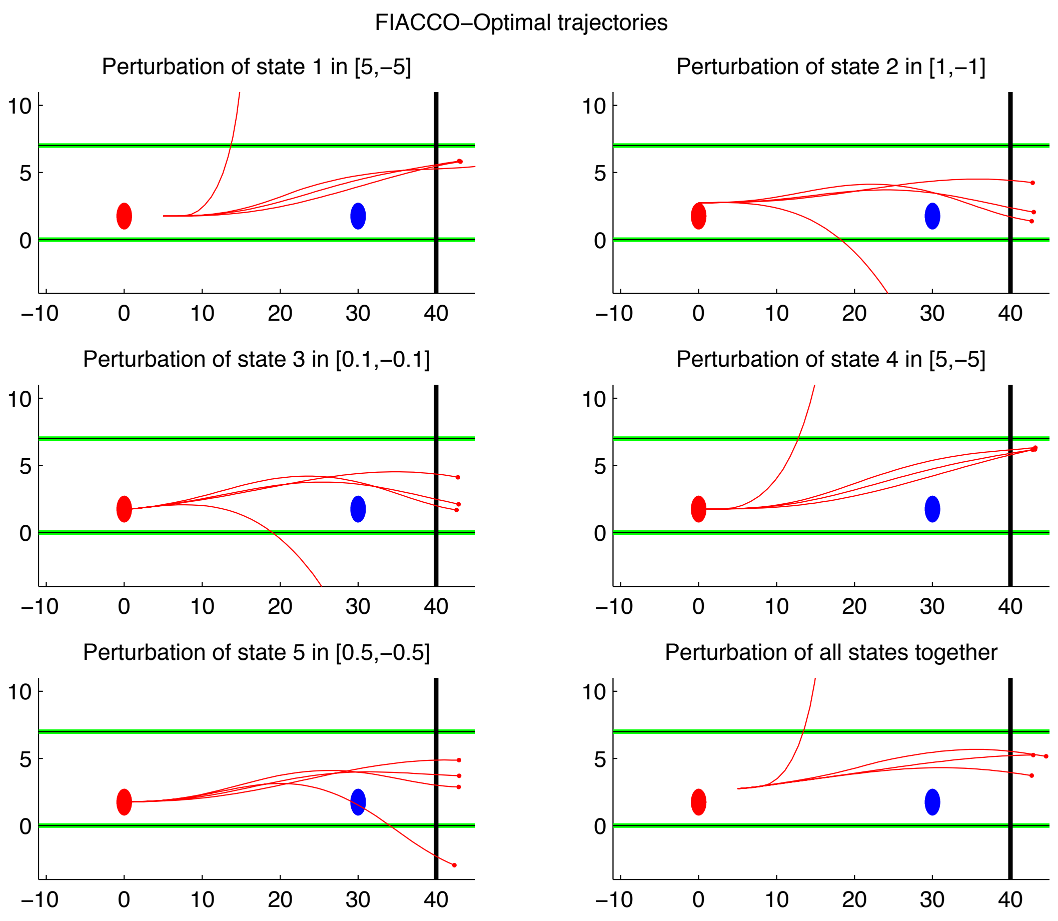


Figure 3.10: Trajectory funnel approximations by Fiacco-Sensitivity, perturbed of $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$ and with a perturbation of all parameters together.

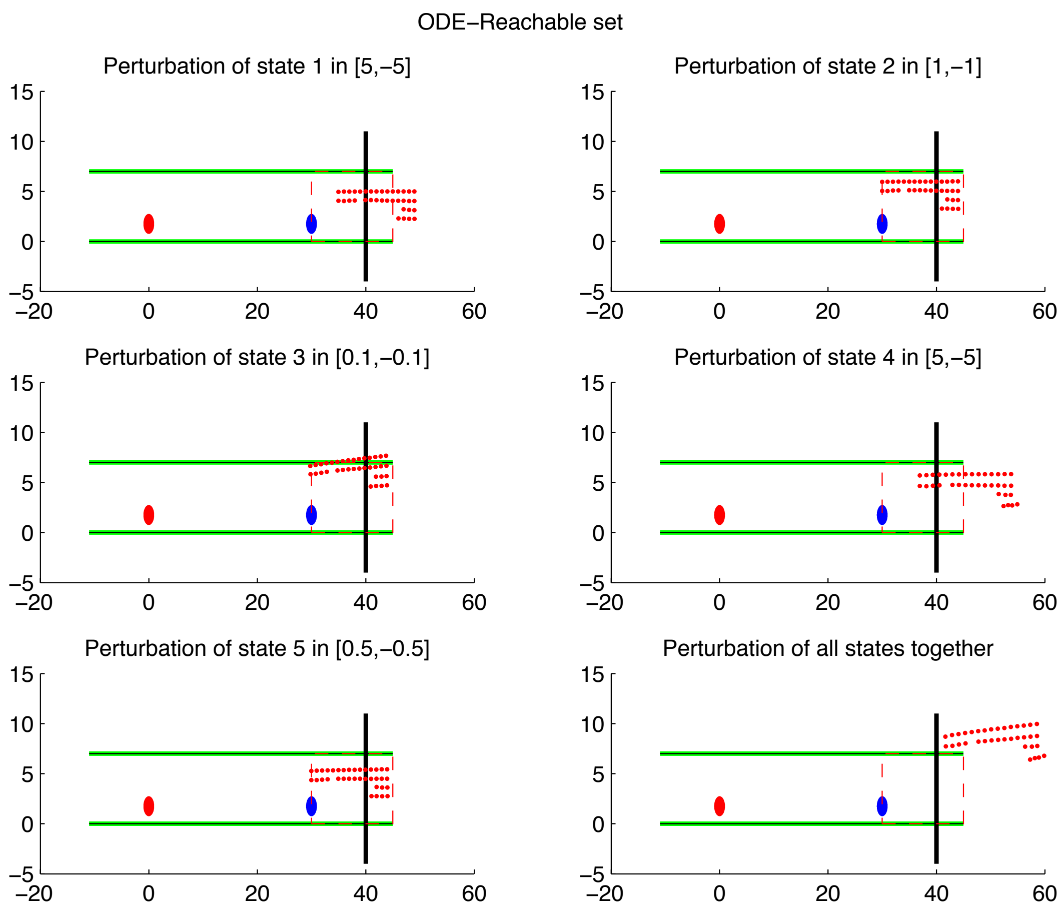


Figure 3.11: Reachable sets approximations by ODE-Sensitivity, perturbed of $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$ and with a perturbation of all parameters together.

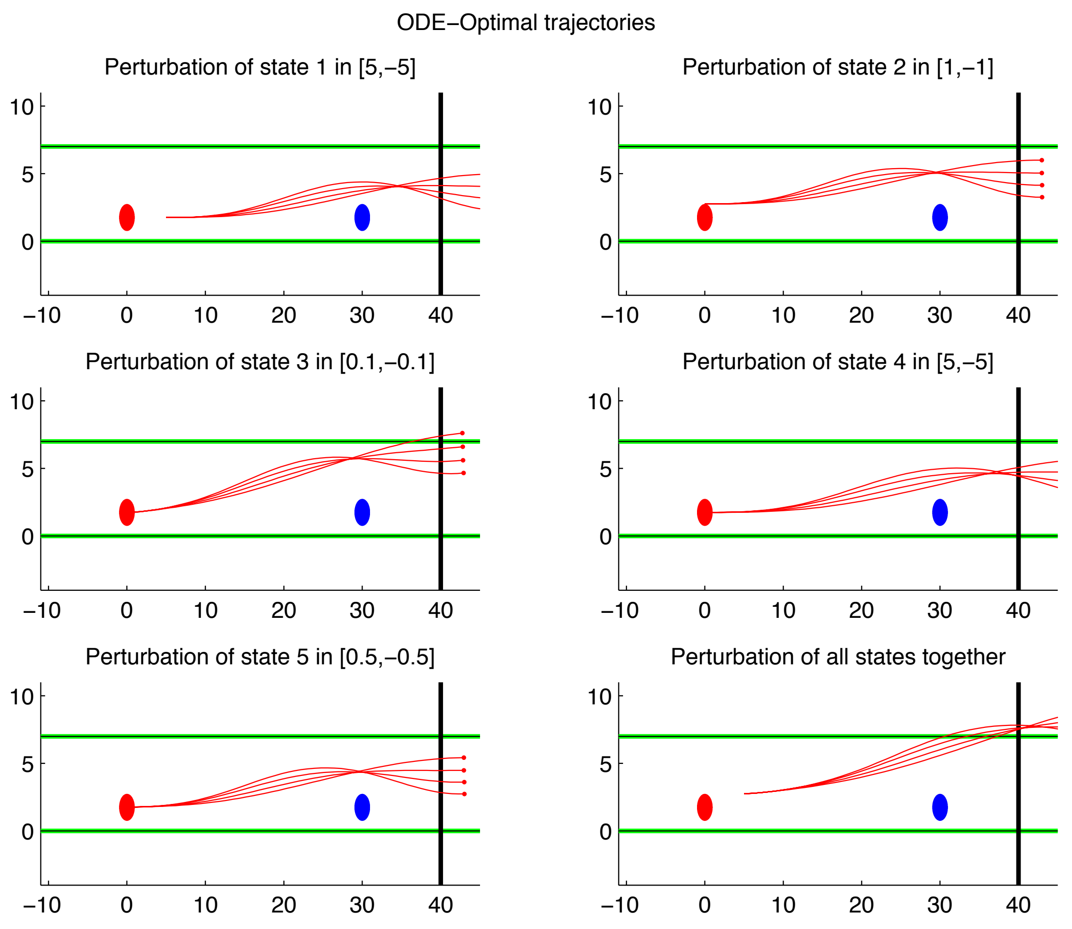


Figure 3.12: Trajectory funnel approximations by ODE-Sensitivity of $p = (5[m], 1[m], 0.1[rad], 5[m/s], 0.5[m/s])$ and with a perturbation of all parameters together.

The ODE-Sensitivity for reachable sets (trajectory funnels) is useful to find the subset of the reachable set (or trajectory funnels), whose points are the final states of those optimal trajectories that does not violate the state constraints. Such a set is called robust reachable set (or robust trajectory funnel) and it is formally defined in 3.4.

Definition 3.4. *Let $\mathcal{FR}^f_{t_f}$ be the reachable set for Problem 3.1 of Chapter II. The* robust reachable set *of level $r \geq 0$ is defined as*

$$\begin{aligned}\widetilde{\mathcal{FR}}^r_{t_f} := \; & \{z_f \in \mathcal{FR}^f_{t_f} \,|\, \exists u \in \mathcal{U}, \exists \tau \in [t_0, t_f] : \\ & z^{\hat{u}}_{z_0} \textit{ is a solution of Problem 3.1 of Chapter II with } z^{\hat{u}}_{z_0}(\tau) = z_f \\ & \textit{and } \forall \xi \in B(z_0, r),\; z^{\hat{u}}_{\xi} \textit{satisfies the state constraints of the Problem 3.1} \\ & \textit{of Chapter II}\}.\end{aligned}$$
(3.21)

The robust trajectory funnel *of level* $r \geq 0$ *is defined as*

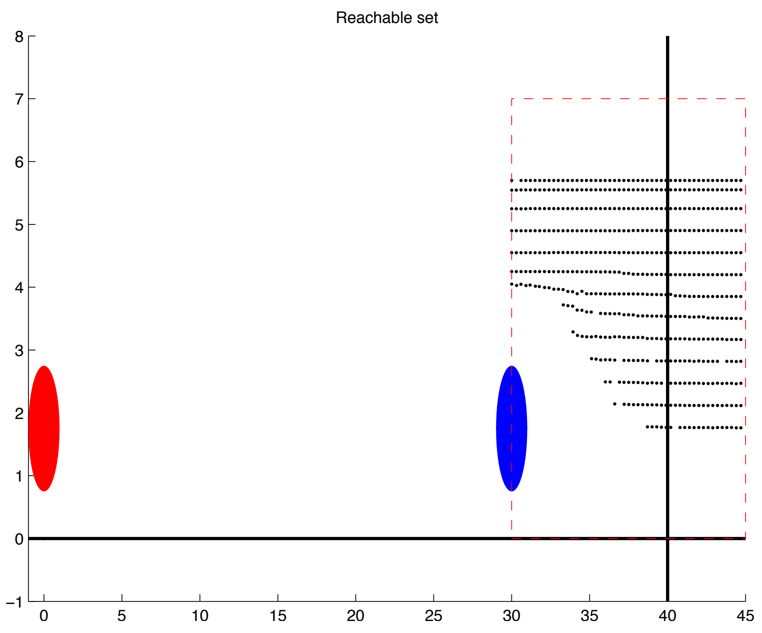
$$\widetilde{\mathcal{TF}}_{t_f}^{r} := \Big\{ z_{z_0}^{u} : \mathbb{R} \to \mathbb{R}^{n_z} \mid \exists u \in \mathcal{U} : z_{z_0}^{u} \textit{ solution of Problem 3.1 of Chapter II and } z_{z_0}^{u}(\tau) \in \widetilde{\mathcal{FR}}_{t_f}^{r}, \exists \tau \in [t_0, t_f] \Big\}. \tag{3.22}$$

Given a grid $\mathbb{G} = \{g_h\}_{h \in \{1,\dots,k\}}$ in the state space, Algorithm 10 computes the robust reachable set (and the robust trajectory funnel) as a subset of the reachable set $\mathcal{FR}_{t_f}^{f}$ such that, given an initial value z_0 and an optimal control $\hat{u}$ for Problem 3.1 of Chapter II, each optimal trajectory $z_{z_0}^{\hat{u}}$ (with $\|g_h - z_{z_0}^{\hat{u}}(\tau)\| \leq Ch$ for a suitable constant C and grid point $g_h \in \mathcal{FR}_{t_f}^{f}$) is admissible for any $\xi \in B(z_0, r)$ with $r > 0$ given. In Algorithm 10, $z_0 := z_0(\hat{p})$ and $\xi := z_0(p)$ for $p \in B(\hat{p}, r)$.

```
input  : $G$ and $Z$ output of Algorithm 9 such that
         $Z(t_k) = (\hat{z}_i(t_k))_{i:g_i \in G}$,
         $\forall t_k \in [t_0, \tau]$,   $\hat{p}, z_0, r$.
output: $\tilde{G}$ robust reachable set,
         $\tilde{Z}$ robust trajectory funnel.
define a grid $P := \{p_h\}_{h=1,\dots,H}$ such that $\xi := z_0(p_h) \in B(z_0, r)$;
for $g_i \in G$ do
    for $t_k \in [t_0, t_f]$ do
        solve system (3.5) and get matrix $S_i(t_k)$;
        for $h = 1, \dots, H$ do
            $\tilde{z}_{i,h}(t_k) := \hat{z}_i(t_k) + S_i(t_k)(\hat{p} - p_h)$;
        end
        if all $(\tilde{z}_h(t_k))_{h=1,\dots,H}$ satisfy state constraints then
            $\tilde{G} := \tilde{G} \cup \{g_i\}$;
            $\tilde{Z}(i,k) := (\tilde{z}_{i,1}(t_k), \dots, \tilde{z}_{i,H}(t_k))$;
        end
    end
end
```

Algorithm 10: Robust reachable set implemented with MATLAB.

The robust reachable set for the scenario modeled in Problem 1.1 is given in Figure 3.13. The robust trajectory funnel is given in Figure 3.14. In this case the constraint violation is minimized and the resulting optimal trajectory is the furthest from the obstacles and the boundary of the road.



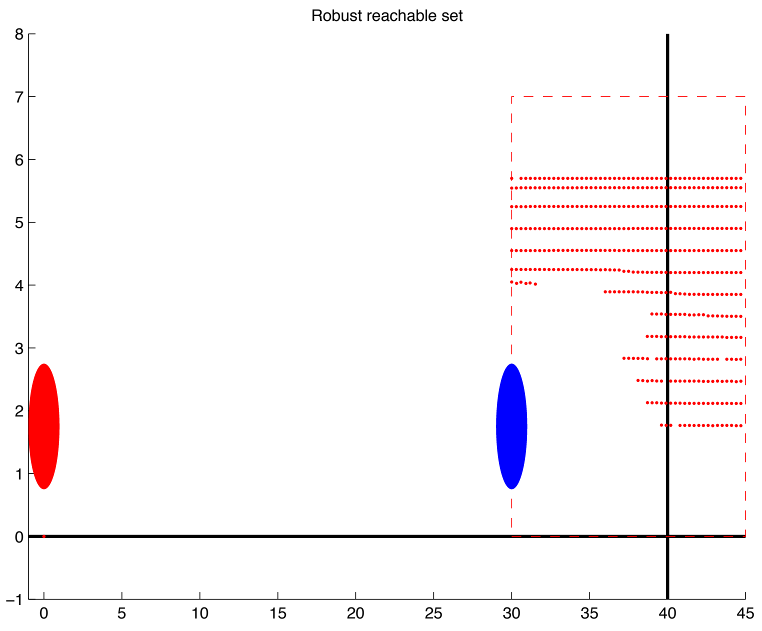


Figure 3.13: The top picture shows the nominal reachable set, the bottom picture shows the robust reachable set for a perturbation of $p = (1[m], 0.1[m], 0.01[rad], 1[m/s], 0.1[m/s])$ and $-p$.

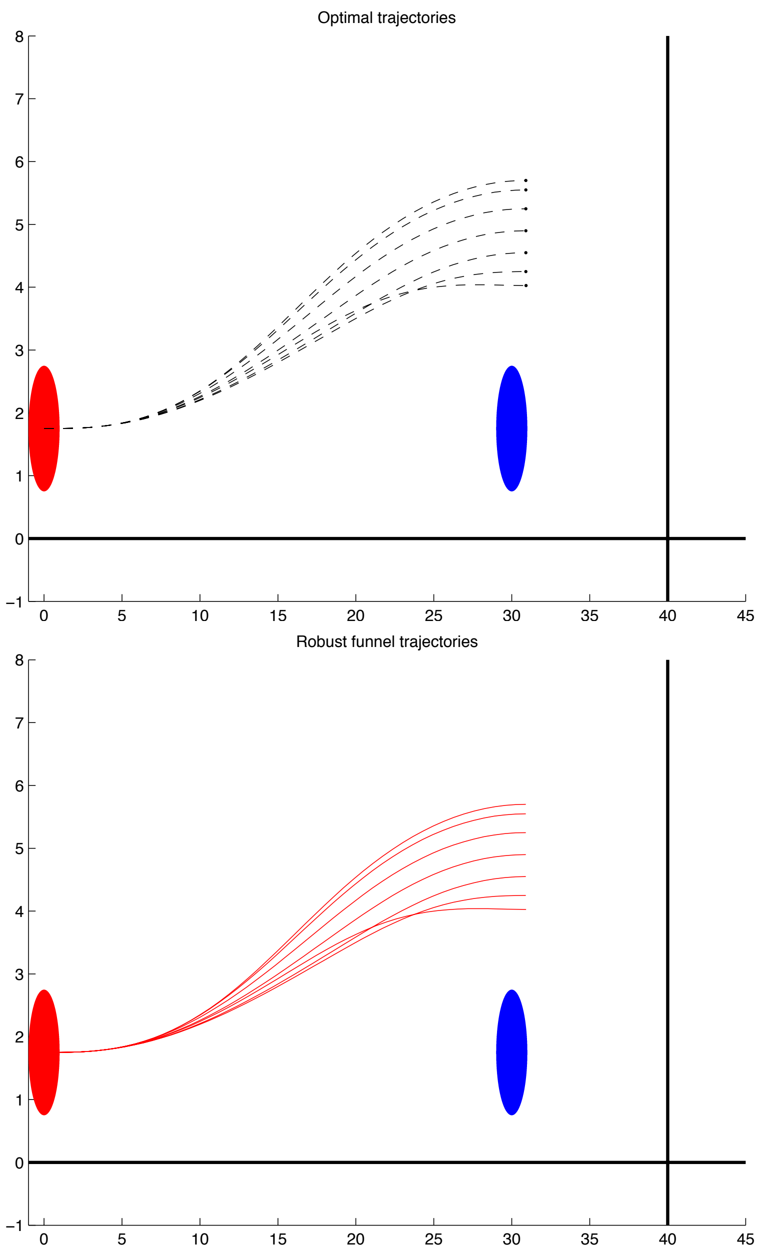


Figure 3.14: The top picture shows the nominal trajectory funnel, the bottom picture shows the robust trajectory funnel for a perturbation of $p = (1[m], 0.1[m], 0.01[rad], 1[m/s], 0.1[m/s])$ and $-p$.

3.3 Radius estimation

The idea of ODE-Sensitivity analysis is that once an optimal control $\hat{u} \in \mathcal{U}$ has been found for Problem 3.1 of Chapter II, and for a given $z_0 \in \mathcal{BR}^f_{t_f}$, then the dependency of the optimal solution $z_{z_0}^{\hat{u}}$ on z_0 is investigated. The set $\widetilde{\mathcal{BR}}_{t_f}^r$ contains all the initial states $z_0 \in \mathcal{BR}_{t_f}^f$ with associated optimal control $\hat{u}$, such that $z_\xi^{\hat{u}}$ satisfies the state and boundary constraints of Problem 3.1 of Chapter II, for all $\xi \in B(z_0, r)$. In Proposition 3.6 the maximum r for a given trajectory $z_{z_0}^{\hat{u}}$ is estimated for problems with boundary constrains only. In particular the constant C is found, such that $r \leq C$ gives a sufficient condition for deciding weather a point in the backward reachable set is also in $\widetilde{\mathcal{BR}}_{t_f}^r$ for problems without state constraints. In the second paragraph an algorithm to obtain a similar estimation of r is derived for optimal control problems with state constraints only.

Radius estimation with boundary constraints only. Here an estimation of the maximum possible initial data perturbation is given, such that the nominal optimal trajectory, for Problem 3.1 of Chapter II with boundary constraints only, is still admissible. This is useful for sensor developers that need to understand the precision of a sensor used in autonomous driving.

Definition 3.5. *Let $\mathcal{BR}_{t_f}^f$ be the backward reachable set for Problem 3.1 of Chapter II with only boundary constraints. The perturbed backward reachable set of level $r \geq 0$ is defined as*

$$\begin{aligned}\widetilde{\mathcal{BR}}_{t_f}^r := \; & \{z_0 \in BR_{t_f}^f, \mid \exists u \in \mathcal{U} : z_{z_0}^u \text{ is a solution of Problem 3.1 of Chapter II} \\ & \text{with only boundary constraints and } \forall \tilde{z}_0 \in B(z_0, r), z_{\tilde{z}_0}^u \text{ satisfies} \\ & \text{the boundary constraints of Problem 3.1 of Chapter II}\}.\end{aligned} \tag{3.23}$$

The next proposition finds a sufficient condition on $r > 0$ small such that $z_0 \in \widetilde{\mathcal{BR}}_{t_f}^r$.

Proposition 3.6. *Let $f : \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}$ be a C^2 and let $L > 0$ denote the uniform Lipschitz constant with respect to its first argument, i.e.*

$$\|f(z, u) - f(\tilde{z}, u))\| \leq L\|z - \tilde{z}\|, \quad \forall u, \tag{3.24}$$

where $z \neq \tilde{z}$. Given $z_0 \in \mathbb{R}^{n_z}$, consider an optimal trajectory $z_{z_0}(t)$ with optimal control $u_{z_0} \in \mathbb{R}^{n_u}$ for Problem 3.1 of Chapter II such that:

$$\begin{aligned} \dot{z_{z_0}}(t) &= f(z_{z_0}(t), u_{z_0}(t)), \quad t \geq 0, \\ z_{z_0}(0) &= z_0. \end{aligned} \tag{3.25}$$

Let Ω be the target set and $d_\Omega : \mathbb{R}^{n_z} \to \mathbb{R}$ be the Lipschitz continuous signed distance function to Ω. Then

$$r \leq -\frac{2}{K} \frac{d_\Omega(z_{z_0}(t_f))}{\|M_{t_f}\|} \Longrightarrow z_0 \in \widetilde{\mathcal{BR}}_{t_f}^r, \tag{3.26}$$

where $\widetilde{\mathcal{BR}}_{t_f}^r$ is in Definition 3.5,

$$M_t := \exp\big(\int_0^t D_z f(z_{z_0}(\tau), u_{z_0}(\tau)) d\tau\big), \quad t > 0 \tag{3.27}$$

and

$$K \geq 1 + \sqrt{1 - \left(2Ce^{4Lt_f} d_\Omega(z_{z_0}(t_f))/\|M_{t_f}\|^2\right)}, \tag{3.28}$$

with

$$\|D^2 f(\cdot, u_{z_0}(t))\|_\infty \leq C. \tag{3.29}$$

Proof. Let z_0 be given and consider an optimal trajectory $z(t) = z_{z_0}(t)$ with control $u = u_{z_0}$:

$$\begin{aligned} \dot{z}(t) &= f(z(t), u(t)), \quad t \geq 0, \\ z(0) &= z_0. \end{aligned} \tag{3.30}$$

Now let $\tilde{z}(t)$ be a solution of

$$\begin{aligned} \dot{\tilde{z}}(t) &= f(\tilde{z}(t), u(t)), \quad t \geq 0, \\ \tilde{z}(0) &= \tilde{z}_0. \end{aligned} \tag{3.31}$$

where $p := \tilde{z}_0 - z_0$, $p \in rB_{\mathbb{R}^{n_z}}$ is small.

Considering the Caratheodory solutions of the problems (3.30) and (3.31), it holds:

$$\tilde{z}(t) - z(t) = \tilde{z}_0 - z_0 + \int_0^t f(\tilde{z}(\tau), u(\tau)) - f(z(\tau), u(\tau)) d\tau. \tag{3.32}$$

For (3.24) then:

$$\begin{aligned} \|\tilde{z}(t) - z(t)\| &\leq \|\tilde{z}(0) - z(0)\| + \textstyle\int_0^t \|f(\tilde{z}(\tau), u(\tau)) - f(z(\tau), u(\tau))\| d\tau \\ &\leq \|p\| + \textstyle\int_0^t L\|\tilde{z}(\tau) - z(\tau)\| d\tau \end{aligned} \tag{3.33}$$

and so for the Gronwall's inequality:

$$\|\tilde{z}(t) - z(t)\| \leq \|p\| e^{Lt} \leq re^{Lt}. \tag{3.34}$$

Assuming that f is at least C^2 twice continuously differentiable in the z variable and the Taylor expansion is:

$$f(\tilde{z}(t), u(t)) - f(z(t), u(t)) = D_z f(z(t), u(t))(\tilde{z}(t) - z(t)) + R_2(z(t)), \tag{3.35}$$

where

$$\|R_2(z(t))\| = \frac{\|D_z^2 f(\cdot, u(t))\|_\infty}{2} \|\tilde{z}(t) - z(t)\|^2. \tag{3.36}$$

Moreover let $C > 0$ be a constant such that

$$\|D^2 f(\cdot, u(t))\|_\infty \leq C, \tag{3.37}$$

then

$$\|R_2(z(t))\| \leq \frac{C}{2} r^2 e^{2Lt}. \tag{3.38}$$

Imposing

$$A_t := D_z f(z(t), u(t)), \quad \text{and} \quad M_t := \exp\big(\int_0^t A_s ds\big) \tag{3.39}$$

and considering (3.30) and (3.31), it holds:

$$\frac{d(\tilde{z}(t) - z(t))}{dt} = D_z f(z(t), u(t))(\tilde{z}(t) - z(t)) + R_2(z(t)) \tag{3.40}$$

Solving (3.40):

$$\begin{aligned} \tilde{z}(t) - z(t) &= \Big(p + \textstyle\int_0^t R_2(z(\tau)) \exp\big(-\int_0^\tau A_s ds\big)\, d\tau\Big) \exp\Big(\int_0^t A_\tau d\tau\Big) \\ &= \Big(p + \textstyle\int_0^t R_2(z(\tau)) \exp\big(-\int_0^\tau A_s ds\big)\, d\tau\Big) M_t \\ &= pM_t + \epsilon(t) \end{aligned} \tag{3.41}$$

where

$$\epsilon(t) := \left(\int_0^t R_2(z(\tau)) \exp\left(-\int_0^\tau A_s ds\right) d\tau\right) M_t \tag{3.42}$$

and, recalling (3.38),

$$\begin{aligned} \|\epsilon(t)\| &\leq \Big(\textstyle\int_0^t \|R_2(z(\tau))\| \exp\big(\int_0^\tau \|D_z f(z(s), u(s))\| ds\big)\, d\tau\Big) \cdot \\ &\quad \exp\Big(\textstyle\int_0^t \|D_z f(z(\tau), u(\tau))\| d\tau\Big) \\ &\leq \tfrac{C}{6L} r^2 e^{4Lt} =: a_t(r). \end{aligned} \tag{3.43}$$

The following implication will be proved:

$$\sup_{e \in B} d_\Omega(z(t_f) + rM_{t_f} e + \epsilon(t_f)) \leq 0 \Rightarrow z_0 \in \widetilde{\mathcal{BR}}^r_{t_f}. \tag{3.44}$$

Let $d_\Omega(\cdot)$ defined as:

$$d_\Omega(x) := \begin{cases} d(x, \Omega), & \text{if } x \in \complement\Omega \\ -d(x, \complement\Omega), & \text{if } x \in \Omega \end{cases}$$

where $\complement\Omega = \mathbb{R}^{n_z}$ and d is the distance between a point and a set in $\mathbb{R}^{n_z}$ defined as:

$$d(x, \Omega) := \inf_{\omega \in \Omega} \|\omega - x\|.$$

By triangular inequality it holds

$$d(x, \Omega) \leq d(x, y) + d(y, \Omega), \qquad y \in \mathbb{R}^{n_z} \text{ and } x \in \complement\Omega, \tag{3.45}$$

$$d(y, \complement\Omega) \leq d(y, x) + d(x, \Omega), \qquad y \in \mathbb{R}^{n_z} \text{ and } x \in \Omega, \tag{3.46}$$

which implies that

$$d_\Omega(x) \leq d_\Omega(y) + d(x, y). \tag{3.47}$$

Therefore,

$$\begin{array}{c}\sup_{e\in B} d_\Omega(z(t_f)+rM_{t_f}e+\epsilon(t_f)) \leq \\ \leq d_\Omega(z(t_f))+r\sup_{e\in B}\|M_{t_f}e+\epsilon(t_f)\| = \\ \leq d_\Omega(z(t_f))+r\sup_{e\in B}\|M_{t_f}e\|+\|\epsilon(t_f)\| = \\ = d_\Omega(z(t_f))+r\|M_{t_f}\|+a_{t_f}(r),\end{array} \tag{3.48}$$

using Equation (3.42) and being $r \geq 0$.

Thus,

$$\begin{array}{cc}d_\Omega(z(t_f))+r\|M_{t_f}\|+a_{t_f}(r)\leq 0 & \Rightarrow \\ \sup_{e\in B} d_\Omega(z(t_f)+rM_{t_f}e+\epsilon(t_f))\leq 0 & \Rightarrow \\ z_0\in\widetilde{\mathcal{BR}}^r_{t_f}. & \end{array} \tag{3.49}$$

The aim is to find a condition on r out of (3.49) which implies $z_0 \in \widetilde{\mathcal{BR}}^r_{t_f}$. It suffices that $r \geq 0$ and:

$$d_\Omega(z(t_f))+r\|M_{t_f}\|+r^2\frac{C}{6L}e^{4Lt_f}\leq 0, \tag{3.50}$$

using (3.43). Let r_1, r_2 be the roots of Equation (3.50) with $r_1 < 0 < r_2$ then:

$$0\leq r\leq \frac{3L\|M_{t_f}\|}{Ce^{4Lt_f}}\left(-1+\sqrt{1+\frac{2C}{3L}e^{4Lt_f}\left(\frac{-d_\Omega(z(t_f))}{\|M_{t_f}\|^2}\right)}\right) \tag{3.51}$$

where $d_\Omega(z(t_f)) < 0$ since $z(t_f)$ is the optimal feasible trajectory for (3.30). In particular if

$$K := 1+\sqrt{1+\frac{2C}{3L}e^{4Lt_f}\left(\frac{-d_\Omega(z(t_f))}{\|M_{t_f}\|^2}\right)}, \tag{3.52}$$

the sufficient condition for $z_0 \in \widetilde{\mathcal{BR}}^r_{t_f}$ is:

$$0\leq r\leq -\frac{2}{K}\frac{d_\Omega(z(t_f))}{\|M_{t_f}\|}. \tag{3.53}$$

□

Radius estimation with state constraints only. In Figure 3.15 the obstacle is denoted as the set $\complement\mathcal{K}$ and it holds that $z(t) \in \mathcal{K}$ for all $t \in [t_0, t_f]$. The radius r is then taken as the maximum radius such that $z_0 \in \widetilde{\mathcal{BR}}^r_{t_f}$. Which means that for a given z_0 and a corresponding optimal trajectory $z^{\hat{u}}_{z_0}$ (red line), all perturbed trajectories $z^{\hat{u}}_\xi$ (black curves) are satisfying the state constraints ($z^{\hat{u}}_\xi(t) \in \mathcal{K}(t), \forall t \in [t_0, t_f]$), for all $\xi \in B(z_0, r)$.

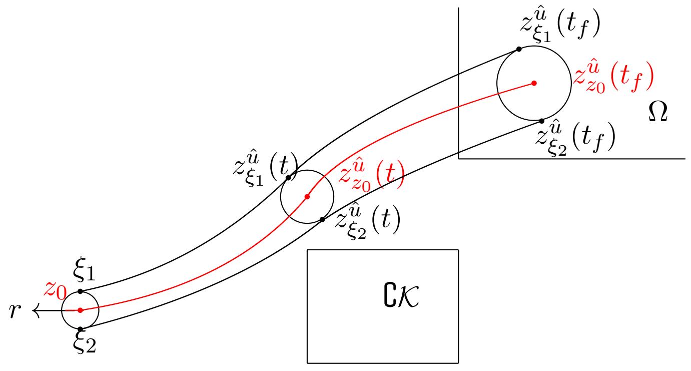


Figure 3.15: Idea of the perturbed reachable set and radius estimation

Observation 3.7. *For r small, in first approximation it holds*

$$z_{B(z_0,r)}(t) = z_{B(z_0,r)}^{\hat{u}}(t) \simeq \hat{z}(t) + rM_tB + o(r), \tag{3.54}$$

where $\hat{z}(t) = z_{z_0}^{\hat{u}}(t)$ is the reference trajectory, M_t is in (3.27) *and B is the unit ball. Equation* (2.85) *of Chapter II is, in first approximation, equivalent to*

$$\sup_{e\in B} \varphi(\hat{z}(t_f) + rM_{t_f}e) \leq 0, \quad \textit{and} \quad \sup_{(0,t_f)} \sup_{e\in B} g(\hat{z}(t) + rM_te) \leq 0,$$
(3.55)

where B is the unit ball. In the same way,

$$\varphi(\hat{z}(t_f)) + r\|M_{t_f}^{\top}\nabla\varphi(\hat{z}(t_f))\| \leq 0 \tag{3.56}$$

and

$$\sup_{(0,t_f)} g(\hat{z}(t)) + r\|M_t^{\top}\nabla g(\hat{z}(t))\| \leq 0 \tag{3.57}$$

Algorithm 11 shows how the radius estimation in Observation 3.7 is computed.

```
input  : $(\hat{z}(t_k), \hat{u}(t_k)) \quad \forall t_k \in [t_0, t_f]$, $\hat{p}$, $z_0$
         state constraints in the form $g(z, t_k) \leq 0$,
         with values in $\mathbb{R}^{n_g}$.
output: $r, r_1, \dots, r_{n_z}$.
for $t_k \in [t_0, t_f]$ do
    $j = 0$;
    solve system (3.5) and get matrix $S(t_k)$;
    find $Dg(t_k) := \left(\frac{\partial g_{n1}}{\partial z_{n_2}}(z(t_k), t_k)\right)_{n_1=1,\dots,n_g,\ n_2=1,\dots,n_z}$;
    compute $DS(t_k) := Dg(t_k) * S(t_k)$;
    $r := +\infty$;
    for $n_1 = 1, \dots, n_g$ do
        if total radius then
            $r = \min(r, -g_{n_1}(z(t_k), t_k)/\|DS_{n_1}(t_k)\|)$;
        else
            for $n_2 = 1, \dots, n_z$ do
                $r_{n_2}(h) = \min(r_{n_2}(h), -g_{n_1}(z(t_k), t_k)/abs(DS_{n_1,n_2}(t_k)))$;
            end
        end
    end
end
```

Algorithm 11: Radius estimation implemented with MATLAB.

Radius estimation of ROCHJ trajectories. To understand better how the total radius works, the scenario in Subsection 2.1 is considered with the following modifications:

- the obstacle initial position is $X_1 = (x_1, y_1) = (0.0[m], -1.75[m])$;
- the reference initial state is $z_0 = (x_0, y_0, \psi_0, v_0) = (-30.0[m], -1.75[m], 0.0, 20.0[m/s])$.

The backward reachable set in a time horizon $T = 2[s]$ and the minimum time optimal trajectory with $\tau = 1.452[s]$ are depicted in Figure 3.16.

Reachable set $\mathcal{BR}_2^f$. (Optimal trajectory also represented with a black line)

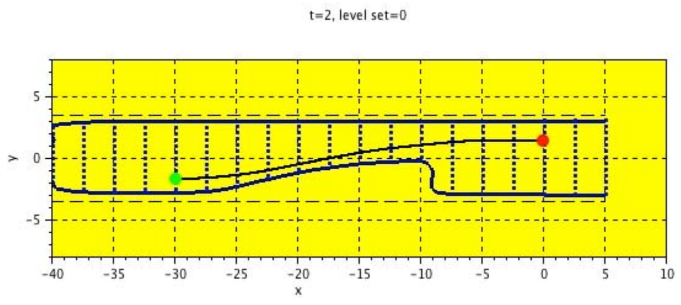


Figure 3.16: Reachable set $\mathcal{BR}_2^f$, obtained with $N_x = 18$ grid points on the interval $[-40, 5]$ and $N_y = 16$ grid points on the interval $[-4, 4]$.

The total radius estimation for the reference trajectory in Figure 3.16 is computed in column *radius* as the minimum over the total radius estimations computed at each time step in column $radius_k$ of Table 3.1. The column g is the state constraint value computed in the state $(x(t(k)), y(t(k)), \psi(t(k)), v(t(k)))$ at time $t(k)$ for the iteration step $k \in [1, \dots, N]$. As one can see, the state constraint is always satisfied since it is always negative as requested in Problem 2.1. Column $\|Dg \cdot S\|$ gives the factor that divides the state constraints value, for each time step $t(k)$, to compute the total radius as shown by the formula in Algorithm 11:

$$r_k = -g(z(t(k)), t(k))/\|DS(t(k))\|, \quad \text{with} \;\; DS(t(k)) = Dg(t(k)) \cdot S(t(k))$$

and thus the total radius will be $r = \min\{r_1, \dots, r_N\}$.

$x(t(k))$	$y(t(k))$	$\psi(t(k))$	$v(t(k))$	$t(k)$	k	g	$\|Dg \cdot S\|$	$radius_k$	$radius$
-30	-1.75	0	20	0	0	-3.5	2.002	1.748	1.748
-29	-1.737	0.025	20	0.05	20	-2.5	2.295	1.089	1.089
-28	-1.7	0.05	20	0.1	40	-2.551	3.732	0.684	0.684
-27	-1.637	0.075	20	0.15	60	-2.653	5.551	0.478	0.478
-26	-1.55	0.1	20.004	0.2	80	-2.806	7.477	0.375	0.375
-25	-1.445	0.114	20.079	0.25	100	-3.002	9.434	0.318	0.318
-24	-1.318	0.139	20.154	0.301	120	-3.235	11.424	0.283	0.283
-23	-1.166	0.164	20.23	0.351	140	-3.519	13.426	0.262	0.262
-22	-0.987	0.189	20.265	0.401	160	-3.856	15.436	0.25	0.25
-21	-0.787	0.2	20.34	0.451	180	-4.247	17.434	0.244	0.244
-20	-0.584	0.2	20.41	0.501	200	-4.653	19.422	0.24	0.24
-19	-0.381	0.2	20.445	0.551	220	-5.058	21.412	0.236	0.236
-18	-0.18	0.192	20.52	0.601	240	-5.467	23.396	0.234	0.234
-17	0.002	0.17	20.594	0.65	260	-5.422	1.799	3.015	0.234
-16	0.186	0.187	20.668	0.7	280	-4.415	1.965	2.247	0.234
-14.999	0.363	0.163	20.741	0.749	300	-3.425	2.111	1.622	0.234
-13.999	0.516	0.146	20.815	0.797	320	-2.432	2.244	1.084	0.234
-12.999	0.661	0.134	20.887	0.846	340	-1.437	2.376	0.605	0.234
-12.549	0.719	0.123	20.92	0.868	349	-1.034	18.086	0.057	0.057
-11.999	0.783	0.11	20.959	0.894	360	-1.091	18.629	0.059	0.057
-10.999	0.886	0.1	21.031	0.942	380	-1.088	18.63	0.058	0.057
-9.999	0.986	0.1	21.098	0.99	400	-1.189	19.628	0.061	0.057
-8.999	1.086	0.1	21.108	1.037	420	-1.289	20.627	0.062	0.057
-7.999	1.186	0.093	21.179	1.085	440	-1.391	21.622	0.064	0.057
-6.999	1.267	0.069	21.25	1.132	460	-1.537	2.905	0.529	0.057
-5.998	1.324	0.046	21.32	1.179	480	-2.525	2.981	0.847	0.057
-4.998	1.358	0.022	21.386	1.226	500	-3.262	51.118	0.064	0.057
-3.998	1.369	0	21.4	1.273	520	-3.262	53.138	0.061	0.057
-2.998	1.369	0	21.47	1.32	540	-3.262	55.137	0.059	0.057
-1.998	1.369	0	21.489	1.366	560	-3.262	57.136	0.057	0.057
-0.998	1.369	0	21.465	1.413	580	-3.262	59.134	0.055	0.055
-0.148	1.369	0	21.446	1.452	597	-3.262	59.834	0.055	0.054

Table 3.1: Time evolution every 20 iterations of the state of the reference vehicle and of the total radius estimation are listed. Iterations 349 and 597 are also exploited for clarification.

Radius estimation of OCPIDDAE trajectories. The radius estimation for the reference trajectory considered in Figure 3.3 is provided straightforward.

```
Minimum constraint violation trajectory:

The radius of parameter 1 is 1.63652204 units
The radius of parameter 2 is 0.45000000 units
The radius of parameter 3 is 0.04150295 units
The radius of parameter 4 is 2.56418685 units
The radius of parameter 5 is 1.08372933 units
The total radius is 0.00209028 units
```

The constraint violation value $w_2 = -0.45$ meters, will allow a larger error in the initial state since the constraints are minimized and not only satisfied. This is shown in the next data, where the radius estimation for the minimum time trajectory in Figure 3.4 is computed and where the constraint violation is fixed to $w_2 = 0.0$ and not optimized.

```
Minimum time trajectory:

The radius of parameter 1 is 0.94731495 units
The radius of parameter 2 is 0.28520537 units
The radius of parameter 3 is 0.03015129 units
The radius of parameter 4 is 1.38143485 units
The radius of parameter 5 is 0.79722614 units
The total radius is 0.00559723 units
```

Chapter V

Verification Tool

Contents

1 The Virtual Test Maker Software.

The software package that collects results of Chapter IV and employes them to verify collision avoidance systems is here explained in details. It is developed in a Linux platform, but it can be extended to Windows systems as well. It is organized in three phases:

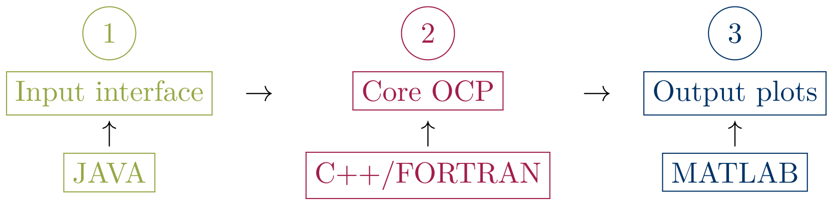


Table 1.1: Structure of Virtual Test Maker.

Explanation of Table 1.1 is given here.

① The JAVA interface is a user friendly interface to define the car traffic scenario parameters and to choose the desired output object.

② The optimization core using ROC-HJ and OCPID-DAE1 software packages is based on C++ and Fortran programming language respectively. It will solve the optimal control problem of the form (3.2) of Chapter II, with constraints of the form (2.85) and (2.86) of Chapter II.

③ The MATLAB plotting helps the visualization of the output data files obtained from the ROC-HJ and OCPID-DAE1 software packages. Moreover a sensitivity analysis is performed.

The Virtual Test Maker (VTM) tool is collected in folder `GUI_0.xx` with structure given in Figure 1.1.

```
🗁Defaults
 ├──🗁BackwardReachableSet
 │   └──🗁ROCHJ
 │       └──…continues in Figure 1.2
 ├──🗁FinalReachableSet
 │   └──🗁OCPIDDAE
 │       └──…continues in Figure 1.2
 ├──🗁OptimalAvoidanceTrajectory
 │   ├──🗁ROCHJ
 │   │   └──…continues in Figure 1.2
 │   └──🗁OCPIDDAE
 │       └──…continues in Figure 1.2
 ├──🗁VerifyAwvAlgorithm
 │   ├──🗁LastPointToBrakeSteer
 │   │   └──…continues in Figure 1.2
 │   ├──🗁OptimalTrajectory
 │   │   └──…continues in Figure 1.2
 │   └──🗁ReachableSet
 │       └──…continues in Figure 1.2
 └──🗁VerifyHotAlgorithm
     ├──🗁MATLAB
     │   └──…continues in Figure 1.2
     └──🗁ROCHJ
         └──…continues in Figure 1.2
```

Figure 1.1: Folder structure behind the user interface. Each level corresponds to a drop-down menu of the interface window in Figure 1.4.

Folders shown in Figure 1.1 contain the source code to answer to a question posed by the user. Each folder corresponds to an answer.

1. The **backward reachable set** is defined in 3.4 of Chapter II and it is computed with ROC-HJ software following Algorithm 5 of Chapter IV.
2. The **final reachable set** is the reachable set defined in 3.3 of Chapter II and it is computed with OCPID-DAE1 software following Algorithm 3 of Chapter IV. The FIACCO-sensitivity and the ODE-sensitivity (Algorithm 9 Chapter IV) are also computed.
3. The **optimal avoidance trajectory** is defined in 3.2 of Chapter II and it is computed with OCPID-DAE1 software following Algorithm 2 of Chapter IV. The FIACCO-sensitivity (Algorithm 6 Chapter IV) and the ODE-sensitivity (Algorithm 7 Chapter IV) are computed. The ROC-HJ software computes minimum time trajectories with a 4D vehicle model and more flexible scenario configurations, see Section 2 of Chapter IV and Algorithm 1 Chapter II.
4. The **verification of CAB algorithm** will be treated in details in Section 2. As shown in Figure 1.1, this question does not have a software choice but a goal choice, the reason for this is that the software used is always OCPID-DAE1 software, but several explicit objectives are accessible. In particular the `OptimalTrajectory` runs Item (2) with OCPID-DAE1 software and `ReachableSet` runs Item (3).
5. The **verification of CABS algorithm** will be treated in details in Section 3. The `ROCHJ` choice is exactly computation at Item (1). The `MATLAB` choice performs the Algorithm 8 of Chapter IV) and radius estimation of Algorithm 11 of Chapter IV).

The `Default` folder structure goes forward as indicated in Figure 1.2, with the choice of a road geometry. Each road geometry contains then the source code for the Core Phase in `C++` or `FORTRAN` folders, the parameter files describing the car traffic scenario as entered from the user friendly interface for the Input Phase in `GUI` folder and the code plotting the output objects and performing the sensitivity analysis for the Output Phase in `MATLAB` folder.

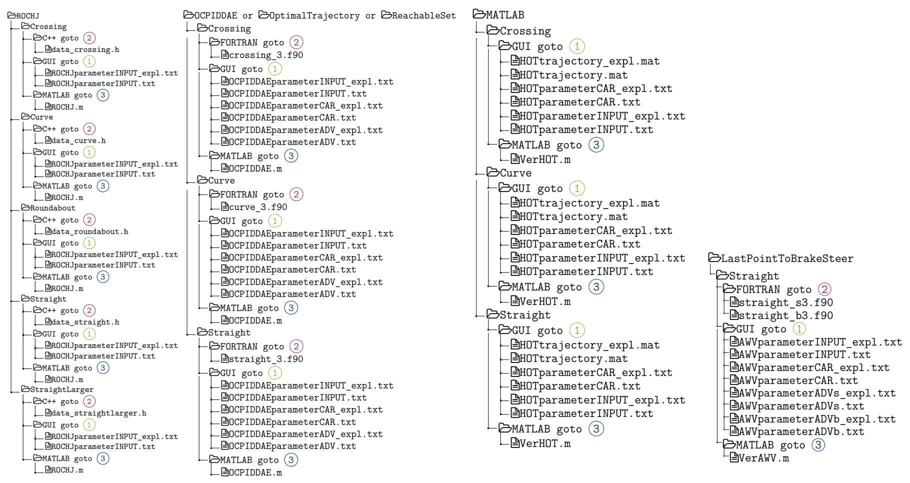


Figure 1.2: First column `ROCHJ` folder structure, second column `OCPIDDAE/OptimalTrajectory/ReachableSet` folder structure, third column `LastPointToBrakeSteer` folder structure, fourth column `MATLAB` folder structure.

In the following paragraphs folders `GUI`, `C++`, `FORTRAN`, `MATLAB` will be discussed in relation to the three phases mentioned at the beginning of this section.

- The data inserted by the user in the window interface, are saved to folders named `GUI` in several files depending on the chosen option among the ones in the `Configuration` drop-down menu. Each file collects internally the parameters in groups that are named as the options of the `Group` drop-down menu.

- ROC-HJ software is contained in each folder named `C++` and it is designed for several road geometries as crossing, curve, roundabout, straight and straight larger. It is used for backward reachable set computations and minimum time trajectories, moreover it is used in the verification of the CABS algorithm. This software belongs to the Core Phase described in ②.

- The OCPID-DAE1 software is contained in folders named `FORTRAN` and is designed for road geometries such as crossing, curve, straight. It is used for reachable sets, optimal trajectories for several minimum criteria and the last point to brake and last point to steer, the latest is used for the verification of CAB algorithm. This software belongs to the Core Phase described in ②.

- Four MATLAB files are used in all questions for plots or computational reasons. The file `OCPIDDAE.m` plots the output of OCPID-DAE1 software, while the file `ROCHJ.m` plots the results of ROC-HJ software. The file `VerAWV.m` shows pictures related to the last point to brake and last point to steer for the verification of CAB algorithm, and the file `VerHOT.m` computes and plots the sensitivity and radius estimation of a CABS trajectory. This software belongs to the Output Phase described in (3).

The first row of Figure 1.3 summarizes the folder structure here presented and relate it to the software architecture in the second row of the same figure.

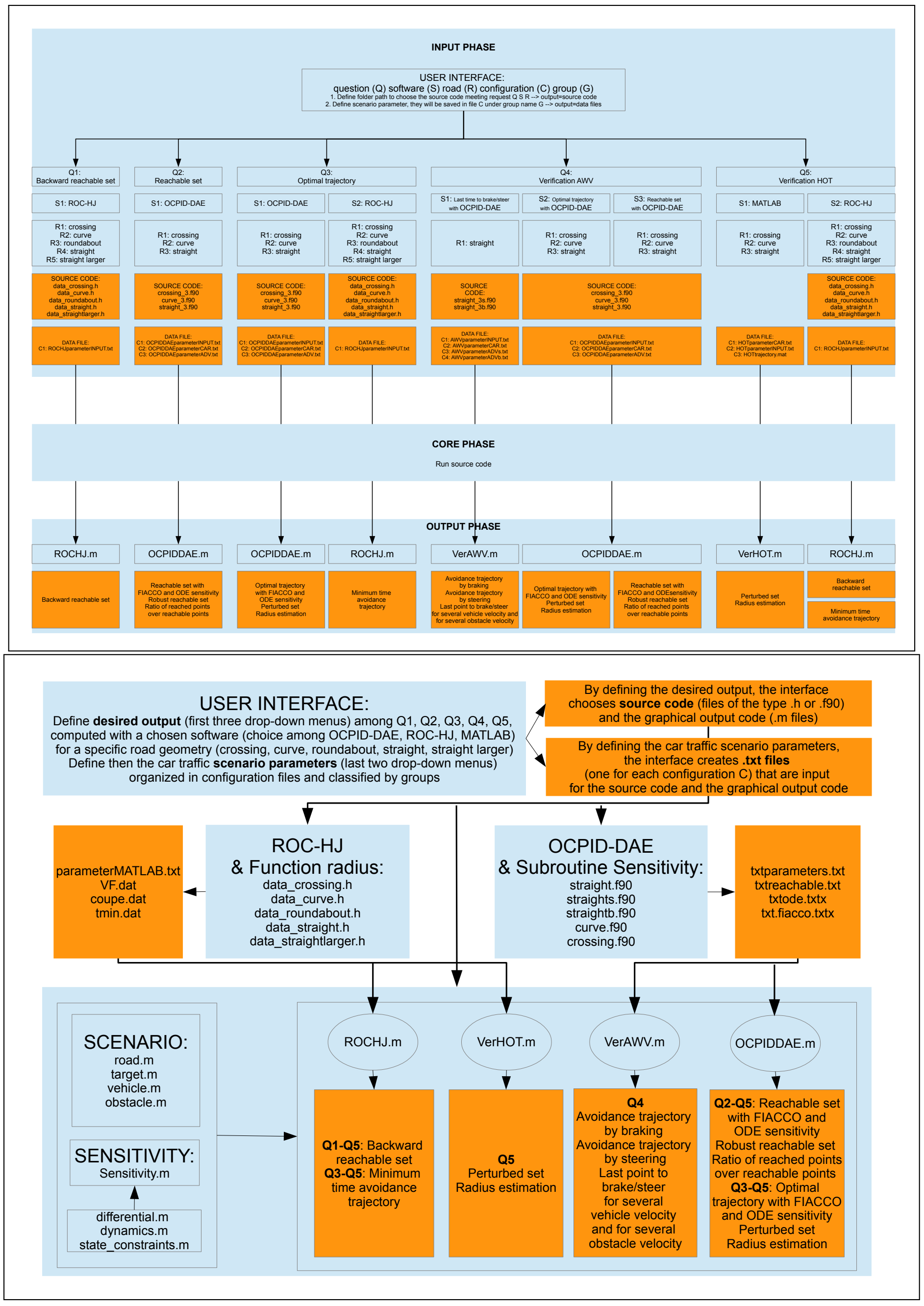
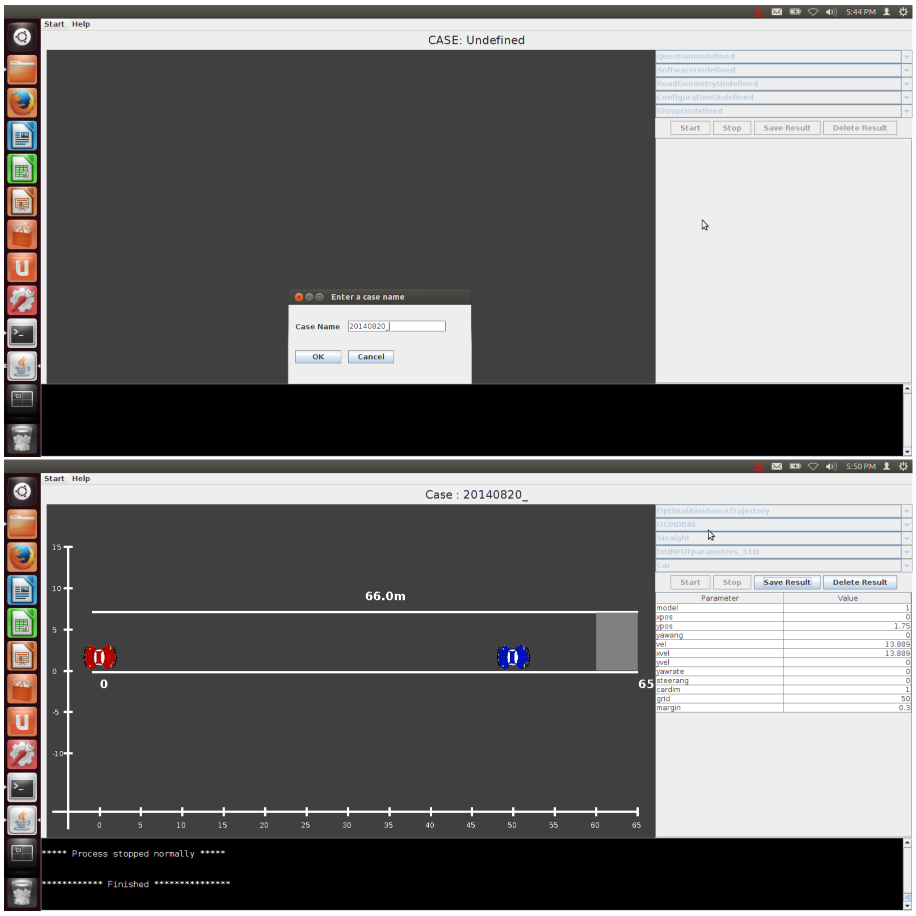

INPUT PHASE
USER INTERFACE:
question (Q) software (S) road (R) configuration (C) group (G)
1. Define folder path to choose the source code meeting request Q S R --> output=source code
2. Define scenario parameter, they will be saved in file C under group name G --> output=data files
Q1: Backward reachable set
Q2: Reachable set
Q3: Optimal trajectory
Q4: Verification AWV
Q5: Verification HOT
S1: ROC-HJ
S1: OCPID-DAE
S1: OCPID-DAE
S2: ROC-HJ
S1: Last time to brake/steer with OCPID-DAE
S2: Optimal trajectory with OCPID-DAE
S3: Reachable set with OCPID-DAE
S1: MATLAB
S2: ROC-HJ
R1: crossing R2: curve R3: roundabout R4: straight R5: straight larger
R1: crossing R2: curve R3: straight
R1: straight
SOURCE CODE: data_crossing.h data_curve.h data_roundabout.h data_straight.h data_straightlarger.h
SOURCE CODE: crossing_3.f90 curve_3.f90 straight_3.f90
SOURCE CODE: straight_3s.f90 straight_3b.f90
DATA FILE: C1: ROCHJparameterINPUT.txt
DATA FILE: C1: OCPIDDAEparameterINPUT.txt C2: OCPIDDAEparameterCAR.txt C3: OCPIDDAEparameterADV.txt
DATA FILE: C1: AWVparameterINPUT.txt C2: AWVparameterCAR.txt C3: AWVparameterADVs.txt C4: AWVparameterADVb.txt
DATA FILE: C1: HOTparameterCAR.txt C2: HOTparameterINPUT.txt C3: HOTtrajectory.mat
CORE PHASE
Run source code
OUTPUT PHASE
ROCHJ.m
OCPIDDAE.m
VerAWV.m
VerHOT.m
Backward reachable set
Reachable set with FIACCO and ODE sensitivity Robust reachable set Ratio of reached points over reachable points
Optimal trajectory with FIACCO and ODE sensitivity Perturbed set Radius estimation
Minimum time avoidance trajectory
Avoidance trajectory by braking Avoidance trajectory by steering Last point to brake/steer for several vehicle velocity and for several obstacle velocity
Perturbed set Radius estimation
USER INTERFACE:
Define desired output (first three drop-down menus) among Q1, Q2, Q3, Q4, Q5, computed with a chosen software (choice among OCPID-DAE, ROC-HJ, MATLAB) for a specific road geometry (crossing, curve, roundabout, straight, straight larger) Define then the car traffic scenario parameters (last two drop-down menus) organized in configuration files and classified by groups
By defining the desired output, the interface chooses source code (files of the type .h or .f90) and the graphical output code (.m files)
By defining the car traffic scenario parameters, the interface creates .txt files (one for each configuration C) that are input for the source code and the graphical output code
parameterMATLAB.txt VF.dat coupe.dat tmin.dat
ROC-HJ & Function radius: data_crossing.h data_curve.h data_roundabout.h data_straight.h data_straightlarger.h
OCPID-DAE & Subroutine Sensitivity: straight.f90 straights.f90 straightb.f90 curve.f90 crossing.f90
txtparameters.txt txtreachable.txt txtode.txtx txt.fiacco.txtx
SCENARIO: road.m target.m vehicle.m obstacle.m
SENSITIVITY: Sensitivity.m
differential.m dynamics.m state_constraints.m
ROCHJ.m
VerHOT.m
VerAWV.m
OCPIDDAE.m
Q1-Q5: Backward reachable set Q3-Q5: Minimum time avoidance trajectory
Q5 Perturbed set Radius estimation
Q4 Avoidance trajectory by braking Avoidance trajectory by steering Last point to brake/steer for several vehicle velocity and for several obstacle velocity
Q2-Q5: Reachable set with FIACCO and ODE sensitivity Robust reachable set Ratio of reached points over reachable points Q3-Q5: Optimal trajectory with FIACCO and ODE sensitivity Perturbed set Radius estimation

Figure 1.3: In first row the folder structure containing the software is shown. In the second row the logic behind such folder is represented. In both pictures the three mentioned phases (input, core, output) are underlined.

Input interface In order to launch the interface from a Linux terminal, the user has to execute the file `startGUI.sh` by typing the command `./startGUI.sh`. This will execute the file `GUI.jar`. After the interface is launched the image pictured in Figure 1.4 will appear. In the left-upper corner the button `case` will allow to start your case simulation and save it in the same named folder.



Parameter	Value
model	1
xpos	0
ypos	1.75
yawang	0
vel	13.889
xvel	13.889
yvel	0
yawrate	0
steerang	0
cardim	1
grid	50
margin	0.3

```
***** Process stopped normally *****

************* Finished ****************
```

Figure 1.4: VTM user interface: to name the case simulation as first step (top picture). A same named folder will be created in folder `GUI_0.xx` with same structure as folder `Default`. After choosing the output and the scenario, the picture on bottom appears where the user can see in the shell on bottom if the run computed a solution. Moreover the user has the option to save the run or delete it.

On the left the chosen scenario will be graphically shown as soon as the scenario parameters are entered on the right side of the window interface. The drop-down menus mirror the folder structure described in Figure 1.1 and contained in the folder `Default`, see Figure 1.3.

- `Question:` The user is asked to choose a question that will define the output. The VTM tool is then accessing the folder with the same question name:

🗁`BackwardReachableSet`
🗁`FinalReachableSet`
🗁`OptimalAvoidanceTrajectory`
🗁`VerifyAwvAlgorithm`
🗁`VerifyHotAlgorithm`

- `Software:` The user specify the software that computes the answer to the previously chosen question and the folder of the same name is accessed by the interface:
  🗁`ROCHJ`
  🗁`OCPIDDAE`
  🗁`MATLAB`

- `RoadGeometry:` As seen in Section 2 of Chapter II, the choice of the road geometry is related to the optimizer (ROC-HJ and OCPID-DAE1). Indeed to different optimizers correspond different regularity requirements on the state constraints. In the third drop-down menu the user can choose the function that models the road state constraint and the folder of the same name is accessed by the interface:
  🗁`Crossing`
  🗁`Curve`
  🗁`Roundabout`
  🗁`Straight`
  🗁`StraightLarger`

- `Configuration:` Each configuration writes a data files inside the `GUI` folder with the parameters specified by the user:
  🗎`ROCHJparameterINPUT.txt`
  🗎`OCPIDDAEparameterINPUT.txt`
  🗎`AWVparameterINPUT.txt`
  🗎`HOTparameterINPUT.txt`
  🗎`OCPIDDAEparameterADV.txt`
  🗎`AWVparameterADVs.txt`
  🗎`AWVparameterADVb.txt`
  🗎`OCPIDDAEparameterCAR.txt`
  🗎`AWVparameterCAR.txt`
  🗎`HOTparameterCAR.txt`
  🗎`HOTtrajectory.mat`

- `Group:` The parameters in each data file in the `GUI` folder are organized in groups. Once a group is chosen, the parameters are listed in the interface and tooltips are shown whenever the user places the mouse pointer on the parameter name.

The detailed description of the `Configuration` files and `Group` data is given straightforward.

- The files

  `ROCHJparameterINPUT.txt`, `OCPIDDAEparameterINPUT.txt`,
  `AWVparameterINPUT.txt`, `HOTparameterINPUT.txt`

  collect the scenario data and includes road margins, obstacle geometry and motion, reference vehicle geometry and initial states, target definition, states and controls boundaries and computational grid refinement. The related files

  `ROCHJparameterINPUT_expl.txt`, `OCPIDDAEparameterINPUT_expl.txt`,
  `AWVparameterINPUT_expl.txt`, `HOTparameterINPUT_expl.txt`

  include tooltips and the definition of each parameter group.

- The files

  `OCPIDDAEparameterADV.txt`, `AWVparameterADVs.txt`,
  `AWVparameterADVb.txt`

  collect advanced data for software OCPID-DAE1 such as states, controls and parameters boundaries and initialization. The related files

  `OCPIDDAEparameterADV_expl.txt`, `AWVparameterADVs_expl.txt`,
  `AWVparameterADVb_expl.txt`

  include tooltips and the definition of each parameter group.

- The files

  `OCPIDDAEparameterCAR.txt`, `AWVparameterCAR.txt`, `HOTparameterCAR.txt`

  collect vehicle data for software OCPID-DAE1 in particular whenever the single track model is chosen. The related files

  `OCPIDDAEparameterCAR_expl.txt`, `AWVparameterCAR_expl.txt`,
  `HOTparameterCAR_expl.txt`

  include tooltips and the definition of each parameter group.

- The file `HOTtrajectory.mat` is a matrix whose column are the states, controls and optimized parameters, and whose rows are their values at each time step. The related file `HOTtrajectory_expl.mat` includes tooltips on the structure of such matrix.

By clicking on the `Start` button the core computation will start, running either ROC-HJ software or OCPID-DAE1 software, depending on the output requested. On the left bottom of the window interface, a terminal script will appear to follow the correctness of the core execution. Once the simulation is launched the user can access three next steps:

```
Start
├─Stop: the user can interrupt the simulation
├─Save Result: the user can save the simulation in a folder under a user
│ chosen name, i.e. YYYYMMDD_xxx
└─Delete Result: the user can delete the run data, for instance whenever
  an error occurred in the core execution
```

The folder `YYYYMMDD_xxx` will have the same structure as the `Dafault` folder depicted in Figure 1.1.

Core Optimal Control Problem By pressing the `Start` button, the interface will start the Core Phase running the source code to solve the optimal control problem. The source code is contained in folders `C++`, `FORTRAN`, `MATLAB` (see Figure 1.2) depending on the chosen software package. Two software packages are computing numerical solution of the optimal control problem defined in 3.1 of Chapter II.

Software package ROC-HJ in folder `C++`, solves n-dimensional Hamilton Jacobi Bellman equations by finite difference methods, or semi-lagrangian methods, via sequential or parallel code. For deeper informations, see [16]. The file `data_xxx.h` is the main input file, it reads the parameter values from `ROCHJparameterINPUT.txt` and contains the functions that define the equation to be solved:

```
C++
└─data_car
  ├─data_crossing.h
  ├─data_curve.h
  ├─data_roundabout.h
  ├─data_straight.h
  └─data_straightlarger.h
```

Figure 1.5: Organization of source file for ROC-HJ software.

The optimal control problem of the form 3.1 of Chapter II, with constraints in the form of (2.85) of Chapter II is implemented in the file interface `data_xxx.h` as follows:

- the dynamics is implemented in function `inline void dynamics` and its gradient is computed in `inline void Dz_dynamics` for radius estimation purposes;

- the Hamiltonian requires functions `inline void compute_Hconst`, `inline double Hnum` for numerical stability;
- the constraints are defined in function `inline double v0`, where target constraints are herein implemented, while state constraints need the auxiliary function `inline double g_obstacle` that requires functions `inline double g_obstacle_point` and `inline double g_vehicle_point` for guaranteeing the avoidance of obstacles-vehicles and the cruise of vehicle on the road; the definition of the road set is contained in function `inline double g_road` and the definition of the obstacle motion is in function `inline void g_obstacle_motion`.

The functions `void compute_radius`, `void compute_radius_d` implemented in the interface file `data_xxx.h` is an extension of the software package. It is computing the radius estimation according to Algorithm 11 of Chapter II. The output files are:

- `VF.dat` contains the final value function v at the end of the computation. The file is structured as follows, on each line :

  ```
  i1 i2 .... in val
  ```

  where `val` corresponds to the value $v(t_f; z_0)$ at mesh point $z_0 = (z_{i_1}, \dots, z_{i_n})$.
- `coupe.dat` contains the projection on a 2d space of a n-dimensional problem with $n > 2$, in this case the plot of `coupe.dat` is a “cut” in some particular 2d plane where the results are visualized.
- `tmin.dat` : minimal time function, structured as `VF.dat`.
- `parameterMATLAB.txt` contains the parameters defined by the user to define the car traffic scenario to plot the solution.

The software package OCPID-DAE1 in folder `FORTRAN` is designed for the numerical solution of optimal control problems and parameter identification problems of the form 3.1 of Chapter II, with constraints in the form of (2.86) of Chapter II, see [48] for details. The file `OCPIDDAE.f90` is the main input file, saved in folder `FORTRAN`. It reads the parameter values from

`OCPIDDAEparameterINPUT.txt`, `OCPIDDAEparameterCAR.txt`,
`OCPIDDAEparameterADV.txt`

and contains the functions that define the equation to be solved. The optimal control problem is implemented in the file interface `OCPIDDAE.f90` as follows:

- `SUBROUTINE OBJ`: objective function;
- `SUBROUTINE DAE`: dynamics;
- `SUBROUTINE NLCSTR`: state constraints modeling obstacle motion and geometry and road geometry;
- `SUBROUTINE BDCOND`: target constraints.

The output files are:

- `OCODE01` contains time, state, control, parameter, state constraints:

$$\begin{pmatrix} t_0 & z_1(t_0) & \dots & z_{n_z}(t_0) & u_1(t_0) & \dots & u_{n_u}(t_0) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ t_N & z_1(t_N) & \dots & z_{n_z}(t_N) & u_1(t_N) & \dots & u_{n_u}(t_N) \end{matrix} \\ \begin{matrix} w_1(t_0) & \dots & w_{n_w}(t_0) & g_1(t_0) & \dots & g_{n_g}(t_0) \\ \vdots & & \vdots & \vdots & & \vdots \\ w_1(t_N) & \dots & w_{n_w}(t_N) & g_1(t_N) & \dots & g_{n_g}(t_N) \end{pmatrix} \tag{1.1}$$

  where $t_0, \dots, t_N$ is the time grid;

- `OCODE02` contains time, sensitivities:

$$\begin{pmatrix} t_0 & \left\{\frac{\partial z_1}{\partial z_{0j}}(t_0) \cdots \frac{\partial z_{n_z}}{\partial z_{0j}}(t_0)\right\}_{j=1,\dots,n_z} & \left\{\frac{\partial z_1}{\partial u_{0j}}(t_0) \cdots \frac{\partial z_n}{\partial u_{0j}}(t_0)\right\}_{j=1,\dots,n_u} \\ \vdots & \vdots & \vdots \\ t_N & \left\{\frac{\partial z_1}{\partial z_{0j}}(t_N) \cdots \frac{\partial z_{n_z}}{\partial z_{0j}}(t_N)\right\}_{j=1,\dots,n_z} & \left\{\frac{\partial z_1}{\partial u_{0j}}(t_N) \cdots \frac{\partial z_n}{\partial u_{0j}}(t_N)\right\}_{j=1,\dots,n_u} \\ \dots & \left\{\frac{\partial z_1}{\partial u_{Nj}}(t_0) \cdots \frac{\partial z_n}{\partial u_{Nj}}(t_0)\right\}_{j=1,\dots,n_u} & \left\{\frac{\partial z_1}{\partial w_j}(t_0) \cdots \frac{\partial z_n}{\partial w_j}(t_0)\right\}_{j=1,\dots,n_w} \\ & \vdots & \vdots \\ \dots & \left\{\frac{\partial z_1}{\partial u_{Nj}}(t_N) \cdots \frac{\partial z_n}{\partial u_{Nj}}(t_N)\right\}_{j=1,\dots,n_u} & \left\{\frac{\partial z_1}{\partial w_j}(t_N) \cdots \frac{\partial z_n}{\partial w_j}(t_N)\right\}_{j=1,\dots,n_w} \end{pmatrix}$$

  (1.2)
  where $t_0, \dots, t_N$ is the time grid and $z(t_0) = z_0 = (z_{01}, \dots, z_{0n_z})$ is the nominal initial value.

- `OCODE03` contains time, derivative of state (only for some integrators);
- `OCODE04` contains number and value of optimization variables;
- `ADJOINT` contains time, state, adjoints, multipliers of discretized state constraints.

The `SUBROUTINE sensitivity` is an extension to the software package. It will use the output files of OCPID-DAE1 software to compute reachable sets, optimal trajectories, following Algorithms 3, 2 of Chapter II. The ODE-sensitivity and FIACCO-sensitivity is computed in `OCODE02` and in the variable `DSOLREALTIME` (that contains the sensitivities of the optimal solution vector z with respect to the parameters p) and in `SUBROUTINE sensitivity` the perturbed optimal trajectory and the perturbed reachable set are computed implementing Algorithms 7, 6, 9 of Chapter II.

The `SUBROUTINE TXT` will save the following data files:

- The file `txtreachable.txt` and all files of the type `txtreachablexx.txt`, contain time, states, controls, parameters for each point of the reachable grid on the xy-plane:

$$\begin{aligned} t(h), &\{\{z(i,h,j), i=1,\dots,N_xN_y\}, j=1,\dots,n_z\}, \\ &\{\{u(i,h,j), i=1,\dots,N_xN_y\}, j=1,\dots,n_u\}, \\ &\{\{w(i,h,j), i=1,,\dots,N_xN_y\}, j=1,\dots,n_w\} \end{aligned} \tag{1.3}$$

  where N_x is the number of grid points on x and N_y is the number of grid points in y, moreover $h=0,\dots,N$ with $t_0,\dots,t_N$ time grid.

- Files `txtfiacco.txt`, `txtode.txt` contain time, states, controls, parameters for each point of the reachable grid on the xy-plane and for each perturbation parameter $p_1,\dots,p_{n_p}$

$$\begin{aligned} t(h), &\left\{\{\{z(i,h,j,k), i=1,\dots,N_xN_y\}, j=1,\dots,n_z\}_{k=1,\dots,n_p}\right\}, \\ &\left\{\{\{u(i,h,j,k), i=1,\dots,N_xN_y\}, j=1,\dots,n_u\}_{k=1,\dots,n_p}\right\}, \\ &\left\{\{\{w(i,h,j,k), i=1,,\dots,N_xN_y\}, j=1,\dots,n_w\}_{k=1,\dots,n_p}\right\} \end{aligned} \tag{1.4}$$

  where N_x is the number of grid points on x and N_y is the number of grid points in y, moreover $h=0,\dots,N$ with $t_0,\dots,t_N$ time grid and n_p is the dimension of the perturbed parameter $p=(p_1,\dots,p_{n_p})$.

- The file `txtparameter.txt` contains the parameters defined by the user to define the car traffic scenario to plot the solution.

Output plots The MATLAB will be manually launched by the user by typing the name of one of the four main files in the MATLAB console. The input files are the Core Phase output and the Input Phase output (orange boxes in Figure 1.3). The graphical output are the answers to the initial question chosen by the user in the JAVA interface.

- The `ROCHJ.m` file will read the output files of ROC-HJ software and will plot the backward reachable set or the optimal trajectory depending on the question given as input in the JAVA interface. The simulations are described in Section 2 of Chapter IV.

- The `OCPIDDAE.m` file will plot the optimal trajectory with states, controls and parameters pictures, the reachable set and the associated FIACCO and ODE perturbation. The `sensitivity.m` file is called by `OCPIDDAE.m` to compute the perturbed reachable set and the radius estimation. The ratio of point reached over the reachable grid points is provided. Numerical examples are given in Section 1 and 3 of Chapter IV.
- The file `VerAWV.m` will give a verification tool for Collision Avoidance by Braking algorithms (CAB). It plots the last point to brake and last point to steer for several obstacle and vehicle velocities. It will be treated in Section 2.
- The file `VerHOT.m` is computing the perturbed set and the radius estimation using `sensitivity.m` file, of a trajectory computed with Collision Avoidance by Braking and Steering algorithm (CABS). The verification procedure for such algorithm is discussed in Section 3.

2 Collision avoidance by braking algorithms

The collision avoidance by braking algorithm is named CAB and gives different kind of warnings to the driver when a collision is likely to occur. The scenario is including one obstacle driving in the direction of the reference vehicle, in front of it. Due to a hard braking maneuver of the obstacle or being the obstacle stuck in the reference vehicle lane, the collision avoidance system warns the driver of the reference vehicle with three different warnings. An acoustical and braking warnings capture the driver attention that hopefully will perform a steering and braking maneuver to avoid the collision. If the driver does not react to these minor warnings, an automatic braking maneuver is performed to avoid the obstacle or at least mitigate the collision with it.

The input of CAB-algorithm is the initial state of the reference vehicle (position, acceleration, velocity) and the initial data of the obstacle in front of it (position, velocity, acceleration). The obstacle motion is also given as input and it can be decided a priori (i.e. linear motion) or real motion data can be given for each time step. The output is a matrix with the obstacle motion data, vehicle motion data and the given warning, for each time step. Warning 0 means that nothing is given, warning 1 corresponds to an acoustical signal, warning 2 is a braking signal and warning 3 performs an automatic braking maneuver until the collision is avoided or mitigated. In Table 2.1 and 2.2 the structure of the output is shown. In particular for Table 2.2 the obstacle motion is simulated, i.e. linear motion is implemented. A first algorithmic mistake is detected by looking at the simulation, that does not correspond to reality since once the obstacle reaches zero velocity the simulation shows that the obstacle drives backwards, a phenomenon that does not happens in real life scenarios.

The output in Tables 2.1 and 2.2 can be summarized with Figure 2.1, where the warning levels are identified with distance intervals between the position of the vehicle and the position of the obstacle.

time	sens	Reaction Time	Warn Level	ttc	Ego vel	Ego lw	Ego delta lw	Ego accelerator	Ego indicator l	Ego indicator r	Dist x	Dist y	Obj velx	Obj vely	Rel vel	Acc x	Acc y	Yaw rate	Stw der	Ego sens	Lat stab	Lon stab	Obj ax	Obj ay
3.67	0.63	1.28	**0**	-1.00	50	0	0	50	0	0	48.89	0	0	0	50	0	0	0	0	0.63	1	1	0	0
3.70	0.64	1.29	**0**	-1.00	50	0	0	50	0	0	48.33	0	0	0	50	0	0	0	0	0.64	1	1	0	0
3.75	0.65	1.30	**0**	-1.00	50	0	0	50	0	0	47.78	0	0	0	50	0	0	0	0	0.65	1	1	0	0
3.78	0.66	1.31	**0**	-1.00	50	0	0	50	0	0	47.22	0	0	0	50	0	0	0	0	0.66	1	1	0	0
3.83	0.66	1.31	**0**	-1.00	50	0	0	50	0	0	46.67	0	0	0	50	0	0	0	0	0.66	1	1	0	0
3.86	0.67	1.32	**0**	-1.00	50	0	0	50	0	0	46.11	0	0	0	50	0	0	0	0	0.67	1	1	0	0
3.91	0.68	1.33	**0**	-1.00	50	0	0	50	0	0	45.56	0	0	0	50	0	0	0	0	0.68	1	1	0	0
3.94	0.68	1.34	**0**	3.75	50	0	0	50	0	0	45.00	0	0	0	50	0	0	0	0	0.68	1	1	0	0
4.02	0.69	1.35	**1**	3.50	50	0	0	50	0	0	43.89	0	0	0	50	0	0	0	0	0.69	1	1	0	0
4.11	0.71	1.37	**1**	3.35	50	0	0	50	0	0	42.78	0	0	0	50	0	0	0	0	0.71	1	1	0	0
4.19	0.72	1.38	**1**	3.25	50	0	0	50	0	0	41.67	0	0	0	50	0	0	0	0	0.72	1	1	0	0
4.27	0.73	1.40	**1**	3.10	50	0	0	50	0	0	40.56	0	0	0	50	0	0	0	0	0.73	1	1	0	0
4.34	0.74	1.41	**1**	2.95	50	0	0	50	0	0	39.44	0	0	0	50	0	0	0	0	0.74	1	1	0	0
4.42	0.75	1.42	**1**	2.85	50	0	0	50	0	0	38.33	0	0	0	50	0	0	0	0	0.75	1	1	0	0
4.50	0.76	1.43	**1**	2.75	50	0	0	50	0	0	37.22	0	0	0	50	0	0	0	0	0.76	1	1	0	0
4.63	0.77	1.44	**1**	2.60	50	0	0	50	0	0	35.56	0	0	0	50	0	0	0	0	0.77	1	1	0	0
4.70	0.78	1.46	**1**	2.50	50	0	0	50	0	0	34.44	0	0	0	50	0	0	0	0	0.78	1	1	0	0
4.78	0.79	1.47	**1**	2.40	50	0	0	50	0	0	33.33	0	0	0	50	0	0	0	0	0.79	1	1	0	0
4.86	0.80	1.48	**1**	2.30	50	0	0	50	0	0	32.22	0	0	0	50	0	0	0	0	0.80	1	1	0	0
4.91	0.80	0.64	**2**	2.50	50	0	0	50	0	0	31.67	0	0	0	50	0	0	0	0	0.80	1	1	0	0
4.98	0.81	0.65	**2**	2.40	50	0	0	50	0	0	30.56	0	0	0	50	0	0	0	0	0.81	1	1	0	0
5.06	0.81	0.65	**2**	2.25	50	0	0	50	0	0	29.44	0	0	0	50	0	0	0	0	0.81	1	1	0	0
5.14	0.82	0.66	**2**	2.15	50	0	0	50	0	0	28.33	0	0	0	50	0	0	0	0	0.82	1	1	0	0
5.25	0.83	0.66	**2**	2.05	50	0	0	50	0	0	27.22	0	0	0	50	0	0	0	0	0.83	1	1	0	0
5.33	0.84	0.67	**2**	1.90	50	0	0	50	0	0	26.11	0	0	0	50	0	0	0	0	0.84	1	1	0	0
5.42	0.84	0.67	**2**	1.80	50	0	0	50	0	0	25.00	0	0	0	50	0	0	0	0	0.84	1	1	0	0
5.50	0.85	0.67	**2**	1.70	50	0	0	50	0	0	23.89	0	0	0	50	0	0	0	0	0.85	1	1	0	0
5.55	0.85	0.68	**2**	1.65	50	0	0	50	0	0	22.78	0	0	0	50	0	0	0	0	0.85	1	1	0	0
5.63	0.86	0.68	**2**	1.55	50	0	0	50	0	0	21.67	0	0	0	50	0	0	0	0	0.86	1	1	0	0
5.73	0.87	0.69	**2**	1.45	50	0	0	50	0	0	20.56	0	0	0	50	0	0	0	0	0.87	1	1	0	0
5.80	0.00	0.00	**3**	1.60	50	0	0	50	0	0	19.44	0	0	0	50	0	0	0	0	0.87	1	1	0	0
5.84	0.00	0.00	**3**	1.50	50	0	0	50	0	0	18.89	0	0	0	50	0	0	0	0	0.87	1	1	0	0
5.89	0.00	0.00	**3**	1.45	50	0	0	50	0	0	18.33	0	0	0	50	0	0	0	0	0.87	1	1	0	0
5.92	0.00	0.00	**3**	1.40	50	0	0	50	0	0	17.78	0	0	0	50	0	0	0	0	0.88	1	1	0	0
5.98	0.00	0.00	**3**	1.35	50	0	0	50	0	0	17.22	0	0	0	50	0	0	0	0	0.88	1	1	0	0
6.03	0.00	0.00	**3**	1.25	50	0	0	50	0	0	16.11	0	0	0	50	0	0	0	0	0.88	1	1	0	0
6.09	0.00	0.00	**3**	1.15	50	0	0	50	0	0	15.56	0	0	0	50	0	0	0	0	0.88	1	1	0	0
6.13	0.00	0.00	**3**	1.10	50	0	0	50	0	0	15.00	0	0	0	50	0	0	0	0	0.89	1	1	0	0
6.14	0.00	0.00	**3**	1.05	50	0	0	50	0	0	14.44	0	0	0	50	0	0	0	0	0.89	1	1	0	0
6.19	0.00	0.00	**3**	1.00	50	0	0	50	0	0	13.89	0	0	0	50	0	0	0	0	0.89	1	1	0	0
6.25	0.00	0.00	**3**	0.95	50	0	0	50	0	0	13.33	0	0	0	50	0	0	0	0	0.89	1	1	0	0
6.27	0.00	0.00	**3**	0.90	50	0	0	50	0	0	12.78	0	0	0	50	0	0	0	0	0.89	1	1	0	0
6.31	0.00	0.00	**3**	0.85	50	0	0	50	0	0	12.22	0	0	0	50	0	0	0	0	0.90	1	1	0	0
6.34	0.00	0.00	**3**	0.80	50	0	0	50	0	0	11.67	0	0	0	50	0	0	0	0	0.90	1	1	0	0
6.39	0.00	0.00	**3**	0.75	50	0	0	50	0	0	11.11	0	0	0	50	0	0	0	0	0.90	1	1	0	0
6.44	0.00	0.00	**3**	0.75	50	0	0	50	0	0	10.56	0	0	0	50	0	0	0	0	0.90	1	1	0	0
6.48	0.00	0.00	**3**	0.70	50	0	0	50	0	0	10.00	0	0	0	50	0	0	0	0	0.90	1	1	0	0
6.52	0.00	0.00	**3**	0.65	50	0	0	50	0	0	9.44	0	0	0	50	0	0	0	0	0.91	1	1	0	0
6.56	0.00	0.00	**3**	0.60	50	0	0	50	0	0	8.89	0	0	0	50	0	0	0	0	0.91	1	1	0	0
6.59	0.00	0.00	**3**	0.55	50	0	0	50	0	0	8.33	0	0	0	50	0	0	0	0	0.91	1	1	0	0
6.63	0.00	0.00	**3**	0.50	50	0	0	50	0	0	7.78	0	0	0	50	0	0	0	0	0.91	1	1	0	0
6.67	0.00	0.00	**3**	0.45	50	0	0	50	0	0	7.22	0	0	0	50	0	0	0	0	0.91	1	1	0	0
6.70	0.00	0.00	**3**	0.40	50	0	0	50	0	0	6.67	0	0	0	50	0	0	0	0	0.92	1	1	0	0
6.75	0.00	0.00	**3**	0.35	50	0	0	50	0	0	6.11	0	0	0	50	0	0	0	0	0.92	1	1	0	0
6.80	0.00	0.00	**3**	0.35	50	0	0	50	0	0	5.56	0	0	0	50	0	0	0	0	0.92	1	1	0	0
6.83	0.00	0.00	**3**	0.30	50	0	0	50	0	0	5.00	0	0	0	50	0	0	0	0	0.92	1	1	0	0
6.89	0.00	0.00	**3**	0.25	50	0	0	50	0	0	4.44	0	0	0	50	0	0	0	0	0.92	1	1	0	0
6.91	0.00	0.00	**3**	0.20	50	0	0	50	0	0	3.89	0	0	0	50	0	0	0	0	0.92	1	1	0	0
6.97	0.00	0.00	**3**	0.15	50	0	0	50	0	0	3.33	0	0	0	50	0	0	0	0	0.92	1	1	0	0
7.00	0.00	0.00	**3**	0.15	50	0	0	50	0	0	2.78	0	0	0	50	0	0	0	0	0.93	1	1	0	0
7.05	0.00	0.00	**3**	0.10	50	0	0	50	0	0	2.22	0	0	0	50	0	0	0	0	0.93	1	1	0	0

Table 2.1: The initial reference vehicle velocity is $50[km/h]$ and the obstacle is fixed. The initial x-distance between vehicle and obstacle is $50[m]$ and the initial y-distance is $0[m]$. The acoustical warning 1 starts at $43.89[m]$ till $32.22[m]$ x-distance between the obstacle and the vehicle, the braking warning 2 starts at $31.67[m]$ till $20.56[m]$ x-distance and the automatic braking maneuver is performed from a x-distance of $19.44[m]$ till the collision is avoided or mitigated ($2.22[m]$ x-distance).

time	sens	Reaction Time	Warn Level	ttc	Ego vel	Ego lw	Ego delta lw	Ego accelerator	Ego indicator l	Ego indicator r	Dist x	Dist y	Obj velx	Obj vely	Rel vel	Acc x	Acc y	Yaw rate	Stw der	Ego sens	Lat stab	Lon stab	Obj ax	Obj ay
18.02	1	1.72	0	-1.00	50	0	0	50	0	0	49.87	0	38.56	0	11.44	0	0	0	0	1.00	1	1	-10	0
18.06	1	1.72	0	-1.00	50	0	0	50	0	0	49.73	0	37.12	0	12.88	0	0	0	0	1.00	1	1	-10	0
18.09	1	1.72	0	-1.00	50	0	0	50	0	0	49.57	0	35.68	0	14.32	0	0	0	0	1.00	1	1	-10	0
18.14	1	1.72	0	-1.00	50	0	0	50	0	0	49.40	0	34.24	0	15.76	0	0	0	0	1.00	1	1	-10	0
18.19	1	1.72	0	-1.00	50	0	0	50	0	0	49.20	0	32.80	0	17.20	0	0	0	0	1.00	1	1	-10	0
18.22	1	1.72	0	-1.00	50	0	0	50	0	0	49.00	0	31.36	0	18.64	0	0	0	0	1.00	1	1	-10	0
18.27	1	1.72	0	-1.00	50	0	0	50	0	0	48.77	0	29.92	0	20.08	0	0	0	0	1.00	1	1	-10	0
18.31	1	1.72	0	-1.00	50	0	0	50	0	0	48.54	0	28.48	0	21.52	0	0	0	0	1.00	1	1	-10	0
18.34	1	1.72	0	-1.00	50	0	0	50	0	0	48.28	0	27.04	0	22.96	0	0	0	0	1.00	1	1	-10	0
18.38	1	1.72	0	-1.00	50	0	0	50	0	0	48.01	0	25.60	0	24.40	0	0	0	0	1.00	1	1	-10	0
18.42	1	1.72	0	4.00	50	0	0	50	0	0	47.72	0	24.16	0	25.84	0	0	0	0	1.00	1	1	-10	0
18.50	1	1.72	1	3.85	50	0	0	50	0	0	47.10	0	21.28	0	28.72	0	0	0	0	1.00	1	1	-10	0
18.58	1	1.72	1	3.70	50	0	0	50	0	0	46.41	0	18.40	0	31.60	0	0	0	0	1.00	1	1	-10	0
18.66	1	1.72	1	3.55	50	0	0	50	0	0	45.66	0	15.52	0	34.48	0	0	0	0	1.00	1	1	-10	0
18.75	1	1.72	1	3.45	50	0	0	50	0	0	44.85	0	12.64	0	37.36	0	0	0	0	1.00	1	1	-10	0
18.83	1	1.72	1	3.30	50	0	0	50	0	0	43.97	0	9.76	0	40.24	0	0	0	0	1.00	1	1	-10	0
18.91	1	1.72	1	3.20	50	0	0	50	0	0	43.03	0	6.88	0	43.12	0	0	0	0	1.00	1	1	-10	0
18.98	1	1.72	1	3.10	50	0	0	50	0	0	42.02	0	4.00	0	46.00	0	0	0	0	1.00	1	1	-10	0
19.06	1	1.72	1	3.00	50	0	0	50	0	0	40.95	0	1.12	0	48.88	0	0	0	0	1.00	1	1	-10	0
19.11	1	0.77	2	2.00	50	0	0	50	0	0	40.39	0	-0.32	0	50.32	0	0	0	0	1.00	1	1	-10	0
19.19	1	0.77	2	1.85	50	0	0	50	0	0	39.23	0	-3.20	0	53.20	0	0	0	0	1.00	1	1	-10	0
19.27	1	0.77	2	1.75	50	0	0	50	0	0	38.00	0	-6.08	0	56.08	0	0	0	0	1.00	1	1	-10	0
19.34	1	0.77	2	1.65	50	0	0	50	0	0	36.70	0	-8.96	0	58.96	0	0	0	0	1.00	1	1	-10	0
19.44	1	0.77	2	1.55	50	0	0	50	0	0	35.34	0	-11.84	0	61.84	0	0	0	0	1.00	1	1	-10	0
19.47	0	0.00	3	1.60	50	0	0	50	0	0	34.64	0	-13.28	0	63.28	0	0	0	0	1.00	1	1	-10	0
19.50	0	0.00	3	1.55	50	0	0	50	0	0	33.92	0	-14.72	0	64.72	0	0	0	0	1.00	1	1	-10	0
19.55	0	0.00	3	1.45	50	0	0	50	0	0	33.19	0	-16.16	0	66.16	0	0	0	0	1.00	1	1	-10	0
19.59	0	0.00	3	1.40	50	0	0	50	0	0	32.44	0	-17.60	0	67.60	0	0	0	0	1.00	1	1	-10	0
19.64	0	0.00	3	1.35	50	0	0	50	0	0	31.67	0	-19.04	0	69.04	0	0	0	0	1.00	1	1	-10	0
19.67	0	0.00	3	1.30	50	0	0	50	0	0	30.89	0	-20.48	0	70.48	0	0	0	0	1.00	1	1	-10	0
19.70	0	0.00	3	1.25	50	0	0	50	0	0	30.09	0	-21.92	0	71.92	0	0	0	0	1.00	1	1	-10	0
19.75	0	0.00	3	1.20	50	0	0	50	0	0	29.27	0	-23.36	0	73.36	0	0	0	0	1.00	1	1	-10	0
19.81	0	0.00	3	1.15	50	0	0	50	0	0	28.44	0	-24.80	0	74.80	0	0	0	0	1.00	1	1	-10	0
19.83	0	0.00	3	1.10	50	0	0	50	0	0	27.59	0	-26.24	0	76.24	0	0	0	0	1.00	1	1	-10	0
19.86	0	0.00	3	1.05	50	0	0	50	0	0	26.73	0	-27.68	0	77.68	0	0	0	0	1.00	1	1	-10	0
19.92	0	0.00	3	1.00	50	0	0	50	0	0	25.85	0	-29.12	0	79.12	0	0	0	0	1.00	1	1	-10	0
19.95	0	0.00	3	0.95	50	0	0	50	0	0	24.96	0	-30.56	0	80.56	0	0	0	0	1.00	1	1	-10	0
19.98	0	0.00	3	0.90	50	0	0	50	0	0	24.04	0	-32.00	0	82.00	0	0	0	0	1.00	1	1	-10	0
20.03	0	0.00	3	0.85	50	0	0	50	0	0	23.12	0	-33.44	0	83.44	0	0	0	0	1.00	1	1	-10	0
20.06	0	0.00	3	0.80	50	0	0	50	0	0	22.17	0	-34.88	0	84.88	0	0	0	0	1.00	1	1	-10	0
20.11	0	0.00	3	0.75	50	0	0	50	0	0	21.22	0	-36.32	0	86.32	0	0	0	0	1.00	1	1	-10	0
20.16	0	0.00	3	0.70	50	0	0	50	0	0	20.24	0	-37.76	0	87.76	0	0	0	0	1.00	1	1	-10	0
20.19	0	0.00	3	0.70	50	0	0	50	0	0	19.25	0	-39.20	0	89.20	0	0	0	0	1.00	1	1	-10	0
20.23	0	0.00	3	0.65	50	0	0	50	0	0	18.24	0	-40.64	0	90.64	0	0	0	0	1.00	1	1	-10	0
20.28	0	0.00	3	0.60	50	0	0	50	0	0	17.22	0	-42.08	0	92.08	0	0	0	0	1.00	1	1	-10	0
20.31	0	0.00	3	0.55	50	0	0	50	0	0	16.18	0	-43.52	0	93.52	0	0	0	0	1.00	1	1	-10	0
20.34	0	0.00	3	0.50	50	0	0	50	0	0	15.12	0	-44.96	0	94.96	0	0	0	0	1.00	1	1	-10	0
20.39	0	0.00	3	0.45	50	0	0	50	0	0	14.05	0	-46.40	0	96.40	0	0	0	0	1.00	1	1	-10	0
20.42	0	0.00	3	0.40	50	0	0	50	0	0	12.97	0	-47.84	0	97.84	0	0	0	0	1.00	1	1	-10	0
20.47	0	0.00	3	0.40	50	0	0	50	0	0	11.86	0	-49.28	0	99.28	0	0	0	0	1.00	1	1	-10	0
20.50	0	0.00	3	0.35	50	0	0	50	0	0	10.74	0	-50.72	0	100.72	0	0	0	0	1.00	1	1	-10	0
20.56	0	0.00	3	0.30	50	0	0	50	0	0	9.61	0	-52.16	0	102.16	0	0	0	0	1.00	1	1	-10	0
20.59	0	0.00	3	0.25	50	0	0	50	0	0	8.46	0	-53.60	0	103.60	0	0	0	0	1.00	1	1	-10	0
20.63	0	0.00	3	0.20	50	0	0	50	0	0	7.29	0	-55.04	0	105.04	0	0	0	0	1.00	1	1	-10	0
20.67	0	0.00	3	0.15	50	0	0	50	0	0	6.11	0	-56.48	0	106.48	0	0	0	0	1.00	1	1	-10	0
20.72	0	0.00	3	0.15	50	0	0	50	0	0	4.91	0	-57.92	0	107.92	0	0	0	0	1.00	1	1	-10	0
20.77	0	0.00	3	0.10	50	0	0	50	0	0	3.69	0	-59.36	0	109.36	0	0	0	0	1.00	1	1	-10	0
20.80	0	0.00	3	0.05	50	0	0	50	0	0	2.46	0	-60.80	0	110.80	0	0	0	0	1.00	1	1	-10	0
20.83	0	0.00	3	-1.00	50	0	0	50	0	0	0.00	0	0.00	0	0.00	0	0	0	0	1.00	1	1	-10	0

Table 2.2: The initial reference vehicle velocity is $50[km/h]$ and the obstacle is driving with initial speed $40[km/h]$ and acceleration $-10[m/s^2]$. The initial x-distance between vehicle and obstacle is $50[m]$ and the initial y-distance is $0[m]$. The acoustical warning 1 starts at $47.10[m]$ till $40.95[m]$ x-distance between the obstacle and the vehicle, the braking warning 2 starts at $40.39[m]$ till $35.34[m]$ x-distance and the automatic braking maneuver is performed from a x-distance of $34.64[m]$ till the collision is avoided or mitigated ($0[m]$ x-distance). In this case the obstacle motion is simulated with a linear motion.

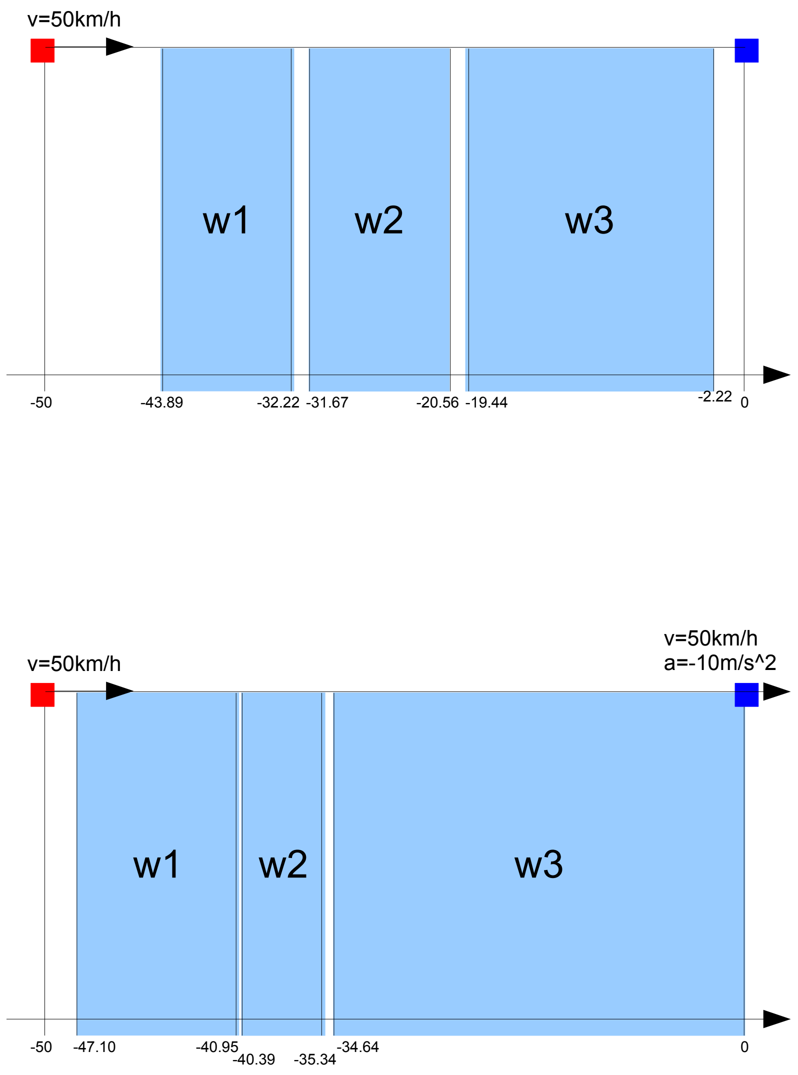


Figure 2.1: First row shows the three CAB warnings for a fixed obstacle. In second row a moving obstacle is considered and the three warnings are shown.

2.1 Last point to brake and last point to steer

To be consistent with the CAB output, VTM tool provides output for comparison for each warning level. The output provided in this section is the last point to brake and the last point to steer for the three CAB warning levels. The criteria to define the warning levels in the VTM tool is based on different ways of imposing box constraints for the control. In particular Table 2.3 summarizes the warning criteria for CAB and VTM.

warning level	CAB output	VTM last point to steer	VTM last point to brake
w1	$[\underline{w}1, \overline{w}1]$	solve Problem 2.1 with $w_\delta \in [-0.1, 0.1] rad/s =: I_1$ $F_B \in [-0.5, 1.5]10^4 N =: J_3$	solve Problem 2.2 with $w_\delta \in [-0.5, 0.5] rad/s =: I_3$ $F_B \in [-0.5, 0.5]10^4 N =: J_1$
w2	$[\underline{w}2, \overline{w}2]$	solve Problem 2.1 with $w_\delta \in [-0.2, 0.2] rad/s =: I_2$ $F_B \in [-0.5, 1.5]10^4 N =: J_3$	solve Problem 2.2 with $w_\delta \in [-0.5, 0.5] rad/s =: I_3$ $F_B \in [-0.5, 0.7]10^4 N =: J_2$
w3	$[\underline{w}3, \overline{w}3]$	solve Problem 2.1 with $w_\delta \in [-0.5, 0.5] rad/s =: I_3$ $F_B \in [-0.5, 1.5]10^4 N =: J_3$	solve Problem 2.2 with $w_\delta \in [-0.5, 0.5] rad/s =: I_3$ $F_B \in [-0.5, 1.5]10^4 N =: J_3$

Table 2.3: The last point to steer associated to warning level i, with $i = 1, 2, 3$ is denoted by $d^s_{w_i}$ and it is defined as $d^s_{w_i} := z_1(t_0) - x_1(t_0)$ where $z_1(t_0)$ and $x_1(t_0)$ are given by solving Problem 2.1 with steering velocity in I_i and braking force in J_3. The last point to brake associated to warning level i, with $i = 1, 2, 3$ is denoted by $d^b_{w_i}$ and it is defined as $d^b_{w_i} := z_1(t_0) - x_1(t_0)$ where $z_1(t_0)$ and $x_1(t_0)$ are given by solving Problem 2.2 with steering velocity in I_3 and braking force in J_i.

The optimal control problem computing the last point to steer and the last point to brake is of the form of Problem 3.2 of Chapter II with constraints in the form (2.86) of Chapter II and with objective function given by (3.3) of Chapter II.

Problem 2.1 (Last point to steer)**.** *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed $t_0 < t_f$. Find states $z : \mathbb{R} \to \mathbb{R}^{n_z}$ absolutely continuous, controls $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u}$ meaurable$\}$ and parameters $w \in W \subset \mathbb{R}^{n_w}$,*

$\tau \in [t_0, t_f]$ such that:

$$\begin{array}{lll}
\min\limits_{z,u\in\mathcal{U},w\in W,\tau\in[t_0,t_f]} & x_1(t_0) & \\
s.t. & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, & \\
& z(t_0) = z_0, & \\
& \textit{target constraint: } z_1(\tau) \geq x_1(\tau),\ z_3(\tau) = 0.0 & \\
& \textit{obstacle avoidance state constraints:} & (2.1)\\
& \quad (z_1(t) - x_i(t))^2 + (z_2(t) - y_i(t))^2 \geq (\ell_i + \ell + \eta)^2 & \\
& \quad \forall i = 1, \ldots, n_{obst} & \\
& \textit{road state constraints: } (\underline{r} + \ell + \eta) \leq z_2(t) \leq (\overline{r} - \ell - \eta) & \\
& \textit{Kamm's circle state constraint: } (1.29) \textit{ of Chapter II} & \\
& \textit{box constraints: } u(t) \in [\underline{u}, \overline{u}],\ \forall t \in [0, \tau] &
\end{array}$$

where $f : [t_0, t_f] \times \mathbb{R}^7 \times \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}^7$ is the single track dynamics given in (1.21) of Chapter II, with $z = (x, y, \psi, v_x, v_y, w_\psi, \delta), u = (w_\delta, F_B)$ and $w = x_1(t_0)$ optimized parameter. The control boundaries $\underline{u}, \overline{u}$ will depend on the chosen warning. The n_{obst} obstacles are modeled as balls of radius ℓ_i and center $(x_i(t), y_i(t))$ moving of linear motion with velocity $v_i(t)$ and constant acceleration a_i. The vehicle is a ball of radius ℓ and center $(z_1, z_2) = (x, y)$. The safety margin between obstacles and vehicle is $\eta = 0.3m$ and the road y boundaries are denoted with $\underline{r} = 0, \overline{r} = 7$.

Problem 2.2 (Last point to brake)**.** *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed $t_0 < t_f$. Find states $z : \mathbb{R} \to \mathbb{R}^{n_z}$ absolutely continuous, controls $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u}$ meaurable$\}$ and parameters $w \in W \subset \mathbb{R}^{n_w}$, $\tau \in [t_0, t_f]$ such that:*

$$\begin{array}{lll}
\min\limits_{z,u\in\mathcal{U},w\in W,\tau\in[t_0,t_f]} & x_1(t_0) & \\
s.t. & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, & \\
& z(t_0) = z_0, & \\
& \textit{target constraint:} & (2.2)\\
& \quad z_1(\tau) = x_1(\tau) - \ell - \ell_1 - \eta,\ z_4(\tau) \leq v_1(\tau) & \\
& \textit{road state constraints: } (\underline{r} + \ell + \eta) \leq z_2(t) \leq (\overline{r} - \ell - \eta) & \\
& \textit{Kamm's circle state constraint: } (1.29) \textit{ of Chapter II} & \\
& \textit{box constraints: } u(t) \in [\underline{u}, \overline{u}],\ \forall t \in [0, \tau] &
\end{array}$$

where $f : [t_0, t_f] \times \mathbb{R}^7 \times \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}^7$ is the single track dynamics given in (1.21) of Chapter II, with $z = (x, y, \psi, v_x, v_y, w_\psi, \delta), u = (w_\delta, F_B)$ and $w = x_1(t_0)$ optimized parameter. The control boundaries $\underline{u}, \overline{u}$ will depend on the chosen warning. The reference vehicle has to stop behind the obstacle with velocity less or equal than the obstacle velocity at final time $v_i(\tau)$. The obstacle length is ℓ_1, while the vehicle length is ℓ and the safety margin between obstacles and vehicle is $\eta = 0.3m$. The road y boundaries are denoted with $\underline{r} = 0, \overline{r} = 7$.

By solving Problems 2.1 and 2.2 for each control interval $[\underline{u}, \overline{u}]$ associated to each warning level w1, w2, w3 (see Table 2.3), the trajectories in Figures 2.2-2.5 are found and the last point to steer/brake is given by $x_1(t_0) - z_1(t_0)$. The simulation is run for $n_{obst} = 1$.

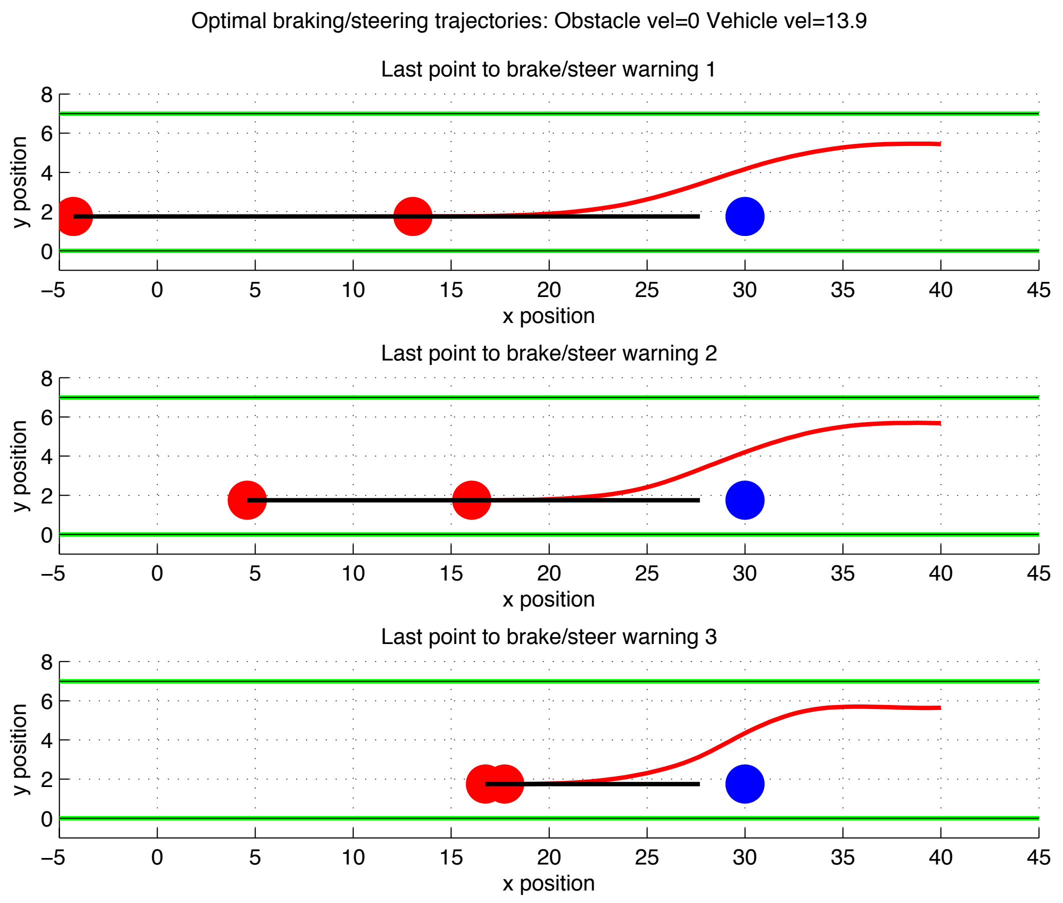


Figure 2.2: The last point to steer and the last point to brake of Table 2.4 for each warning level. Optimal braking/steering trajectories: Obstacle vel= $0[m/s]$ Vehicle vel= $13.9[m/s]$.

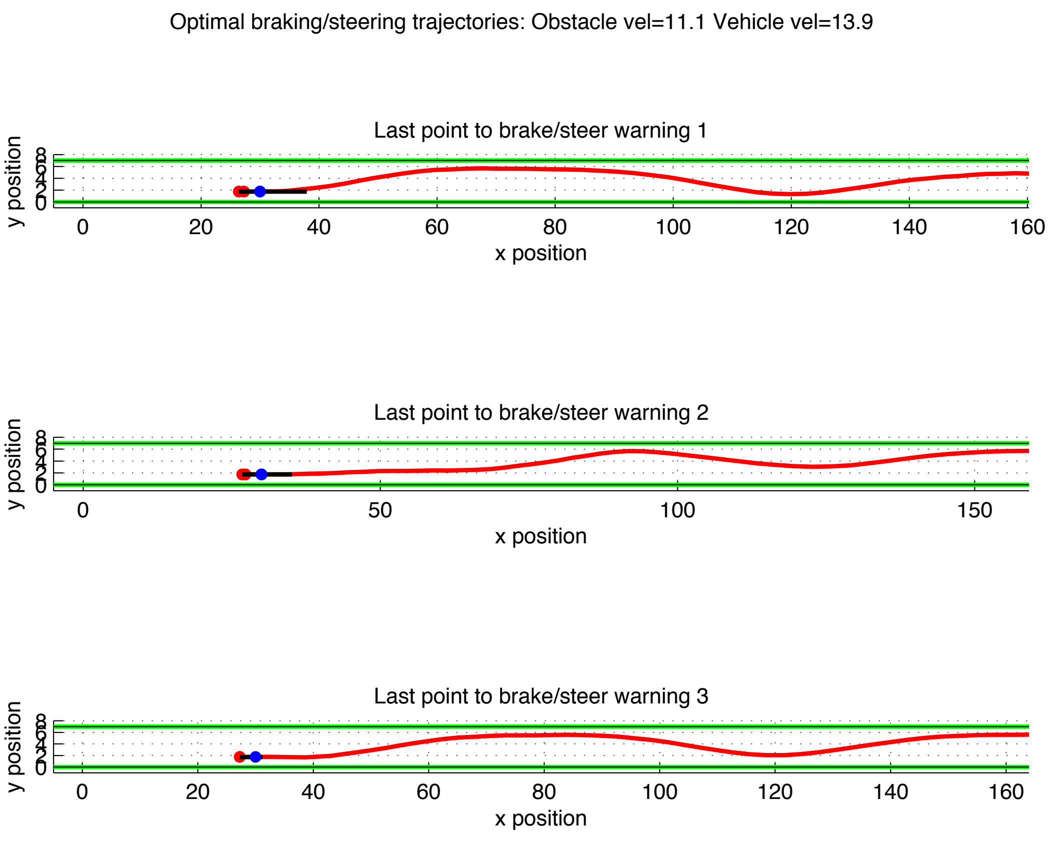


Figure 2.3: The last point to steer and the last point to brake of Table 2.4 for each warning level. Optimal braking/steering trajectories: Obstacle vel= 11.1$[m/s]$ Vehicle vel= 13.9$[m/s]$.

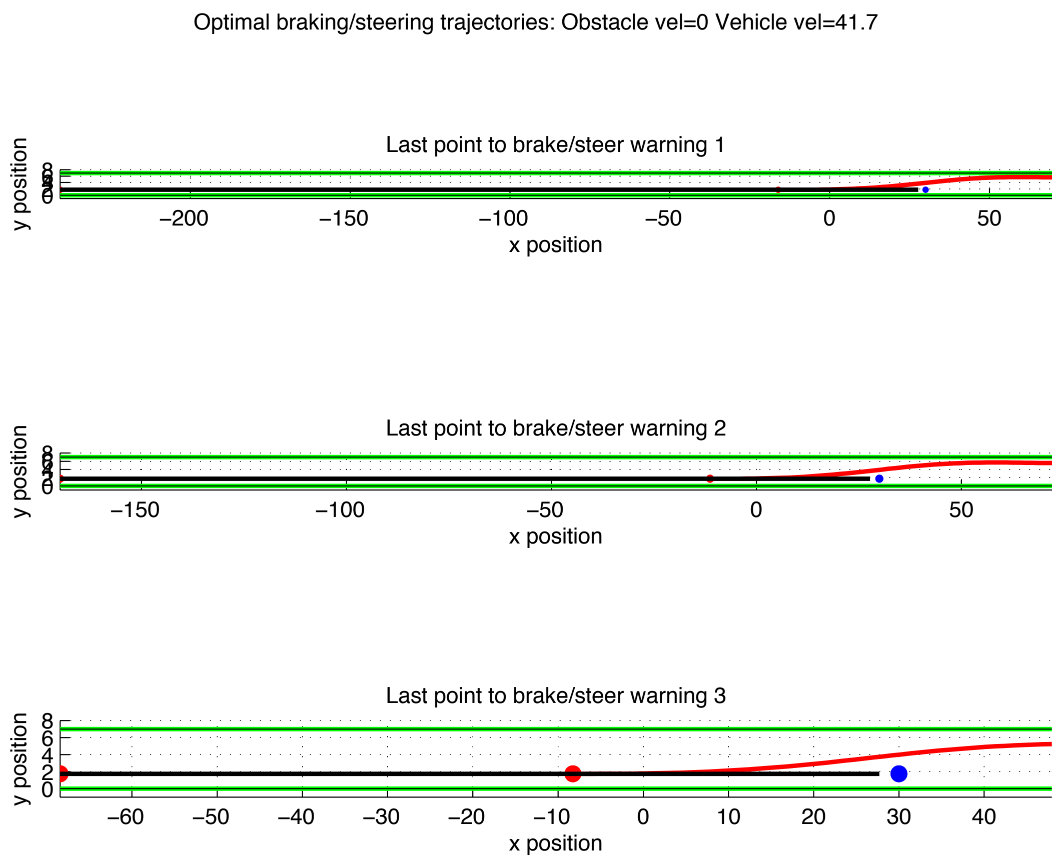


Figure 2.4: The last point to steer and the last point to brake of Table 2.4 for each warning level. Optimal braking/steering trajectories: Obstacle vel= $0[m/s]$ Vehicle vel= $41.7[m/s]$.

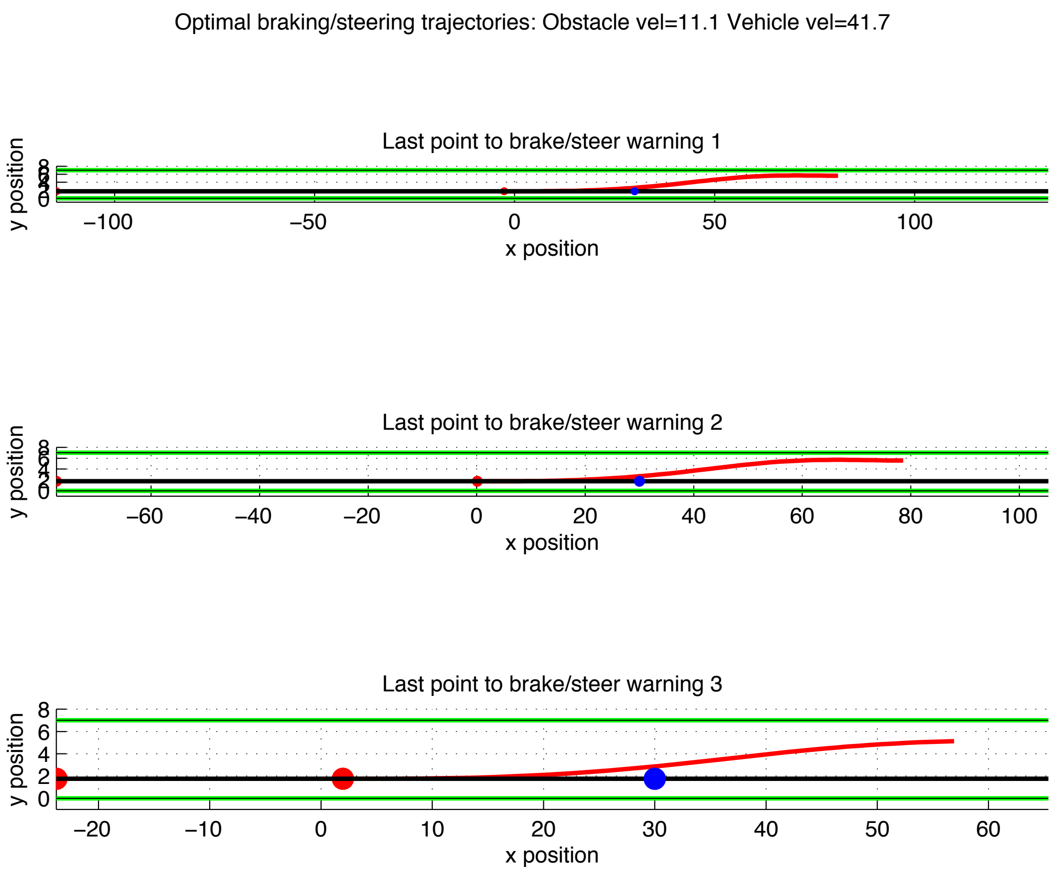


Figure 2.5: The last point to steer and the last point to brake of Table 2.4 for each warning level. Optimal braking/steering trajectories: Obstacle vel= $11.1[m/s]$ Vehicle vel= $41.7[m/s]$.

Following the structure of Table 2.3, Table 2.4 compares the last point to steer and the last point to brake for each warning level, with the distance intervals given by CAB-algorithm for each warning level. Four case studies are computed for combinations of two different vehicle initial velocities and two different obstacle initial velocities.

vel car [km/h]	vel obst [km/h]	warning level	CAB output	VTM last point to steer d^s	VTM last point to brake d^b
50	0	w1	[43.8889, 32.222]	16.950	34.280
		w2	[31.6667, 20.5556]	13.950	25.406
		w3	[19.4444, 2.22222]	12.270	13.251
50	40	w1	[7.55556, 5.11111]	2.730	3.588
		w2	[5, 2.77778]	2.720	3.233
		w3	[2.55556, 2.11111]	2.720	2.743
150	0	w1	[130, 100]	46.105	270.744
		w2	[98.3333, 75]	41.298	199.872
		w3	[73.3333, 2.72853]	38.258	98.441
150	40	w1	[108.333, 83.8889]	32.605	144.549
		w2	[82.6667, 54.5556]	29.880	107.448
		w3	[53.3333, 3.22222]	28.033	53.778

Table 2.4: Four cases where the distances given CAB-algorithm for each warning level are compared with the last point to steer and the last point to brake for each warning level. For high vehicle velocities the warning level distance is always less than the last point to brake. For low vehicle velocities the CAB-algorithm applies a prudential strategy in the sense that the warning is given always for bigger distances than the last point to brake.

In Figures 2.6, 2.7, 2.8 for each warning level, the last point to steer (red line) and the last point to brake (black line) are plotted for several vehicle initial velocity. One can notice that for low velocities the last point to brake occurs later than the last point to steer and thus the braking maneuver is more convenient than the steering maneuver. In Figure 2.6 a fixed obstacle is considered while in Figure 2.7 the obstacle constant velocity is $15m/s$ and in Figure 2.8 is $30[m/s]$.

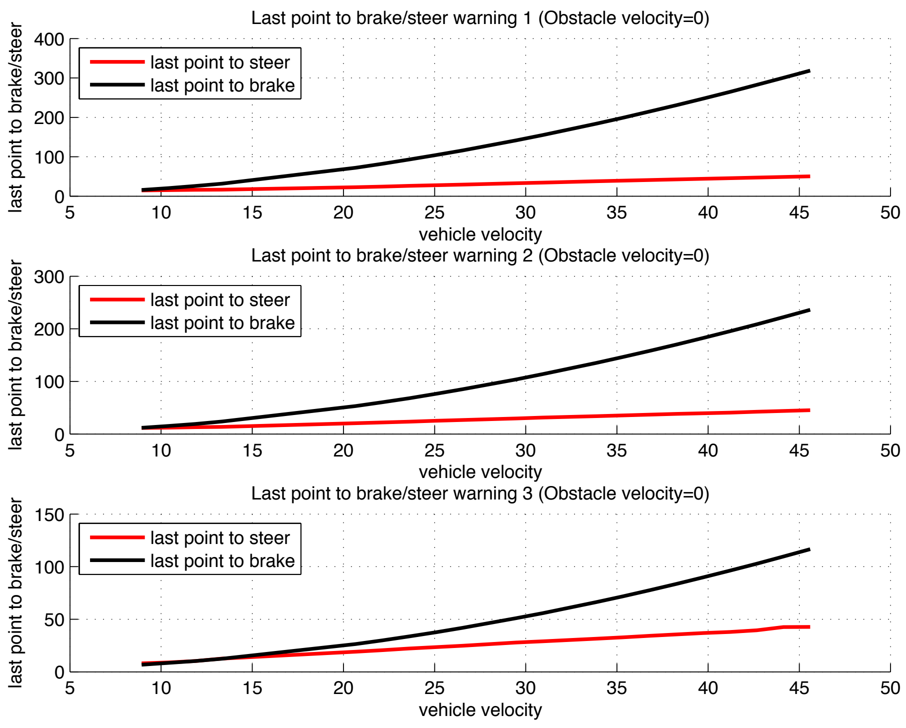


Figure 2.6: The last point to steer and the last point to brake are function of the initial velocity of the reference vehicle. In first row, plots are computed for warning level 1 i.e. for values of the steering velocity in the interval $[-0.1, 0.1]$ and braking force in the interval $[-0.5, 0.5]$ 10^4N. Similarly second and third row are computed for warning level 2 and 3 respectively, with steering velocity in the intervals $[-0.2, 0.2]$, $[-0.5, 0.5]$ and braking force in the interval $[-0.5, 0.7]$ 10^4N, $[-0.5, 1.5]$ 10^4N, respectively. The obstacle is fixed in this particular case.

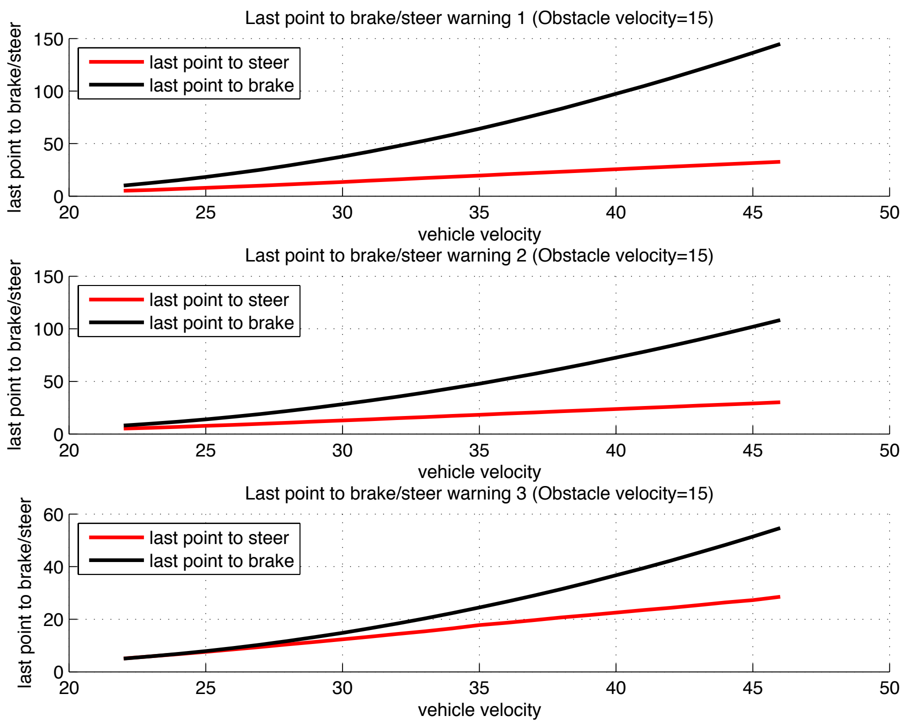


Figure 2.7: The last point to steer and the last point to brake are function of the initial velocity of the reference vehicle. In first row, plots are computed for warning level 1 i.e. for values of the steering velocity in the interval $[-0.1, 0.1]$ and braking force in the interval $[-0.5, 0.5]$ $10^4 N$. Similarly second and third row are computed for warning level 2 and 3 respectively, with steering velocity in the intervals $[-0.2, 0.2]$, $[-0.5, 0.5]$ and braking force in the interval $[-0.5, 0.7]$ $10^4 N$, $[-0.5, 1.5]$ $10^4 N$, respectively. The obstacle has constant velocity $15 m/s$ in this particular case.

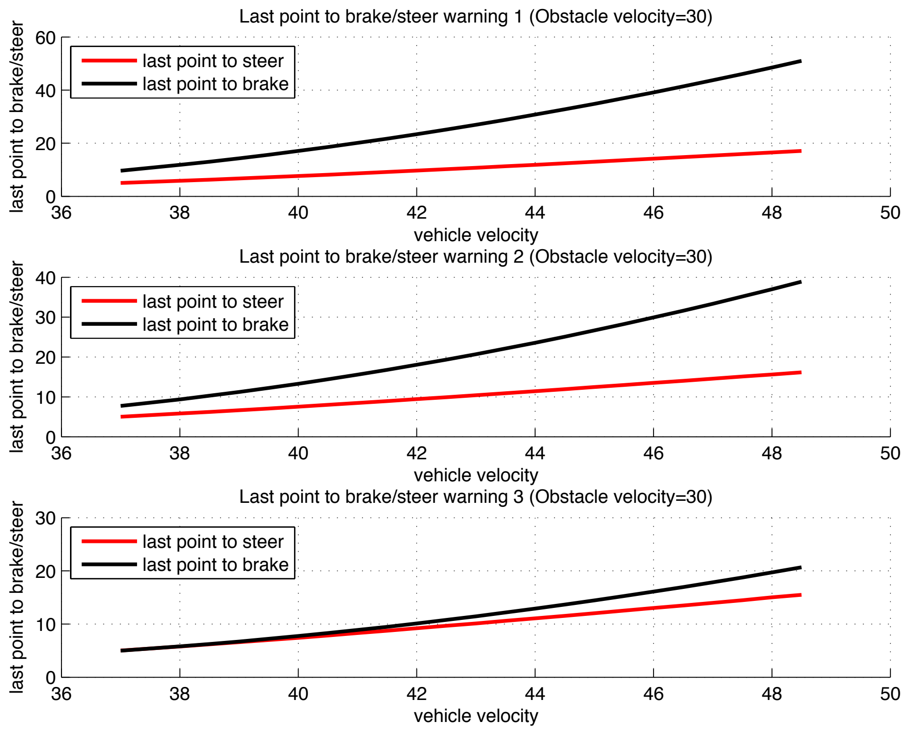


Figure 2.8: The last point to steer and the last point to brake are function of the initial velocity of the reference vehicle. In first row, plots are computed for warning level 1 i.e. for values of the steering velocity in the interval $[-0.1, 0.1]$ and braking force in the interval $[-0.5, 0.5]$ 10^4N. Similarly second and third row are computed for warning level 2 and 3 respectively, with steering velocity in the intervals $[-0.2, 0.2]$, $[-0.5, 0.5]$ and braking force in the interval $[-0.5, 0.7]$ 10^4N, $[-0.5, 1.5]$ 10^4N, respectively. The obstacle has constant velocity $30m/s$ in this particular case.

The last point to brake and the last point to steer are plot as functions of the vehicle initial velocity and the relative initial velocity between vehicle and obstacle, for warning levels 1 and 2 in Figures 2.9, for warning level 3 in Figure 2.10. The warning level is given by imposing different box constraints on the controls, as done previously in this section. The more the warning increase the more the control intervals increase, allowing bigger capabilities to the vehicle. As a consequence of this, the last time to brake and the last time to steer decrease while increasing the warning level.

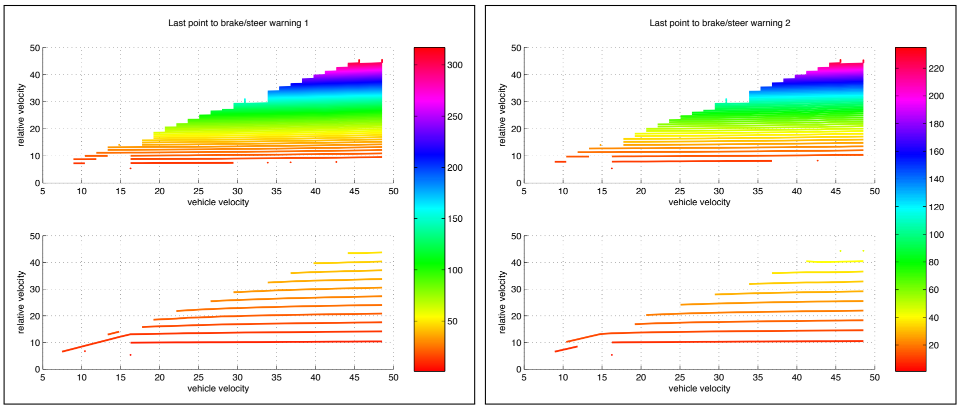


Figure 2.9: The last time to brake defined as initial distance between the reference vehicle and obstacle is given in first row, while the last point to steer is given in second row. The left picture refers to warning level 1 (for the last point to steer $w_\delta \in [-0.1, 0.1]$ and for the last point to brake $F_B \in [-0.5, 0.5]$), while the right picture refers to warning level 2 (for the last point to steer $w_\delta \in [-0.2, 0.2]$ and for the last point to brake $F_B \in [-0.5, 0.7]$). By fixing the relative velocity, the last time to brake and the last time to steer for high vehicle velocity is constant, however for small velocities the last point to steer decreases by increasing the vehicle and obstacle velocity. For high relative velocities the last point to brake is higher than the last point to steer, thus steering maneuvers are preferred over braking maneuvers.

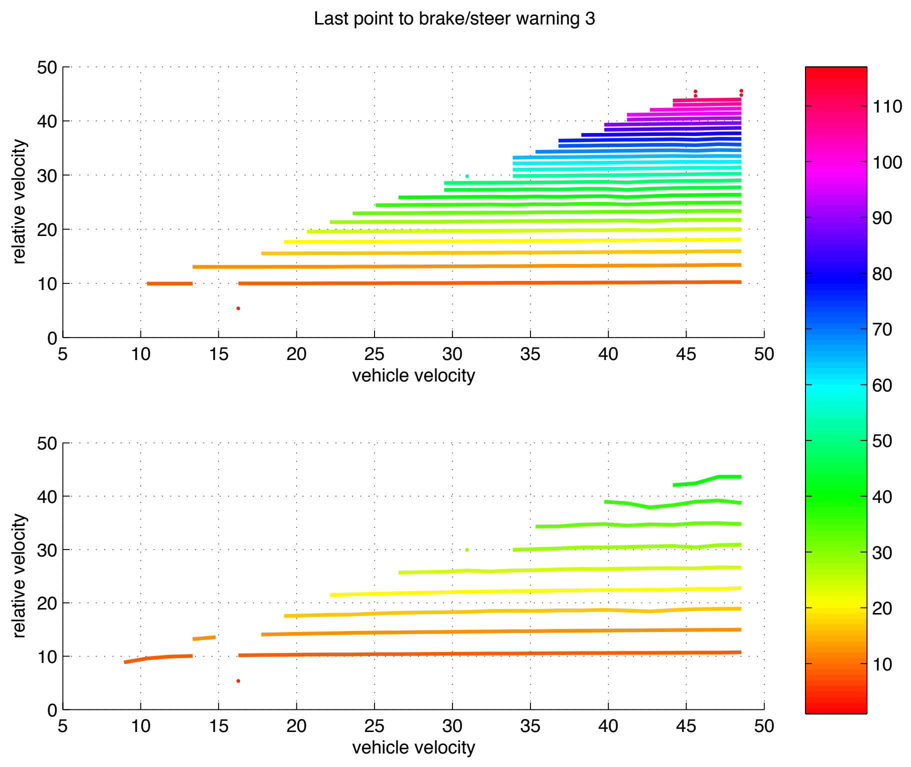


Figure 2.10: For warning level 3 (for the last point to steer $w_\delta \in [-0.5, 0.5]$ and for the last point to brake $F_B \in [-0.5, 1.5]$), the last time to brake defined as initial distance between the reference vehicle and obstacle is given in first row, while the last point to steer is given in second row. By fixing the relative velocity, the last time to brake and the last time to steer for high vehicle velocity is constant. For high relative velocities the last point to brake is higher than the last point to steer, thus steering maneuvers are preferred over braking maneuvers. However for low relative velocities the last point to steer is higher than the last point to brake.

2.2 Reachable set for each warning level

The following optimal control problem is solved to find the reachable set for a given car traffic scenario where CAB algorithm can be tested.

Problem 2.3. *Let* $[t_0, t_f] \subset \mathbb{R}$ *be a non-empty and bounded interval with fixed* $t_0 < t_f$*. Find states* $z : \mathbb{R} \to \mathbb{R}^{n_z}$ *absolutely continuous, controls* $u \in \mathcal{U} = \{u \,|\, u :$

$[t_0, t_f) \to U \in \mathbb{R}^{n_u}$ meaurable$\}$ and parameter $\tau \in [t_0, t_f]$ such that:

$$
\begin{array}{lll}
\min\limits_{z,u\in\mathcal{U},\tau\in[t_0,t_f]} & \left\| \begin{pmatrix} z_1(\tau) \\ z_2(\tau) \end{pmatrix} - \begin{pmatrix} g_h^x \\ g_h^y \end{pmatrix} \right\|_2 & \\
s.t. & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, & \\
& z(t_0) = z_0, & \\
& \textit{target constraint: } z_3(\tau) = 0.0 & (2.3) \\
& \textit{obstacle avoidance state constraints:} & \\
& \quad (z_1(t) - x_i(t))^2 + (z_2(t) - y_i(t))^2 \geq (\ell_i + \ell + \eta)^2 & \\
& \quad \forall i = 1, \dots, n_{obst} & \\
& \textit{road state constraints: } (\underline{r} + \ell + \eta) \leq z_2(t) \leq (\overline{r} - \ell - \eta) & \\
& \textit{Kamm's circle state constraint: } (1.29) \textit{ of Chapter II} &
\end{array}
$$

where $f : [t_0, t_f] \times \mathbb{R}^7 \times \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}^7$ is the single track dynamics given in (1.21) of Chapter II, with $z = (x, y, \psi, v, w_\psi, \delta), u = (w_\delta, F_B)$. The n_{obst} obstacles are modeled as balls of radius ℓ_i and center $(x_i(t), y_i(t))$ moving with linear motion with velocity $v_i(t)$ and constant acceleration a_i. The vehicle is a ball of radius ℓ and center $(z_1, z_2) = (x, y)$. The safety margin between obstacles and vehicle is $\eta = 0.3[m]$ and the road y boundaries are denoted with $\underline{r} = 0, \overline{r} = 7$. The target area is defined by the condition on the vehicle yaw angle which guarantee that the vehicle at the final time τ is driving parallel to the road.

The reachable set is computed recursively solving the optimal control problem 2.3 for each point g_h in the grid

$$\{g_h\}_{h=1,\dots,H} =: \mathbb{G}_H \subset [\underline{X}, \overline{X}] \times [\underline{Y}, \overline{Y}] \subset \mathbb{R}^2 \tag{2.4}$$

defined as:

$$g_h = \begin{pmatrix} g_{h_x}^x \\ g_{h_y}^y \end{pmatrix} = \begin{pmatrix} \underline{X} + \frac{\overline{X} - \underline{X}}{H_x}(h_x - 1) \\ \underline{Y} + \frac{\overline{Y} - \underline{Y}}{H_y}(h_y - 1) \end{pmatrix}, \tag{2.5}$$

for $h_x = 1, \dots, H_x, h_y = 1, \dots, H_y$ and $H = H_x H_y, h = h_x h_y$.

For the numerical simulations the first case of Table 2.4 is considered, where the initial vehicle velocity is $50km/h$ and the single obstacle is fixed. For each warning level distance intervals computed by CAB algorithm (see Table 2.4 column 4 of the first case study), the reachable set is provided in Figure 2.11. This means to solve Problem 2.3 for an x initial distance between vehicle and obstacle within the warning intervals, i.e. $d_{w_i} = (x_1 - z_1(t_0)) \in [\underline{w}_i, \overline{w}_i]$ for $i = 1, 2, 3$. The ratio of number of reachable points over the number of grid points is given for each warning level:

```
CASE v0_vehicle=50km/h v0_obstacle=0km/h:
The ratio of reachable points over the target grid is 0.6
for warning level 1
The ratio of reachable points over the target grid is 0.6
for warning level 2
```

```
The ratio of reachable points over the target grid is 0.4
for warning level 3
```

The ratio can also be used to construct new warning level definitions since it is decreasing with the increasing of the warning level.

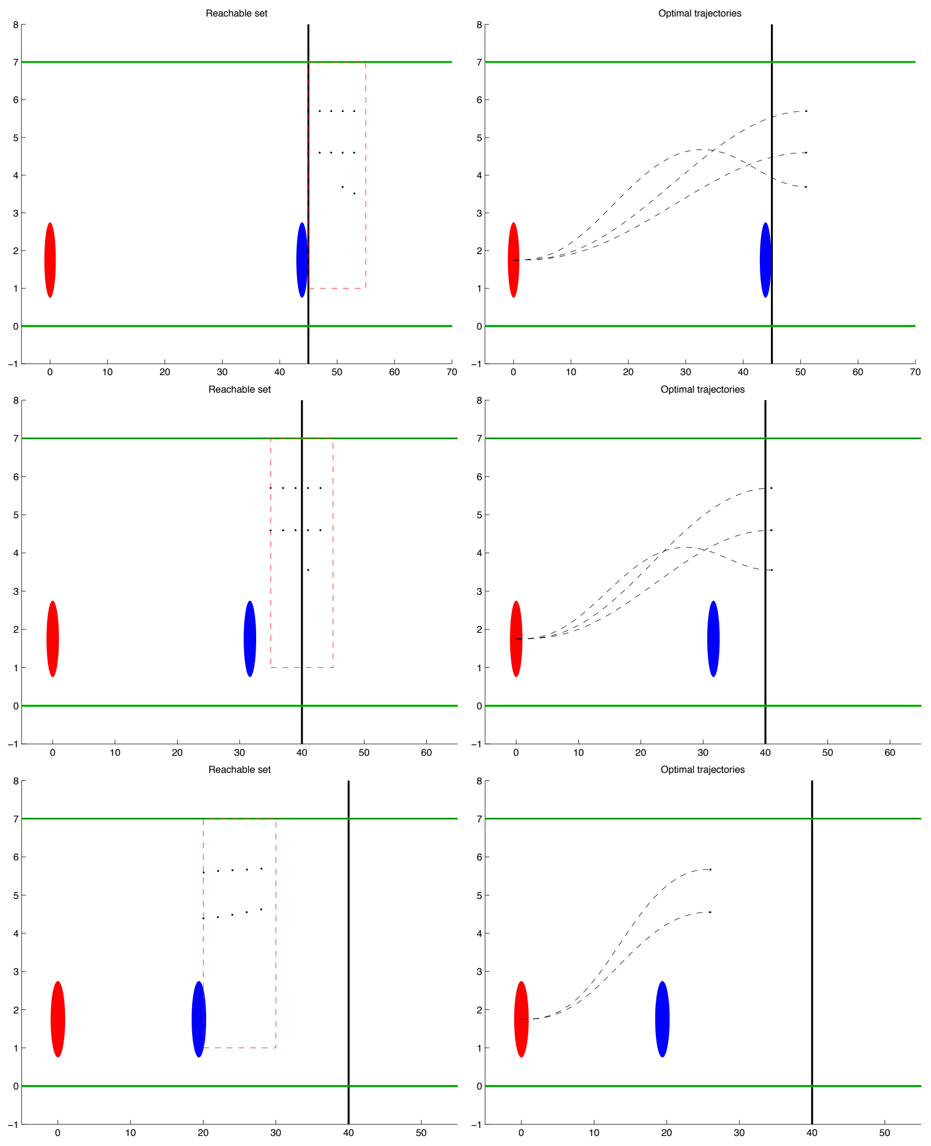


Figure 2.11: On the left the reachable set is computed, on the right a funnel of trajectories is plot for a fiber of the reachable set. Each row correspond to a different warning level.

The robust reachable set associated to the reachable set in Figure 2.11 is non-empty only for very small intervals of the perturbations, see Figure 2.12.

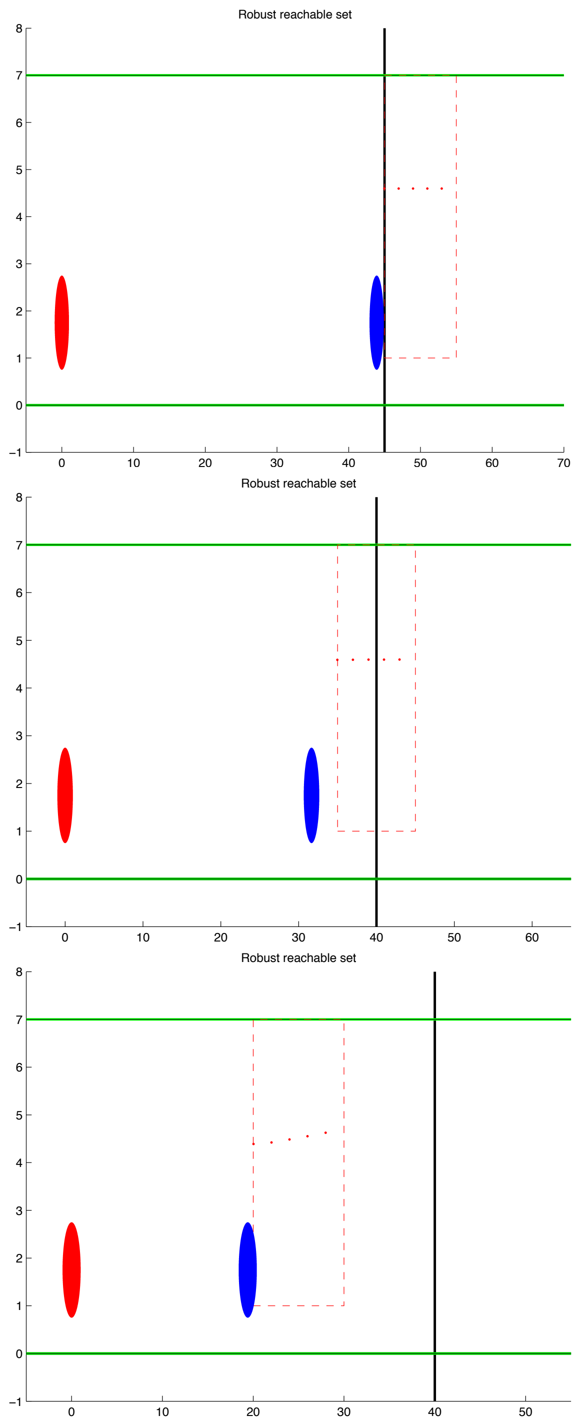


Figure 2.12: The robust reachable set is computed for perturbations from $p = (1, 0.1, 0.01, 1, 0.1)$ to $-p$. Each picture correspond to a different warning level starting from warning level 1 in the right picture.

2.3 Optimal trajectory for each warning level

The following optimal control problem is solved to find minimum time trajectories for a given car traffic scenario where CAB-algorithm can be tested.

Problem 2.4. *Let $[t_0, t_f] \subset \mathbb{R}$ be a non-empty and bounded interval with fixed $t_0 < t_f$. Find states $z : \mathbb{R} \to \mathbb{R}^{n_z}$ absolutely continuous, controls $u \in \mathcal{U} = \{u \,|\, u : [t_0, t_f) \to U \in \mathbb{R}^{n_u}$ measurable$\}$ and parameter $\tau \in [t_0, t_f]$ such that:*

$$\begin{array}{lll}
\displaystyle\min_{z,u\in\mathcal{U},\tau\in[t_0,t_f]} & \tau & \\
s.t. & \dot{z}(t) = f(t, z(t), u(t)) \qquad a.e.\ t \in [t_0, \tau] \subset \mathbb{R}, & \\
& z(t_0) = z_0, & \\
& \textit{target constraint: } z_1(\tau) > x_{target},\ z_3(\tau) = 0.0 & \\
& \textit{obstacle avoidance state constraints:} & \\
& \qquad (z_1(t) - x_i(t))^2 + (z_2(t) - y_i(t))^2 \geq (\ell_i + \ell + \eta)^2 & \\
& \qquad \forall i = 1, \ldots, n_{obst} & \\
& \textit{road state constraints: } (\underline{r} + \ell + \eta) \leq z_2(t) \leq (\overline{r} - \ell - \eta) & \\
& \textit{Kamm's circle state constraint: } (1.29) \textit{ of Chapter II} &
\end{array} \tag{2.6}$$

where $f : [t_0, t_f] \times \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}^7$ is the single track dynamics given in (1.21) of Chapter II, with $z = (x, y, \psi, v_x, v_y, w_\psi, \delta), u = (w_\delta, F_B)$. The n_{obst} obstacles are modeled as balls of radius ℓ_i and center $(x_i(t), y_i(t))$ moving with linear motion with velocity $v_i(t)$ and constant acceleration a_i. The vehicle is a ball of radius ℓ and center $(z_1, z_2) = (x, y)$. The safety margin between obstacles and vehicle is $\eta = 0.3m$ and the road y boundaries are denoted with $\underline{r} = 0, \overline{r} = 7$. The target area is defined by the condition on the vehicle yaw angle which guarantee that the vehicle at the final time τ is driving parallel to the road, moreover the vehicle can stop in the half-space defined by $x > x_{target}$.

For the numerical simulations the first case of Table 2.4 is considered, where the initial vehicle velocity is $50[km/h]$ and the unique obstacle is fixed. The interval computed by CAB-algorithm (see Table 2.4 column 4 of the first case study), the minimum time trajectory is provided in Figures 2.13 for warning level 1, 2.14 for warning level 2, 2.15 for warning level 3. Thus the Problem 2.4 is solved three times, each time imposing $d_{w_i} = z_1(t_o) - x_1 \in [\underline{w}_i, \overline{w}_i]$ for $i = 1, 2, 3$. The ODE-Sensitivity information is in Figure 2.16 and Fiacco-Sensitivity information is in Figure 2.17.

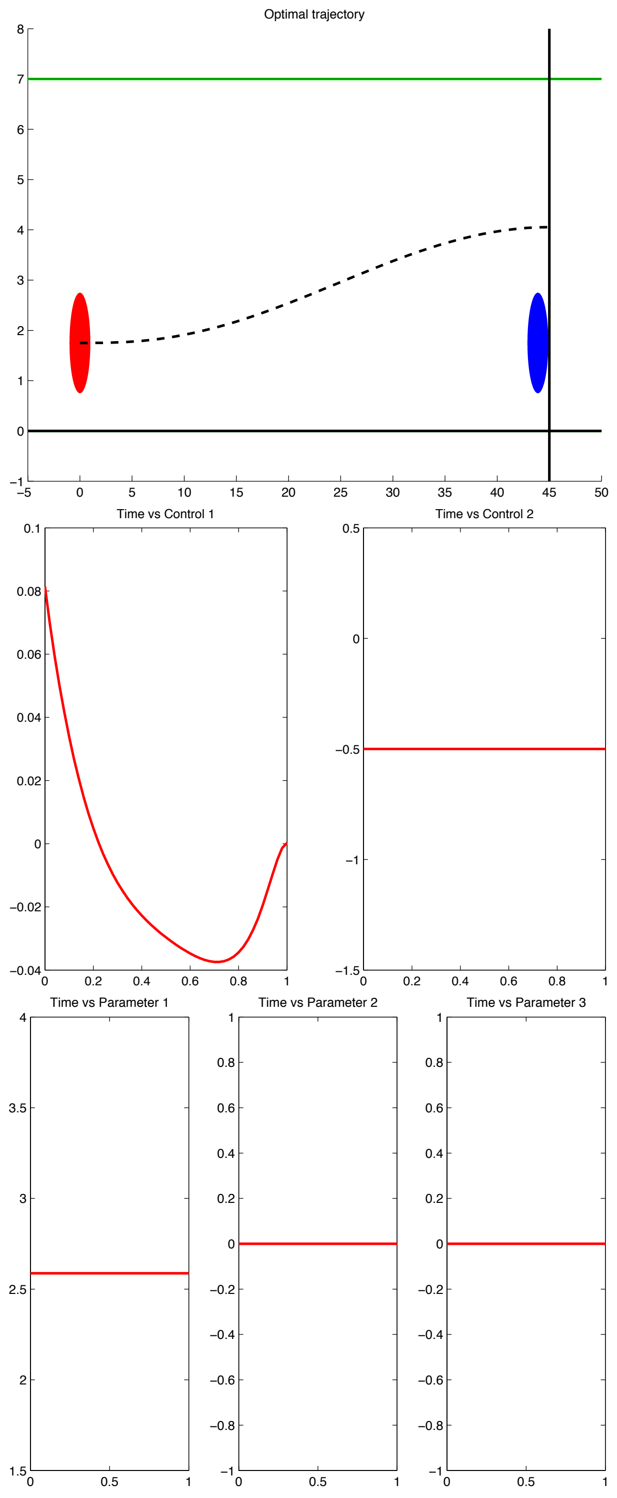


Figure 2.13: Warning level 1. From top to bottom, optimal trajectory, controls and parameters are given.

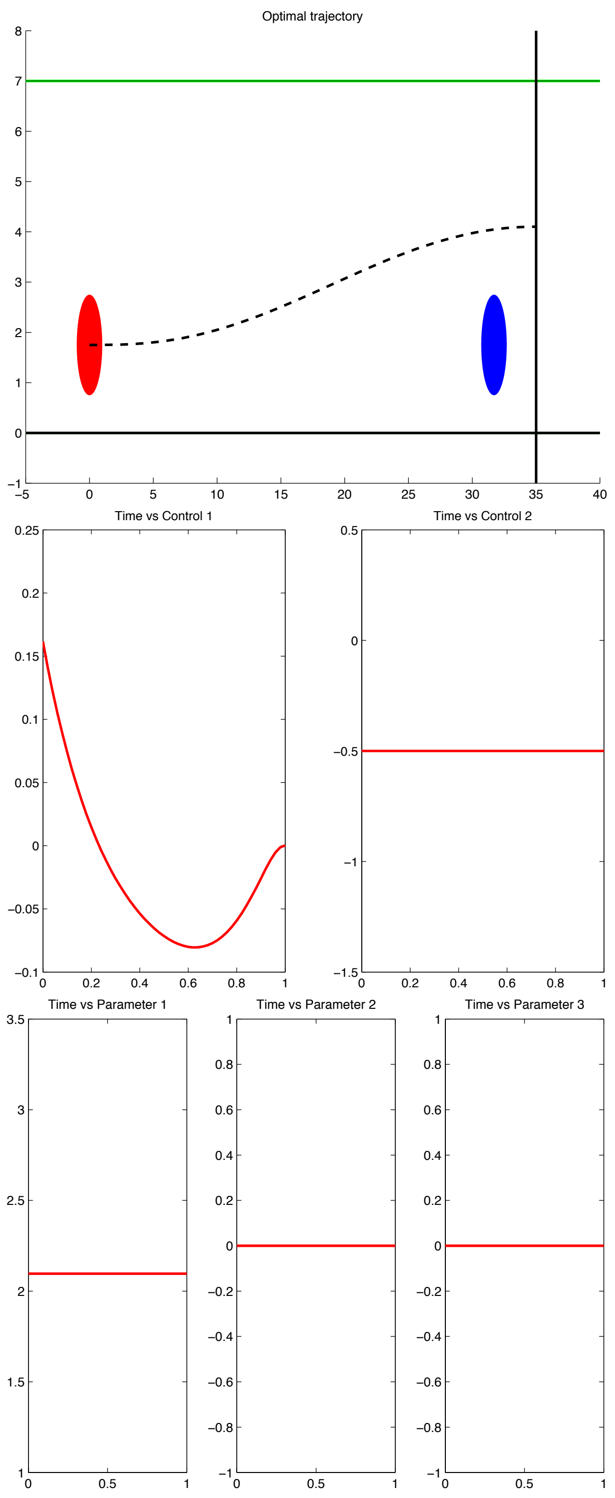


Figure 2.14: Warning level 2. From top to bottom, optimal trajectory, controls and parameters are given.

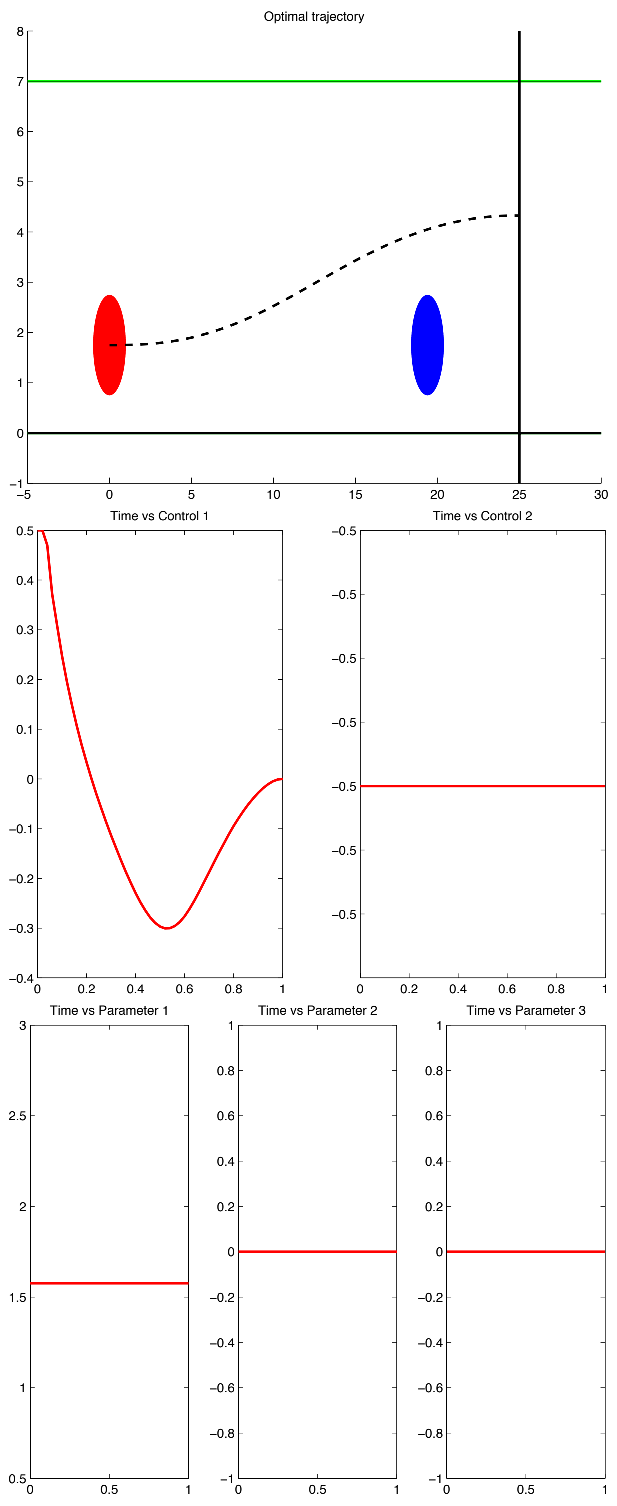


Figure 2.15: Warning level 3. From top to bottom, optimal trajectory, controls and parameters are given.

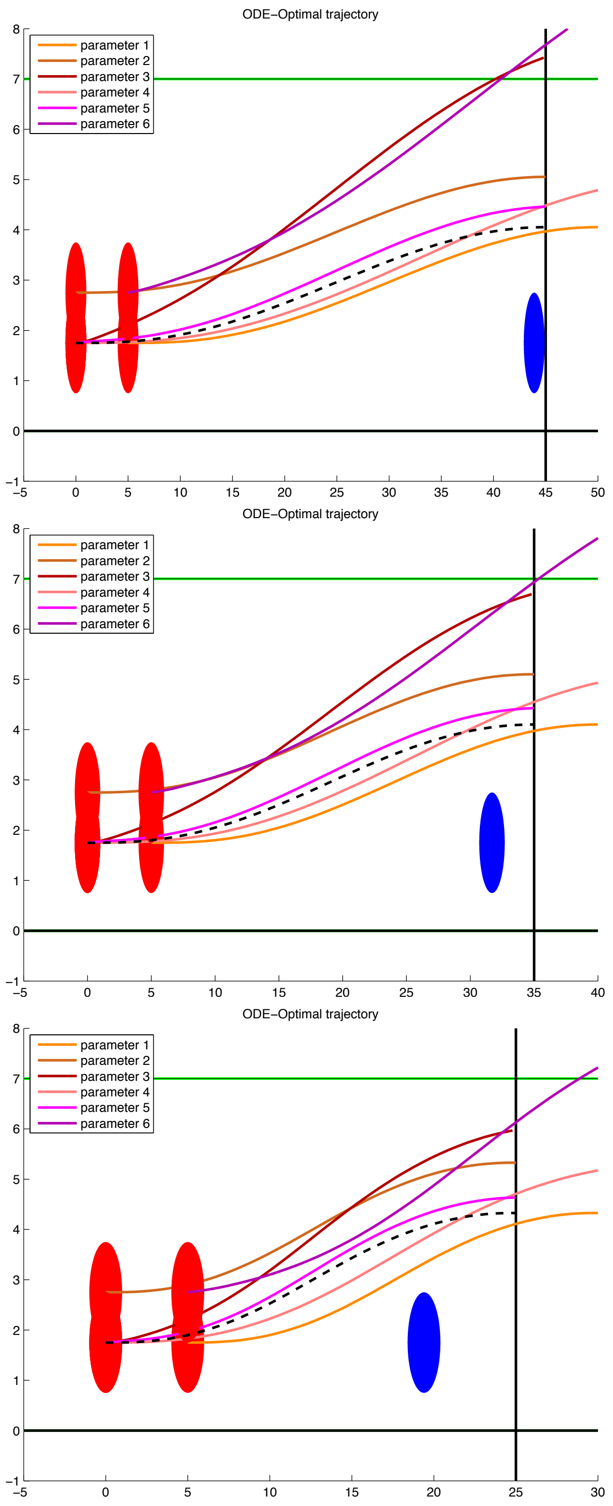


Figure 2.16: The ODE-perturbed trajectory is plot for a perturbation of $p = (5, 1, 0.1, 5, 0.5)$ for each warning level.

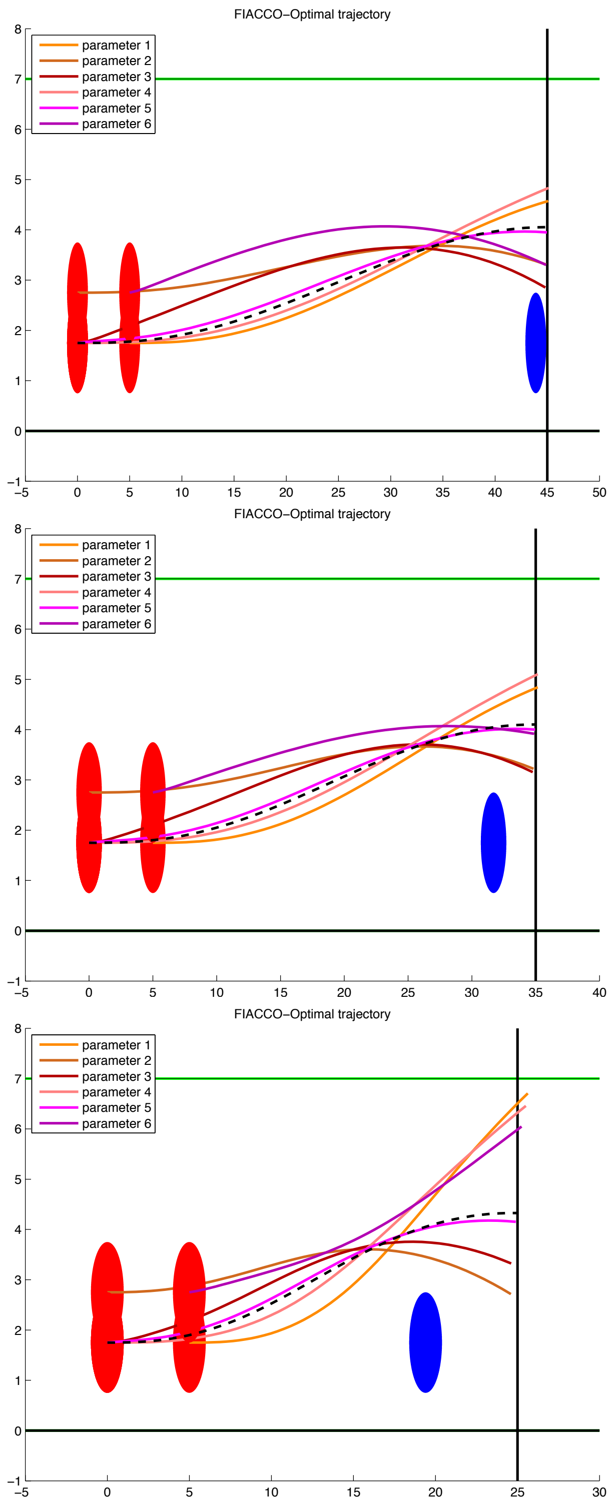


Figure 2.17: The Fiacco-perturbed trajectory is plot for a perturbation of $p = (5, 1, 0.1, 5, 0.5)$ for each warning level.

The radius is given straightforward:

```
Radius estimation for the optimal trajectory at warning level 1:
The radius of parameter 1 is 1.11282078 units
The radius of parameter 2 is 0.28044771 units
The radius of parameter 3 is 0.02146816 units
The radius of parameter 4 is 1.04972374 units
The radius of parameter 5 is 1.12156770 units
The total radius is 0.00512426 units


Radius estimation for the optimal trajectory at warning level 2:
The radius of parameter 1 is 1.08799803 units
The radius of parameter 2 is 0.28052736 units
The radius of parameter 3 is 0.02412671 units
The radius of parameter 4 is 1.13457006 units
The radius of parameter 5 is 1.17160768 units
The total radius is 0.00689615 units


Radius estimation for the optimal trajectory at warning level 3:
The radius of parameter 1 is 0.89301410 units
The radius of parameter 2 is 0.28668384 units
The radius of parameter 3 is 0.03296789 units
The radius of parameter 4 is 1.35522279 units
The radius of parameter 5 is 0.99717206 units
The total radius is 0.00940189 units
```

3 Collision avoidance by braking and steering algorithms

The CABS-algorithm deals with traffic complexity in urban surroundings and it is implemented in new assistance systems and safety systems with the aim to reduce the number of accidents. The algorithm is designed for finding real time trajectories for collision avoidance scenarios by parametric sensitivity analysis of a non-linear optimization problem. Parametric sensitivity analysis has been presented in Chapter III and is based on the idea that once a nominal optimal trajectory is found, then one can find real time approximation of optimal trajectories with small perturbation of the initial state in the non-linear optimal control problem whose solution is the nominal trajectory. The non-linear optimal control problem adopts the single track dynamics in (1.26) of Chapter II with seven states and two controls. The objective function is a linear combination of several functions that model the total time of maneuver, the length of the maneuver, the total velocity over the time horizon, the total slip angle over time horizon and the opposite of the minimum distance between vehicle and obstacle over the time step. Such function is minimized not over the controls, but over three time steps t_1, t_2, t_3 which are the switching times where the steering velocity control w_δ changes its value. The steering velocity is defined as the piecewise continuous function in Figure 3.1. The braking force is defined according to the Kamm's circle law (1.29) of Chapter II in such a way that the equality holds. Box constraints are imposed for the controls and the states.

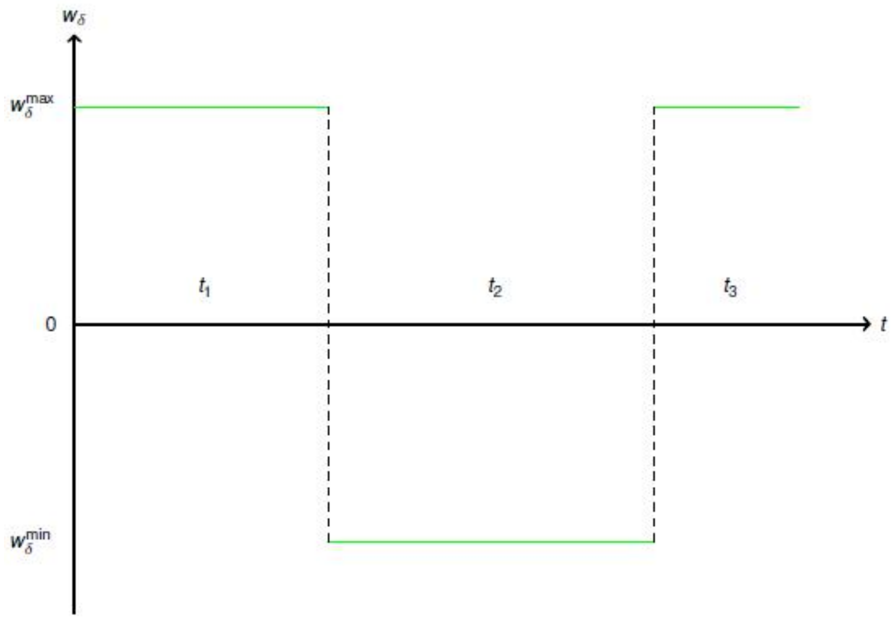


Figure 3.1: Definition of the steering velocity control w_δ.

The scenario (see Figure 3.2) implemented for the CABS-algorithm requests to avoid an obstacle driving in front of the reference vehicle. The two objects have the same velocity direction and the same yaw angle at initial time t_0. Due to the lack of data from the sensors in the reference vehicle, the obstacle geometry cannot be detected entirely, only its width is available. For these reasons in the model the obstacle set is seen as an offset to overcome at the final time. Thus terminal constraints are involving the final slip angle, steering angle, and yaw angle, as well as the final position of the vehicle. The road geometry is not considered and the obstacle motion is given a priori as a linear motion, i.e. no system identification techniques are applied to predict the obstacle motion.

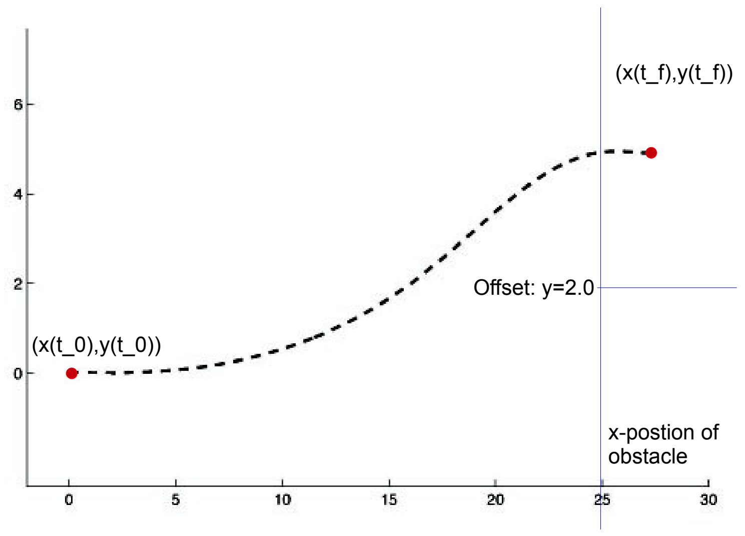


Figure 3.2: CABS algorithm scenario.

The CABS-algorithm input is then the offset, the x obstacle position and the initial state of the dynamical system modeling the reference vehicle motion. The output is composed of an optimal avoidance trajectory described by the seven states (vehicle position (x, y), vehicle velocity v, vehicle yaw angle ψ, vehicle yaw rate w_{ψ}, vehicle side slip angle α, vehicle steering angle δ) and two controls (vehicle steering velocity w_{δ} and vehicle braking force F_B) of Figures 3.3 and 3.4 is considered.

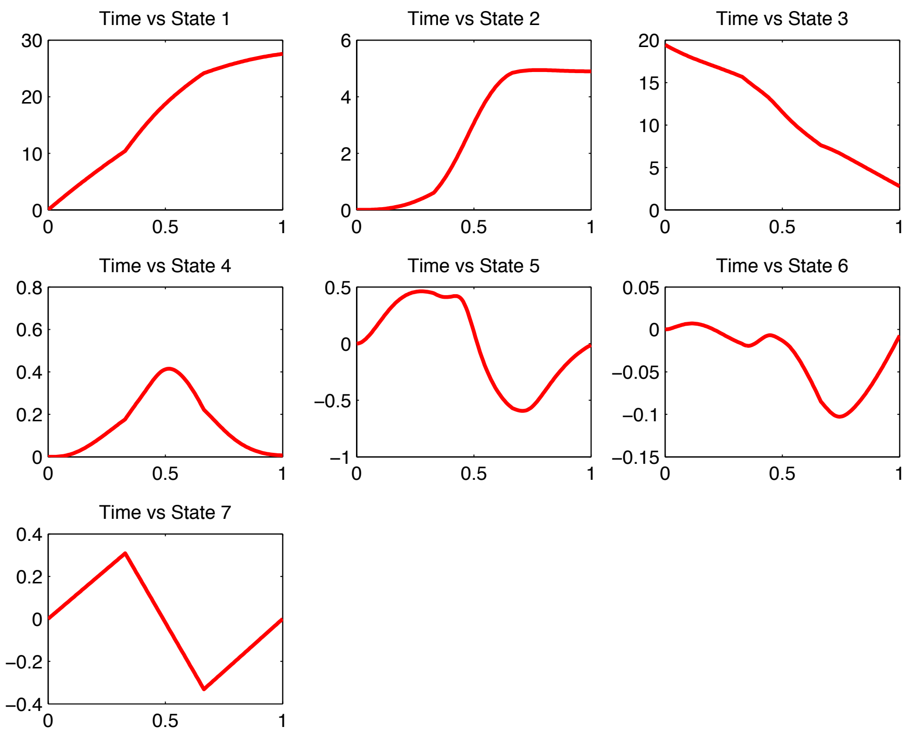
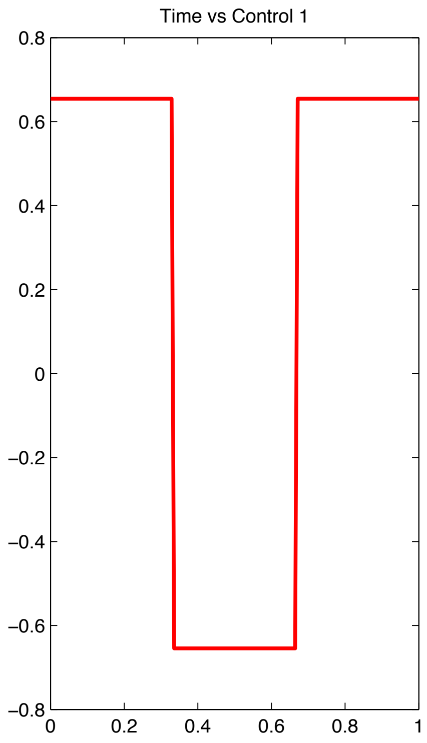


Figure 3.3: CABS trajectory-states $x, y, v, \psi, w_\psi, \alpha, \delta$.



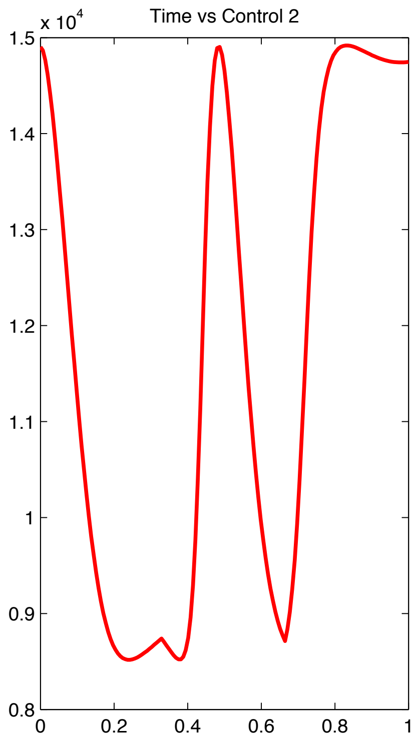


Figure 3.4: CABS trajectory-controls w_δ, F_B.

In the next simulations three scenarios are considered all with a straight road with lower bound -2 meters and upper bound 8 meters.

1. A single fixed circular obstacle of radius 1 meter and position $(25, 0)$ has to be avoided. The circular vehicle of radius 1 meter has initial state $z(0) = (0, 0, 19.4, 0, 0, 0, 0)$.
2. A fixed circular obstacle of radius 1 meter with initial position $(25, 0)$ has to be avoided by the circular vehicle of radius 1 meter with initial state $z(0) = (0, 0, 19.4, 0, 0, 0, 0)$. Another circular obstacle of radius 1 meter with initial position $(2, 6)$ and constant velocity $25m/s$ is overtaking the reference vehicle.
3. A circular obstacle of radius 1 meter with initial position $(25, 0)$ and moving of linear motion with initial velocity $10m/s$ and acceleration $-6m/s^2$ has to be avoided by the circular vehicle of radius 1 meter with initial state $z(0) = (0, 0, 19.4, 0, 0, 0, 0)$. Another circular obstacle of radius 1 meter with initial position $(40, 6)$ initial velocity $25m/s$ and initial acceleration $-6m/s^2$ is approaching the reference vehicle by driving in the opposite direction.

3.1 Sensitivity study

The starting point to construct a verification procedure for the CABS-algortihm, is to compute the ODE perturbed trajectories with respect to the nominal trajectory computed with CABS-algorithm (see Section 3.1 of Chapter IV for details). In Figure 3.5 the nominal trajectory is plotted for scenario described in Item (1). The perturbed trajectory set is plotted in Figure 3.6 and all the perturbed trajectories are collision free, indeed the considered perturbation value for each state is within the limit shown by the maximum perturbation radius estimation of Table 3.1. If perturbations are outside the ball of radius the maximum radius estimation, then the perturbed trajectory set is composed by trajectories that are leading to a collision or leaving the road, see Figure 3.7.

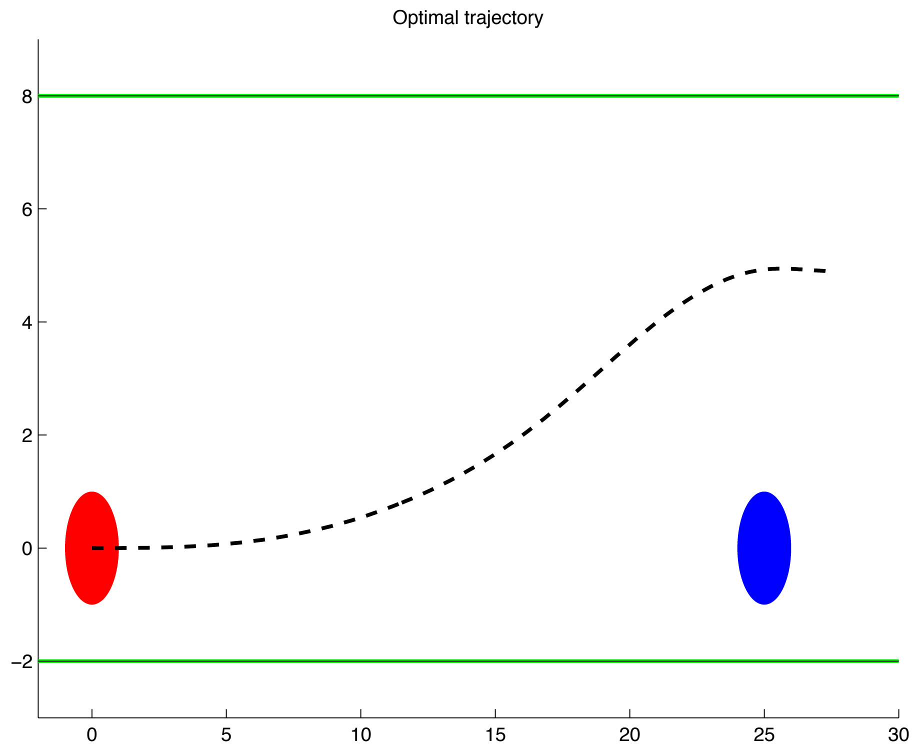


Figure 3.5: Sensitivity comparison-nominal trajectory.

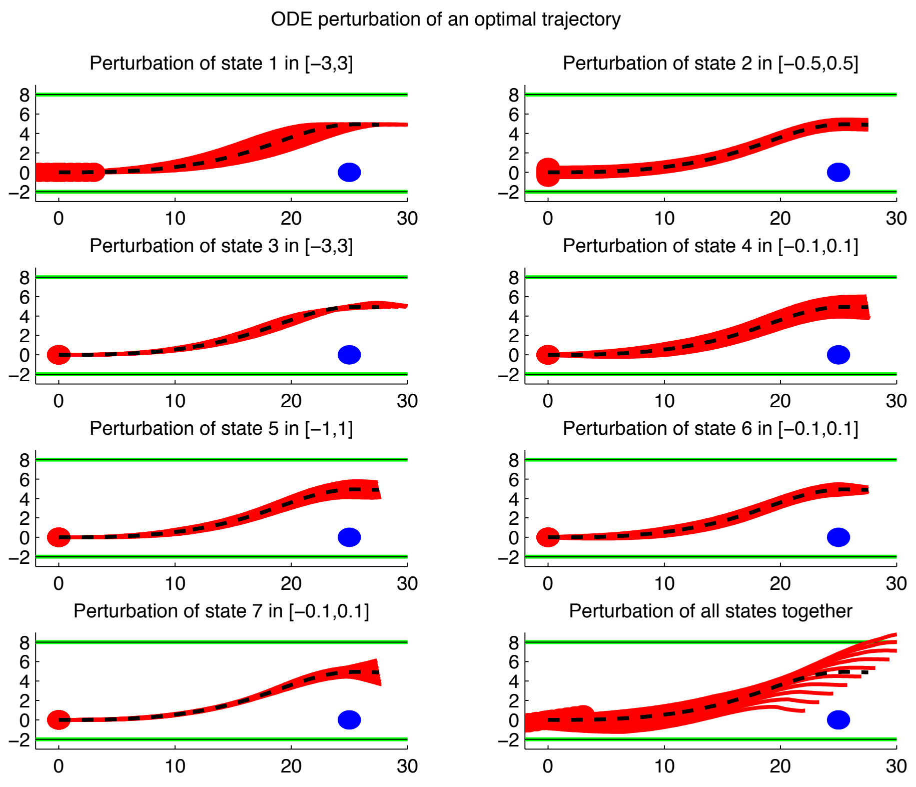


Figure 3.6: Sensitivity comparison-perturbation within the ball with maximum radius estimation boundaries and center in the nominal initial state.

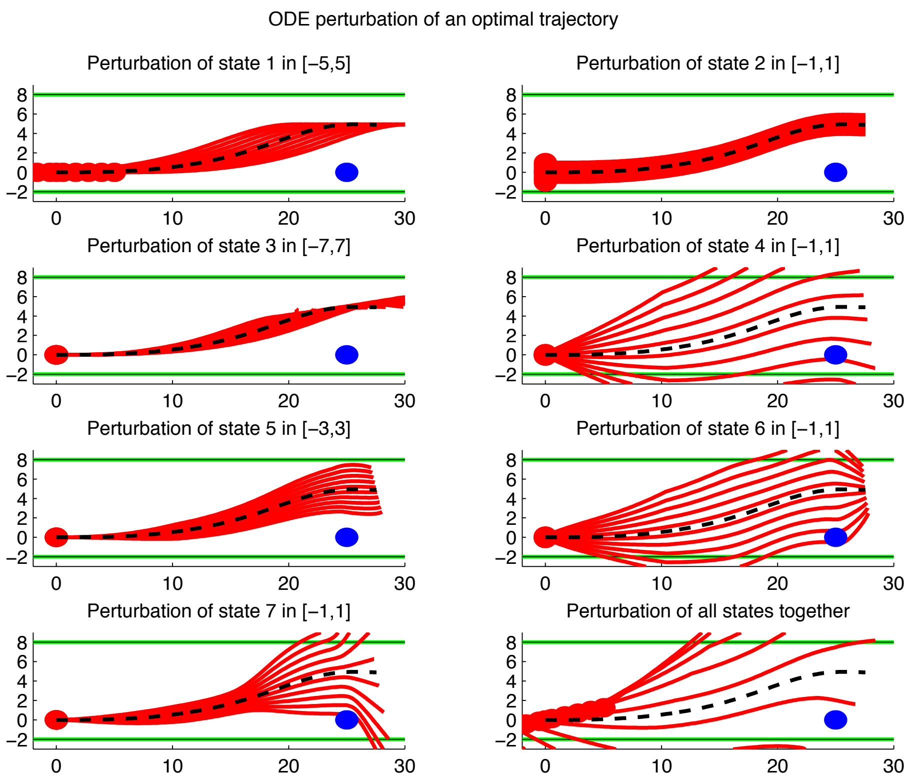


Figure 3.7: Sensitivity comparison-perturbation outside the ball with maximum radius estimation boundaries and center in the nominal initial state.

Table 3.1 gives the maximum radius estimation for ODE-perturbation of the initial value of the nominal optimal trajectory such that the perturbed trajectories are collision free. Roughly speaking, if the sensor detecting the initial state of the reference vehicle has tolerance within the computed radius in Table 3.1 then the nominal optimal trajectory is collision free even if affected by sensor errors in data detection. The Table shows two radius estimation in this particular case of verification of a CABS-trajectory. The first column is including the Kamm's circle state constraint, the second column does not take into account the state constraint and this leads to bigger radius estimation. The reason why this happens is because in the computation of the CABS-trajectory, the Kamm's circle state constraints is satisfied as equality, and thus as soon as the perturbation affects the equation, the condition is not satisfied. Thus if the state constraints depends on the state whose initial value is perturbed, there is no margin of perturbation for having this constraint still satisfied.

For the scenario described in Item (2) and the scenario described in Item (3) the optimal trajectories are given in Figures 3.8 and 3.9. The obstacles are moving and at the final time a crash is occurring. Thus the CABS-trajectory cannot be applied in scenarios where there is more than one obstacle, since in many cases this will lead

	with Kamm's circle state constraint	without Kamm's circle state constraint
`The radius of parameter 1 is`	`0.00000000 units`	`3.39492756 units`
`The radius of parameter 2 is`	`0.00000000 units`	`0.70000000 units`
`The radius of parameter 3 is`	`0.00000000 units`	`5.02035598 units`
`The radius of parameter 4 is`	`0.00000000 units`	`0.15962142 units`
`The radius of parameter 5 is`	`0.00000000 units`	`2.09258572 units`
`The radius of parameter 6 is`	`0.00000000 units`	`0.31363956 units`
`The radius of parameter 7 is`	`0.00000000 units`	`0.14582771 units`
`The total radius is`	`0.00000000 units`	`0.00683953 units`

Table 3.1: Maximum radius estimation with and without Kamm's circle constraints.

to a collision.

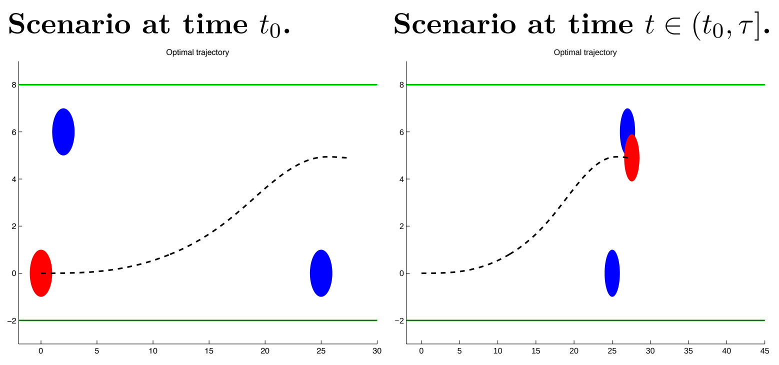


Figure 3.8: The scenario in Item (2) nominal trajectory and perturbed trajectory sets.

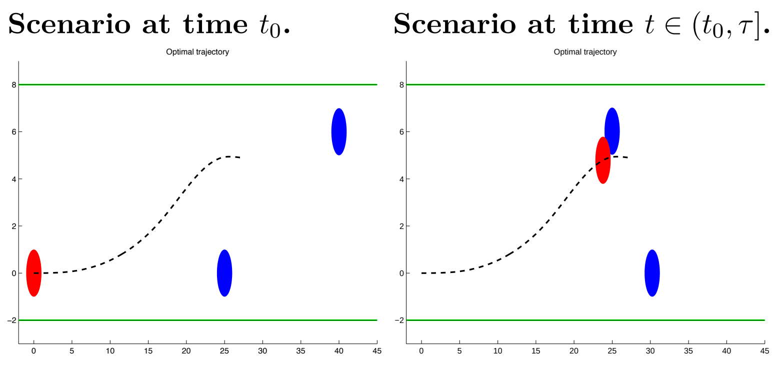


Figure 3.9: The scenario in Item (3) nominal trajectory and perturbed trajectory sets.

3.2 Comparison with backward reachable set

In Figure 3.10 the backward reachable set (see Definition 3.4 of Chapter II) is computed for scenarios in Items (1), (2), (3). In the computations, the reference vehicle as well as the obstacles are modeled as squares of dimension 2 meters. The vehicle dynamics is the 4d model in 1.31 of Chapter II since this gives the possibility to look at a larger number of trajectories. For scenarios in Items (1) and (2) the initial state considered here is in the computed reachable set, thus an optimal trajectory exists from such initial state. For scenario in Item (1) this trajectory is the CABS-trajectory in Figure 3.5, however for the scenario in Item (2) the CABS-trajectory is not collision free (see Figure 3.8 top-right picture), so the trajectory suggested by the VTM tool should be adopted. The scenario in Item (3) does not have a collision free trajectory from the given initial state and a collision will for sure happen. In this case the collision avoidance system based on the CABS-algorithm should not activate and a mitigation algorithm should be applied.

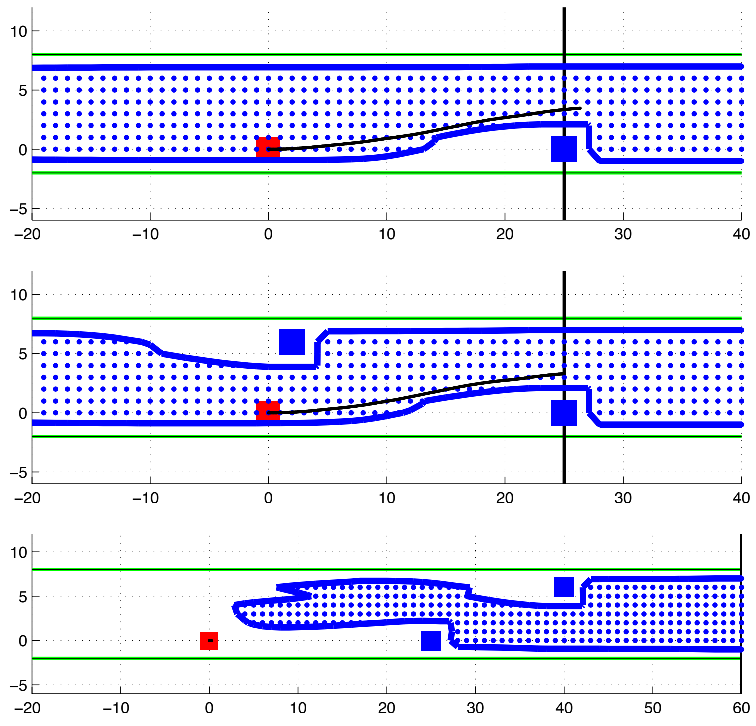


Figure 3.10: In first and second row the backward reachable set and a minimum time trajectory from the initial state $z_0 = (0, 0, 19.4, 0, 0, 0, 0)$ are computed for the scenario in Item (1) and the scenario in Item (2) respectively. The backward reachable set for the scenario in Item (3) is computed in the last row, where it is clear that an optimal trajectory from the initial state $z_0 = (0, 0, 19.4, 0, 0, 0, 0)$ leads always to a collision.

Chapter VI

Future Directions

Contents

A selection of possible future research directions is here presented. Industrial applications and problem formulations with different theoretical approaches are discussed for improving and developing collision avoidance systems.

The implementation in real car of algorithms based on optimal control approaches attracts the car industry attention. The advantages of a collision avoidance algorithm based on optimal control techniques is the possibility to deal with complex scenarios, and thanks to this to develop precise avoidance maneuvers. However the more complex the scenario is the slower are the computational times to find an optimal solution. The real time implementation is tricky and requires deep numerical analysis skills.

A second important issue in collision avoidance is the modelization of sensor data uncertainties or model errors. The use of stochastic optimal control can be an interesting tool to understand how these errors affect the problem and to obtain solutions that are robust with respect to such uncertainties. In this chapter the stochastic term is only introduced in the initial value problem obtained by fixing the control to a given optimal function.

The last research direction presented in this chapter tries to improve the physical model presented in Chapter II. The simplification made in Chapter II of considering an a priori known obstacle motion is too strong. In real life car traffic scenarios the obstacle motion is unknown and has a strong influence on the failure of the collision avoidance system. The obstacle motion depends on many factors, as human or environmental, leading to a very complex modelization problem involving several academic subjects, ranging from psychology to data learning. A game theory approach can be a compromise between the assumption of a pre-defined obstacle motion and the one fully faithful to the complexity of a real life scenario. Therefore a car traffic scenario can be considered as a game between obstacles and vehicle. In the worse case scenario a non-cooperative game has to be considered, for the best case scenario a cooperative game is required.

1 Real time implementation of collision avoidance trajectories

Implementation in cars of collision avoidance algorithms via optimal control techniques is an attractive field in car industry. Such success is due to the fact that optimal control problems model a wide rage of typologies of car traffic scenario, with more than one obstacle and several road geometries, considering also the capabilities of the reference vehicle. The idea here shown concerns the basic scenario in collision avoidance. An obstacle has to be overcome by a car in a straight road and no other obstacles are involved. It appears that such a scenario is commonly happening in many real life situations and many data has been collected. A comparison with human driver trajectories is extremely useful to have a feedback on the correctness of the model. Moreover a look at real life trajectories will lead to a better understanding of the maneuver performance criteria, and thus of the choice of the objective function. Since optimal control solvers do not give real time solutions, the idea is to implement in the car the optimal solutions found off-line sorted by initial data and scenario typology.

The experimental data used in the following examples are collected by the Driver Assistance System department in Volkswagen Research. Four real life situations are considered where a vehicle overtakes an obstacle in a straight road.

First Example. Using the notation of Problem 3.1 of Chapter II the initial state is:

$$\begin{aligned} z(0) &= (x(0), y(0), \psi(0), v_x(0), v_y(0), w_\psi(0), \delta(0)) \\ &= (0, 1.75, 0, 21, 0, -0.0027, 0.05) \end{aligned} \tag{1.1}$$

where (x, y) is the vehicle center of gravity, ψ is the yaw angle, (v_x, v_y) are the velocity components in $[m/s]$, w_ψ is the yaw rate, and δ is the steering angle. The obstacle initial position is $(15.42, 1.10)$ moving with acceleration $0.74[m/s^2]$ and initial velocity $19.35[m/s]$. In the simulations the obstacle and the vehicle are considered as squares of dimension $2[m]$ and the road has width $7[m]$. The final condition imposes that the maneuver terminates at time t_f such that the x-position of the vehicle at time t_f equals the one of the obstacle.

In Figure 1.1 the real trajectory and the velocity value are compared with the ones of simulated data for different objective functions. It appears that the experimental data trajectory is a combination between the minimal initial distance and the minimal time solution.

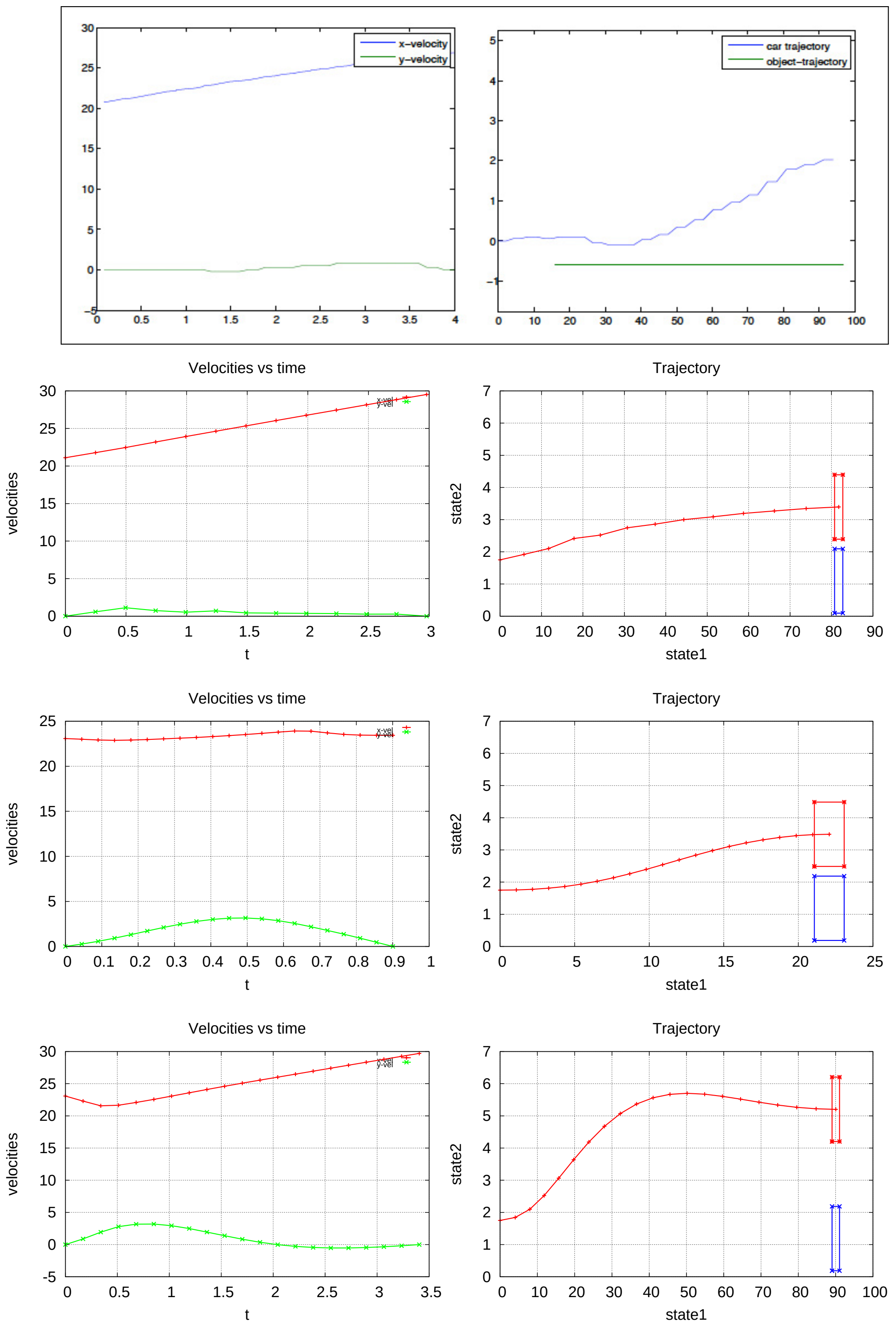


Figure 1.1: The first column shows the velocity component values v_x, v_y in $[m/s]$, while the second column plots the trajectory in the (x, y) plane. The first row shows the data related to real life scenarios. The second road is the minimal time solution. The third row is the solution obtained by minimizing the initial x-distance between vehicle and obstacle. The last row solution is obtained by minimizing the steering effort.

Second Example. In this case the obstacle is smaller and slower than in the first example, moreover it is really close to the lower bound of the road. Thus:

$$z(0) = (x(0), y(0), \psi(0), v_x(0), v_y(0), w_\psi(0), \delta(0))$$

$$= (0, 1.75, 0, 20.38, 0, 0.0076, 0.08).$$

The obstacle initial position is $(13.55, 0.75)$ moving with acceleration $0.67[m/s^2]$ and initial velocity $17.13[m/s]$. In the simulations the obstacle and the vehicle are considered as squares of dimension $1[m]$ and $2[m]$ respectively, and the road has width $7[m]$. As in the first example, the final condition imposes that the maneuver terminates at time t_f such that the x-position of the vehicle at time t_f equals the one of the obstacle.

In Figure 1.2 the real trajectory and the velocity value are compared with the ones of simulated data for different objective functions. It appears that the experimental data trajectory is well approximated by the minimal time solution.

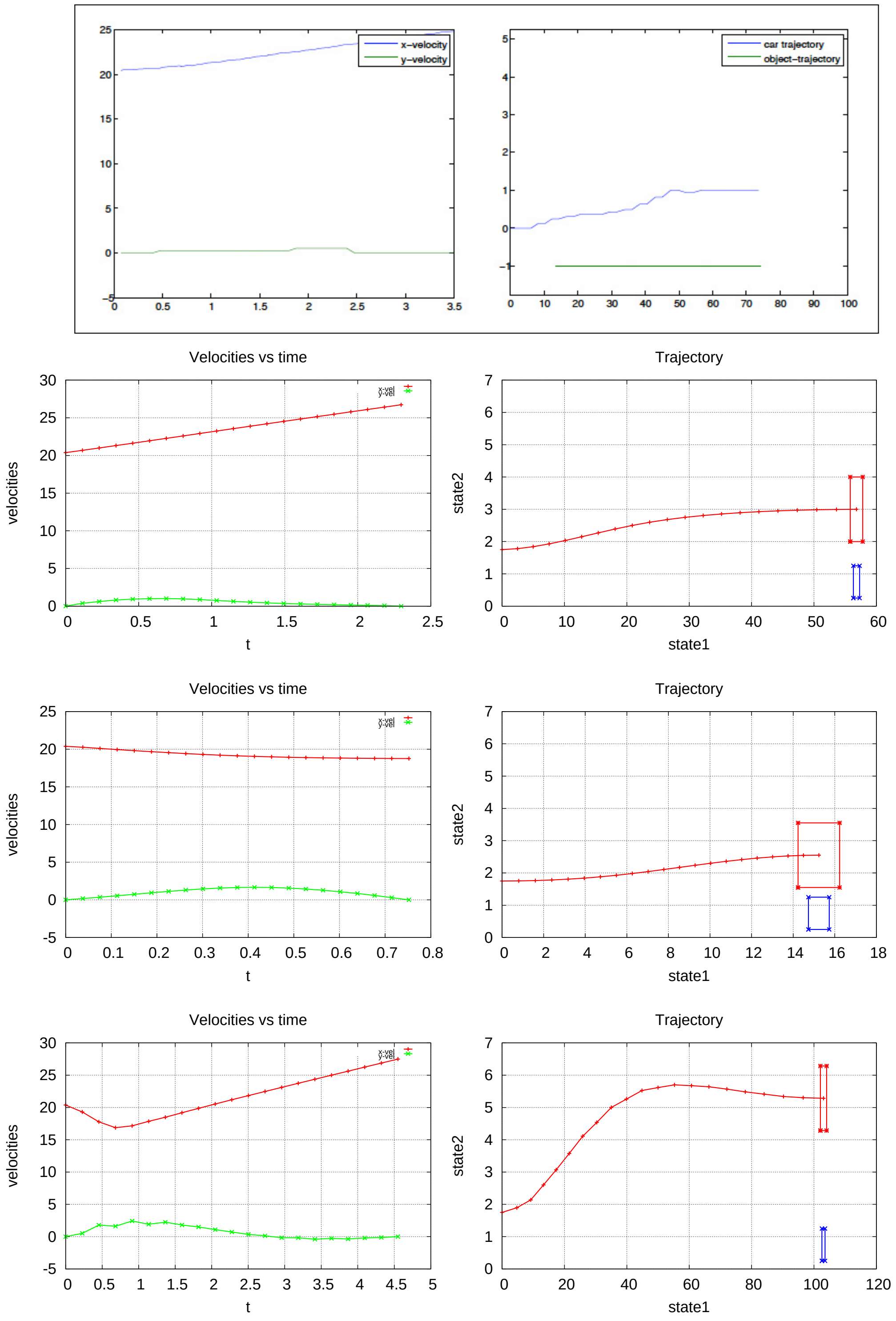


Figure 1.2: The first column shows the velocity component values v_x, v_y in $[m/s]$, while the second column plots the trajectory in the (x, y) plane. The first row shows the data related to real life scenarios. The second road is the minimal time solution. The third row is the solution obtained by minimizing the initial x-distance between vehicle and obstacle. The last row solution is obtained by minimizing the steering effort.

Third Example. The initial vehicle state is:

$$
\begin{aligned}
z(0) &= (x(0), y(0), \psi(0), v_x(0), v_y(0), w_\psi(0), \delta(0)) \\
&= (0, 1.75, 0, 24.4, 0, -0.019, 0.0).
\end{aligned}
$$

The obstacle initial position is $(14.74, 0.72)$ moving with acceleration $0.51[m/s^2]$ and initial velocity $21.79[m/s]$. In the simulations the obstacle and the vehicle are considered as squares of dimension $2[m]$, and the road has width $7[m]$. As in the first example, the final condition imposes that the maneuver terminates at time t_f such that the x-position of the vehicle at time t_f equals the one of the obstacle.

In Figure 1.3 the real trajectory and the velocity value are compared with the ones of simulated data for different objective functions. It appears that the experimental data trajectory is well approximated by the minimal time solution and the solution minimizing the steering effort.

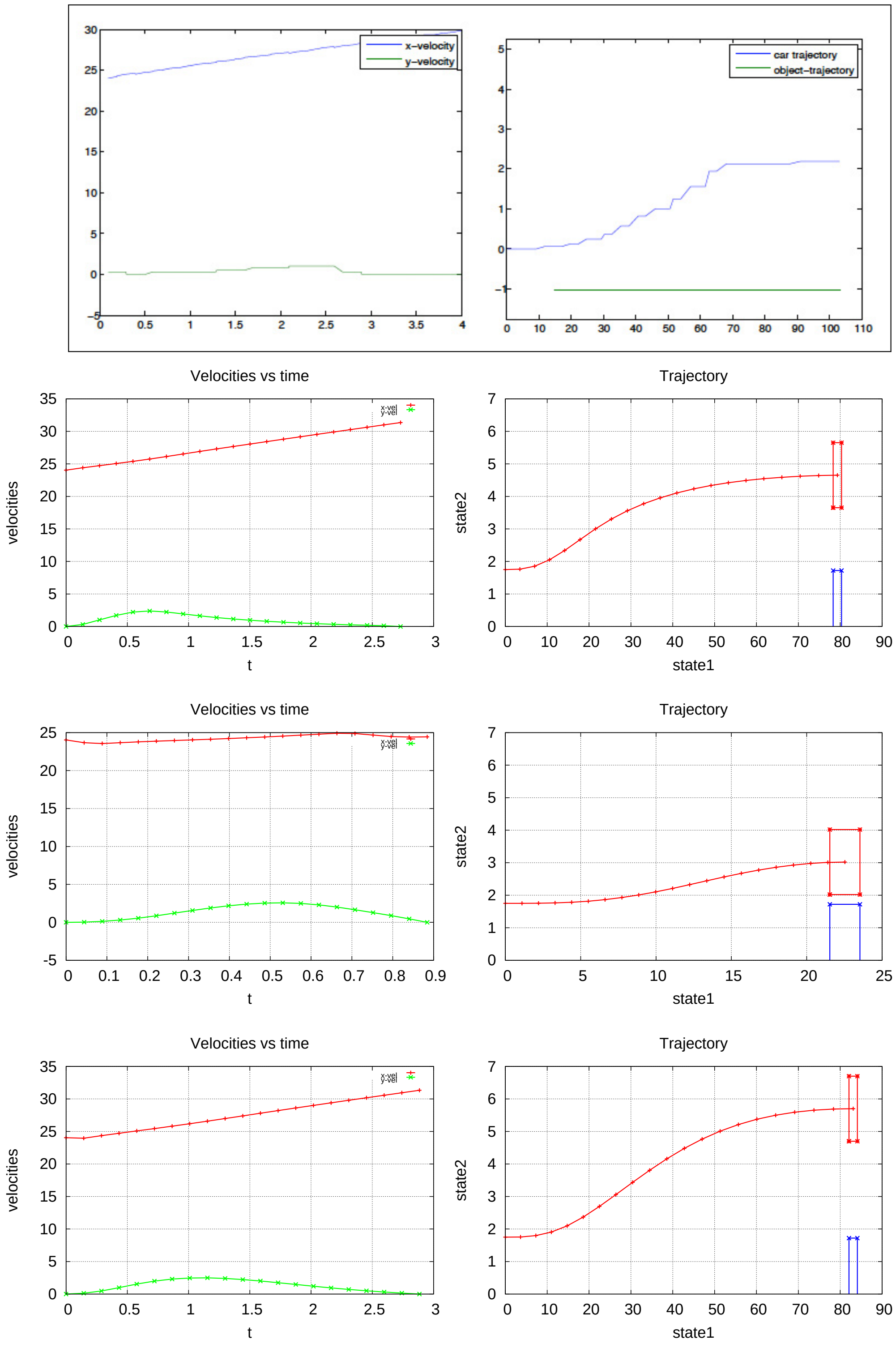


Figure 1.3: The first column shows the velocity component values v_x, v_y in $[m/s]$, while the second column plots the trajectory in the (x, y) plane. The first row shows the data related to real life scenarios. The second road is the minimal time solution. The third row is the solution obtained by minimizing the initial x-distance between vehicle and obstacle. The last row solution is obtained by minimizing the steering effort.

Fourth Example. In this case the obstacle is smaller and slower than in the first example. The initial vehicle state is:

$$\begin{aligned} z(0) &= (x(0), y(0), \psi(0), v_x(0), v_y(0), w_\psi(0), \delta(0)) \\ &= (0, 1.75, 0, 20.57, 0, -0.001, 0.05). \end{aligned}$$

The obstacle initial position is $(7.2, 1.2)$ moving with acceleration $0.40[m/s^2]$ and initial velocity $19.14[m/s]$. In the simulations the obstacle and the vehicle are considered as squares of dimension $1[m]$ and $2[m]$ respectively, and the road has width $7[m]$. As in the first example, the final condition imposes that the maneuver terminates at time t_f such that the x-position of the vehicle at time t_f equals the one of the obstacle.

In Figure 1.4 the real trajectory and the velocity value are compared with the ones of simulated data for different objective functions. It appears that the experimental data trajectory is well approximated by the solution obtained minimizing the steering effort.

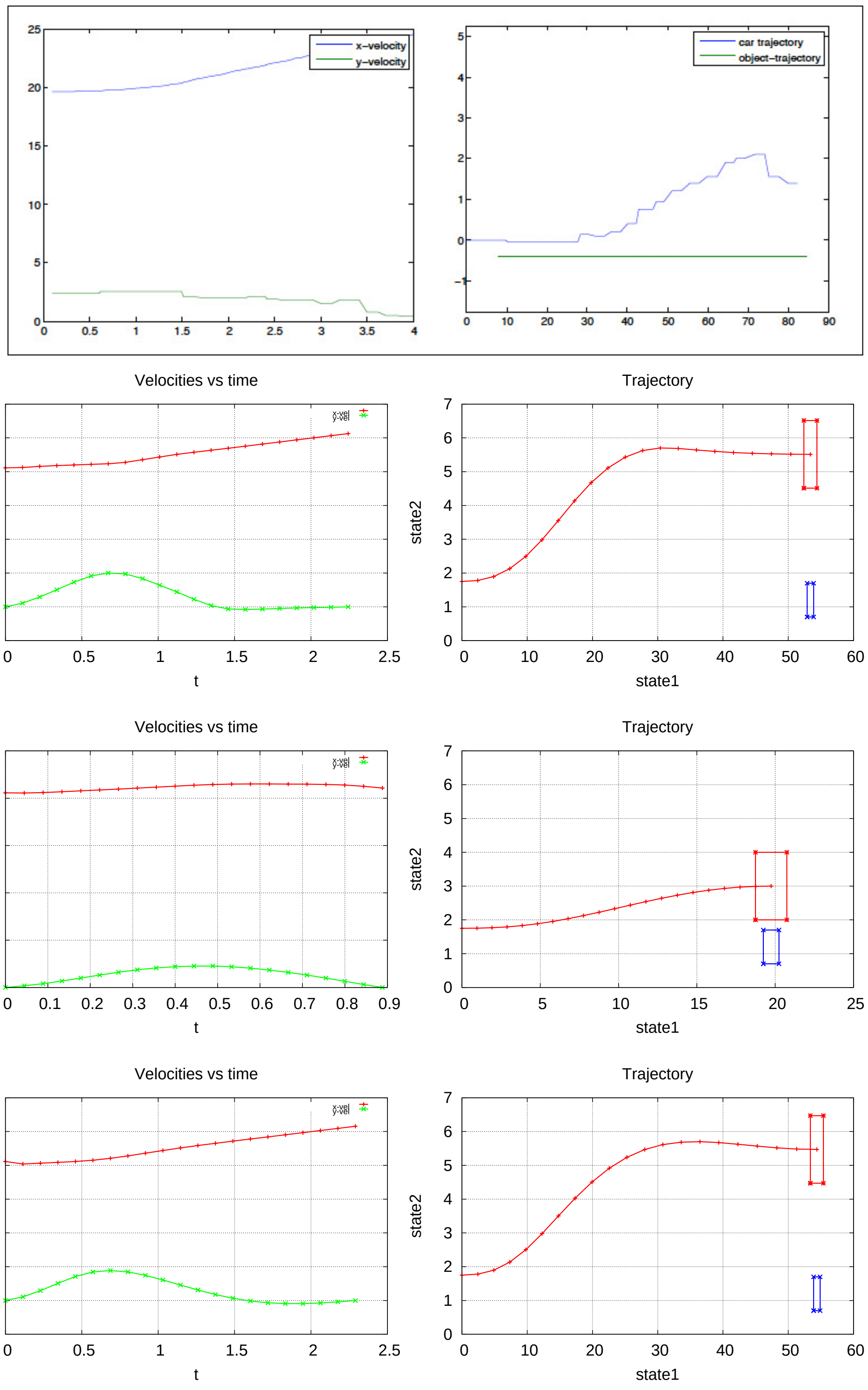


Figure 1.4: The first column shows the velocity component values v_x, v_y in $[m/s]$, while the second column plots the trajectory in the (x, y) plane. The first row shows the data related to real life scenarios. The second road is the minimal time solution. The third row is the solution obtained by minimizing the initial x-distance between vehicle and obstacle. The last row solution is obtained by minimizing the steering effort.

The four examples show that all objectives may occur, largely depending on the driver behavior. Thus to implement optimal trajectories in real car, the driver behavior needs to be detected and the corresponding avoidance trajectory can be applied. For instance for an aggressive driver, a minimum initial distance trajectory is more suitable and it avoids the risk of nuisance warnings. While for an "average" driver a minimum steering effort trajectory is more appropriate to avoid panic attacs in the driver.

2 Stochastic approach

Given an initial state $z_0 \in \mathbb{R}^{n_z}$, $n_z \geq 1$ integer, for any control policy $u \in \mathcal{U} = L^\infty([t_0, t_f], U)$, $z^u_{z_0}(\cdot)$ denotes the Caratheodory solution of the following dynamical system

$$\begin{aligned} &\dot{z}(t) = f(z(t), u(t)), \quad a.e.\ t \geq t_0, \\ &z(t_0) = z_0. \end{aligned} \tag{2.1}$$

Let $\hat{u} \in \mathcal{U}$ such that the function $z^{\hat{u}}_{z_0}$ satisfies the following constraints:

$$z^{\hat{u}}_{z_0}(t_f) \in \mathcal{C}, \tag{2.2}$$

with $\mathcal{C}$ non empty closed set of $\mathbb{R}^{n_z}$.

As seen in Section 3.3 of Chapter IV, it is possible to estimate the maximum $r > 0$ such that for all $\xi \in B(z_0, r)$, $z^{\hat{u}}_{\xi}$ solves (2.1) with $z(t_0) = \xi$, and (2.2). A similar estimation for r is here found if a stochastic term for modeling errors in the initial data is introduced.

Let the stochastic initial value problem in (2.3) be considered.

$$\begin{aligned} &\dot{Z}(t) = f(Z(t), \hat{u}(t)), \quad a.e.\ t \geq t_0, \\ &Z(t_0) = z_0 + \varepsilon W =: \xi. \end{aligned} \tag{2.3}$$

where W is for instance a random variable with law $N(0,1)$ and $\hat{u} \in \mathcal{U}$ given such that (2.2) holds.

Then the perturbed trajectory starts with the value $Z^\epsilon_0 = z_0 + \epsilon W$ where W is an $N(0,1)$ random variable. Thus

$$Z^0_{t_0} = z_0 + \int_0^{t_0} f(Z_t)dt \tag{2.4}$$

and

$$Z^\epsilon_{t_0} = Z^\epsilon_0 + \int_0^{t_0} f(Z^\epsilon_t)dt. \tag{2.5}$$

By difference,

$$|Z^\epsilon_{t_0} - Z^0_{t_0}| \leq |Z^\epsilon_0 - z_0| + \int_0^{t_0} L|Z^\epsilon_t - Z^0_t|dt, \tag{2.6}$$

where L is the Lipschitz constant of the dynamics f. Then by the Gronwall Lemma

$$|Z^\epsilon_{t_0} - Z^0_{t_0}| \leq e^{Lt_0}|Z^\epsilon_0 - z_0| \leq \ \epsilon \ e^{Lt_0}|W|, \quad \forall t_0 \geq 0. \tag{2.7}$$

Assuming that $Z^0_{t_f} \in \mathcal{C}$, the probability to reach the target for the perturbed trajectory is computed. More precisely, the size of ϵ (which measures the size of the initial perturbation) needs to be found, such that, for instance,

$$\mathbb{P}[Z^\epsilon_{t_f} \in \mathcal{C}] \geq \Delta = 1 - a, \quad a \text{ small} \tag{2.8}$$

where here Δ corresponds to the confidence to be achieved.

If

$$d_{\mathcal{C}}(Z^0_{t_f}) = -\delta < 0 \tag{2.9}$$

(where $d_{\mathcal{C}}$ denotes the signed distance to $\mathcal{C}$), then $B(Z^0_{t_f}, \delta) \subset \mathcal{C}$, and

$$\begin{array}{rcl} \mathbb{P}[Z^\epsilon_{t_f} \in \mathcal{C}] & \geq & \mathbb{P}[Z^\epsilon_{t_f} \in B(Z^0_{t_f}, \delta)] \\ & = & \mathbb{P}[|Z^\epsilon_T - Z^0_{t_f}| \leq \delta] \\ & \geq & \mathbb{P}[\epsilon\, e^{Lt_f}|W| \leq \delta] \\ & \geq & \mathbb{P}[|W| \leq e^{-Lt_f}\frac{\delta}{\epsilon}] = 1 - \mathbb{P}[|W| \geq e^{-Lt_f}\frac{\delta}{\epsilon}] \end{array} \tag{2.10}$$

Now let $b := e^{-Lt_f}\frac{\delta}{\epsilon}$. Assuming that δ/ϵ is large (or ϵ sufficiently small with respect to δ), then the following holds

$$\begin{array}{rcl} \mathbb{P}[|W| \geq b] & = & \frac{1}{(2\pi)^{d/2}} \int_{\|z_0\| \geq b} e^{-\|z_0\|^2/2} dz_0 \\ & = & (2\pi)^{-d/2} S_{d-1} \int_b^\infty r^{d-1} e^{-r^2/2} dr \\ & \sim & (2\pi)^{-d/2} S_{d-1} b^{d-2} e^{-b^2/2}, \quad b \to \infty \end{array} \tag{2.11}$$

where S_{d-1} is the unit surface in $\mathbb{R}^d$, and W random variable with values in $\mathbb{R}^d$.

Thus in order to have (2.8), b has to be such that:

$$(2\pi)^{-d/2} S_{d-1} b^{d-2} e^{-b^2/2} \sim a, \tag{2.12}$$

so

$$-b^2/2 + o(b) \sim \log a, \tag{2.13}$$

and therefore

$$b \sim \sqrt{-2\log a}. \tag{2.14}$$

To conclude

$$\epsilon \sim e^{-Lt_f} \frac{\delta}{\sqrt{-2\log a}} \tag{2.15}$$

For instance, if $a = 0.01$ the value $\frac{1}{\sqrt{-2\log a}} \equiv 0.33$ is obtained.

3 Game theory approach

In this section a pedestrian motion and a car motion subject to uncertainties are modelled as differential games.

3.1 Models

Car Model. The dynamics of the car is given by

$$\begin{array}{rcl} z'_C(t) & = & f_C(z_C(t), u_C(t)), \text{ a.e in } [0,T] \\ z_C(0) & = & z_{C,0} \end{array} \tag{3.1}$$

where

- The time interval $[t_0, t_f]$ is fixed.
- The state is $z_C := (x_C, y_C, v_C, \psi_C)$; (x_C, y_C) are the coordinates of the center of gravity of the car, v_C is the velocity function and ψ_C is the steering angle function.
- The control is $u_C := (a_C, w_{\psi C})$; a_C is the acceleration function of the vehicle and $w_{\psi C}$ is the steering angular velocity function.
- The function f_C is defined as:

$$\begin{array}{rcl} x'_C(t) & = & v_C(t)\cos(\psi_C(t)), \\ y'_C(t) & = & v_C(t)\sin(\psi_C(t)), \\ v'_C(t) & = & a_C(t), \\ \psi'_C(t) & = & w_{\psi C}(t). \end{array} \tag{3.2}$$

- The vector of initial state $z_{C,0}$ is constant.

Perturbed Car Model. If perturbations in the dynamics or in the initial state are considered then the (3.1) can be written as (3.3). Perturbations in the dynamics f_C and in the initial value $z_C(0)$ are modeled by the function $p = (p_1, p_2) \in \mathcal{P} = L^\infty([t_0, \infty], P)$, $P \subset \mathbb{R}^2$ and for $i = 1, 2$, $p_i(t) \in \mathcal{B}(\gamma_{p_i}, B_{p_i})$, the ball of radius γ_{p_i} centered in B_{p_i}.

The dynamics of the car is, in this case, given by

$$\begin{array}{rcl} z'_C(t) & = & f_C(z_C(t), u_C(t)) + p_1(t), \text{ a.e in } [t_0, t_f] \\ z_C(0) & = & z_{C,0} + p_2(t) \end{array} \tag{3.3}$$

where $z_C, u_C, f_C, z_{C,0}$ are defined as before and the time interval $[t_0, t_f]$ is fixed.

Pedestrian Model. The dynamics of the pedestrian is given by

$$\begin{array}{rcl} z'_P(t) & = & f_P(z_P(t), u_P(t)), \text{ a.e in } [t_0, t_f] \\ z_P(0) & = & z_{P,0} \end{array} \tag{3.4}$$

where

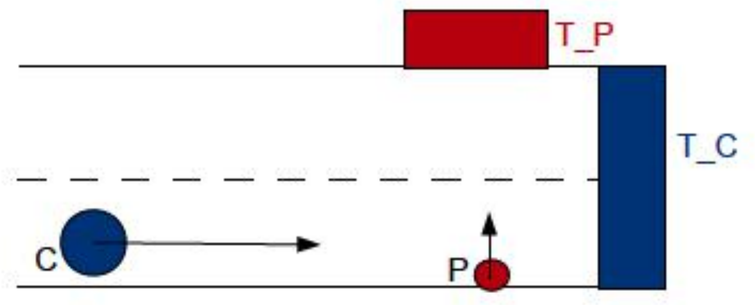
- The time interval $[t_0, t_f]$ is fixed.
- The state is $z_P := (x_P, y_P)$, the coordinates of the center of gravity of the pedestrian.
- The control is $u_P := (v_P^x, v_P^y)$; v_P^x and v_P^y are the components of the pedestrian's velocity in the x-axis and in the y-axis respectively.
- The function f_P is defined as:

$$\begin{array}{rcl} x'_P(t) & = & v_P^x(t), \\ y'_P(t) & = & v_P^y(t). \end{array} \tag{3.5}$$

- The vector of initial state $z_{P,0}$ is constant.

3.2 Differential game

Pedestrian-Car. The first game is between a suicide pedestrian P and a car C. The pedestrian is modeled as a ball of radius r_P constant and centered in $(x_P(t), y_P(t))$, the car is modeled as a ball of constant radius r_C and center $(x_C(t), y_C(t))$.



- **Terminal state constraints.** They want to reach their own target, $T_C = [\underline{x}_C, +\infty] \times [-\infty, +\infty]$ for the car and $T_P = [\underline{x}_P, \overline{x}_P] \times [\underline{y}_P, +\infty]$ for the pedestrian, in a safe position:

$$\begin{array}{l} \underline{x}_P \leq x_P(t_f) \leq \overline{x}_P, \\ y_P(t_f) \geq \underline{y}_P, \\ x_C(t_f) \geq \underline{x}_C, \\ \psi_C(t_f) = 0, \end{array} \tag{3.6}$$

  where $\underline{x}_P$, $\overline{x}_P$, $\underline{y}_P$ and $\underline{x}_C$ are constants.

- **Control constraints.** The controls are bounded as follow:

$$\begin{array}{l} \underline{u}_P \leq u_P(t) \leq \overline{u}_P, \\ \underline{u}_C \leq u_C(t) \leq \overline{u}_C, \end{array} \tag{3.7}$$

  where $\underline{u}_P$, $\overline{u}_P$, $\underline{u}_C$ and $\overline{u}_C$ are constants.

- **State constraints.** The car C has to stay on the road for the whole path, until it reaches the target T_C, so:

$$\underline{y}_R \leq y_C(t) \leq \overline{y}_R, \tag{3.8}$$

  and $\underline{y}_R, \overline{y}_R$ are the lower and upper bound of a straight road respectively.

  Moreover the crash is modelled with the following state constraint:

$$\begin{array}{l} \alpha + (x_P(t) - x_C(t))^2 + (y_P(t) - y_C(t))^2 \\ -(r_P(t) + r_C(t))^2 \geq 0, t \in [t_0, t_f], \\ \alpha \geq 0, \end{array}$$

  α is defined in the next item.

- **Differential equations.**

$$\begin{array}{rcl} z'_P(t) & = & f_P(z_P(t), u_P(t)), \text{ a.e in } [t_0, t_f] \\ z_P(t_0) & = & z_{P,0} \end{array} \tag{3.9}$$

and

$$\begin{aligned} z'_C(t) &= f_C(z_C(t), u_C(t)), \text{ a.e in } [t_0, t_f] \\ z_C(t_0) &= z_{C,0}. \end{aligned} \tag{3.10}$$

Moreover the following equation tells that α is a constant state:

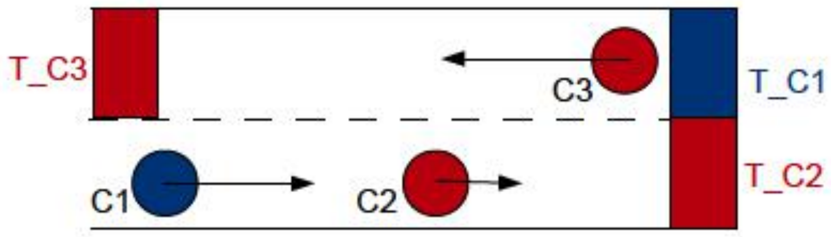
$$\alpha' = 0. \tag{3.11}$$

- **Objective function.** While C is trying to avoid P, P is trying to crash with C, so the equation

$$\int_{t_0}^{t_f} \max\{0, (r_P + r_C)^2 - (x_P(t) - x_C(t))^2 - (y_P(t) - y_C(t))^2\}^2 dt \tag{3.12}$$

is minimized over u_C and maximized over u_P in a zero-sum game.

Car1-Car2-Car3. In this game two cars $C2, C3$, modeled as balls of radius r_{C2}, r_{C3} centered in $(x_{C2}(t), y_{C2}(t))$, $(x_{C3}(t), y_{C3}(t))$, try to hit car $C1$, modeled as a ball of radius r_{C1} and center $(x_{C1}(t), y_{C1}(t))$. The radius r_{C1}, r_{C2}, r_{C3} are constant.



- **Terminal state constraints.** All the cars involved want to reach their own target, $T_{C1} = [\underline{x}_{C1}, +\infty] \times [-\infty, +\infty]$, $T_{C2} = [\underline{x}_{C2}, +\infty] \times [-\infty, \overline{y}_{C2}]$, $T_{C3} = [-\infty, \overline{x}_{C3}] \times [\underline{y}_{C3}, +\infty]$, in a safe position:

$$\begin{array}{lll} x_{C1}(t_f) \geq \underline{x}_{C1}, & \psi_{C1}(t_f) = 0, & \\ x_{C2}(t_f) \geq \underline{x}_{C2}, & y_{C2}(t_f) \leq \overline{y}_{C2}, & \psi_{C2}(t_f) = 0, \\ x_{C3}(t_f) \leq \overline{x}_{C3}, & y_{C3}(t_f) \geq \underline{y}_{C3}, & \psi_{C3}(t_f) = 0, \end{array} \tag{3.13}$$

where $\underline{x}_{C1}$, $\underline{x}_{C2}$, $\overline{y}_{C2}$, $\overline{x}_{C3}$ and $\underline{y}_{C3}$ are constants.

- **Control constraints.** The controls are bounded as follow:

$$\underline{u}_C \leq u_{Ci}(t) \leq \overline{u}_C, \quad \text{for all} \quad i \in \{1, 2, 3\} \tag{3.14}$$

where $\underline{u}_C$ and $\overline{u}_C$ are constants.

- **State constraints.** It can appear that in this kind of game almost any constellation will lead to a collision, i.e. the car 1 cannot avoid it. However if the following constraints are impose an improvement on the avoidance possibilities is expected. Each car Ci has to stay on the road for the whole path, until it reaches the target T_{Ci}, so:

$$\underline{y}_R \leq y_{Ci}(t) \leq \overline{y}_R, \quad \text{for all} \quad i \in \{1,2,3\} \tag{3.15}$$

and $\underline{y}_R, \overline{y}_R$ are the lower and upper bound of a straight road respectively.

Moreover the crash is modelled with the following state constraint:

$$\begin{array}{l} \alpha + (x_{C2}(t) - x_{C1}(t))^2 + (y_{C2}(t) - y_{C1}(t))^2 \\ -(r_{C2} + r_{C1})^2 \geq 0, \\ \alpha + (x_{C3}(t) - x_{C1}(t))^2 + (y_{C3}(t) - y_{C1}(t))^2 \\ -(r_{C3} + r_{C1})^2 \geq 0, \\ \alpha \geq 0, \end{array}$$

α is a constant state. Also $C2$ does not have to crash with $C3$:

$$(x_{C2}(t) - x_{C3}(t))^2 + (y_{C2}(t) - y_{C3}(t))^2 - (r_{C2} + r_{C3})^2 \geq 0. \tag{3.16}$$

- **Differential equations.** Objects $C1, C2, C3$ are following the car model described in (3.1):

$$\begin{array}{rcl} z'_{Ci}(t) &=& f_{Ci}(z_{Ci}(t), u_{Ci}(t)), \text{ a.e in } [t_0, t_f] \\ z_{Ci}(t_0) &=& z_{Ci,0} \end{array} \tag{3.17}$$

for $i \in \{1,2,3\}$. As before the following equation tells that α is a constant state:

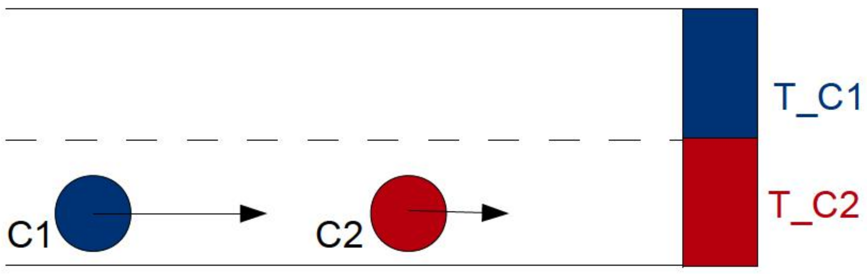
$$\alpha' = 0. \tag{3.18}$$

- **Objective function.** While $C1$ is trying to avoid $C2$ and $C3$, $C2$ and $C3$ are trying to crash with $C1$, so:

$$\int_{t_0}^{t_f} \max\{0, (r_P + r_C)^2 - (x_P(t) - x_C(t))^2 - (y_P(t) - y_C(t))^2\}^2 dt \tag{3.19}$$

is minimized over u_{C1} and maximized over u_{C2}, u_{C3} in a zero-sum game.

Perturbation-Car. In this game car $C1$, modeled as a ball of fixed radius r_1 and center $(x_1(t), y_1(t))$, tries to avoid car $C2$, a balls of fixed radius r_2 and center $(x_2(t), y_2(t))$, moving with a given motion. Errors in the dynamics and in the initial state are affecting $C1$ and in particular they play against it.



- **Terminal state constraints.** Object $C1$ wants to reach the target, $T_{C1} = [\underline{x}_{C1}, +\infty] \times [-\infty, +\infty]$, in a safe position:

$$\begin{array}{l} x_{C1}(t_f) \geq \underline{x}_{C1} \\ \psi_{C1}(t_f) = 0, \end{array} \tag{3.20}$$

where $\underline{x}_{C1}$ is a constant.

- **Control constraints.** The control is bounded as follow:

$$\underline{u}_{C1} \leq u_{C1}(t) \leq \overline{u}_{C1}, \tag{3.21}$$

where $\underline{u}_{C1}$ and $\overline{u}_{C1}$ are constants.

- **State constraints.** $C1$ has to stay on the road for the whole path, until it reaches the target T_{C1}, so:

$$\underline{y}_R \leq y_{C1}(t) \leq \overline{y}_R \tag{3.22}$$

and $\underline{y}_R, \overline{y}_R$ are the lower and upper bound of a straight road respectively. Moreover the crash is modelled with the following state constraint:

$$\begin{array}{l} \alpha + (x_{C2}(t) - x_{C1}(t))^2 + (y_{C2}(t) - y_{C1}(t))^2 \\ -(r_{C2} + r_{C1})^2 \geq 0, \\ \alpha \geq 0, \end{array}$$

α is a constant state. We have to specify that $x_{C2}(t)$ and $y_{C2}(t)$ are such that:

$$\begin{array}{l} x_{C2}(t) = x_{C2}(t_0) + v_{C2}(t_0)t + \frac{1}{2}a_{C2}t^2, \\ y_{C2}(t) = y_{C2}(t_0) \end{array} \tag{3.23}$$

where $(x_{C2}(t_0), y_{C2}(t_0))$ is the initial position of $C2$, $v_{C2}(t_0)$ is the initial velocity of $C2$ and a_{C2} is its constant acceleration.

- **Differential equations.** Objects $C1$ is following the car model described in (3.3):

$$\begin{array}{rcl} z'_{C1}(t) & = & f_C(z_{C1}(t), u_{C1}(t)) + p_1(t), \text{ a.e in } [t_0, t_f] \\ z_{C1}(t_0) & = & z_{C1,0} + p_2(t). \end{array} \tag{3.24}$$

As before the following equation is added which tells that α is a constant state:

$$\alpha' = 0. \tag{3.25}$$

- **Other constraints.** As in (3.3):
  - Perturbations in the dynamics f_C are modeled by the vector
$$p_1(t) \in \mathcal{B}(\gamma_{p_1}, B_{p_1}), \tag{3.26}$$
    the ball of radius γ_{p_1} centered in B_{p_1}.
  - Perturbations in the initial value $z_C(t_0)$ are modeled by the vector
$$p_2(t) \in \mathcal{B}(\gamma_{p_2}, B_{p_2}), \tag{3.27}$$
    the ball of radius γ_{p_2} centered in B_{p_2}.
- **Objective function.** While $C1$ is trying to avoid $C2$ and reach the target T_{C1} staying on the road, p is trying to make $C1$ violating the constraints, so:
$$\int_{t_0}^{t_f} \max\{0, (r_P + r_C)^2 - (x_P(t) - x_C(t))^2 - (y_P(t) - y_C(t))^2\}^2 dt \tag{3.28}$$
  is minimized over u_{C1} and maximized over p.

Bibliography

[1] Althoff, M., Stursberg, O., and Buss, M. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *Proceedings of 47th IEEE Conference on Decision and Control* (2008), pp. 4042–4048.

[2] Althoff, M., Stursberg, O., and Buss, M. Model-based probabilistic collision detection in autonomous driving. *IEEE Transactions Intelligent Transportation Systems 10* (2009), 299–310.

[3] Ascher, U. M., Mattheij, R. M., and Russell, R. D. *Numerical solution of boundary value problems for ordinary differential equations.* SIAM, 1995.

[4] Aubin, J. P. *Viability theory.* Springer, 2009.

[5] Aubin, J.-P., Bayen, A. M., and Saint-Pierre, P. *Viability theory: New directions.* Springer, 2011.

[6] Aubin, J.-P., and Frankowska, H. The viability kernel algorithm for computing value functions of infinite horizon optimal control problems. *Journal of Mathematical Analysis and Applications 201* (1996), 555–576.

[7] Aubin, J.-P., and Frankowska, H. *Set-valued analysis.* Springer, 2009.

[8] Aubin, J.-P., and Saint-Pierre, P. Viability kernels and capture basins for analyzing the dynamic behavior: Lorenz attractors, Julia sets, and Hutchinson's maps. In *Differential equations, chaos and variational problems.* Springer, 2008, pp. 29–47.

[9] Baier, R., and Gerdts, M. A computational method for non-convex reachable sets using optimal control. In *Proceedings of the European Control Conference* (2009), pp. 23–26.

[10] Baier, R., Gerdts, M., and Xausa, I. Approximation of reachable sets using optimal control algorithms. *Preprint 35* (2011).

[11] Bardi, M., and Capuzzo-Dolcetta, I. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations.* Springer, 2008.

[12] Barron, E., and Ishii, H. The Bellman equation for minimizing the maximum cost. *Nonlinear Analysis: Theory, Methods & Applications 13* (1989), 1067–1090.

[13] Bellman, R. Dynamic programming and Lagrange multipliers. *National Academy of Sciences of the United States of America 42* (1956), 767–769.

[14] Betts, J. T. *Practical methods for optimal control and estimation using nonlinear programming.* SIAM, 2010.

[15] Bock, H. G., and Plitt, K.-J. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings of IFAC World Congress* (1984).

[16] Bokanowski, O., Désilles, A., and Zidani, H. Roc-hj solver numerical parallel library for solving hamilton-jacobi equations. http://uma.ensta-paristech.fr/files/ROC-HJ/, 2011.

[17] Bokanowski, O., Forcadel, N., and Zidani, H. Reachability and minimal times for state constrained nonlinear problems without any controllability assumption. *SIAM Journal on Control and Optimization 48* (2010), 4292–4316.

[18] Bokanowski, O., and Zidani, H. Minimal time problems with moving targets and obstacles. In *Proceedings of 18th IFAC World Congress* (2011), pp. 2589–2593.

[19] Bonneuil, N. Computing the viability kernel in large state dimension. *Journal of Mathematical Analysis and Applications 323* (2006), 1444–1454.

[20] Bonneuil, N. Maximum under continuous-discrete-time dynamic with target and viability constraints. *Optimization 61* (2012), 901–913.

[21] Büskens, C. Users's guide for the fortran subroutines bdsco, nudocccs, kollokat, diroptim and dirmesch. Tech. rep., Universität Münster, Germany, 1994.

[22] Büskens, C. *Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer-und Zustands-Beschränkungen.* PhD thesis, Universität Münster, Germany, 1998.

[23] Büskens, C., and Maurer, H. SQP-methods for solving optimal control problems with control and state constraints: Adjoint variables, sensitivity analysis and real-time control. *Journal of computational and applied mathematics 120* (2000), 85–108.

[24] Büskens, C., and Maurer, H. Sensitivity analysis and real-time optimization of parametric nonlinear programming problems. In *Online optimization of large scale systems.* Springer, 2001, pp. 3–16.

[25] Butcher, J. C. *Numerical methods for ordinary differential equations.* John Wiley & Sons, 2008.

[26] Cardaliaguet, P., Quincampoix, M., and Saint-Pierre, P. Optimal times for constrained nonlinear control problems without local controllability. *Applied Mathematics and Optimization 36* (1997), 21–42.

[27] Cardaliaguet, P., Quincampoix, M., and Saint-Pierre, P. Numerical schemes for discontinuous value functions of optimal control. *Set-Valued Analysis 8* (2000), 111–126.

[28] Clarke, F. H. *Optimization and nonsmooth analysis.* SIAM, 1990.

[29] Coelingh, E., Jakobsson, L., Lind, H., and Lindman, M. Collision warning with auto brake: A real-life safety perspective. In *Proceedings of the 20th International Technical Conference on the Enhanced Safety of Vehicles* (2007).

[30] Coron, J.-M. *Control and nonlinearity.* American Mathematical Society, 2009.

[31] Desaraju, V., Ro, H. C., Yang, M., Tay, E., Roth, S., and Del Vecchio, D. Partial order techniques for vehicle collision avoidance: Application to an autonomous roundabout test-bed. In *Proceedings of IEEE International Conference on Robotics and Automation* (2009), pp. 82–87.

[32] Désilles, A., Zidani, H., and Crück, E. Collision analysis for an UAV. In *Proceedings of AIAA Guidance, Navigation, and Control Conference* (2012).

[33] Deuflhard, P., Pesch, H.-J., and Rentrop, P. A modified continuation method for the numerical solution of nonlinear two-point boundary value problems by shooting techniques. *Numerische Mathematik 26* (1976), 327–343.

[34] Distner, M., Bengtsson, M., Broberg, T., and Jakobsson, L. City safety - a system addressing rear-end collisions at low speeds. In *Proceedings of 21st International Technical Conference on the Enhanced Safety of Vehicles* (2009).

[35] Doi, A., Butsuen, T., Niibe, T., Takagi, T., Yamamoto, Y., and Seni, H. Development of a rear-end collision avoidance system with automatic brake control. *JSAE Review 15* (1994), 335–340.

[36] Donzé, A., and Maler, O. Systematic simulation using sensitivity analysis. In *Hybrid systems: Computation and control.* Springer, 2007, pp. 174–189.

[37] Fabien, B. C. DSOA: The implementation of a dynamic system optimization algorithm. *Optimal Control Applications and Methods 31* (2010), 231–247.

[38] Falcone, M., Giorgi, T., and Loreti, P. Level sets of viscosity solutions: Some applications to fronts and rendez-vous problems. *SIAM Journal on Applied Mathematics 54* (1994), 1335–1354.

[39] Falcone, M., and Soravia, P. Appendix A. Numerical solution of dynamic programming equations - Appendix B. Nonlinear $\mathcal{H}_\infty$ control. In *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations.* Birkhäuser, 1997, pp. 471–574.

[40] Fayolle, P.-A., Pasko, A., and Schmitt, B. SARDF: Signed approximate real distance functions in heterogeneous objects modeling. In *Heterogeneous objects modelling and applications.* Springer, 2008, pp. 118–141.

[41] Fiacco, A. V. *Introduction to sensitivity and stability analysis in nonlinear programming.* Elsevier, 1983.

[42] Frankowska, H., and Plaskacz, S. Semicontinuous solutions of Hamilton-Jacobi-Bellman equations with degenerate state constraints. *Journal of Mathematical Analysis and Applications 251* (2000), 818–838.

[43] Frankowska, H., and Quincampoix, M. Viability kernels of differential inclusions with constraints: Algorithm and applications. *Journal of Mathematical Systems, Estimation, and Control 1* (1991), 371–388.

[44] Fujita, Y., Akuzawa, K., and Sato, M. Radar brake system. *JSAE Review 16* (1995).

[45] Gerdts, M. Direct shooting method for the numerical solution of higher-index DAE optimal control problems. *Journal of Optimization Theory and Applications 117* (2003), 267–294.

[46] Gerdts, M. Solving mixed-integer optimal control problems by branch&bound: A case study from automobile test-driving with gear shift. *Optimal Control Applications and Methods 26* (2005), 1–18.

[47] Gerdts, M. A variable time transformation method for mixed-integer optimal control problems. *Optimal Control Applications and Methods 27* (2006), 169–182.

[48] Gerdts, M. Ocpid-dae1 optimal control and parameter identification with differential-algebraic equations of index 1. http://www.optimal-control.de, 2011.

[49] Gerdts, M. *Optimal control of ODEs and DAEs.* Walter de Gruyter, 2012.

[50] Gerdts, M., Henrion, R., Hömberg, D., and Landry, C. Path planning and collision avoidance for robots. *Numerical Algebra, Control and Optimization 2* (2012), 437–463.

[51] Gerdts, M., Karrenberg, S., Müller-Bessler, B., and Stock, G. Generating locally optimal trajectories for an automatically driven car. *Optimization and Engineering 10* (2009), 439–463.

[52] Gerdts, M., and Xausa, I. Avoidance trajectories using reachable sets and parametric sensitivity analysis. In *System modeling and optimization.* Springer, 2013, pp. 491–500.

[53] Girsanov, I. V. *Lectures on mathematical theory of extremum problems.* Springer, 1972.

[54] Hargraves, C. R., and Paris, S. W. Direct trajectory optimization using nonlinear programming and collocation. *AIAA Journal of Guidance, Control, and Dynamics 10* (1987), 338–342.

[55] Hart, J. C. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer 12* (1996), 527–545.

[56] Hartl, R. F., Sethi, S. P., and Vickson, R. G. A survey of the maximum principles for optimal control problems with state constraints. *SIAM Review 37* (1995), 181–218.

[57] Hattori, Y., Ono, E., and Hosoe, S. Optimum vehicle trajectory control for obstacle avoidance problem. *IEEE/ASME Transactions Mechatronics 11* (2006), 507–512.

[58] Hentschel, M., Wulf, O., and Wagner, B. A hybrid feedback controller for car-like robots-combining reactive obstacle avoidance and global replanning. *Integrated Computer-Aided Engineering 14* (2007), 3–14.

[59] Hsu, C., Liang, C., Ke, L., and Huang, F. Verification of on-line vehicle collision avoidance warning system using DSRC. *World Academy of Science, Engineering and Technology 55* (2009), 377–383.

[60] Hui, N. B., Mahendar, V., and Pratihar, D. K. Time-optimal, collision-free navigation of a car-like mobile robot using neuro-fuzzy approaches. *Fuzzy Sets and Systems 157* (2006), 2171–2204.

[61] Ioffe, A. D., Tikhomirov, V. M., and Makowski, K. *Theory of extremal problems.* Elsevier, 2009.

[62] Joseph, N. W. Agilent engineering excellence program: Collision avoidance system.

[63] Jula, H., Kosmatopoulos, E. B., and Ioannou, P. A. Collision avoidance analysis for lane changing and merging. *IEEE Transactions Vehicular Technology 49* (2000), 2295–2308.

[64] Kim, Z. Robust lane detection and tracking in challenging scenarios. *IEEE Transactions Intelligent Transportation Systems 9* (2008), 16–26.

[65] Kopischke, S. *Entwicklung einer Notbremsfunktion mit Rapid-Prototyping-Methoden.* PhD thesis, TU Braunschweig, Germany, 2000.

[66] Lachner, R., Breitner, M. H., and Pesch, H. J. Real-time collision avoidance: Differential game, numerical solution, and synthesis of strategies. In *Advances in dynamic games and applications.* Springer, 2000, pp. 115–135.

[67] Landry, C., Gerdts, M., Henrion, R., and Hömberg, D. Path-planning with collision avoidance in automotive industry. In *System modeling and optimization*. Springer, 2013, pp. 102–111.

[68] Lee, K., and Peng, H. Evaluation of automotive forward collision warning and collision avoidance algorithms. *Vehicle System Dynamics 43* (2005), 735–751.

[69] Lygeros, J. On reachability and minimum cost optimal control. *Automatica 40* (2004), 917–927.

[70] Maidens, J. N., Kaynama, S., Mitchell, I. M., Oishi, M. M., and Dumont, G. A. Lagrangian methods for approximating the viability kernel in high-dimensional systems. *Automatica 49* (2013), 2017–2029.

[71] Mitchell, I. M. A toolbox of level set methods. Tech. rep., University British Columbia, Vancouver, BC, Canada, 2004.

[72] Mitchell, I. M. Comparing forward and backward reachability as tools for safety analysis. In *Hybrid systems: Computation and control*. Springer, 2007, pp. 428–443.

[73] Mitchell, I. M., Bayen, A. M., and Tomlin, C. J. A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions Automatic Control 50* (2005), 947–957.

[74] Nilsson, J., Fredriksson, J., and Odblom, A. C. Verification of collision avoidance systems using reachability analysis. In *Proceedings of 19th IFAC World Congress* (2014), pp. 10676–10681.

[75] Oberle, H. J., and Grimm, W. *BNDSCO: A program for the numerical solution of optimal control problems*. Institut für Angewandte Mathematik, 2001.

[76] Osher, S., and Shu, C.-W. High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations. *SIAM Journal on Numerical Analysis 28* (1991), 907–922.

[77] Otte, D., Krettek, C., Brunner, H., and Zwipp, H. Scientific approach and methodology of a new in-depth investigation study in germany called GIDAS. In *Proceedings of International Technical Conference on the Enhanced Safety of Vehicles* (2003), p. 10.

[78] Pacejka, H. B., and Bakker, E. The magic formula tyre model. *Vehicle system dynamics 21* (1992), 1–18.

[79] Pesch, H. J., and Bulirsch, R. The maximum principle, Bellman's equation, and Carathéodory's work. *Journal of Optimization Theory and Applications 80* (1994), 199–225.

[80] Pontryagin, L. S. *Mathematical theory of optimal processes.* CRC Press, 1987.

[81] Posa, M., and Tedrake, R. Direct trajectory optimization of rigid body dynamical systems through contact. In *Algorithmic foundations of robotics X.* Springer, 2013, pp. 527–542.

[82] Pytlak, R. Runge-Kutta based procedure for the optimal control of differential-algebraic equations. *Journal of Optimization Theory and Applications 97* (1998), 675–705.

[83] Quincampoix, M., and Saint-Pierre, P. An algorithm for viability kernels in Hölderian case: Approximation by discrete dynamical systems. *Journal of Mathematical Systems Estimation and Control 8* (1998), 17–30.

[84] Rockafellar, R. T. *Convex analysis.* Princeton University Press, 1997.

[85] Rockafellar, R. T., Wets, R. J.-B., and Wets, M. *Variational analysis.* Springer, 1998.

[86] Ross, I. M., and Fahroo, F. A unified computational framework for real-time optimal control. In *Proceedings of 42nd IEEE Conference on Decision and Control* (2003), pp. 2210–2215.

[87] Saint-Pierre, P. Approximation of the viability kernel. *Applied Mathematics and Optimization 29* (1994), 187–209.

[88] Seiler, P., Song, B., and Hedrick, J. K. Development of a collision avoidance system. *Development 4* (1998), 17–22.

[89] Sivaraman, S., and Trivedi, M. M. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions Intelligent Transportation Systems 14* (2013), 1773–1795.

[90] Sivaraman, S., and Trivedi, M. M. Dynamic probabilistic drivability maps for lane change and merge driver assistance. *IEEE Transactions Intelligent Transportation Systems 15* (2014), 2063–2073.

[91] Soner, H. M. Optimal control with state-space constraint I. *SIAM Journal on Control and Optimization 24* (1986), 552–561.

[92] Subchan, S., and Zbikowski, R. *Computational optimal control: Tools and practice.* John Wiley & Sons, 2009.

[93] Sussmann, H. Geometry and optimal control. In *Mathematical control theory.* Springer, 1999, pp. 140–198.

[94] Teo, K., Jennings, L., Lee, H., and Rehbock, V. The control parameterization enhancing transform for constrained optimal control problems. *The Journal of the Australian Mathematical Society. Series B. Applied Mathematics 40* (1999), 314–335.

[95] Tilove, R. B. Local obstacle avoidance for mobile robots based on the method of artificial potentials. In *Proceedings of IEEE International Conference on Robotics and Automation* (1990), pp. 566–571.

[96] Tomlin, C., Mitchell, I., and Ghosh, R. Safety verification of conflict resolution manoeuvres. *IEEE Transactions Intelligent Transportation Systems 2* (2001), 110–120.

[97] Trélat, E. *Contrôle optimal: Théorie & applications.* Vuibert, 2005.

[98] Vinter, R. *Optimal control.* Springer, 2010.

[99] Volkswagen. Self-study Programme 470 - the Touareg 2011 - electrics/electronics - ACC/front assist. Tech. rep., Volkswagen AG, 2011.

[100] Volkswagen. Self-study Programme 488 - the Passat 2011 - ACC/front assist. Tech. rep., Volkswagen AG, 2011.

[101] Volkswagen. Self-study Programme 516 - the Golf 2013 - driver assist systems design and function. Tech. rep., Volkswagen AG, 2013.

[102] Winner, H. Darmstaedter kolloquium "mensch und fahrzeug". Tech. rep., TU Darmstadt, 2011.

[103] Wright, S., and Nocedal, J. *Numerical optimization.* Springer, 1999.

[104] Xausa, I., Baier, R., Bokanowski, O., and Gerdts, M. Computation of safety regions for driver assistance systems by using a Hamilton-Jacobi approach. *Submitted* (2014), 22 pp.

[105] Xausa, I., Baier, R., Gerdts, M., Gonter, M., and Wegwerth, C. Avoidance trajectories for driver assistance systems via solvers for optimal control problems. In *Proceedings of 20th International Symposium on Mathematical Theory of Networks and Systems* (2012).

[106] Yang, L., Yang, J. H., Feron, E., and Kulkarni, V. Development of a performance-based approach for a rear-end collision warning and avoidance system for automobiles. In *Proceedings of IEEE Intelligent Vehicles Symposium* (2003), pp. 316–321.

[107] Zeidan, V. Second order admissible directions and generalized coupled points for optimal control problems. *Nonlinear Analysis: Theory, Methods & Applications 25* (1996), 479–507.